Universiteit Leiden

Opleiding Informatica

First Steps towards

Hybrid Quantum-Classical Optimization

Name: Dapeng Wang

Date: 20/09/2019

1st supervisor: Prof. dr. Thomas Bäck
2nd supervisor: Dr. Florian Neukart

MASTER THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

First Steps towards Hybrid Quantum-Classical Optimization

Universiteit
Leiden
Leiden Institute of
Advanced Computer Science

Dapeng Wang

LIACS

Leiden University

A thesis submitted for the degree of

Computer Science

2019 September

Abstract

After the breakthroughs of quantum computers and quantum algorithms, quantum neural network and quantum-assisted optimization are expected to be general-purpose methods and applications. Recently, significant breakthroughs in the field of quantum computing are specialized quantum computers such as D-Wave, and quantum optimization algorithms such as quantum approximate optimization algorithm(QAOA). In this master thesis, three hypotheses are proposed, and the corresponding experiments are finished, which show the results for the hypotheses. First, a regression model is organized for microstructure battery dataset, and it is driven by the classical neural network. Then I turn to quantum approaches. Hypothesis supposes there should be quantum computing approaches which can accelerate the training process and get a high accuracy value, either using a quantum neural network or quantum assisted algorithms. The results show that for machine learning tasks, specific quantum computing approaches outperform some conventional approaches, such as the classical neural network. However, due to the current quantum hardware limitations, large-scale quantum neural networks and large-scale quantum circuits are not available in contemporary technology.

Keywords Quantum Computing, Quantum Assisted Optimization, Quantum Approximate Optimization Algorithm(QAOA), Quantum Neural Network.

Acknowledgements

This paper was completed under the guidance and supervision of my mentor, Dr. Florian Neukart. He gave me a model of broad and profound knowledge, serious scientific attitude and rigorous academic spirit, which has had a profound and lasting impact on me. From the selection of the subject to the final completion of the project, Dr. Florian Neukart has always given me careful and patient guidance, which not only enabled me to establish ambitious academic goals, but also made me understand many of the life. I would like to express my utmost sincere gratitude and heartfelt thanks to Dr. Florian Neukart.

Secondly, the successful completion of this paper is inseparable from the care and help of Prof. Dr. Thomas Bäck and the co-workers in San Francisco Code: Lab, Volkswagen Group of America. I want to thank them for guidance and help. Thanks for their sincere dedication, which has made me more successful in my studies, and made me feel the warmth in my life.

The life of a master student is coming to an end. In the two years of school, I cherished the opportunity to study in a university, worked diligently, and greatly improved my ability to solve problems. I am grateful to the professors for teaching knowledge to me and all the helpful suggestions from my classmates. I appreciate all the experience during this time, either in Leiden or in San Francisco, which I will never forget.

Once again, I want to thank my parents who have cultivated me. They not only provide me with a warm home, but also prepare sailboats for me to sail at any time. I appreciate all the things that my parents have done for me. It is their supports that give me the opportunity to study, and it is their love that encourages me and makes me optimistic and confident. My parents' health and safety is my greatest wish.

Finally, I want to thank all the specialists experts who took the time to participate in the paper review and defense. Thank you!

Contents

List of Figures

List of Tables

1

Introduction

Recently, researchers and industry have identified that classical computing chips may hit an inflection point[10, 72], where computational power is limited by the laws of thermodynamics and quantum effects on transistors on microprocessors which are hitting limits in regards to miniaturization and enumeration on computing chips[16, 69].

In the meantime, since the 1980s, physicists, computer scientists, and other research communities have identified quantum computing[12, 34], which uses the quantum effects of entanglement and superposition as a way to perform computation, as a possible way to mitigate these limits on computational power[34, 36]. These have resulted in a boom in new fields of research into quantum algorithms and hardware which leverage these approaches and effects[20, 27, 66, 84, 86]. The current state of the art has seen emerging applications and algorithms on Noisy Intermediate-Scale Quantum devices (NISQ) for combinatorial optimization problems[23, 35, 85], and machine learning[14, 32, 74, 75, 99, 103], as well as benchmarking and testing current hardware and algorithms[14, 28].

At the same time, over the past 10-15 years, the field of Artificial Intelligence and Machine Learning has undergone a renaissance with the advent of increased access to data and computational resources, namely graphics processing units (GPUs) for parallel computation[90]. These advances have resulted in state of the art performance in computer vision, planning, reinforcement learning, and natural language understanding amongst other domains[50].

1.1 History of Quantum Computing

The history of quantum computing began in the early 20^{th} century, when physicists began to realize that they had lost control of reality. In the 20^{th} century, there were two major crises in physics. One of them was the 'Ultraviolet Disaster'[100]. Physicists discussed a quarter of a century and created the modern physics theory of quantum mechanics[15, 29, 33]. What is quantum mechanics? Quantum mechanics is a discipline in physics that studies the laws governing the motion of microscopic materials. Whether quantum computing can be realized depends on the degree of control over microscopic particles. Therefore, more in-depth research on quantum computing is needed. The concept of quantum computing and quantum computers was first proposed by the famous physicist Feynman in 1981[34]. By representing information in superposition [18], where bits could take on a value of 0 and 1, 0 or 1, a quantum computer could simulate systems that would not be possible for a classical machine. The fundamental theories and blueprints of quantum computers formed in the 1980s and 1990s still guide Google and others who are engaged in this technology.

The principle of quantum computing should actually be divided into two parts. Part of it is the physical principle and physical realization of quantum computers, the other part is quantum algorithms. In the early 1990s, a suite of algorithms was proposed for these quantum machines, which theoretically showed speedups for procedures such as searching databases and factoring integers[28, 34].

Quantum mechanics studies the field of physic and the behavior of most essential and smallest parts of our universe at the subatomic level[39]. Quantum Computers is a straightforward physical system, which locates in a discrete domain, not in the continuous domain[19]. Quantum computers are made up of a finite number of units, each of which is a two-state-system, which also can be considered as the most straightforward Quantum Mechanic System. Getting a feeling for how formalism can be applied to actual phenomena is the main difficulty in Quantum Mechanics. Solving oversimplified abstract models of real physical systems, to which the quantum formalism can be applied[61].

In 1936, there were only a few people thought of a programmable computer, after Turing came up with the idea with an application to the Entscheiidungs problem[91], machines executing billions of operations per second, which is so crazy. Even after transistor, no one thought digital computers would be as ubiquitous and useful as they are today. According to Moore's law, the feature size on silicon chips is cut in half every two years[71], the force behind this prophecy is that, if we blindly extrapolate it, around 2,050 feature size of computers would need to be the size of an atom[8]. Even today, molecular-size transistors exist, which is not very reliable and far from being used. This is an indication towards, computers with components of atomic size. However, energies considerably higher than in atomic systems required. If the computer becomes diminutive size, we need to learn to engineer such systems, such as molecular transistors[102], and these systems function precisely under the laws of **Quantum Theory**[19]. And we can use quantum computers to achieve better efficiency at specific tasks, as is shown in figure 1.1, with the input size increases, the quantum system performs a linear increase while the classical system performs an exponential increase, and there are two kinds of constraints for the linear growth, one of them is the size of the input set, another is the limitation of classical system hardware or quantum system hardware.

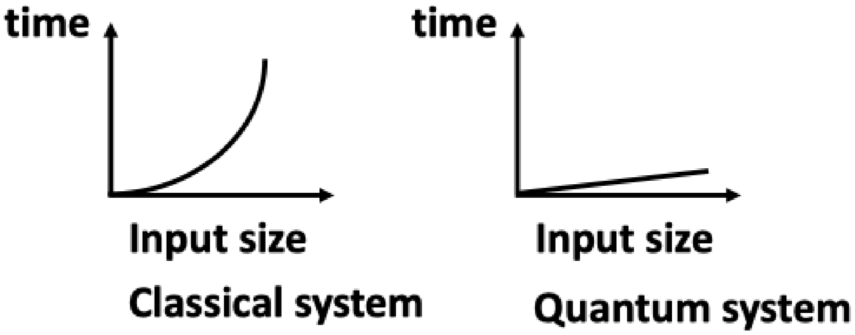


Figure 1.1: Comparison between Classical System and Quantum System. - Running time

1.2 State of Research

This thesis will focus on two kinds of research: a construction of quantum neural network and a trial of solving the quantum assisted optimization problem, comparing the performance of the classical neural network and the quantum neural network with the same dataset, and presenting the results of the quantum assisted optimization problem.

1.3 Thesis Structure

The thesis structure is organized as follows: Section 2 will introduce some related work on quantum computing, quantum algorithms, and the evolution of quantum hardware. This Section will introduce what was done in other papers, the relationship between the previous work and current topics. Section 3 will give some definitions, explanations, and examples of the quantum mechanics which are used in this thesis. Section 4 will present a microstructure battery dataset and Section 5 will discuss the problem statements and some hypotheses about this research. Section 6 includes four kinds of different experiments which can provide experiment results. Conclusion and future work will be discussed in Section 7.

2

Related Work

In this chapter, some related work about quantum computing will be presented, such as quantum mechanics, adiabatic quantum computing, the universal gate quantum computing and the evolution of quantum hardware.

2.1 Quantum Mechanics

In contrast to the classical computer, in which the smallest unit, the bit, can only take the values 0 or 1, the quantum computer uses co-called qubits(quantum bits). Qubits can simultaneously assume these values 0 and 1. This is called superposition[63]. In the case of several Qubits, there is the possibility of interlocked states. A classical computer with N bits has a state space with 2^N elements, a quantum computer with N qubits, but a state space with 2^N dimensions, which corresponds to any combination of all possible states of the classical computer[95]. The dynamics of isolated quantum systems are linear, and roughly a quantum computer can simultaneously compute all the function values of a function(this is the 'parallelism'). At the end of a calculation, however, measurement has to be carried out which delivers only one result from all possible results. The challenge of algorithms for quantum computers is to carry out transformations such that the quantum computer provides the answer to a question[51].

There are different technologies to build quantum computers. The qubit can be prepared by the two-dimensional subspace of the state space as of superconducting islands with a Josephson Function[59, 97](SQUID - superconducting quantum interference device), or of atoms/irons are shown(in a case), or of quantum wells or the polarization fo a photon. On the one hand, qubits must be actively shielded from the environment. Otherwise, decoherence destroys the quantum state directly. On the other hand, they must be checked with other qubits and interact in the preparation and the measurement with the surroundings during the bill[83]. Without quantum error correction, no realistic calculation is carried out. In the quantum error correction, raw physical qubits are grouped into error-corrected logical qubits[44, 89]. With the latter then, the calculations are performed.

For the construction of a quantum computer, there are various paradigms. Both are the subject of research and development:

- Quantum Gate Model. Accounting on quantum circuits.
- Adiabatic theorem. Statement about the change of energy potential i.e. by quantum annealing.

Significantly, the Adiabatic Quantum Computing is basically equivalent to polynomial quantum gate model. In certain applications, a quantum computer ahs an exponential advantage compared to classical computers. Known quantum algorithms include Shor for the prime factorization or Grover for the database search. The necessary hardware on which to run these algorithms but not yet exist or is only at the research stage. The development of quantum algorithms is operated in the coming years with more emphasis to identify new applications with the goal.

Possible applications include[14, 32, 74, 75, 99, 103]:

- Pattern recognition, machine learning, optimization problems.
- Development of materials and catalysts by quantum simulation or calculation of the properties of materials by means of a quantum computer.
- Support for research and development of artificial intelligence

A first objective could be the proof of 'quantum supremacy,' i.e., a quantum computer solving specific problems faster than the fastest available classical computers[17]. In a universal quantum computer about 50 logical qubits are necessary for achieving this target without auxiliary bits. This goal could be achieved already in 5–10 years. Looking ahead, the current technology is advanced already massive, and it is expected that further increases in speed over a hybrid quantum-classical approach to gain importance[77].

Regarding industrial optimization problems, adiabatic quantum computing or quantum annealing systems are already up-and-coming today[67, 74, 75]. When adiabatic quantum calculation by the adiabatic evolution of a qubit-system, is implemented. The system is simply initialized in the ground state[1] of a Hamiltonian. Thereafter, the system is adiabatically developed towards the 'problem' -Hamiltonian whose place slowly enough maps the solution of the optimization problems. After the adiabatic theorem, the system(described by one over time changing Hamiltonian) remains in the ground state, as long as the development takes place slowly enough. Optimization problems, and thus most of the problems in machine learning (including pattern recognition, natural language understanding, forecasting model, clustering, computer version) can thus be expressed as energy value problems[41, 70]. This means that the quantity to be minimized (about the difference from the current to the required output of algorithm) is represented as an energy surface. The minimum energy state (which is also the optimal solution), the system can always find adequate representation problem is, therefore, the output of the calculation. Currently, classical optimization allows us now in a position to find a solution. However, the quantum annealer hardware and algorithm are in, and through the use of tunneling play local minima not matter - the optimum is always found[82]. This is not the case with classical algorithms. In addition, the time required of a classical algorithm of the FLOP/s[2] the used hardware and the possible parallelization of the algorithms depends. A quantum annealer allows you to find the solution instantaneously[73].

In quantum annealers, therefore, problems that can be very interesting for diverse industries can be represented[14, 32, 74, 75, 99, 103]: optimization problems, learning, and stochastic simulations. Even the prime-problem can be solved with the quantum annealer but using a different algorithm than the Shor's[80].

Literally, the quantum computer is expected to be more powerful than the classical ones in some cases. The strengths of quantum computers are the generation of random numbers[49], the search for the minimum of unordered series[7], and the problem of

[1] The lowest energy state.

[2] In computing, floating-point operations per second (FLOPS, flops or flop/s) is a measure of computer performance, from Wikipedia.

node connection in graph theory[58], etc.. Scientists have designed a variety of quantum algorithms to solve problems which are difficult for classical computers. These quantum algorithms are practically applied in the transportation[73], medical[1], and financial markets[42] in the past decades. Also, quantum computers can be used to help design the catalyst for clean energy, understand enzymes in living organisms, and discover new solar cell materials or high-temperature superconductor materials. Its strength lies in the powerful computing power of existing traditional computers.

As we all know, classical computers are currently using binary-based system[13]. Through the binary counting method, the two terms of a bit 0 and bit 1 are represented in a classical computer. However, quantum computers are entirely different. Basic unit of a quantum computer is called qubit, and the information stored in the qubit can be both 0 and 1. Therefore, a qubit can represent two numbers of 0 and 1 at the same time. Two qubits can represent $0, 1, 2, 3$ simultaneously, so N qubits can represent 2^N numbers, as N increases, the ability to represent information will increase exponentially, it means the quantum runs at a speed that is N times the traditional computer.

2.2 Gate Model Quantum Computing

Gate model quantum computing is also called the universal quantum computing model, and the workflow of gate model quantum computing[38] can be illustrated in Fig 2.1. Suppose we have a problem definition, for example, the traveling salesman problem, which aims at finding the shortest way of visiting n cities. After the problem definition, we need to find a quantum algorithm which can solve the problem. In this thesis, Quantum Approximate Optimization Algorithm(QAOA)[30] was used. Such quantum algorithms are assigned into a quantum circuit, which consists of the quantum gates and unitary operations, and the details of quantum gates and unitary operations will be introduced in Section 3. After completing the quantum circuit, the quantum compiler allocates sets of quantum gates into quantum hardware and connects two qubits. The connection is the interaction between two qubits, and it is not the actual connection between two qubits. Once the quantum compilation is done, we can execute the program on the quantum processing unit(QPU) or on the quantum simulator[37].

As the field worked towards theoretical developments, technical developments gained momentum in the early 2000s. Many companies around the world are putting their

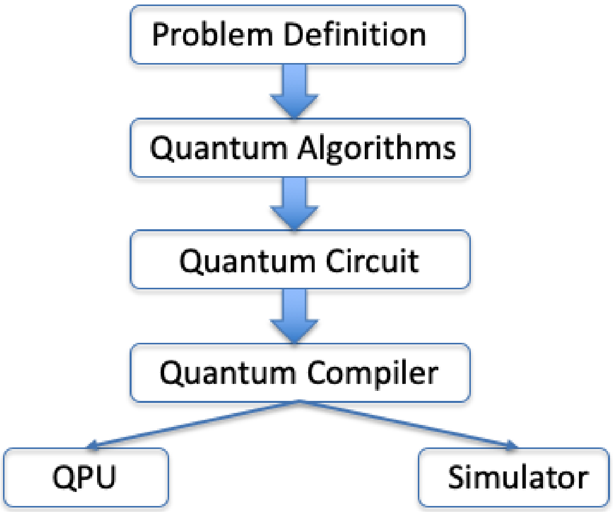


Figure 2.1: Workflow - Gate Model Quantum Computing

efforts in building a gate model quantum computer, such as Alibaba[5], Baidu[9], Google[43], IBM Q[53], Intel[55], Microsoft[68], etc.. D-Wave, a Canadian quantum computing company, formed partnerships with leading academic institutions, and in 2011 made commercially available a quantum annealing system, the D-wave One, after making some demonstrations using the Orion prototype in 2007 and 2009[54]. The annealing system was based on an earlier paper by Farhi[31], where they described an algorithm for solving instances of satisfiability problems by following a time evolution from an initial Hamiltonian to a final Hamiltonian which represented the solution to the encoded satisfiability problem[64].

