Shahjalal University of Science and Technology
Department of Computer Science and Engineering

Fundamental Machine Learning Routines as Quantum Algorithms on a Superconducting Quantum Computer

Sristy Sangskriti	Protik Nag
Reg. No.: 2015331019	Reg. No.: 2015331022
4th year, 2nd Semester	4th year, 2nd Semester

Department of Computer Science and Engineering

Supervisor

Summit Haque

Lecturer

Department of Computer Science and Engineering

8th February, 2020

Fundamental Machine Learning Routines as Quantum Algorithms on a Superconducting Quantum Computer

A Thesis submitted to the

Department of Computer Science and Engineering

Shahjalal University of Science and Technology

Sylhet - 3114, Bangladesh

in partial fulfillment of the requirements for the degree of

Bachelor of Science in Computer Science and Engineering

By

Sristy Sangskriti

Reg. No.: 2015331019

4th year, 2nd Semester

Protik Nag

Reg. No.: 2015331022

4th year, 2nd Semester

Department of Computer Science and Engineering

Supervisor

Summit Haque

Lecturer

Department of Computer Science and Engineering

8th February, 2020

Recommendation Letter from Thesis Supervisor

The thesis

entitled *Fundamental Machine Learning Routines as Quantum Algorithms on a Superconducting Quantum Computer*

submitted by the students

1. Sristy Sangskriti

2. Protik Nag

is a record of research work carried out under my supervision and I, hereby, approve that the report be submitted in partial fulfillment of the requirements for the award of their Bachelor Degrees.

Signature of the Supervisor:

Name of the Supervisor: Summit Haque

Date: 8th February, 2020

Certificate of Acceptance of the Thesis

The Research Project

entitled *Fundamental Machine Learning Routines as Quantum Algorithms on a Superconducting Quantum Computer*

submitted by the students

1. Sristy Sangskriti

2. Protik Nag

on 8th February, 2020

is, hereby, accepted as the partial fulfillment of the requirements for the award of their Bachelor Degrees.

Head of the Dept.	**Chairman, Exam. Committee**	**Supervisor**
Mohammad Abdullah Al Mumin	Dr Mohammad Reza Selim	Summit Haque
Professor & Head	Professor	Lecturer

Abstract

The Harrow-Hassidim-Lloyd algorithm is intended for solving the system of linear equations on quantum devices. The exponential advantage of the algorithm comes with four caveats. We present a numerical study of the performance of the algorithm when these caveats are not perfectly matched. We observe that, between diagonal and non-diagonal matrices, the algorithm performs with higher success probability for the diagonal matrices. At the same time, it fails to perform well on lower or higher density sparse Hermitian matrices. Again, Quantum Support Vector Machine algorithm is a promising algorithm for classification problem. We have found out that it works better with binary classification problem than multi-label classification problem. And there are many opportunities left for improving the performance.

Keywords: Quantum Computer, Machine learning, Qubit, Superposition, Spin, Entanglement, Quantum support vector machine.

Acknowledgements

Contents

List of Tables

List of Figures

Chapter 1

Introduction

As humans become increasingly dependent on machines, we are observers to a modern revolution that is assuming control over the world, and that will be the inevitable future of Machine Learning. Machine learning algorithms are developed in such a way that they can process lots of data together and train a model to make decisions. But in order to speed up this process we need a new technology called quantum computers which will make revolutionary change in this field.

1.1   The Beginning

Quantum computing, an applied science of quantum mechanical phenomena, was introduced with the notion of simulating things that a traditional computer can not do. Quantum computers are accelerated machines that can compute exponentially faster than classical computers. New algorithms for solving difficult problems using quantum computers have been being developed for years. One of these problems are systems of linear equations.

Quantum computer can be a life saver in the field of machine learning and the vast of AI deserves a big change. Quantum machine learning will help promote the computational power to handle problems that are too complex too solve. Some discussable fields of quantum machine learning algorithms are finding eigenvalues and eigenvectors of large matrices, analysis of classical machine learning algorithms on quantum computers etc.