As for universal gate quantum computing, the problem is being able to RandD chips and test them in an efficient way to improve coherence[21] (length of the information is stored and can be manipulated) and qubit reliability[70]. The all silicon chip breakthroughs[1] indicate that standard micro-facilities can be used to create quantum processor units, which leads towards cheaper and less specialized facilities for qubit manufacturing[21]. The companies that put their efforts in quantum mechanics can design a universal gate quantum computing chips which are specialized for their purposes.

[1] https://www.smh.com.au/technology/australian-researchers-make-quantum-computing-breakthrough-paving-way-for-worldfirst-chip-20151005-gk1bov.html

2.3 Quantum Annealing

When people talk about quantum computing, we literally talk about several different patterns. The most common paradigm is the gate model quantum computing, which is discussed in Section 2.2. Another most common paradigm is quantum annealing. There is the difference between quantum annealing and adiabatic quantum computing: the adiabatic quantum computing is a universal quantum computing paradigm. However, quantum annealing solves a more specific problem rather than a universal problem, which is much more easier[3]. The technology of quantum annealing computers is up to $2,000$ qubits in 2018. However, the gate model quantum computers can only handle less than 100 qubits. D-Wave has built superconducting quantum annealers in 2016[24], and this company has the record for the number of $2,048$ qubits on D-Wave 2000Q quantum computer. Recently, IARPA[1] launched a project to build a superconducting quantum annealers, and QNNcloud[2] also implemented a quantum optics which can deal with a coherent Ising model[11].

Over the past few years, quantum annealing inspired some gate-model algorithms which work on current near-term quantum computers[13, 103]. The workflow of quantum annealing is slightly different from the gate model quantum computing. Instead of using a quantum circuit in gate model quantum computers, quantum annealing uses the classical Ising model, which is a necessary form if researchers intend to use quantum annealing as an approach. However, quantum annealers also suffer from the limitation of connectivity[2, 4], in this case, a minor graph embedding is required, which combines some physical qubits together to form a logical qubit. The workflow of quantum annealers[37, 38] is shown in Fig 2.2:

[1] The Intelligence Advanced Research Project Activity, website:`https://www.iarpa.gov/index.php/about-iarpa`

[2] QNNcloud is a cloud service that enables using a Quantum Neural Network(QNN), website:`https://qnncloud.com/`

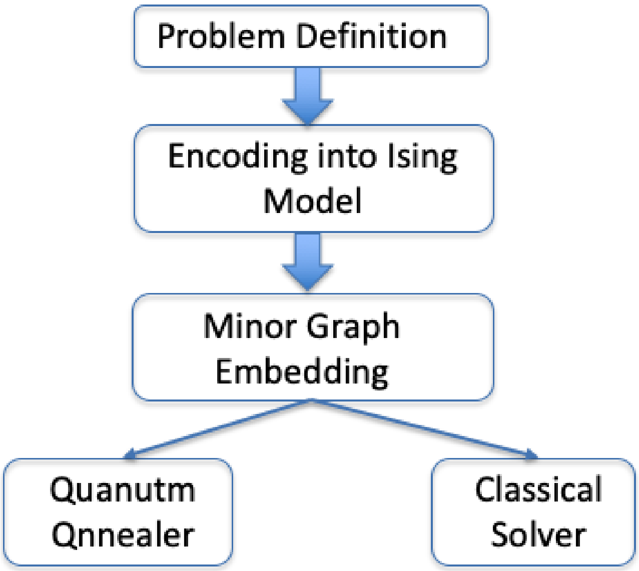


Figure 2.2: Workflow - Quantum Annealing

2.4 Quantum Hardware Evolution

As for the hardware of a quantum simulator, the existing implementation is really endless[57]. Mainstream schemes of the implementing quantum simulators are light quantum[6, 78], NMR[56], optical cavity[45], ion trap[88], superconductivity[65], etc.. But in fact, there is not a general-purpose quantum computer like a classical computer. There are two kinds of existing practical hardware:

- Quantum annealing machine.
- Quantum simulator.

The technical details of the quantum annealing machine are more complicated, except for the D-Wave[103] and a few collaborators, no one is familiar with this field of research. Currently, quantum annealing machine is the only quantum computer with commercial value. The restriction of D-Wave's quantum annealing machine is that it can only solve the Quadratic Unconstrained Binary Optimization(QUBO) problem, but there are quite a few classical problems that can be reduced to the QUBO problem, so it still meaningful for quantum computing researches. The bottleneck of the current calculation is mainly how to quickly convert to a QUBO problem with a classical computer, rather than the calculation speed of D-Wave.

We usually say that quantum computers mainly refers to quantum simulators. The so-called simulation is to control a quantum system which use classical methods to simulate quantum processes. CIRQ[43] is one of the quantum simulators which invented in this way by Google. And recently, Google starts a project called TFQuantum[43], which is also a quantum simulator and will be used in my researches. There is not any hope for general quantum computer in a short time, but if the simulation machine is big enough, quantum simulation systems can help a lot in some research field such as chemistry and biology, the quantum simulator can simulate some macromolecules, such as proteins, which cannot be simulated on a classical computer. Google's quantum supremacy is mainly aimed at this problem, the simulation system of a 49-qubits quantum system is as large as the simulation on Taihu Lake[1][81]. Thus, Google calls 50 qubits quantum computer as quantum supremacy, which is more than all the existing classic computers.

[1] This is a super classical computer whose actual performance is 93.0146PFLOP/S.

3

Preliminaries

In this chapter, some basic quantum mechanics will be recommended, which can build up the theory of quantum computing, and also some of them will be used in the later sections.

3.1 Quantum States

Consider a simple assumption, a coin is a 2-side-level system, and it is either head or tail without the situation of the coin standing on the ground. A quantum state is a probability distribution, whose simplest case has two states, which is called as *qubit*. In quantum computing, people always write a quantum state in the form of a column vector, and the label of a quantum state is called a ket in the Dirac notation[1]. Suppose there is a qubit $|\psi\rangle$ and it can be written in the form of column vector in formula 3.1:

$$|\psi\rangle = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \tag{3.1}$$

In other words, a *ket* is just a column vector, which can be understood as the stochastic vector in the classical case. In quantum computing theory, the ket is used for representing a quantum state, and the difference to classical probability distributions and stochastic vectors is the normalization constraint, the square sum of their absolute values adds up to 1:

$$\sqrt{|a_0|^2 + |a_1|^2} = 1 \tag{3.2}$$

[1] Dirac notation can describe quantum states in quantum mechanics, each quantum state is described as a state vector in Hilbert Space, namely the ket $|\psi\rangle$ and its conjugate transpose, bra $\langle\psi|$[96].

In the equantion, $a_0, a_1 \in \mathbb{C}$. In the formula 3.2, the normalizing factor is in the l_2 norm rather than l_1 norm. The components of quantum state vector are called as *probability amplitudes*, which are complex values, thus the diagram of the *probability amplitudes* are no longer restricted to the positive orthant[72].

Here comes to 2-qubits quantum system, $|0\rangle$ and $|1\rangle$ are the canonical basis vectors in two dimensions:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.3}$$

Formula 3.3 is also called the computational basis in quantum computing. According to the formula 3.1 and formula 3.3, an arbitrary qubit state in this computational basis is:

$$|\psi\rangle = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = a_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + a_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = a_0|0\rangle + a_1|1\rangle \tag{3.4}$$

Formula 3.4 is the expansion of an arbitrary qubit in a computational basis, and it is called *superposition*. Superposition is a combinational quantum state, we can obtain the outcome 0 with a probability of $|a_0|^2$, and he outcome 1 with a probability of $|a_1|^2$, this can be explained in detail as the Born rule[60].

3.2 Entanglement

In the previous section, we have already known that quantum states are probability distributions, which are normed in the l_2 norm. If involving more qubits, we can see an important quantum effect, which is called *entanglement*[52]. Given two qubits which are written in formula 3.5:

$$|\psi\rangle = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}, |\psi'\rangle = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \tag{3.5}$$

and the product of $|\psi\rangle$ and $|\psi'\rangle$ is in the form of formula 3.6:

$$|\psi\rangle \otimes |\psi'\rangle = \begin{bmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{bmatrix} \tag{3.6}$$

$|\psi\rangle$ and $|\psi'\rangle$ are the column vectors which describe two qubits, and $\otimes$ is the tensor product, in the field of quantum computing, it is the same as the Kronecker product.

Suppose there are two registers q_0 and q_1, each register can be assigned one qubit, and both of the qubits are in the $|0\rangle$ state, then the entanglement of these two qubits is $|0\rangle \otimes |0\rangle$, whose abbreviation is $|00\rangle$. By using the similar definition, we can get the states $|01\rangle$, $|10\rangle$ and $|11\rangle$. These four states form the computational basis of a four-dimensional complex space $\mathbb{C}^2 \otimes \mathbb{C}^2$.

But for some $|\psi\rangle$ and $|\psi'\rangle \in \mathbb{C}^2$, the quantum state cannot be written as $|\psi\rangle \otimes |\psi'\rangle$. Given a state in formula 3.7:

$$|\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{3.7}$$

the vector space is in $\mathbb{C}^2 \otimes \mathbb{C}^2$, suppose $|\phi^+\rangle$ can be written as formula 3.8:

$$|\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \begin{bmatrix} a_0b_0 \\ a_0b_1 \\ a_1b_0 \\ a_1b_1 \end{bmatrix} = a_0b_0|00\rangle + a_0b_1|01\rangle + a_1b_0|10\rangle + a_1b_1|11\rangle \tag{3.8}$$

$|01\rangle$ and $|10\rangle$ are not in the left side of the formula 3.8, thus the coefficients must be zero:

$$a_1b_0 = 0, a_0b_1 = 0 \tag{3.9}$$

To satisfy the equation 3.9, one of the coefficients(a_0, a_1, b_0, b_1) should be 0, but this will lead to conflicts. Because $a_1b_1 = 1$, a_1 cannot be 0. So b_0 must be 0, but $a_0b_0 = 1$, thus the state $|\phi^+\rangle$ cannot be written as product. Those states that cannot be written as a product are called entangled states. It is a mathematical form of describing a phenomenon of strong correlations between random variables that exceed what is possible classically. Entanglement plays a central role in countless quantum algorithms.

3.3 Measurement

Measurement is a vital concept in quantum mechanics, suppose it is a sample from a probability distribution, and the outcome can be produced with a certain probability. Measurement is the connection between the quantum world and the classical world, we cannot directly observe quantum states, but we can get statistics about the quantum states by using measurement[72].

Before introducing measurement in detail, we need to know a notation of ket, that is *bra*. It is the conjugate transpose of a ket, which means the bra is a row vector. Contrast with formula 3.1, the bra form of $|\psi\rangle$ is shown in formula 3.10:

$$\langle\psi| = \begin{bmatrix} a_0 & a_1 \end{bmatrix} \tag{3.10}$$

When a bra is written followed by a ket, that is the dot product, which can be written in the form of formula 3.11. Since quantum states are normalized, the inner product of any quantum state with itself is always 1:

$$\langle\psi|\psi\rangle = a_0^2 + a_1^2 = 1 \tag{3.11}$$

If one wants to calculate the outer product of two vectors, for example, $|\psi\rangle\langle\psi|$, then it should be a matrix in formula 3.12:

$$|\psi\rangle\langle\psi| = \begin{bmatrix} a_0a_0 & a_0a_1 \\ a_1a_0 & a_1a_1 \end{bmatrix} \tag{3.12}$$

A measurement is the canonical basis in a quantum mechanics, it is an operator-valued variable. The measurement contains two projections, $|0\rangle\langle 0|$ and $|1\rangle\langle 1|$, where $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\langle 0| = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $\langle 1| = \begin{bmatrix} 0 & 1 \end{bmatrix}$. If one needs a scalar value of applying a projection on a vector, we need to add bra to the left. For example, given a quantum state $|\gamma\rangle$, the scalar for the quantum state is $\langle\gamma|0\rangle\langle 0|\gamma\rangle$, which is called the expectation value of the operator $|0\rangle\langle 0|$.

3.4 Unitary Matrix

The evolution of stochastic vectors can be represented by a stochastic matrix, the evolution of quantum states can also be represented by unitary matrix[62]. Unitary evolution is a closed system, it is a quantum system that is isolated from the environment. But nowadays quantum computers are open quantum systems that have uncontrolled and unpredictable interactions with the environment.

A unitary matrix's conjugate transpose is its inverse, this is the unique property of a unitary matrix. It means a matrix U is unitary if formula 3.13 is satisfied:

$$UU^\dagger = U^\dagger U = \mathbb{I} \tag{3.13}$$

where $\dagger$ means the conjugate transpose, and $\mathbb{I}$ is the identity matrix.

A more general expression of a 2 × 2 unitary matrix is[62, 98]:

$$U = \begin{bmatrix} a & b \\ -e^{i\varphi}b^* & e^{i\varphi}a^* \end{bmatrix}, \quad |a|^2 + |b|^2 = 1 \tag{3.14}$$

The formula 3.14 depends on 4 kinds of real parameters[40]:

- The phase of a.
- The phase of b.
- The relative magnitude between a and b.
- The angle φ.

3.5 Gate Model

So far, we have understood the notation of quantum mechanics and quantum computing. In this section, we will meet some basics of gate-model quantum computing, which is also regarded as universal quantum computing.

Circuits are composed of qubit registers, on which the quantum gates are applied. In quantum simulators, such as Cirq and TFQuantum[43], the index of qubit registers starts from 0, but we often say qubit 0, qubit 1, etc., that refers to the registers which contain a qubit state, for example, qubit 1 on the register can be in a qubit state of $|1\rangle$. In quantum computing, any unitary operation can be represented by quantum elementary gates, for the current research, three types of quantum gates are enough. Table 3.1 shows some common quantum gates, and all of them are unitary matrices[25].

In fact, in Table 3.1, most of the quantum gates are single-qubit operations, and the NOT gate is the only non-trivial single-bit gate. The CNOT gate and SWAP gate are two-qubit gates in the table, and it requires two qubits interactions to create the entanglement. Fig 3.1 shows a simple example of applying quantum gates on a quantum circuit:

The Hadamard gate creates a superposition $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ in qubit 0, which controls an X gate on qubit 1. Because qubit 0 is in the equal superposition after Hadamard gate, it will not apply the X gate for $|0\rangle$ but it will apply the X gate for $|1\rangle$. By using this approach, we can create the final state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, which entangles two qubit registers.

Table 3.1: Quantum logic gates.

Gate	Name	Matrix
X	Pauli-X or Not gate	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Y	Pauli-Y gate	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Z	Pauli-Z gate	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
H	Hadamard gate	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Rx(θ)	Rotation around X	$\begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$
Ry(θ)	Rotation around Y	$\begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$
CNOT, CX	Controlled-NOT gate	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
SWAP	Swap gate	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

3.6 Hamiltonian

Hamiltonian gives a description of a system evolving with time[94], the form of this equation is expressed by the Schr$\ddot{o}$dinger equation[26]:

$$i\hbar\frac{d}{dt}|\psi(t)\rangle = H|\psi(t)\rangle \tag{3.15}$$

where $\hbar$ is the reduced Planck constant. In the previous section, unitary operators are introduced for evolving quantum states, and if we solve Schr$\ddot{o}$dinger equation for a time t, the unitary operation can be:

$$U = e^{\frac{-iHt}{\hbar}} \tag{3.16}$$

where the Hamiltonian operator does not depend on time. Formula 3.16 indicates that every unitary in Table 3.1 has some underlying Hamiltonian. The Schr$\ddot{o}$dinger equation

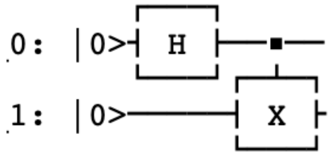


Figure 3.1: Quantum gates on quantum circuit - Simple example

is the time-dependent variant, and the state depends on time. Thus, we can describe the energy of the system in the form[37]:

$$H|\psi\rangle = E|\psi\rangle \tag{3.17}$$

where E is the total energy of the system. In a thermal state, the energy of the samples follows a Boltzmann distribution[50], and the distribution is defined as:

$$P\left(E_i\right) = \frac{e^{-E_i/T}}{\sum_{j=1}^{M} e^{-E_j/T}} \tag{3.18}$$

where E_i is energy, and M is the total number of possible energy levels. When a quantum system is at zero temperature, the entire probability mass concentrates on the lowest energy level, which is the ground state energy of a quantum system[8]. When the quantum system is at high temperature, all the energy levels have equal probabilities.

4

Cathode Microstructure Dataset

In this chapter, a new microstructure dataset will be introduced, the collinearity and correlation matrix of the dataset will be analyzed. The purpose of collinearity analysis and correlation analysis is to drop irrelevant data, which makes the research more accurate.

4.1 Representative Volume Element

Currently, some simulated data for 120 cathode microstructures are generated as input parameters, which are obtained from [87]. Microstructures in the form of 3D volumes sliced into a series of images as well as the respective discharge curves, which can be seen in Fig 4.1:

The input parameters for each microstructure consist of microstructure statistics as well as the electrochemical model. Batteries' microstructures are stored as a series of $25\mu m \times 25\mu m \times 25\mu m$ cubes with a voxel size of $0.1\mu m$. The output parameters for each microstructure are the specific energy and specific power, and the results of each P2D simulation is stored as a time series of voltages.

Representative Volume Element(RVE) is shown in Fig 4.2. The RVE consists of three important parts which are called Electrolyte, Active Material(e.g., NMC) and Binder, which are shown in Fig 4.3.

The properties[87] of a volume element are shown in table 4.1.

where ϵ is the active material(AM) volume percentage, μ is active material particle diameter, ϵ_b is the binder volume percentage, constant 6% of active material, volume

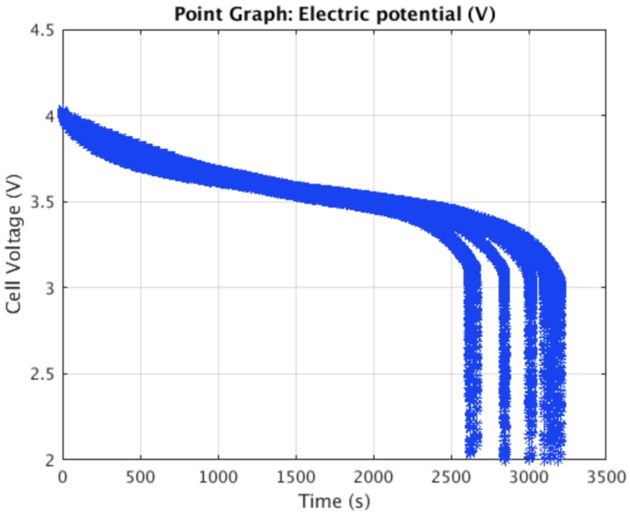


Figure 4.1: Respective Dischard Curves -

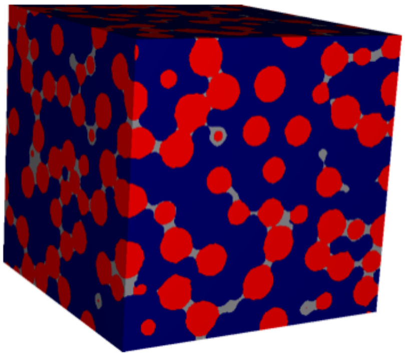


Figure 4.2: Representative Volume Element -

Table 4.1: Microtructure properties.

Property	mean	std	min	max
ϵ / %	59.6	6.3	41.0	69.6
μ / μm	9.4	3.9	3.7	15.0
AM distrubution		random		
ϵ_b / %	11.2	1.2	7.7	13.1
Binder distribution		between AM particles		
$D_{1,eff}/10^{-11}m^2s^{-1}$	3.005	1.41	1.10	7.97
$k_{1,eff}/10^{-2}Sm^{-1}$	12.2	5.6	4.4	31.8
A/10^5m^{-1}	1.7	0.9	0.6	4.4

percentage changes slightly. $D_{1,eff}$ is the effective diffusion of the electrolyte, $k_{1,eff}$ is the effective conductivity of the electrolyte, A is the specific surface between eletrolyte and active material.

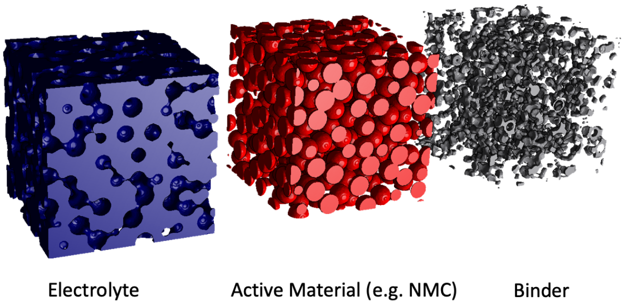


Figure 4.3: 3 Elements of RVE -

where ϵ is the active material(AM) volume percentage, μ is active material particle diameter, ϵ_b is the binder volume percentage, constant 6% of active material, volume percentage changes slightly. $D_{1,eff}$ is the effective diffusion of the electrolyte, $k_{1,eff}$ is the effective conductivity of the electrolyte, A is the specific surface between the electrolyte and active material.

4.2 P2Dmodel Dataset

P2Dmodel[87] contains all microstructure statistics data of 120 kinds of Representative Volume Elements, which is described in table 4.2.

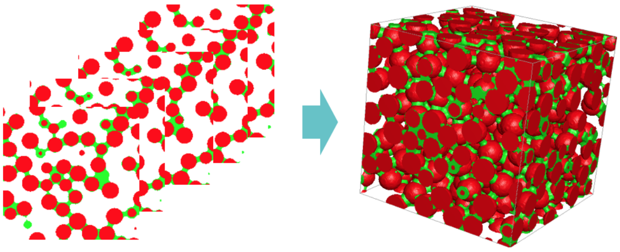


Figure 4.4: Artificially Generate Data - 250 slices of a RVE

Table 4.2: Microstructure statistics data of RVE.

Column	Name	Description
1	Index	ID
2	Microstructure Name	Name of the corresponding GeoDict microstructure
3-9	a_pos – epsl_pos	microstructure statistics (input)
10-12	L - C Rate	electrochemical model (input)
13	gamma	classification, is parameter set feasible (calculated)
14	termination	flag - simulation successful (1), simulation error termination (0)
15	mincl	Simulation result
16	Specific Energy	Simulation result: specific energy (output)
17	Specific Power	Simulation result: specific power (output)

4.3 F Test

F-test is the most commonly used alias called joint hypotheses test. It is a test of the statistical value obeying the F-distribution under the null hypothesis (H0). It is usually used to analyze statistical models. It uses more than one parameter to determine whether all or part of the parameters in the model are suitable for estimating. The name F test was named after American mathematician and statistician George W. Snedecor[93], which aims to commemorate British statistician and biologist Ronald Aylmer Fisher. Fisher invented this test and F allocation in the 1920s[92], originally called the Variance Ratio[48].

4.3.1 Calculation Formula

Let $X = x_1, x_2, ..., x_n$ and $Y = y_1, y_2, ..., y_n$ be two independent parameter sequences that follow a normal distribution, then the mean of the two sequences is expressed as formula 4.1 and 4.2.

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.1}$$

$$\overline{Y} = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{4.2}$$

So we can find the variance of the two sequences as formula 4.3 and formula 4.4.

$$S_X^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{X})^2 \tag{4.3}$$

$$S_Y^2 = \frac{1}{m-1}\sum_{i=1}^{m}(y_i - \overline{Y})^2 \tag{4.4}$$

So the formula 4.5 is for calculating F distribution:

$$F = \frac{S_X^2}{S_Y^2} \tag{4.5}$$

4.4 P Test

The calculation of the p-value is inseparable from the hypothesis test. The p-value is a degressive indicator of the credible level of the result. The larger the p-value, the less we can assume that the association of the variables in the sample is a reliable indicator of the correlation of the variables in the population. The p-value is the probability of making an error that is effective as a result of the observation. For example, p=0.05 reminds 5% of the variable association in the sample may be due to chance. That is to say, there is no correlation between any variables in the population. We repeat a similar experiment and find that there is one experiment in about 20 trials. The variable correlation we have studied will be equal to or stronger than our experimental results. This is not to say that if there is an association between the variables, we can get the same result of 5% or 95% of the number. When the variables in the population are related, the possibility of repeated research and invention association can be related to the statistical efficiency of the design. In many research areas, a p-value of 0.05 is generally considered to be the boundary level at which a fault can be accepted.

4.5 P Test and F Test in P2Dmodel Parameters

The P-value and F-value of the microstructure statistics data in P2Dmodel can be seen in table 4.3.

Table 4.3: P value and F value of the microstructure statistics data.

specificPower[W/kg]			specificEnergy[Wh/kg]	
Feature	F-Score	P-Value	F-Score	P-Value
$a\_pos[1/m]$	78.108136	1.105269e-14	159.514374	1.174570e-23
$rp\_pos[meter]$	161.939201	7.008335e-24	668.944158	1.905756e-50
$Dl\_pos[m^2/s]$	266.840967	4.492096e-32	234.992748	7.490189e-30
$Kl\_pos[S/m]$	266.840570	4.492370e-32	234.987753	7.496472e-30
$epss\_pos[-]$	97963.857274	4.005366e-174	967.824599	1.049151e-58
$epsb\_pos[-]$	98016.156575	3.881350e-174	967.793863	1.050906e-58
$epsl\_pos[-]$	98006.611166	3.903690e-174	967.850132	1.047695e-58
gamma	166.166155	2.879584e-24	148.779077	1.222202e-22
mincl[mol/mˆ3]	269.961117	2.784768e-32	231.449600	1.361654e-29

The p-value is a decreasing indicator of the credibility of the result. The larger the p-value, the less we can think about the association of the variables in the sample, which is a reliable indicator of the correlation of the variables in the population. The p-value is the probability of making an overall representation of the observations as valid. As we can see in table 4.3, the F value indicates the significance of the whole fitting equation. Thus, we can get the relationship among the features and specific power/energy. In the table, we get big F-Value and small P-Value, it means the input parameters(features) can fit the outputs(specific energy and specific power) well[46]. The larger the F, the more significant the equation is, and the better the degree of the fitting among the statistical values and specific energy/power. The p-value measures the difference between the statistical values(input parameters) and specific energy/power. And the p-value is less than 0.01, indicating that the difference between the two groups is extremely significant. At first glance, we can build a model which uses linear regression analysis among the statistical values.

4.6 Correlation Matrix

P related variables, find the correlation coefficient between the two variables, a total of $C_P^2 = \frac{P(P-1)}{2}$ correlation coefficients. If they are arranged in sequence according to the order of the variables, they are arranged into a matrix of numbers, which is called the correlation matrix. The commonly used letter R is indicated.

Correlation coefficient matrix mainly depends on the coefficient. Generally, ±0.5 or more is considered to have a weak correlation, and ±0.8 or more is considered to be relatively high, and greater than ±0.9 indicates a high correlation. All correlations here refer to linear correlations. Each variable has three values for each other, correlation coefficient, statistic value, and p-value. Mainly to see the correlation coefficient, the p-value can be used to determine whether the correlation coefficient is significantly zero.

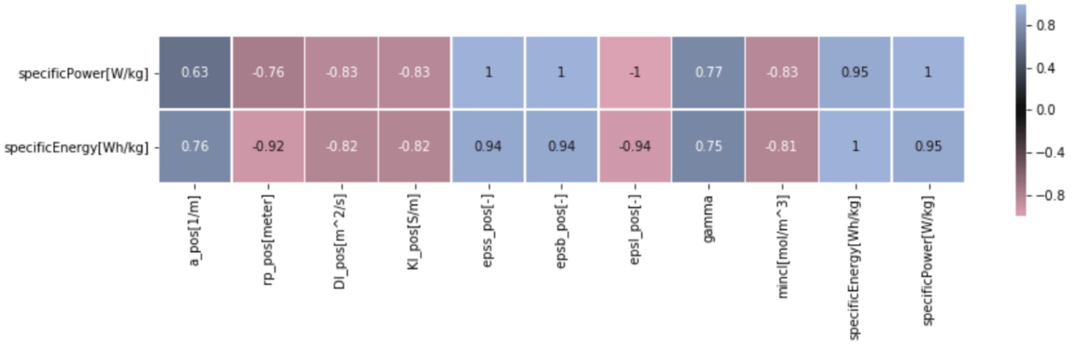


Figure 4.5: Correlation Matrix - Statistical values and Specific Energy/Power

According the the correlation matrix in Fig 4.5, almost all the statistic values in P2Dmodel($a\_pos[1/m]$, $rp\_pos[meter]$, $Dl\_pos[m^2/s]$, $Kl\_pos[S/m]$, $epss\_pos$[-], $epsb\_pos$[-], $epsl\_pos$[-], gamma, mincl[mol/mˆ3]) have a strong correlation with the specific energy and specific power.

5

Problem Statement

In this chapter, a statement of the research background will be presented, and three hypotheses for this research are proposed.

5.1 Research Background

In the development of emerging technology products such as drones and electric vehicles, the importance of battery technology is becoming increasingly prominent. Given that driverless technology is being integrated into almost the entire transportation sector, battery technology will also be the key to success in the future. In recent years, with the development of cutting-edge technology such as artificial intelligence, the unmanned technology has been fully popularized in the three-dimensional land, sea and air. Unmanned vehicles, unmanned aerial vehicles, unmanned ships and flying cars have mushroomed. The growth is gratifying. At the same time, since the development of energy-saving concepts and battery technology, pure electric vehicles have also become the mainstream trend. As a result, unmanned technology and pure electric concepts are accelerating integration, and pure electric unmanned vehicles will ”rule” future traffic. Under this trend, the importance of battery technology will undoubtedly increase. For purely electric vehicles, the current battery technology is sufficient to ensure commercial demand due to battery life requirements. However, unmanned aircraft, especially large unmanned flying vehicles, unmanned passenger aircraft, etc., have high requirements for endurance, and battery technology constraints will be fatal.

A negative electrode, a positive electrode, and some electrolytes make up the most straightforward battery. Negatively charged electrons flow from the negative electrode to the positive electrode through the electrolyte, forming a current. The negative electrode is usually made of a lithium metal oxide, and the negative electrode is called a lithium-ion battery. Lithium-ion batteries are most commonly used because they have the highest energy capacity and can fit into a small space like your phone. Lithium-ion batteries have three times the energy density of conventional rechargeable batteries during charging and discharging. Most lithium-ion batteries consist of a positive graphite electrode and a liquid organic electrolyte. In order to avoid a short circuit between the positive and negative electrodes in the battery, a small piece of thin permeable polypropylene (a plastic) is blocked between the two poles. If the barrier is cracked or eroded, the positive and negative electrodes will contact, and the battery will heat up extremely quickly. The battery is also filled with a flammable electrolyte that will ignite when it is hot and can be quickly excited by a short circuit. If the plug is uncovered, the liquid electrolyte will leak. Aims at solving the above problems, researchers in the Volkswagen Group are looking for solid electrolytes as an alternative.

Specific goals and objectives will be to find better materials for the cathodes and anodes of batteries and to improve the state-of-art in quantum-assisted machine learning. This project is part of the effort to predict properties of materials from their structure, and ultimately, to generate new structures that exhibit the desired properties. Currently, we have some simulated data for 120 cathode microstructures generated from input parameters, microstructures in the form of 3D volumes slice into a series of images as well as the respective discharge curves.

5.2 Hypotheses

Melanie's team[1] did a research on using a 3D neural network, whose inputs are microstructure images, and predict specific energy and specific power, which can be illustrated in Fig 5.1.

As Melanie's team are using classical neural network as a training approach and they achieved some results which are not bad, I prefer trying some other approaches in

[1] Dr. Melanie Senn, Nasim Souly, Prateek Agrawal, Nikhil George, Alex Alekseyenko. Website: http://vwiecc.com/

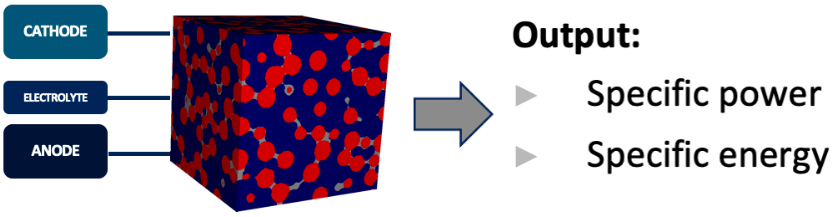


Figure 5.1: IECC's approach - Melanie's team

my thesis project:

- Hypothesis 1. I recommend using statistic features such as $a\_pos[1/m]$, rp_pos, etc., which are introduced in Dataset Section, it shows there are relations between statistic features and specific energy/power. So it will be a regression model, whose inputs are the statistic features and outputs are the specific energy/power. The purpose is to find if there is a model that can fit the dataset, and if the model can predict accurately on the test set.

- Hypothesis 2. As Melanie's team are using a classical neural network as a training approach to predict the specific energy and specific power, I recommend using a quantum neural network as a training approach to predict specific energy and specific power. The aim of this hypothesis is to figure out if we can use a quantum neural network to find a better combination of cathode, electrolyte and anode, which can produce high specific power and specific energy. The experiments are in Section 6.2 and Section 6.3, which started with a small quantum neural network which was applied on XOR problem, and organised on a larger dataset, such as Minist dataset and microstructure dataset.