Besides, traditional computers increases their processing power by expanding the quantity of

transistors which are the essential part of the basic structure of classical computers. A transistor can just hold one state at a time i.e. switch on or off, an ever increasing number of transistors is necessary to process on every single possible states. So the best way to amplify computing power is to increase the number of transistors and to minify them. The mostly developed chips today consist of a huge number of transistors on a single chip. But there is also a limitation to how little they can get. Till now there is no quantum effect of their sizes but someday it will impossible to make transistors any smaller [3].

1.2 Scope

System of linear equations is of great importance in basically every field of science and engineering and also in finance, economics. These systems seem to appear in the area of solving differential or partial equations and also in machine learning problems such as regression. The HHL algorithm can be used as a subroutine in a bigger quantum algorithm. It is a fundamental algorithm which can further be used to support many quantum machine algorithms [4–10]. But due to the ever increasing amount of data, it becomes too difficult to acquire a solution by processing this huge amount of data. This situation requires a great deal of computational power and precious time.

Although quantum computers can accelerate the processing for some problems than classical ones, there are many parts which still contain some uncertainty or restrictions [11]. We can mention the quantum phase estimation algorithm [12] in this context. The eigenvalue estimation of hermitian matrix is one major step of the HHL algorithm. Hermitian matrix is discussed in the latter chapter. As the phase estimation system is embedded in this algorithm there are also some uncertainty of getting the exact output from this algorithm.

But quantum computers may not supplant classical computers. Quantum computers are supposed to be able to work on specific kinds of problems like large optimizations. They will also be expensive to run. But quantum computers have exponential potential over classical computers. When classical computers use backtracking in order to find a path through a tangle, quantum computer parellelly search through every possible path to find the destination. By adding more qubits to a quantum computer the power of the computer can be increased at an exponential rate.

Machine Learning algorithms such as neural networks sometimes require too much time to train

due to lack of computational power. It has become a necessity to accelerate the training process and making complex algorithms feasible. To solve such problems related to computational complexity, quantum computing is introduced. Using Support Vector Machine algorithms on quantum computers for helpful classification of data and dealing with complex problems may be a small step.

Quantum computers can be used to do prime factorization in no time and this unleash demolition on cyber security. Quantum computing can also be applied to understand nano particles, designing chemicals and drugs, discovery about the universe, in renewable energy and so on. It may revolutionize every aspect of our society.

1.3   Motivation

Machine learning is a common platform to almost every field of science because it helps to make decisions and gives an overview of the scenery. But the base of machine learning lies in mathematics. The field of mathematics has reached beyond the scope classical mechanics and mathematics gives new ideas to improve machine learning algorithms. But to implement these algorithms and ideas scientists need greater computational power which classical computers can not meet always or even if they can process the ideas taking too much time. So an era of quantum computers is coming toward us.

Chapter 2

Statement of the Problems and Summary of Results

This whole thesis work can be split into two major parts. First we summarize the statements and then give a summary of the results.

2.1 A Numerical Study of the Harrow-Hassidim-Lloyd Algorithm for Solving Systems of Linear Equations

The Harrow-Hassidim-Lloyd algorithm is used to solve systems of linear equations. The details about this algorithm is described in the latter chapter 3. In our contribution, we have decided to benchmark this algorithm. We have tried to find out the flaws and have provided some corner cases of this algorithm. We have measured the output it returns at different cases. Our goal of this contribution is to discover how the HHL algorithm performs on various matrices. The algorithm has been examined with different sizes of matrices, with differently dense matrices and with both diagonal and non-diagonal matrices. We have learnt that the HHL algorithm performs well with small sized, low density, diagonal matrices. Otherwise, the performance starts to decline. Chapter 6 contains more details about the experiment we have conducted to come to this conclusion.