- Hypothesis 3. As neural networks always take a long time to find an optimum, a wonder is that whether a quantum assisted method can accelerate the process of an optimization problem. QAOA has excellent potential in optimizing problem[30], the recommendation is using QAOA as an optimizer in updating the weights, and nobody did this before. This experiment will be shown in Section 6.4, which started with a simple neural network and applied QAOA as an update rule, then try QAOA optimizer on a larger dataset.

6

Experiments

In this chapter, four kinds of different experiments will be explained in detail, and the results can be used as the evidence for the hypotheses in Section 5.

6.1 Regression Analysis

This is the first experiment which implemented a small, simple neural network which can be used for regression analysis on the microstructure battery dataset.

6.1.1 Problem Statement

In the first research, regression analysis was to find a model or equations which can fit the dataset, if a regression model can be found over the statistical values, as we introduced in table 4.2, column 3 to 13 can be used as inputs of my model, then we can make predictions such as specific energy and specific power, which are the outputs. The aim is to find whether there is a model which can fit on the test set.

So far, there were only 120 examples of the dataset, so 90 of the dataset were used as training data, the rest 30 of the dataset were test set.

6.1.2 Approaches

I analyzed the relationship among the statistical values in P2Dmodel and the specific energy/power, which can be seen in Fig 6.1. As can be seen in Fig 6.1, almost all the statistic values have linear relationships with specific energy and specific power, which can be the persuasive evidence for using linear regression.

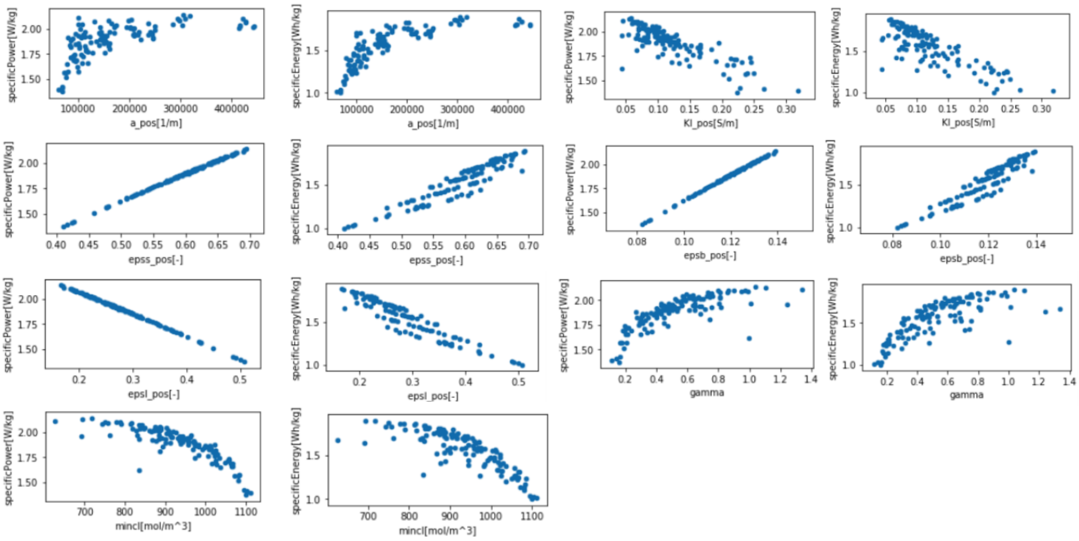


Figure 6.1: Co-linearity - Statistical values and Specific Energy/Power

As the aim is to find a linear regression which can model the relationship between input variables and output variables. The linear regression is defined as formula 6.1:

$$Y = a + bx \tag{6.1}$$

where Y is the output variables(specific energy and specific power) in my approach, which are dependent variables. X is the input variables(statistical values in P2Dmodel), which are the independent variables. The slope of a linear regression line is represented by b, and a is the intercept of Y.The definitions of a and b can be shown in formula 6.2 and formula 6.3.

$$a = \frac{\sum y \sum x^2 - \sum x \sum xy}{n \sum x^2 - (\sum x)^2} \tag{6.2}$$

$$b = \frac{n \sum xy - \sum x \sum y}{n \sum x^2 - (\sum x)^2} \tag{6.3}$$

According to the the formula 6.1 to formula 6.3, I built a 5-layers neural network in TensorFlow, it will be explained in the next section.

6.1.3 Network Structure

A 5-layers neural network was build to find the linear regression between input variables and output variables, which has 7 input nodes and 2 output nodes, the structure of the neural network can be seen in Fig 6.2.

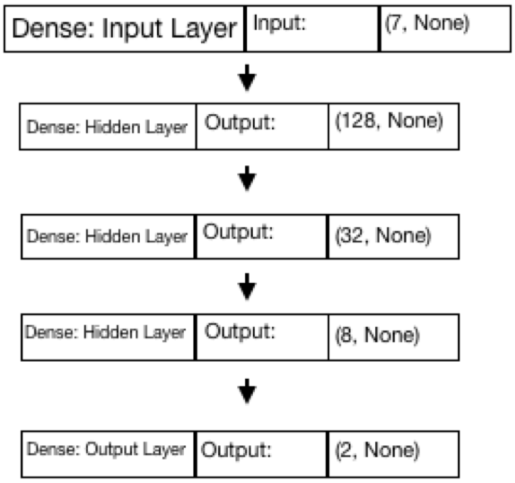


Figure 6.2: Neural Network - Structure

As can be seen in Fig 6.2, multiple hidden layers are actually multi-level abstractions of input features. The ultimate goal is to partition different types of data better linearly.

6.1.4 Result and Analysis

The loss of the regression model can be seen in Fig 6.3. Although the loss on the test set was terrible, but with the epoch increasing, the loss on the test set decreased and reached a stable loss value.

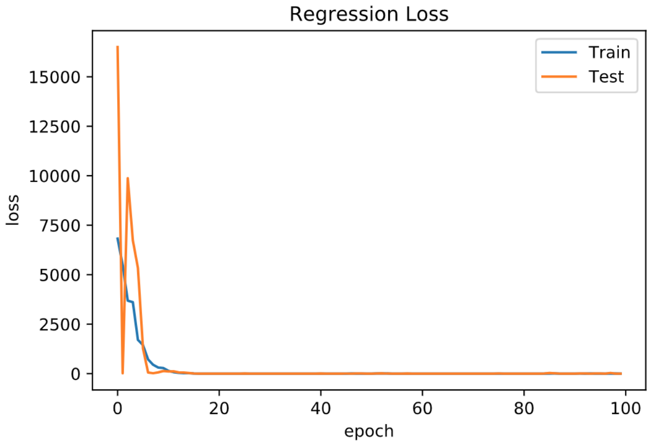


Figure 6.3: Regression Model - Loss

In the beginning, the result seems to be good enough, but think more about it, I have an idea that the model might be overfitting. As the size of the dataset is very small, which only has 90 training dataset, and after 20 epochs, the regression loss values are the same on the training set and test set, which is ridiculous. I suppose there are two reasons which make this phenomenon. One reason is that the size of the data set is too small for training a good model. Another reason could be overfitting, as I have a small size of the dataset and there were too much training parameters in the neural network. And it can also be explained in another way, a decision boundary or a neuron is a linear division. Multiple neurons are multiple linear divisions, and multiple linear divisions are continually approaching the decision boundary. This process can be imagined as an integration process. A decision boundary is composed of multiple linear partitions. If the number of neurons is much larger, this will result in a lot of linear divisions, which will distort the decision boundary, basically the over-fitting.

6.2 Quantum Neural Network

In this second experiment, a quantum neural network was built on the quantum simulator, which can solve some simple problems.

6.2.1 Problem Statement

In this experiment, I start with a small and straightforward problem–XOR problem, a hybrid quantum-classical approach was applied on this problem, and the aim is to figure out whether the quantum assisted approach can accelerate the training process if setting a threshold for the loss value, or within a fixed iteration. After trying the XOR problem, I want to figure out if this approach can be used for a larger dataset, which means a much more general situation in real life.

6.2.2 Dataset

There are two kinds of the dataset used in this experiment: the XOR dataset, downsampled dataset.

The XOR dataset is straightforward, and it consists of binary values which can be illustrated in table 6.1. This is a straightforward dataset, and XOR problem can be solved by using a classical neural network, but in this experiment, a quantum method was used for solving XOR problem.

Table 6.1: XOR Dataset.

Input Data	Output Label
0 0	0
0 1	1
1 0	1
1 1	0

6.2.3 Approaches

The parametrized quantum logic is prepared by quantum fourier transform, and preparing the data in data points in a TensorFlow compute graph use it to generate the data prep circuit in sampler to prepend a quantum representation of the data on to the circuit logic outlined in l3 and then using backpropagation (without CV), optimize the parameters of the classical-quantum hybrid net.

6.2.3.1 Reason of Using QFT

In 1994, Shor created a quantum form of the Fourier Transform[79]. Based on Shor's work, a quantum version of the Fast Fourier Transform was developed by Coppersmith[22], which enabled the implementation of QFT on a quantum machine. Without any doubt, QFT is the landmark of quantum algorithms, and it is also a crucial part of Shor's algorithm and many other quantum algorithms[76]. QFT can improve the complexity of a classical algorithm from $O(NlogN)$ to $O(log^2N)$, because of its speed-up performance, QFT was chosen as an approach in this experiment.

6.2.3.2 Quantum Fourier Transform

In quantum mechanics, the state vectors for qubits are just a vector of complex numbers, and the classical Discrete Fourier Transform can not be applied to any state vectors. Given a state vector $|\psi\rangle$ which defined as:

$$|\psi\rangle = \sum_{j=0}^{N-1} a_j|j\rangle = \begin{pmatrix} a_0 \\ \vdots \\ a_{N-1} \end{pmatrix} \tag{6.4}$$

So the Discrete Fourier Transform(DFT) of this state can be computed as:

$$F|\psi\rangle = \sum_{k=0}^{N-1} b_k|k\rangle \tag{6.5}$$

where

$$b_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} a_j e^{\frac{2\pi ijk}{N}} \tag{6.6}$$

It has been proved that the operation is unitary[104], so it can be implemented in quantum mechanics. Consider the 2-qubit state:

$$|\psi\rangle = a_{00}|00\rangle + a_{01}|01\rangle + a_{10}|10\rangle + a_{11}|11\rangle \tag{6.7}$$

which has $N = 4$, according to formula 6.6, the b_0, b_1, b_2 and b_3 can be defined by:

$$b_0 = \frac{1}{2}\sum_{j=0}^{3} a_j = \frac{1}{2}\left(a_{00} + a_{01} + a_{10} + a_{11}\right) \tag{6.8}$$

$$b_1 = \frac{1}{2}\sum_{j=0}^{3} a_j e^{\frac{2\pi ij}{4}} = \frac{1}{2}\left(a_{00} + a_{01}e^{\frac{i\pi}{2}} + a_{10}e^{i\pi} + a_{11}e^{\frac{3i\pi}{2}}\right) \tag{6.9}$$

$$b_2 = \frac{1}{2}\sum_{j=0}^{3} a_j e^{\frac{4\pi ij}{4}} = \frac{1}{2}\left(a_{00} + a_{01}e^{i\pi} + a_{10}e^{2i\pi} + a_{11}e^{3i\pi}\right) \tag{6.10}$$

$$b_3 = \frac{1}{2}\sum_{j=0}^{3} a_j e^{\frac{6\pi ij}{4}} = \frac{1}{2}\left(a_{00} + a_{01}e^{\frac{3i\pi}{2}} + a_{10}e^{3i\pi} + a_{11}e^{\frac{9i\pi}{2}}\right) \tag{6.11}$$

Writing $\omega = e^{\frac{\pi i}{2}}$, noting that $\omega^0 = 1$, $\omega^1 = i$, $\omega^2 = -1$, $\omega^3 = -i$, $\omega^4 = e^{2\pi i} = 1$ and $e^{\frac{9i\pi}{2}} = e^{\frac{\pi i}{2}} = i$, so the 2-qubit QFT can be re-write in matrix form which is shown in formula 6.12:

$$F = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 \\ 1 & \omega^2 & 1 & \omega^2 \\ 1 & \omega^3 & \omega^2 & \omega \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix} \tag{6.12}$$

and this matrix form can be easily shown to be a unitary operator. A fairly simple circuit that performs this transformation can be built on a quantum circuit simulator, which is explained in detail in Section 6.2.3.3.

6.2.3.3 Quantum Circuit

The implement of the Quantum Fourier Transform on a quantum circuit simulator is quite simple, basic quantum gates and elements were introduced in Section 3, but it is necessary to introduce some new gates which are used in the quantum circuit. Moreover, this quantum circuit is a quantum simulator on Cirq[43].

- The controlled-R_k gate: controlled-R_k gate can apply a relative phase change to $|1\rangle$, the matrix form of this operator is:

$$R_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{\frac{2\pi i}{2}} \end{pmatrix} \tag{6.13}$$

The diagram of a controlled-R_k gate on a quantum circuit is shown in Fig 6.4:

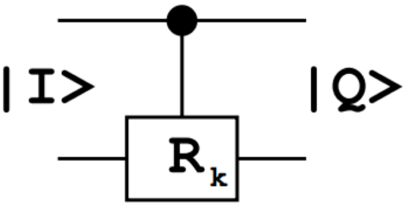


Figure 6.4: The controlled-R_k gate - on a quantum circuit

- The controlled-X gate: in the basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, the matrix form of this operator is:

$$Controlled - X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{6.14}$$

The diagram of a controlled-X gate on a quantum circuit is shown in Fig 6.5:

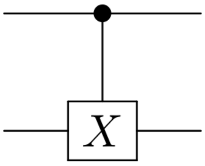


Figure 6.5: The controlled-X gate - on a quantum circuit

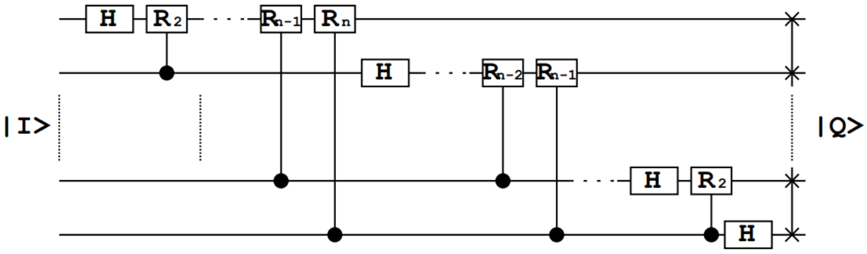


Figure 6.6: QFT on Quantum Circuit - General Situation

The circuit performing the QFT is drawn generally in Fig 6.6:

In this experiment, the quantum circuit for a QFT process should be customized. The quantum circuit consists of four layers and one readout layer, acting on four qubits with some quantum gates. The first layer of the quantum circuit is shown in Fig 6.7: In this first layer, an H gate was applied on qubit 2. One CPhase gate was applied between qubit 1 and 2 with the rotation angle is $\frac{\pi}{2}$, another CPhase gate was applied between qubit 0 and 2 with the rotation angle is $\frac{\pi}{4}$. In the second layer, an H gate was applied on qubit 1, and a CPhase gate was applied between qubit 0 and 1 with the rotation angle is $\frac{\pi}{2}$. In the third layer, only an H gate was applied on qubit 0. The last layer is active layer, which used QAOA algorithm for optimization, so the complete QFT circuit is shown in Fig 6.8.

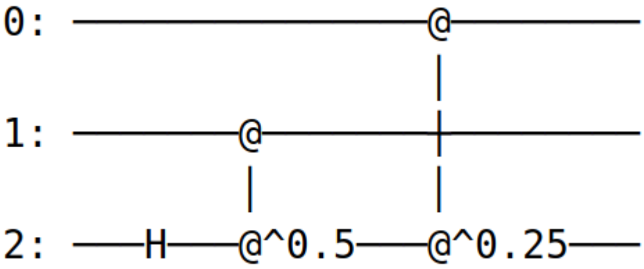
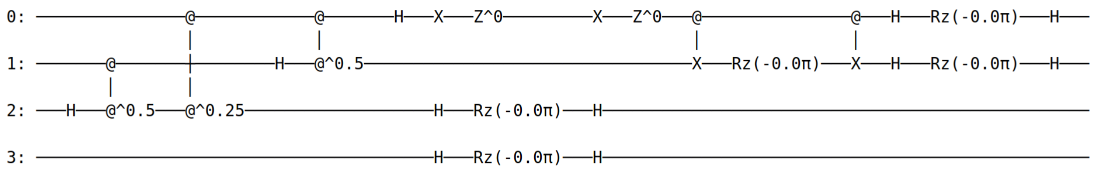


Figure 6.7: QFT Circuit - First Layer



```
0: ────────────────@────────────@───────H───X───Z^0─────────X───Z^0───@────────────────@───H──Rz(-0.0π)────H────
                   │            │                                     │                │
1: ────────@───────┼────────H───@^0.5─────────────────────────────────X───Rz(-0.0π)────X───H──Rz(-0.0π)────H────
           │       │
2: ────H───@^0.5───@^0.25───────────────────H───Rz(-0.0π)───H───────────────────────────────────────────────────

3: ─────────────────────────────────────────H───Rz(-0.0π)───H───────────────────────────────────────────────────
```

Figure 6.8: QFT Circuit - Complete Circuit

6.2.4 Result and Analysis

As a comparison, the XOR problem was executed on a quantum circuit simulator and a classical neural network, which is implemented in TensorFlow. The result can be seen in Table 6.10:

Table 6.2: MSE comparasion - Quantum approach and classical approach.

Iteration	MSE-Quantum XOR	MSE-Classical XOR
0	0.25441742	0.9210175
10	0.2435121	0.8736283
20	0.19666877	0.8526401
30	0.13149635	0.8117458
40	0.038574576	0.7800300
50	0.0075638234	0.7565701
60	0.0029614838	0.7393677
70	0.0018134008	0.7267133
80	0.001371848	0.7173568
90	0.0011428597	0.7104138

As can be seen in Table 6.10, after the same value of iteration which was 100, the Mean Square Error(MSE) of classical approach was 0.71041, the MSE of quantum approach was 0.00114, which was almost 617 times faster of the classical approach. If the classic approach with TensorFlow converges to the same MSE of quantum approach, it will need more than 5, 000 iterations.

Fig 6.9 shows that MSE changes on each iteration by using quantum approach. As can be seen in Fig 6.9, the MSE decreased sharply between iteration 20 and 40, and converged to a small stable value which is 0.00114. As for a comparison, Fig 6.10 illustrated the dynamic changes in each iteration. The initial error of classical approach was higher than the quantum approach, and the convergence speed of the classical

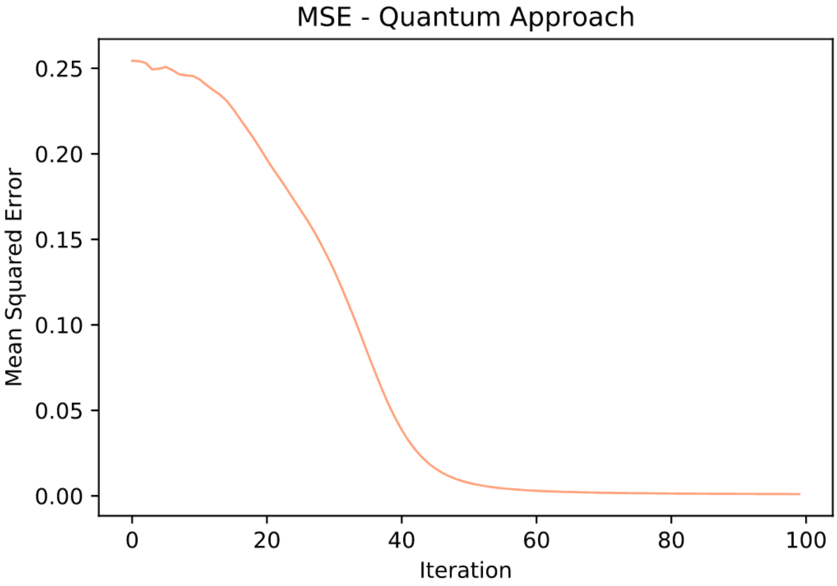


Figure 6.9: MSE - Quantum Approach

approach was slower than the quantum approach. In 100 iterations, the convergence of quantum approach was much quicker than the classical approach, and at the same time, the quantum approach achieved a much smaller MSE value, which was a great result. Fig 6.11 shows the dynamic changes by using classical approach, as can be seen in the figure, even if the model is executed for $20,000$ iterations, it cannot reach the MSE of quantum approach.

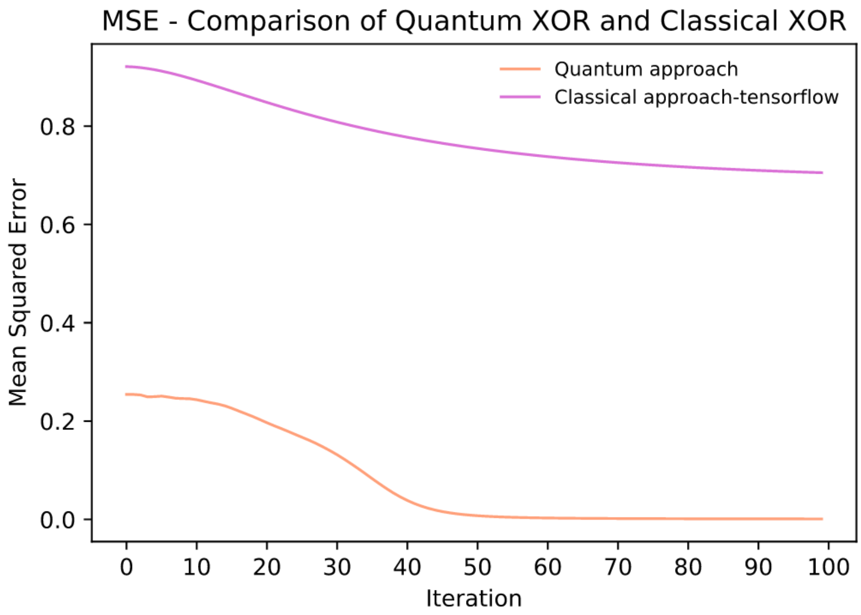


Figure 6.10: Comparision - MSE of quantum approach and classical approach

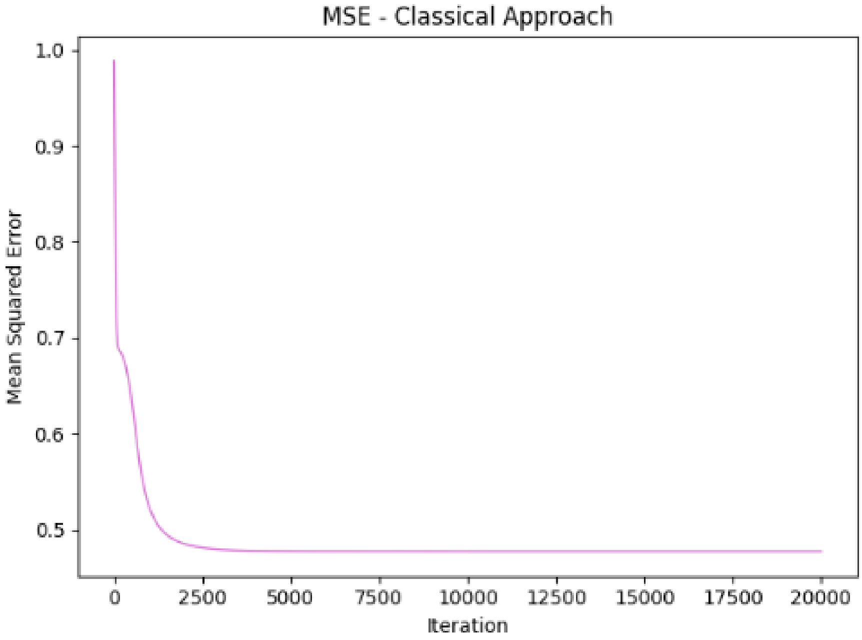


Figure 6.11: MSE - Classical Approach

6.2.5 Conclusion

Using the quantum method to solve XOR problem is a trial, the aim was to figure out whether the quantum approach can accelerate the training speed and get a small error at the same time(on a quantum simulator in this experiment). The research results show that the quantum approach can accelerate the training speed, and compared to the classical approach, the quantum approach can achieve a small MSE with the same iteration, which is only 100 iterations. This is definitely an exciting and inspiring result, which shows the potential of using quantum computing in an optimization problem. XOR is a small and simple problem. This trial shows using quantum approach is feasible. In the next step, the quantum approach will be used in a larger dataset, which is Mnist dataset.

6.3 Quantum Binary Classification

In the previous experiment, the simple XOR problem was solved by using quantum computing approach, which got an accelerate on the training process and resulted in a minimal MSE value. Further research was done on this research pathway, which built a classifier by using quantum computing approach.

6.3.1 Problem Statement

In this experiment, Mnist dataset was used, which is a bigger and more complex dataset than the XOR dataset. And a quantum computing approach was used on this dataset, trying to find out whether the quantum computing approach is workable for a more significant problem.

6.3.2 Dataset

The original Mnist handwritten digit dataset consists of $60,000$ training examples and $10,000$ testing examples, it is a subset of a more extensive set from NIST, and the digits have been resized in a fixed-size by size-normalizing, and the size is 28×28. However, due to the limitation of the quantum circuit in Cirq, the limitation means the quantum simulator on Cirq can handle 17 qubit quantum computer, in other words, it should be 16-bit data with one readout bit. So it is necessary to downsample the digit image to a size of 10 bits, which will be in total 10 data points on the circuit and one readout bit. If using one readout, labeling ten digits is impossible, instead of predicting ten labels, two digits were picked up, say 6 and 9, and reduce the dataset which only consists of digit 6 and digit 9, and testing if the quantum neural network can distinguish these two different digits.

6.3.3 Approaches

Farhi did a research on representing subset parity and subset majority[31], which can be used for quantum classification. In his paper, they represented subset majority with the unitary operator with β set to $0.9\pi/n$, where β was the rotate angle of a quantum operator, and n was the number of commuting two-qubit operators in a quantum circuit. This method can also be used for classification, but only the binary classification. If a quantum approach is proposed, all the classical data should be represented in a quantum definition, such as the superposition and quantum gates. In the following sections, I will introduce the details of the superposition and quantum gates in a binary classification problem.

6.3.3.1 Superposition Representation

The aim is to build a quantum neural network with TFQuantum, the input data and output data are supposed to be binary values, which is the limitation of the quantum neural network. The aim of the quantum binary classifier is distinguishing digit 6 and 9, and the samples are divided into some sample labeled as $+1$ and -1, then the quantum states can be written in formula 6.15:

$$|+1\rangle = N_+ \sum_{z:l(z)=1} e^{i\varphi_z}|z,1\rangle \quad |-1\rangle = N_- \sum_{z:l(z)=-1} e^{i\varphi_z}|z,1\rangle \tag{6.15}$$

where N_+ and N_- represent the normalization factors, N_+ and N_- are set to 0 in Farhi's research[30], so they are all set to 0. The way of preparing classical data for quantum computation is bitstring encoding, z is the string of input variables, $|z,1\rangle$ is the representation of a classical dataset of N-length bitstrings, where $z \in 0,1^n$. A quantum unitary(introduced in next part) that encodes these bitstrings into binary labeled eigenstates according to $U(z) = |z\rangle$. After converting the classical data into quantum states, each of the quantum states has a label, either is $+1$ or -1. $+1$ represents the digit 6, and -1 represents the digit 9.

In this experiment, different input samples are combined, which forms a superposition state. Then, picking a loss function and calculate the gradient on each epoch, the loss function will be used to compare the readout qubit's expectation value $\langle Y_{n+1}\rangle$ to the class of a data point, where $y \in \{0,1\}$. The loss function is defined in formula 6.16:

$$loss(\vec{\theta}, z) = \frac{1}{n}\sum_{i=1}^{n}(\left\langle z_i, 1 \middle| U^\dagger(\vec{\theta})Y_{n+1}U(\vec{\theta}) \middle| z_i, 1\right\rangle - l(z_i))^2 \tag{6.16}$$

where n is the number of labels. $\left\langle z_i, 1 \middle| U^\dagger(\vec{\theta})Y_{n+1}U(\vec{\theta}) \middle| z_i, 1\right\rangle$ is the ground truth in quantum computation basis, it is a list of arrays and consist of information from dataset object about the desired output from the same quantum circuit, in other words, it is the true label in a classical neural network. $l(z_i)$ is a list of readout expectations from the quantum circuit, which can be considered as the predicted labels in a classical neural network.

6.3.3.2 Quantum Unitaries

Besides the loss function in a superposition representation, some unitary operations should be associated with labels, based on quantum computational basis states, and the label unitary can be written in formula 6.17:

$$U_l \left| z, z_{n+1} \right\rangle = \exp\left(i \frac{\pi}{4} l(z) X_{n+1} \right) \left| z, z_{n+1} \right\rangle \tag{6.17}$$

where the $l(z)$ can only be $+1$(represents digit 6) or -1(represents digit 9), and it can be regarded as a diagonal operator in the quantum computational basis. This is an abstract way of representing the classical labels in a quantum circuit.

Reed-Muller[101] can represent any Boolean function and re-write the label unitary 6.17 as a product of 2 qubit unitaries. For convenient expression, the boolean variables b_i can be defined as formula 6.18:

$$b_i = \frac{1}{2} \left(1 - z_i \right) \tag{6.18}$$

Starting from bits b_1 to b_n, the product of two qubit unitaries can be written in formula 6.19:

$$b = a_0 \oplus (a_1 b_1 \oplus a_2 b_2 \oplus \ldots a_n b_n) \oplus (a_{12} b_1 b_2 \oplus a_{13} b_1 b_3 + \ldots) \oplus \ldots \oplus a_{123} \ldots b_1 b_2 \ldots b_n \tag{6.19}$$

If one intends to use B to represent b, which is the diagonal operator in the quantum computation basis, the label unitary can be written in the form of formula 6.20, the terms in B can commute with each other by multiplying X_{n+1}:

$$U_l = \exp\left(i \frac{\pi}{4} X_{n+1} \right) \exp\left(-i \frac{\pi}{2} B X_{n+1} \right) \tag{6.20}$$

6.3.3.3 Quantum Circuit

For the purpose of constructing the quantum circuit model, the circuit ansatz for binary classification can be written as :

$$U(\vec{\theta}) = \exp\left(i \frac{\beta}{2} \sum_j \theta_j Z_j X_{n+1} \right) \tag{6.21}$$

A simple quantum neural network with linear activations is used for determining the classification results. For supervised learning, this quantum circuit is supposed to be trained on a series of labeled inputs, which are prepared in the dataset process. In TFQuantum, the **PlaceholderLayer** should be inserted before $U(\theta)$, which will be resolved for each sample that is taken during optimization.

In the beginning, β is set to a fixed value, and **10 qubits** was chosen as operators in the quantum circuit, each operator are the product of three values: β, Pauli-Z gate and Pauli-X gate, the details can be seen in Algorithm 1:

Algorithm 1 Construct quantum operators

```
procedure QUANTUM OPERATORS($i$)                ▷ $i$ is the subscript index of quantum operators
    $\beta \leftarrow \frac{-1.0*(0.9*\pi/6.0)}{2.0}$
    $i \leftarrow 0$
    for $i \leq 9$ do
        operator = $\beta\times$ PauliZ($i$) $\times$ PauliX(10)
    end for
    return operators                            ▷ operators consist of a list of operator($i$)
end procedure
```

When the quantum operators are prepared, an exponential layer can be constructed according to the formula 6.21 so that the quantum circuit can be constructed according to Algorithm 2:

Algorithm 2 Construct quantum circuit

```
procedure QUANTUM CIRCUIT($i$)
    prev_layers ←PlaceholderLayer
    parameters ← 0                               ▷ This is a list
    operators ← Quantum operators($i$)
    active_layer ← ExponentialLayer(prev_layers, parameters, active_layer)
    return active_layer          ▷ This is the complete layer on a quantum circuit
end procedure
```

The **PlaceholderLayer** is a layer in TFQuantum[43] allowing for dynamic data input to a quantum circuit, and it is a special layer that can be used to insert layers into the circuit just before compilation. The **PlaceholderLayer** is useful for allowing layers representing training data to be inserted anywhere in the circuit. **Quantum operators**(i) can be processed in Algorithm 1. The **ExponentialLayer** constructs a parametric layer from a sequence of operator exponentials, which obeys the rule of formula 6.21. The role of using ExponentialLayer is to append a layer $\exp(-i * x * term)$ for each term in operators. Procedure quantum operators in Algorithm 1 are the initial operators, which are specified in the list operators, by qubit paulis or standard paulis. Parameter variables representing initial operator coefficients are given in the list parameters, which is processed in line 3 of Algorithm 2.

The quantum circuit can be constructed according to the above steps, which is illustrated in Fig 6.12. As can be seen in Fig 6.12, each quantum operator from 0 to 9 is connected to the operator 10, and for the purpose of binary classification, the quantum neural network architecture returns specify qubit $n + 1$ as the binary classifier readout, which is in the expectation value $\langle Y_{n+1} \rangle$. After completing the quantum circuit, TFQuantum can compile this logic down to a Cirq circuit, but the **PlaceholderLayer** is not visible in Fig 6.12, since it is empty until a specific layer of circuit logic is drawn from the dataset during training. Moreover, the way of preparing a classical dataset into a quantum dataset has been introduced in Section 6.3.3.1.