2.2 Comparative Study of Different Quantum Machine Learning Algorithm for Solving a Binary Classification Problem

Support Vector Machine (SVM) is a traditional classical machine learning approach, where vectors are constructed through the problem space such that they divide up the problem data into separate classifications. We expect quantum computers to be able to determine those vectors faster than classical hardware. The reason behind this expectation is explained in the chapter 3. Though, quantum hardwares are not at a stage where the performance outpaces that of classical hardwares currently. Our contribution shows results of an experiment of implementing the quantum support vector machine algorithm for a binary classification problem and for a multiple class problem and analyze those results with the classical ones.

2.3 Summary of Results

In a nutshell, we can divide our research contribution into two parts. One is to benchmark the HHL algorithm and another one is to run a quantum machine learning algorithm on a standard dataset and correlate the results with a classical one.

At first, while working on the HHL algorithm, we have observed some key points. We have found out that this algorithm performs better with smaller-sized matrices and the performance declines dramatically when the number of equations increases. Also, it functions better with lower density matrices. As the density increases, both performance and efficiency reduces. All the graphs are provided in chapter 7. Again, to solve higher dimensional system of linear equations, it requires larger circuit, which likewise reduces the productivity and becomes more susceptible to inaccuracy. Surprisingly, this algorithm works better on diagonal matrices than the non-diagonal ones.

In the later part, we have experimented the quantum subpart of the Support Vector Machine algorithm. We have used two datasets named the Breast Cancer Dataset [13] and the Iris Dataset [14]. Description of these datasets are presented in the chapter 6. We have applied QSVM on these datasets and measured the accuracy. Also, we have run the classical SVM and gathered the results. We have studied both of the procedures and have found out that quantum devices do not perform well regarding Support Vector Machine algorithm yet. They are still far away from the

state-of-the-art classical performance. But the performance can be stepped up by conducting the same experiments several times and taking the finest output. How the performance increases as the number of shots increases has been explained in chapter 7. Though this kind of approach can be inefficient in terms of memory and time.

Chapter 3

Preliminary Concepts

Before we start to discuss about the quantum system and its components, we need some basic knowledge about some mathematical terms and theorems. These are complex numbers, qubits, superposition and so on. Then we discuss about the research of other people integrating machine learning on quantum computers.

3.1   Complex Vector Space

Complex number is the base of quantum computing. Complex number is the field of mathematics where lessons of quantum computing starts from. Although complex numbers were practised because of mathematical curiousity, scientists gradually understood that complex numbers are omnipresent and very much important.

A *complex vector space* is nonempty set of vectors $\mathbb{V}$ which satisfies addition, negation and scalar multiplication and a distinct element zero vector $0 \in \mathbb{V}$ in the set [1].

If $\mathbb{B} = \{V_0, V_1, ..., V_{n-1}\}$ is a set of vectors in complex vector space $\mathbb{V}$ so that every $V \in \mathbb{V}$ can be written as a linear combination of vectors from $\mathbb{B}$ and $\mathbb{B}$ is linearly independent i.e. $0 = c_0 \cdot V_0 + c_1 \cdot V_1 + ... + c_{n-1} \cdot V_{n-1}$ where $c_0 = c_1 = ... = c_{n-1} = 0$, then $\mathbb{B} \in \mathbb{V}$ is called a *basis* of $\mathbb{V}$. A *canonical basis* or *standard basis* is comprised of a number of functions $f_j (j = 0, 1, 2, 3, ...)$ where $f_j = \begin{cases} 1, & if\ j = n, \\ 0, & otherwise \end{cases}$.

A change of basis matrix or *transition matrix* is another matrix. It refers to the change from

some basis $\mathbb{B}$ to some basis $\mathbb{E}$ for any vector V such that $V_E = M_{E \leftarrow B} \cdot V_B$, where $M_{E \leftarrow B}$ is the transition matrix.