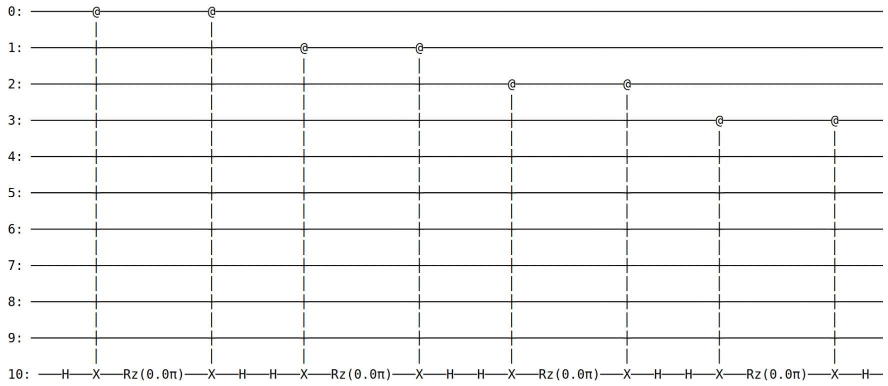


Figure 6.12: Quantum Circuit - Partial Quantum Neural Network

After finishing the preparation, model training should be taken on the agenda. The *BasicModel* in TFQuantum provides an architecture for constructing a basic quantum neural network, and this is a development tool similar to Tensorflow. In order to build a quantum neural network in TFQuantum, three requirements should be fulfilled:

- Define a quantum circuit logic. The quantum circuit logic has been defined in detail in Section 6.3.3.3.

- One readout per circuit to be used in optimization. Readout scheme is the expectation value $\langle Y_{n+1} \rangle$, standard in literature, $\langle \psi | Y_{n+1} | \psi \rangle$ should be estimated. The aim is to get basis measurement information and statistics for a hamiltonian. The optimizer is ADAM Optimizer, it is a common optimizer that has efficient calculation and low memory requirements.
- A loss function. The loss function is defined in formula 6.16, which is in a superposition representation.

6.3.4 Result and Analysis

Table 6.3 shows the statistics data of the quantum neural network:

Table 6.3: Statistics data of QNN model.

Optimizer	ADAM
Learning Rate	0.1
Batch Size	1
Iterations	400
Number of learnable params	10
Circuit depth	50
Loss Function	$loss(\vec{\theta}, z) = \frac{1}{n} \sum_{i=1}^{n} (\left\langle z_i, 1 \middle\vert U^{\dagger}(\vec{\theta}) Y_{n+1} U(\vec{\theta}) \middle\vert z_i, 1 \right\rangle - l(z_i))^2$

Also, the MSE value on each iteration is shown in Fig 6.13:

In general, the MSE value decreased on each iteration, but between iteration 200 and 250, it increased sharply, one guess is that it was a normal situation in a training process, after iteration 300, the MSE values converged to a small value around 0.1. Also, the model was tested on a test set, the accuracy was 0.54, which is not as good as a conventional approach, such as training with TensorFlow, the reason should be the downsampled input dataset, which cannot be compared to training with an original Mnist dataset. In my perspective, this result is good enough for convincing that a quantum neural network can achieve a binary classification.

6.3.5 Conclusion

In this experiment, a quantum classification was achieved on the downsamples Mnist dataset. The quantum circuit was only 50, and the number of learnable parameters

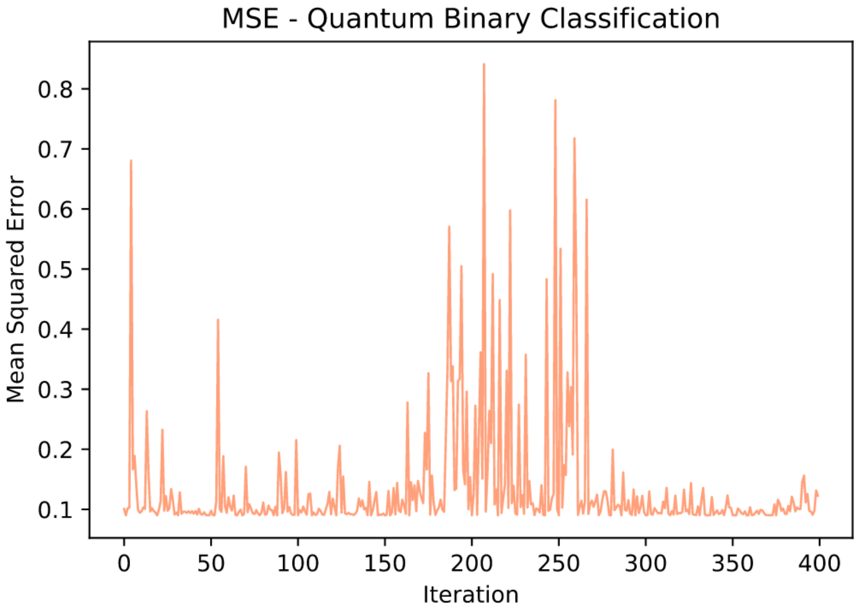


Figure 6.13: MSE - Quantum Binary Classification

was only 10, which was much smaller than the number of parameters on a conventional neural network. Also, this quantum classifier only took a few seconds on a quantum simulator, as we all know, the speed on a simulator is much slower than the speed on a physical computer. A belief is that if we can process this problem on a physical quantum computer, the training process will need less time. If there are enough qubits on a physical quantum computer, higher accuracy can be achieved with the original Mnist dataset, which will be exciting progress on quantum computing.

Another trial was to downsample the microstructure battery material dataset, but the results were basically none of use. As is mentioned in the dataset, the quantum simulator only has 17 qubits accessible, but the size of microstructure battery material dataset is 250∗250, which far exceeds the size of a quantum circuit simulator. Moreover, even if the microstructure battery material dataset was downsampled to 4∗4, which lost a large number of details, such as the connections among the cathode, the electrolyte, and the anode. Although the model can classify digit 6 and 9 on a downsampled Mnist dataset, however, it cannot train a model on the microstructure battery material dataset, neither the original dataset or the downsampled dataset.

6.4 Quantum Assisted Optimization

This experiment is a little different from the previous three experiments. This experiment focused on a shallow-circuit variational algorithm for gate model quantum computers–the quantum approximate optimization algorithm(QAOA), which was inspired by quantum annealing.

6.4.1 Problem Statement

In this experiment, the purpose is to deal with optimizing weights in a classical neural network by using a quantum algorithm. It can be understood as a quantum assisted optimization problem. In Section 3, Farhi's quantum approximate optimization algorithm was introduced, but the limitation of Farhi's QAOA is that it can only process bit instance, either the input and output. As we all know, the weight values are always float values, and they are barely integers. Thus, the aim of this experiment is to come up with an approach, which can apply QAOA on float values, and figure out if QAOA helps the optimization process in a classical neural network.

6.4.2 Dataset

Based on the experiments in the previous section, the quantum circuit simulator on TFQuantum is limited to 17 qubits, and if researchers want to use the microstructure battery material dataset, the number of the classical neural network which processes such dataset is $5,316,002$, this is a large number of learnable parameters which cannot be allocated on a quantum circuit simulator.

So, at this point, I also start with a straightforward dataset. The intention to know if QAOA can help with the optimization process. The dataset is shown in Table 6.4.

Table 6.4: Simple Dataset.

Input			Output
0	0	1	0
0	1	1	1
1	0	1	1
1	1	1	0

The input dataset only consists of 3 bits of binary values, and the output value is either 0 or 1, this is a quite simple dataset, the purpose of using this dataset is to simplify the influence of a complex dataset and focus on using QAOA in updating the weights.

The second dataset was Iris Dataset[1], it first appeared in the famous British statistician and biologist Ronald Fisher's 1936 paper, it is used to introduce linear discriminant analysis. In this data set, three different species of Iris are included: Iris Setosa, Iris Versicolour and Iris Virginica. Each class collected 50 samples, so this data set contains a total of 150 samples. The data set measures four characteristics of all 150 samples:

- Sepal length.
- Sepal width.
- Petal length.
- Petal width.

The units of the above four features are all centimeters (cm). In the experiment, these four features were used as inputs, and the outputs are three species of Iris: Iris Setosa, Iris Versicolour and Iris Virginica. Part of the original Iris dataset can be seen in Table 6.5:

Table 6.5: Example of Iris dataset.

Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.1	3.5	1.4	0.2	Setosa
4.9	3.0	1.4	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor
4.9	2.4	3.3	1.0	Versicolor
6.3	3.3	6.0	2.5	Virginica
7.6	3.0	6.6	2.1	Virginica

However, the neurons on the neural network cannot accept 'Setosa' or 'Versicolor' or 'Virginica' as input datatype. Moreover, these species are converted to 3-bits strings

[1] Iris flower dataset, from wiki https://en.wikipedia.org/wiki/Iris_flower_data_set.

with binary values, which is an intuitive way for processing in neural networks. The bit encoding of Iris species can be seen in Table 6.6:

Table 6.6: Bit string of species.

Species	Bit String
Setosa	1, 0, 0
Versicolor	0, 1, 0
Virginica	0, 0, 1

6.4.3 Approaches

In this section, three approaches will be introduced, and they can help complete this experiment. A neural network was built without TensorFlow, only using numpy build the neural networks for the simple dataset and Iris dataset. In the neural network structure, QAOA was used as an optimizer, which can update the weight in the neural networks.

6.4.3.1 Neural Network on Simple Dataset

The simple neural network was built manually, which consists of three layers. There are three steps in this simple neural network:

- Forward Propagation. Calculate the loss value by using input matrix and weight matrix.
- Back Propagation. According to the chain rule (derivation), calculate the contribution value of each weight to the loss function.
- Update weights. Adjust weight matrix according to the contribution value, in order to minimize loss function.

Fig 6.14 illustrates the network structure of this simple neural network:

In this neural network, the input data X is in the shape of $(3 * 4)$, and output data y is in the shape of $(4 * 1)$. W_0 and W_1 are the weight matrix of the neural network, and they are initialized randomly with the shape of $(3 * 4)$ and $(4 * 1)$. In the process of

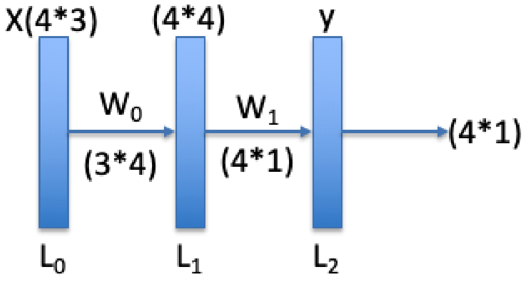


Figure 6.14: Network Structure - Simple Neural Network

forwarding propagation, the sigmoid function is used as an activation function, which is defined as formula 6.22:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{6.22}$$

So the derivative function of sigmoid function can be written as formula 6.23:

$$f'(x) = f(x)(1 - f(x)) \tag{6.23}$$

Formula 6.24 shows a way of calculating the loss function in this simple neural network, where y is the true label and l_2 is the predicted lable. In this way, the weight matrix can be updated according to the loss values in each layer.

$$loss = y - l_2 \tag{6.24}$$

6.4.3.2 Neural Network on Iris Dataset

After encoding species into 3-bits bitstrings, a three-layers neural network was built, which has 4 nodes a bias node in the input layer, 7 nodes and a bias node in the hidden layer, 3 nodes in the output layer, the structure of the neural network can be seen in Fig 6.15:

The activation function is also sigmoid function, which is defined in formula 6.22. And the loss function is the same function which is defined in formula 6.24, which can be used for the backpropagation process.

6.4.3.3 Background of QAOA

Farhi came up with a Quantum Approximate Optimization Algorithm[30]. As is shown in Fig 6.16, QAOA creates an iterative loop between the quantum and the classical

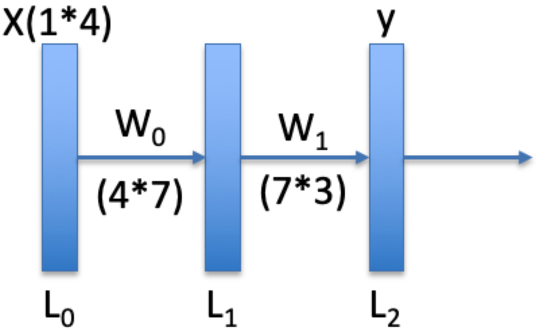


Figure 6.15: Network Structure - Iris Neural Network

processing hardware units, it is a classical-quantum hybrid algorithm. The basic idea is straightforward: run a short sequence of gates where some gates are parametrized, then readout the results, make an adjustment to the parameters on a classical computer, such as ADAM optimizer, and repeat the calculation with the new parameters on the quantum hardware.

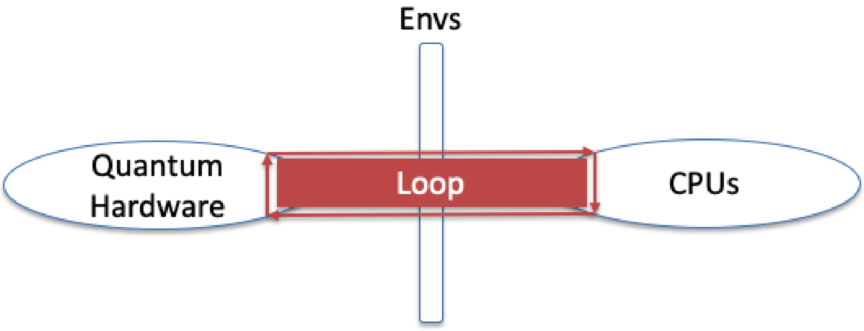


Figure 6.16: QAOA - Iterative Loop

QAOA approximates the adiabatic pathway on a gate model computer, the adiabatic pathway is discretized into p steps, where $p \geq 1$[30]. Step p influences the precision of the algorithm, each discrete time step i has two parameters: β_i and γ_i. The classical optimization algorithm does an optimization on these parameters based on the observed energy at the end of a run on the quantum hardware. The circuit depth grows linearly with step p, at the worst situation, it is p times, in general, step p is a fixed value during a process of running. Suppose we want to discretize the time-dependent formula 6.25 under an adiabatic condition:

$$H(t) = (1-t)H_0 + tH_1 \tag{6.25}$$

For time t_0, the unitary can be splitted as formula 6.26:

$$U\left(t_0\right) = U\left(H_0, \beta_0\right) U\left(H_1, \gamma_0\right) \tag{6.26}$$

Thus, for each time t, we can split the unitary operators as formula 6.26. Eventually, the adiabatic pathway can be discretized into p chunks, as can be illustrated in formula 6.27:

$$U = U\left(H_0, \beta_0\right) U\left(H_1, \gamma_0\right) \ldots U\left(H_0, \beta_p\right) U\left(H_1, \gamma_p\right) \tag{6.27}$$

By the end of the time evolution t_p, QAOA can approximate the adiabatic pathway as shown in Fig 6.17:

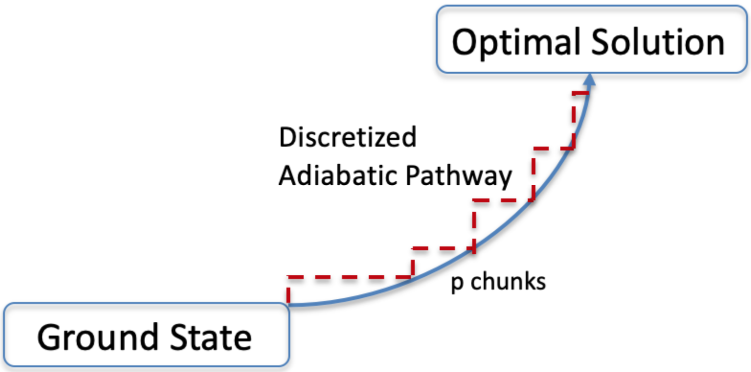


Figure 6.17: Discretized Adiabatic Pathway - p steps

A cost Hamiltonian function(objective function) and a mixing Hamiltonian function are required if using QAOA, the cost Hamiltonian function can be encoded into a set of unitary operators, and the unitary operators can be located on a quantum circuit, either on a quantum simulator or a quantum computer. In Farhi's paper, an objective function is defined on n bit string, which can be seen in formula 6.28:

$$H_C = \sum_{\alpha=1}^{m} C_\alpha(z) \tag{6.28}$$

where z is the a bit string. If z satisfies α, $H_1 = 0$ and 0 otherwise. Then a unitary operator $U(H_1, \gamma)$ is defined as formula 6.29:

$$U(H_C, \gamma) = e^{-i\gamma H_C} = \prod_{\alpha=1}^{m} e^{-i\gamma H_{C\alpha}} \tag{6.29}$$

where γ is restricted to 0 and 2π.

Following is the definition of a mixing Hamiltonian function, the simplest mixing Hamiltonian function is shown in formula 6.30:

$$H_M = -\sum_{j=1}^{n} \sigma_j^x \tag{6.30}$$

which is the sum of σ_j^x operators. Then the unitary operator $U(H_0, \beta)$ is defined as formula 6.31:

$$U(H_M, \beta) = e^{-i\beta H_M} = \prod_{j=1}^{n} e^{-i\beta\sigma_j^x} \tag{6.31}$$

where β is restricted to 0 and π.

For time step t_0, the unitary operators can be written as formula 6.32:

$$U\left(t_0\right) = U\left(H_M, \beta_0\right) U\left(H_C, \gamma_0\right) \tag{6.32}$$

if continuing the time split operation from t_0 to t_p, the evolution over step p creates such p chunks of this unitary operation, which can be written in formula 6.33:

$$|\gamma, \beta\rangle = U\left(H_M, \beta_0\right) U\left(H_C, \gamma_0\right) \ldots U\left(H_M, \beta_p\right) U\left(H_C, \gamma_p\right) \tag{6.33}$$

One important thing is that the cost Hamiltonian and mixing Hamiltonian are non-commuting operators, the mixing Hamiltonian drives the state towards an equal superposition, whereas the cost Hamiltonian seeks the ground state.

6.4.3.4 QAOA Update Rule

Because the input of QAOA is an n bit string with a cost Hamiltonian function H_1, it cannot be used for updating the weight values in a classical neural network. If one intends to use QAOA in an experiment, some modifications are needed. First modification is the cost Hamiltonian function, which is also called objective function.

Consider a continuous optimization problem, the aim is to find a minimum x^* which satisfies the equation 6.34:

$$f\left(x^*\right) \approx \min_{x \in \mathbb{R}} f(x) \tag{6.34}$$

$f(x)$ can be regarded as an objective function, so the problem is finding a state which minimizes the value of a cost Hamiltonian:

$$\hat{H}_C = f(\hat{x}) \tag{6.35}$$

In the experiment, MSE loss function was used as the cost Hamiltonian function, as defined in formula 6.36:

$$f(\hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{6.36}$$

where y_i is actual label and $\tilde{y}_i$ is the predicted label, n is the number of training dataset. The mixing Hamiltonian function is the kinetic energy in formula 6.37. This mixing Hamiltonian function is widely used in [47], other choices are also possible to use.

$$\hat{H}_M = \frac{1}{2}\sum_{j=1}^{N}\hat{p}_j^2 := \frac{1}{2}\hat{p}^2 \tag{6.37}$$

According to the Heisenberg representation[26], if $H_{t/\hbar}$ is replaced with mixing Hamiltonian function 6.37, the position operator can be written in formula 6.38:

$$e^{i\beta\hat{p}^2/2}\hat{x}e^{-i\beta\hat{p}^2/2} = \hat{x} + \beta\hat{p} \tag{6.38}$$

By using the same idea, the cost operator can be written in the form of momentum:

$$e^{i\gamma f(\hat{\alpha})}\hat{p}e^{-i\gamma f(\hat{x})} = \hat{p} - \gamma\nabla f(\hat{x}) \tag{6.39}$$

So the momentum is the negative gradient of the cost function. Then, a combination of the cost Hamiltonian(formula 6.39) and mixing Hamiltonian(6.38) is shown in formula 6.40:

$$\hat{x} \to \hat{x} + \beta\hat{p} - \gamma\beta\nabla f(\hat{x}) \tag{6.40}$$

Formula 6.40 is the gradient descent rule with momentum, each part of the wavefunction is updated by descending in the direction of its local gradient, with an additional momentum-dependent displacement. $\hat{x}$ is the value that needs to be updated, in the experiment, the weights on the neural network should be updated according to this quantum method. Thus, formula 6.40 can also be written in the form of formula 6.41:

$$\hat{w} \to \hat{w} + \beta\hat{p} - \gamma\beta\nabla f(\hat{w}) \tag{6.41}$$

Then the QAOA unitaries can be written as 6.42:

$$\hat{U}(\vec{\gamma}, \vec{\beta}) = \prod_{j=1}^{P} e^{-i\beta_j\hat{H}_M}e^{-i\gamma_j\hat{H}_C} \tag{6.42}$$

where $\gamma = \{\gamma_j\}_{j=1}^{p}$ and $\beta = \{\beta_j\}_{j=1}^{p}$. In this equation, β is the rotate angle which determine the strengths of the shifts in momentum, and γ is the learning rate on the quantum circuit, this learning rate is different from the learning rate in ADAM optimizer. Suppose the initial state is $|\Psi_0\rangle$ over the weights, and the output state $|\Psi_{\gamma,\beta}\rangle$ can be produced by applying QAOA unitaries on the initial state, which is shown in formula 6.43:

$$|\Psi_{\gamma,\beta}\rangle = \hat{U}(\gamma,\beta)\,|\Psi_0\rangle \tag{6.43}$$

6.4.4 Results and Analysis

This experiment was run on a quantum simulator(Cirq[43]), QAOA step p was set to 6. ADAM optimizer was used on a classical computer, the learning rate of ADAM optimizer was 0.005, and the loop was executed 100 times. QAOA update rule was used in specific iterations, which is 20 and 21, both in the simple neural network and Iris neural network. Except for iteration 20 and 21, the weights were updated in backpropagation.

Fig 6.18 illustrates the trend of MSE value of simple neural network during the whole process. As can be seen in Fig 6.18, MSE value decreases during the 100 iterations, which confirms that the simple neural network works well. The zoomed figure shows the MSE value on 20^{th} and 21^{th} iteration. Because the QAOA update rule was used on 20^{th} and 21^{th} iteration, the MSE value decreased sharply on both of these iterations. This experiment was executed many times and resulted in similar results, which convinced that QAOA update rule helped the simple neural network find an optimum faster and more accurate.

The trend of MSE value of Iris neural network is shown in Fig 6.19. MSE value decreases during the 100 iterations, and this result confirms that the three layers neural network works well on Iris dataset. The zoomed figure on the right side shows the MSE value on 20^{th} and 21^{th} iteration, although the QAOA update rule was applied during both iterations, MSE value did not change on either of these two iterations. This result shows that the QAOA update rule in Section 6.4.3.4 does not work on Iris dataset. One possible reason might be the inappropriate cost function, a proper cost function is needed, and it should be converted to a superposition representation. Another possible reason of this phenomenon may be the update rule is not generally correct, the QAOA update rule works on a small and simple neural network with a small size of weight matrix, but when the size of the neural network and the weight matrix grow more

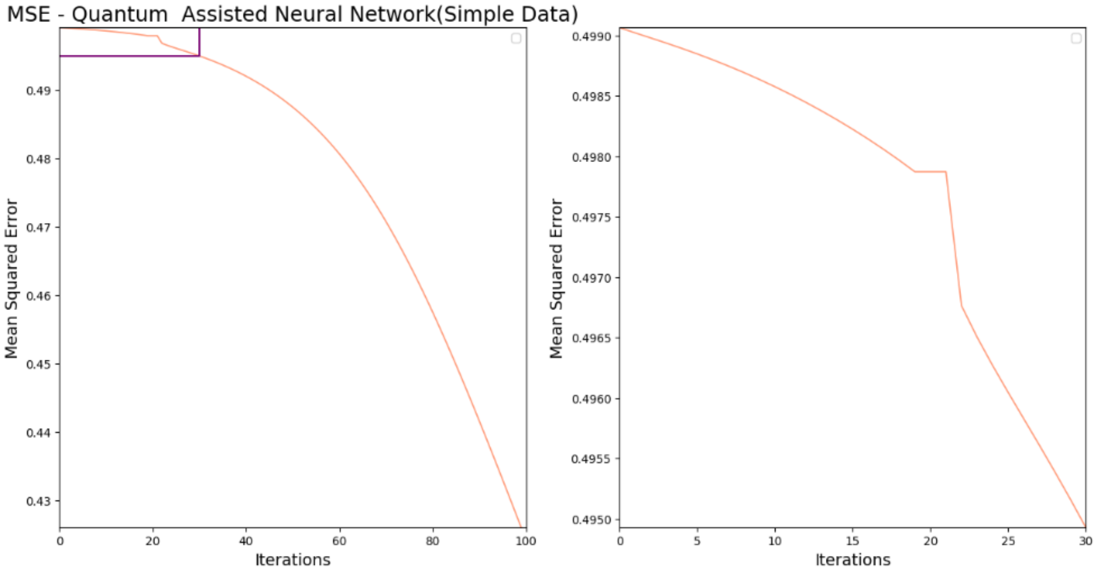


Figure 6.18: MSE - Simple Neural Network

significant, this QAOA update rule is invalid for other kinds of neural network and problems.

6.4.5 Conclusion

Due to the limitation on a quantum virtual simulator, it is slow and cannot be executed many times, and the quantum approach was only used on 20^{th} and 21^{th} iteration. The ideal setup should be a threshold of the MSE value, once the MSE value increases, the quantum approach can be executed to avoid traps during the training process. In the beginning, I wanted to use a customized optimizer in TensorFlow, and it is not easy putting the data point on the tensors and quantum circuits back and forth, probably I did not find the correct way dealing with the tensors and quantum circuits in TFQuantum. Thus only in specific iterations, the quantum approach was executed.

In this experiment, the quantum approach was effective during the training of simple neural network, which accelerated the process of optimization, but it didn't show the ability to avoid traps during training, the aim of using Iris dataset was to investigate a more complex and circuit-handleable dataset, which may cause traps during the training process, and I could figure out whether the quantum approach helps or not. Unfortunately, the QAOA update rule did not work on such problem, the reason

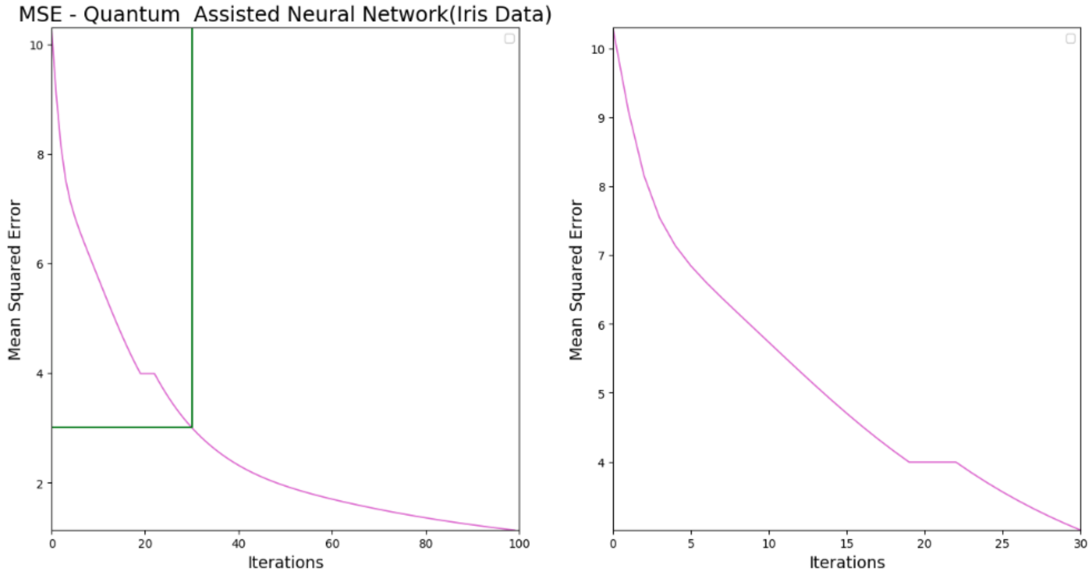


Figure 6.19: MSE - Iris Neural Network

could be the inappropriate weight-update rule, although the QAOA update rule was applicable on the simple dataset, it was not applicable to other situations, which is kind of failure in some aspect.

7

Conclusion and Future Work

In this chapter, conclusions will be presented based on the experimental results and some future works that can be done.

7.1 Conclusion

Here the research hypotheses presented in Section 5 can be answered, and draw conclusions on the following subjects: whether the quantum neural network can help accelerate the training process and achieve a higher accuracy compared to classical approaches, whether the quantum approximating optimization algorithm can speed up during specific training process and converge to an optima.

Based on the experiment in Section 6.2 and 6.3, I can conclude that a parametrized Quantum Fourier Transform can speed up the training process with fewer iterations, as is shown in Table 6.10, under the same iteration value which is 100, quantum assisted XOR neural network achieved the MSE value as 0.00114 at the end of the training process, but the classical XOR neural network only got an MSE value of 0.7104. It will cost more than $20,000$ iterations when the classical XOR neural network converges to an MSE value lower than 0.001. This experiment showed that a quantum neural network has a powerful ability to speeding up the convergence process. However, due to the limitation of current quantum hardware and the number of qubits, we can not use a quantum neural network as a complete solution for a large dataset. At present, all that can be achieved is a binary classification with the downsampled dataset.

A new trial was discussed in Section 6.4. It was an experiment about quantum assisted optimization problem, a quantum approximate optimization algorithm(QAOA) was used as an update rule, which was used in specific iterations. The idea was at first applied to a simple neural network which received binary values as input dataset, and then it was applied to a neural network which received Iris dataset as inputs. The results in Section 6.4.4 indicated that QAOA showed its potential ability in accelerating optimization problem during specific iterations, but if the size of the neural network became larger, which means the size of weight matrix exceeded the size of a quantum circuit, this tentative QAOA update rule failed in optimization.

Overall, the hypotheses in Section 5 were answered during the research, although a regression model was found for the microstructure battery dataset, the regression model should be overfitting because of the lack of training sets and redundant parameters. A parametrized quantum neural network was organized, which can successfully accelerate the training process of a problem and get higher accuracy. However, for a more complex problem, the ability of a quantum neural network not equal to its ambition, so far, it can only deal with binary classification problems. A QAOA update rule was proposed in Section 6.4.3.4 and applied it on a small problem, even if the showed improvements during specific iterations, this approach could not be a generalized method for solving such problems.

7.2 Future Work

As future directions, an interested is in checking whether these approaches can be run on a real quantum computer or a quantum simulator which has enough qubits for use. Google will announce a 72-qubit superconducting quantum chip in this year or next year[43], which is an encouraging and exciting event for many researchers.

Secondly, I still have doubt about the QAOA update rule, I suppose something wrong with the definition in my research, such as the choice of the cost function, and the representation of superposition in a quantum circuit, etc.. It is a severe problem to answer clearly within several months only, my research was a trial which showed that this kind of approach might not be feasible. This kind of research can be a stepping stone for further studies in using QAOA as an optimizer, and I will also keep adjusting this approach for more extensive uses.

References

[1] A. A. Abd El-Latif, B. Abd-El-Atty, and M. Talha. **Robust Encryption of Quantum Medical Images**. *IEEE Access*, **6**:1073–1081, 2018. 8

[2] Steven H. Adachi and Maxwell P. Henderson. **Application of Quantum Annealing to Training of Deep Neural Networks.** *CoRR*, **abs/1510.06356**, 2015. 10

[3] Tameem Albash and Daniel A. Lidar. **Adiabatic Quantum Computing**. *Rev. Mod. Phys. 90, 015002 (2018)*, 2016. 10

[4] Tameem Albash, Walter Vinci, and Daniel A. Lidar. **Simulated Quantum Annealing with Two All-to-All Connectivity Schemes**. *Phys. Rev. A 94, 022327 (2016)*, 2016. 10

[5] ALIBABA. **Quantum Lab.** *Alibaba Damo Academy*, 2018. 9

[6] A.Pais. **Einstein and the Quantum Theory**. *American Physical Society*, 1979. 11

[7] Srinivasan Arunachalam and Ronald de Wolf. **Guest Column: A Survey of Quantum Learning Theory.** *SIGACT News*, **48**(2):41–67, 2017. 7

[8] Dave Bacon. **Quantum Computing. Introduction and Basics of Quantum Theory**. *University of Washington*, 2018. 3, 19

[9] BAIDU. **Quantum Computing**. *Baidu Research*, 2018. 9

[10] Nilesh Barde, Debajit Thakur, Pranav P Bardapurkar, Sanjaykumar N Dalvi, S. N. Mor Arts, and Bhagyashri Sarda. **Consequences and Limitations of Conventional Computers and their Solutions through Quantum Computers**. 2011. 1

[11] Christian Bauckhage, Eduardo Brito, Kostadin Cvejoski, César Ojeda, Rafet Sifa, and Stefan Wrobel. **Ising Models for Binary Clustering via Adiabatic Quantum Computing.** In *EMMCVPR*, **10746**, pages 3–17, 2017. 10

[12] Paul Benioff. **The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines**. *Journal of Statistical Physics*, **22**:563–591, 1980. 1

[13] K. Bertels, I. Ashraf, R. Nane, S. Varsamopoulos, A. Mouedenne, A. Sarkar, and N. Khammassi. **Quantum Computer Architecture: Towards Full-Stack Quantum Accelerators**, 2019. 8, 10

[14] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. **Quantum machine learning**. *Nature*, **549**(7671):195–202, 9 2017. 1, 6, 7

[15] Niels Bohr. **Bohr Theory of Hydrogen**. *Physics2000*, 1915. 2

[16] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven. **Characterizing quantum supremacy in near-term devices**. *Nature Physics*, **14**:595–600, April 2018. 1

[17] Sergio Boixo, Sergei Isakov, Vadim Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J. Bremner, John Martinis, and Hartmut Neven. **Characterizing Quantum Supremacy in Near-Term Devices**. *Nature Physics*, **14**:595–600, 2018. 6