Now, a *Hadamard matrix H* is the transition matrix from the canonical basis $\left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\}$ to this other basis $\left\{ \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} \right\}$ in $\mathbb{R}^2$.
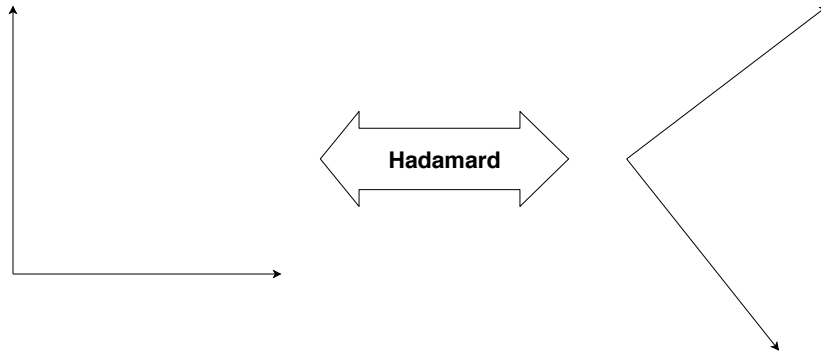
Figure 3.1: The Hadamard matrix shown as a transition between two bases [1].

In figure 3.1 the transition between two hadamard matrices is depicted.

A Hadamard matrix multiplied by itself gives the identity matrix. The transition back to the canonical basis is a Hadamard matrix too. So Hadamard matrix has the property of "reversibility". Hadamard matrices play a major role in quantum computing.It is used in data encryption, signal processing, data compression algorithms. It is a crucial part of Shor's algorithm [15].

3.2   Qubit and Superposition

As depicted in the book of "Quantum Computing For Computer Scientists" [1], in classical computing system we measure information using *bit*. But in two dimensional quantum system information is measured by quantum bit or *qubit*. In classical binary system there are two bits 0 and 1. These two bits can either indicate presence or absence of electricity or denote true or false or maybe refer to switch on or off. So a bit is a system where there are only two possible states. We can represent these states by 2-by-1 matrices such as state $|0\rangle = \begin{matrix} 0 \\ 1 \end{matrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{matrix} 0 \\ 1 \end{matrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Now in quantum world, a qubit can be in one state and in other state simultaneously i.e. a qubit

can represent true and false or state $|0\rangle$ and $|1\rangle$ at the same time. We can represent such a qubit by a 2-by-1 matrix with complex numbers $|1\rangle = \begin{matrix} 0 \\ 1 \end{matrix} \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}$ where $|c_0|^2 + |c_1|^2 = 1$. A qubit becomes a bit after being measured. One of these $|c_x|^2$ is to be interpreted as the probability that after measuring the qubit it will be found in state $|x\rangle$ where $x = 0, 1$.

Thus one qubit represents two different states, two qubits can take on any of the four probable states 01, 11, 10, or 00. So n qubits can represent 2^n different states. This concept of superposition is the key for quantum computers to be much more powerful than classical computers. The key idea is to represent much more information using less computing power.

Let us suppose that a particle can be only be detected at one of the equally spaced points $x_0, x_1, ..., x_{n-1}$ of a line where $x_1 = x_0 + \delta x, x_2 = x_1 + \delta x, ...$ with some fixed increment δx. The particle being at point x_i is denoted as $|x_i\rangle$ (Dirac ket notation). Each of these basic states is associated with an n dimensional complex column vector generally denoted as $[c_0, c_1, ..., c_{n-1}]^T$. Now, an arbitrary state denoted as $|\psi\rangle$ is a linear combination of $|x_0\rangle, |x_1\rangle, ..., |x_{n-1}\rangle$

$$|\psi\rangle = c_0 |x_0\rangle + c_1 |x_1\rangle + ... + c_{n-1} |x_{n-1}\rangle \tag{3.1}$$

where $c_0, c_1, ... c_{n-1}$ are complex amplitudes. This $|\psi\rangle$ can be described as the overlap of n waves each contributing with amplitude c_i. We call this $|\psi\rangle$ a **superposition** of the basic states, which represents the particle as being in all locations $x_0, x_1, ... x_{n-1}$ at the same time.The probability of the particle being at point x_i after being observed is

$$p(x_i) = \frac{|c_i|^2}{||\psi\rangle|^2} = \frac{|c_i|^2}{\sum_j |c_j|^2} \tag{3.2}$$