[18] Juergen E. Braun and Michael Griebel. **On a Constructive Proof of Kolmogorov's Superposition Theorem**. *Constructive Approximation*, **30**:653–675, 2009. 2

[19] Jeffrey Bub. **Quantum mechanics is about quantum information**. *Foundations of Physics*, **35**:4, 2005. 2, 3

[20] Cristian S. Calude and Elena Calude. **The Road to Quantum Computational Supremacy**. *arXiv:1712.01356*, 2017. 1

[21] Quantum Computing. **What's the difference between quantum annealing and universal gate quantum computers?** 2018. Available from: `https://www.amarchenkova.com/2016/02/28/quantum-annealing-vs-universal-gate-quantum-computer/`. 9

[22] D. Coppersmith. **Tech. Rep. IBM Research Report.** *An approximate Fourier transform useful in quantum factoring*, 1994. 37

[23] Tatjana Curcic. **Optimization with Noisy Intermediate-Scale Quantum devices (ONISQ)**. *Research Funding*, 2019. 1

[24] D-Wave. **The D-Wave 2000Q Quantum Computer**. *Tech Collateral*, 2016. 10

[25] Pierre-Luc Dallaire-Demers and Frank K. Wilhelm. **Quantum gates and architecture for the quantum simulation of the Fermi-Hubbard model**. *Phys. Rev. A*, **94**:062304, Dec 2016. 17

[26] E.Schr ¨odinger. **About Heisenberg Uncertainty Relation**. *Physics-Mathematical Section*, pages 296–303, 1930. 18, 59

[27] Kai DeLorenzo, Shelby Kimmel, and R. Teal Witter. **Applications of the quantum algorithm for st-connectivity**. *arXiv:1904.05995*, 2019. 1

[28] David Deutsch. **Quantum computational networks**. 1989. 1, 2

[29] Albert Einstein. **Einstein's Light Quanta**. *Quanta*, 1905. 2

[30] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. **A Quantum Approximate Optimization Algorithm**. *arXiv:1411.4028*, 2014. 8, 31, 46, 55, 56

[31] Edward Farhi and Hartmut Neven. **Classification with Quantum Neural Networks on Near Term Processors**. *arXiv:1802.06002*, 2018. 9, 45

[32] D. V. Fastovets, Yu. I. Bogdanov, B. I. Bantysh, and V. F. Lukichev. **Machine learning methods in quantum computing theory**. *arXiv:1906.10175*, 2019. 1, 6, 7

[33] Richard P. Faynman. **There's Plenty of Room at the Bottom**. *Engineering and Science magazine*, **XXIII**:5, 1960. 2

[34] Richard P. Faynman. **Simulating physics with computers**. *International Journal of Theoretical Physics*, **21**:467–488, 1982. 1, 2

[35] Sebastian Feld and Claudia Linnhoff-Popien. **Quantum Technology and Optimization Problems**. *Springer International Publishing*, 2019. 1

[36] Richard P. Feynman. **Quantum Mechanical Computers**. *Foundations of Physics*, **16**(6):507–531, 1986. 1

[37] Mark Fingerhuth, Tomás Babej, and Peter Wittek. **Open source software in quantum computing.** *CoRR*, **abs/1812.09167**, 2018. 8, 10, 19

[38] Mark Fingerhuth, Tomáš Babej, and Peter Wittek. **Open source software in quantum computing**. *arXiv:1812.09167*, 2018. 8, 10

[39] Christopher A. Fuchs. **Quantum Mechanics as Quantum Information (and only a little more)**. *arXiv:quant-ph/0205039*, 2002. 2

[40] Ziemowit Fuhr, Hartmut; Rzeszotnik. **A note on factoring unitary matrices**. *Linear Algebra and its Applications*, 2018. 17

[41] ANDRÁS GILYÉN, SRINIVASAN ARUNACHALAM, AND NATHAN WIEBE. **Optimizing quantum optimization algorithms via faster quantum gradient computation.** In *SODA*, pages 1425–1444. SIAM, 2019. 7

[42] CARLOS PEDRO GONCALVES. **Quantum financial economics — risk and returns**. *Journal of Systems Science and Complexity*, **26**(2):187–200, Apr 2013. 8

[43] GOOGLE. **Quantum**. *Google AI*, 2019. 9, 12, 17, 39, 48, 60, 64

[44] DANIEL GOTTESMAN. **An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation**. *arXiv:0904.2557*, April 2009. 6

[45] PHILIPPE GRANGIER, GEORGES REYMOND, AND NICOLAS SCHLOSSER. **Implementations of Quantum Computing Using Cavity quantum Electrodynamics Schemes**. *Fortschr. Phys.*, **48**:9–11,859–874, 2000. 11

[46] SANDER GREENLAND, STEPHEN J. SENN, KENNETH J. ROTHMAN, JOHN B. CARLIN, CHARLES POOLE, STEVEN N. GOODMAN, AND DOUGLAS G. ALTMAN. **Statistical tests, P values, confidence intervals, and power: a guide to misinterpretations**. *European Journal of Epidemiology*, **31**(4):337–350, Apr 2016. 26

[47] STUART HADFIELD, ZHIHUI WANG, BRYAN O'GORMAN, ELEANOR G. RIEFFEL, DAVIDE VENTURELLI, AND RUPAK BISWAS. **From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz**. *Algorithms*, **34**, 2017. 59

[48] DAVID J. HAND. **Statistical Concepts: A Second Course, Fourth Edition by Richard G. Lomax, Debbie L. Hahs-Vaughn**. *International Statistical Review*, **80**(3):491–491, December 2012. 24

[49] MIGUEL HERRERO-COLLANTES AND JUAN CARLOS GARCIA-ESCARTIN. **Quantum random number generators**. *Rev. Mod. Phys.*, **89**:015004, Feb 2017. 7

[50] GEOFFREY E. HINTON. **A Practical Guide to Training Restricted Boltzmann Machines**. *Neural Networks: Tricks of the Trade: Second Edition*, pages 599–619, 2012. 1, 19

[51] A. S. HOLEVO. **Bounds for the Quantity of Information Transmitted by a Quantum Communication Channel**. *Probl. Peredachi Inf.*, **9**:3, 1973. 5

[52] RYSZARD HORODECKI, PAWEŁ HORODECKI, MICHAŁ HORODECKI, AND KAROL HORODECKI. **Quantum entanglement**. *Rev. Mod. Phys.*, **81**:865–942, Jun 2009. 14

[53] IBM. **IBM Q**. *The future is quantum*, 2019. 9

[54] D-WAVE SYSTEMS INC. **D-wave official website**. 2018. 9

[55] INTEL. **Quantum Computing**. *Intel news room*, 2019. 9

[56] J. A. Jones and M. Mosca. **Implementation of a Quantum Algorithm to Solve Deutsch's Problem on a Nuclear Magnetic Resonance Quantum Computer**. *J.Chem.Phys. 109 (1998) 1648-1653*, 1998. 11

[57] Yoshito Kanamori, Seong-Moo Yoo, W Pan, and F.T. Sheldon. **A short survey on quantum computers**. *International Journal of Computers and Applications*, **28**, 01 2006. 11

[58] Hirotada Kobayashi, François Le Gall, Harumichi Nishimura, and Martin Rötteler. **General Scheme for Perfect Quantum Network Coding with Free Classical Communication**. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris Nikoletseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming*, pages 622–633, Berlin, Heidelberg, 2009. 8

[59] V. M. Krasnov. **Stacked Josephson junction SQUID**. *Physica C 368 (2002) 246-250*, 2001. 5

[60] Nicolaas P. Landsman. *Born Rule and its Interpretation*, pages 64–70. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. 14

[61] Federico Laudisa and Carlo Rovelli. **Relational Quantum Mechanics**. *Metaphysics Research Lab, Stanford University*, 2013. 2

[62] Edward Li, Chi-Kwong; Poon. **Additive decomposition of real matrices**. *Linear and Multilinear Algebra*, 2002. 16, 17

[63] Tongcang Li and Zhang-Qi Yin. **Quantum superposition, entanglement, and state teleportation of a microorganism on an electromechanical oscillator**. *Science Bulletin 2016, 61(2):163-171*, 2015. 5

[64] D-Wave User Manual. **Problem Solving Handbook**. 2018. Available from: `https://cloud.dwavesys.com/`. 9

[65] Matteo Mariantoni, H. Wang, T. Yamamoto, M. Neeley, Radoslaw C. Bialczak, Y. Chen, M. Lenander, Erik Lucero, A. D. O'Connell, D. Sank, M. Weides, J. Wenner, Y. Yin, J. Zhao, A. N. Korotkov, A. N. Cleland, and John M. Martinis. **Implementing the Quantum von Neumann Architecture with Superconducting Circuits**. *Science 334, 61-65 (2011)*, 2011. 11

[66] Ayesha Tasnim Marufa Rahmi, Debakar Shamanta. **Basic Quantum Algorithms and Applications**. *International Journal of Computer Applications*, **56**, 2012. 1

[67] Catherine C. McGeoch. *Adiabatic Quantum Computation and Quantum Annealing: Theory and Practice.* Morgan & Claypool Publishers, 2014. 7

[68] MICROSOFT. **Quantum Labs**. *Microsoft Quantum*, 2019. 9

[69] M. Mohseni, P. Read, H. Neven, S. Boixo, V. Denchev, R. Babbush, A. Fowler, V. Smelyanskiy, and J. Martinis. **Commercialize quantum technologies in five years**. *Nature*, **543**:171–174, March 2017. 1

[70] Nikolaj Moll, Panagiotis Barkoutsos, Lev S Bishop, Jerry M Chow, Andrew Cross, Daniel J Egger, Stefan Filipp, Andreas Fuhrer, Jay M Gambetta, Marc Ganzhorn, Abhinav Kandala, Antonio Mezzacapo, Peter Müller, Walter Riess, Gian Salis, John Smolin, Ivano Tavernelli, and Kristan Temme. **Quantum optimization using variational algorithms on near-term quantum devices**. *Quantum Science and Technology*, **3**(3):030503, jun 2018. 7, 9

[71] Gordon E. Moore. **Cramming more components onto integrated circuits**. *Electronics*, **38**(8), April 1965. 3

[72] Marius Nagy and Selim G. Akl. **Quantum computing: beyond the limits of conventional computation**. *International Journal of Parallel, Emergent and Distributed Systems*, **22**(2):123–135, 2007. 1, 14, 15

[73] Florian Neukart, Gabriele Compostella, Christian Seidel, David von Dollen, Sheir Yarkoni, and Bob Parney. **Traffic Flow Optimization Using a Quantum Annealer**. *Frontiers in ICT*, **4**:29, 2017. 7, 8

[74] Florian Neukart, David Von Dollen, and Christian Seidel. **Quantum-Assisted Cluster Analysis on a Quantum Annealing Device**. *Frontiers in Physics*, **6**:55, 2018. 1, 6, 7

[75] Florian Neukart, David Von Dollen, Christian Seidel, and Gabriele Compostella. **Quantum-Enhanced Reinforcement Learning for Finite-Episode Games with Discrete State Spaces**. *Frontiers in Physics*, **5**:71, 2018. 1, 6, 7

[76] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, New York, NY, USA, 10th edition, 2011. 37

[77] Harumichi Nishimura and Masanao Ozawa. **Computational complexity of uniform quantum circuit families and quantum Turing machines.** *Theor. Comput. Sci.*, **276**(1-2):147–181, 2002. 7

[78] Benjamin Perez-Garcia, Melanie McLaren, Sandeep K. Goyal, Raul I. Hernandez-Aranda, Andrew Forbes, and Thomas Konrad. **Quantum computation with classical light: implementation of the Deutsch-Jozsa Algorithm**. *arXiv:1510.03365*, 2015. 11

[79] P.W.Shor. **Algorithms for quantum computation: discrete logarithms and factoring.** *S. Goldwasser (Ed.), Proceedings of the 35th Annual Symposium on the Foundations of Computer Science, IEEE Computer Society, Los Alamitos.*, pages 124–134, 1994. 37

[80] P.W.Shor. **Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer.** *SIAM J. Comput.*, **26**(5):1484–1509, 1997. 7

[81] Bo-Qiang Qin, Pengzhu Xu, Qinglong Wu, Liancong Luo, and Yunlin Zhang. *Environmental Issues of Lake Taihu, China*, **581**, pages 3–14. 05 2007. 12

[82] Supriya Raheja, Reena Dadhich, and Smita Rajpal. **An Optimum Time Quantum Using Linguistic Synthesis for Round Robin Scheduling Algorithm**. *CoRR*, **abs/1203.2247**, 2012. 7

[83] Maximilian Schlosshauer. **Decoherence, the measurement problem, and interpretations of quantum mechanics**. *Reviews of Modern Physics*, **76**(4):1267–1305, February 2005. 6

[84] J.P.Singh Shaktikanta Nayak, Sitakanta Nayak. **A POSSIBLE APPLICATION OF QUANTUM ALGORITHM FOR MARKET PRICE PREDICTION**. *Journal of Global Research in Computer Science*, **4**, 2013. 1

[85] R. Shaydulin, H. Ushijima-Mwesigwa, C. F. A. Negre, I. Safro, S. M. Mniszewski, and Y. Alexeev. **A Hybrid Approach for Solving Optimization Problems on Small Quantum Computers**. *Computer*, **52**(6):18–26, June 2019. 1

[86] Vidya Raj C. M. S. Shivakumar. **Applying Quantum Algorithm to Speed Up the Solution of Hamiltonian Cycle Problems**. *International Conference on Intelligent Information Processing*, **228**, 2006. 1

[87] Presentation Slides. **Cathode Microstructure Simulation for Machine Learning Application**. *Konzerthaus Berlin*, 2019. Available from: `https://www.konzerthaus.de`. 21, 23

[88] Andrew M. Steane. **The ion trap quantum information processor**. *Applied Physics B*, **64**:623–643, 1997. 11

[89] Kok Chuan Tan, S. Omkar, and Hyeonseok Jeong. **Coherence as a Unit Resource for Quantum Error Correction**. *arXiv:1704.07572*, 2017. 6

[90] CARLOS TRUJILLO and JORGE GARCIA-SUCERQUIA. **GRAPHICS PROCESSING UNITS: MORE THAN THE PATHWAY TO REALISTIC VIDEO-GAMES**. *DYNA*, **78**:164 – 172, 08 2011. 1

[91] Alan Turing. **On Computable Numbers, with an Application to the Entscheidungsproblem**. *Proceedings of the London Mathematical Society*, **42**(1):230–265, 1936. 2

[92] Graham J. G. Upton. **Fisher's Exact Test**. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, **155**(3):395–402, 1992. 24

[93] James A. Walsh. **George W. Snedecor and William G. Cochran. Statistical Methods. (6th ed.)**. *Educational and Psychological Measurement*, **29**(2):546–549, 1969. 24

[94] Jianwei Wang, Stefano Paesani, Raffaele Santagati, Sebastian Knauer, Antonio A. Gentile, Nathan Wiebe, Maurangelo Petruzzella, Jeremy L. O'Brien, John G. Rarity, Anthony Laing, and Mark G. Thompson. **Experimental quantum Hamiltonian learning**. *Nature Physics*, **13**(6):551–555, 2017. 18

[95] C. Wetterich. **Quantum computing with classical bits**. *arXiv:1806.05960*, 2018. 5

[96] Wiki. **Bra-Ket Notation**. *Wikipedia*, 2019. 13

[97] Wiki. **Josephson effect**. *Wikipedia*, 2019. 5

[98] Wiki. **Unitary Matrix**. *Wikipedia*, 2019. 17

[99] Peter Wittek. **Quantum Machine Learning: What Quantum Computing Means to Data Mining**. *ELSEVIER*, 2014. 1, 6, 7

[100] Phil Woodruff and Alex Bradshaw. **Ultraviolet catastrophe?** *Physics World*, **11**(1):17–20, jan 1998. 2

[101] Ahmed Younes and Julian Miller. **Representation of Boolean Quantum Circuits as Reed-Muller Expansions**. **7**:431–444, 2003. 47

[102] A Zahir, Syed Azhar Ali Zaidi, Azzurra Pulimeno, Mariagrazia Graziano, Danilo Demarchi, Guido Masera, and Gianluca Piccinini. **Molecular transistor circuits: From device model to circuit simulation**. *IEEE/ACM International Symposium on Nanoscale Architectures*, pages 129–134, 07 2014. 3

[103] Zhikuan Zhao, Alejandro Pozas-Kerstjens, Patrick Rebentrost, and Peter Wittek. **Bayesian Deep Learning on a Quantum Computer**. *arXiv:1806.11463*, 2018. 1, 6, 7, 10, 11

[104] Sisi Zhou, Thomas Loke, J. Izaac, and Jb Wang. **Quantum Fourier Transform in Computational Basis**. *Quantum Information Processing*, **16**, 11 2015. 38