3.3 Spin and Entanglement

Even if quantum particles are isolated by long distance, they can collaborate with one another or the effect of the movement of one particle can affect another one. This phenomenon is called quantum entanglement.

Spin is basic property of quantum particles and has no classical analogue. In 1922 the Stern-Gerlach Experiment [1] showed that an electron in the presence of a magnetic field behaves as
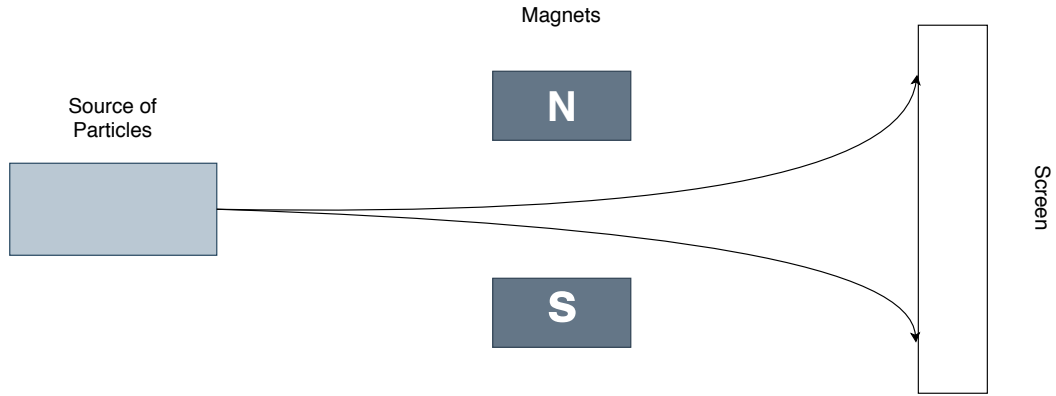
Figure 3.2: The Stern-Gerlach experiment [1].

though it were a charged spinning top and acts as a small magnet. As shown in Figure 3.2 the source shoots beams of electrons through a magnetic field which is nonhomogeneous. The electrons are supposed to hit vertically directly into the screen. Bur they get splitted by the magnetic field into two opposite direction with opposite spin. Some electrons spin one way and some spin the opposite way. Now the electrons do not have an internal magnetic structure, they are just charged particles. From this experiment it is shown that the spinning particle when being measured in a specific direction can only be found either clockwise or anticlockwise state. For each of this given directions there exists two basic spin states. For example, for vertical axis these states are spin up $|\uparrow\rangle$ and spin down $|\downarrow\rangle$. So the generate state becomes the superposition of of up and down i.e.

$$|\psi\rangle = c_0 |\uparrow\rangle + c_1 |\downarrow\rangle \tag{3.3}$$

where c_0 and c_1 are respectively the amplitudes of finding the particle in the up state and down state.

Now that we have discussed about "spin", if one particle of a pair is chosen to be in spin up state while being measured, at that point this choice is conveyed to the next connected particle that now acquires the opposite state spin "down". Thus quantum entanglement allows far-off qubits to interact immediately with each other.

3.4   Quantum Logic Gates

In classical computing we have logical gates such as AND, OR, NOT gates which work on bits. But in the quantum world, only reversible gates work as quantum logical gates. These gates are represented by Unitary matrices and they work on qubits [1]. Operations that are not measurements such as NOT gate and Identity gates are reversible. OR gate is not reversible because given an output of $|1\rangle$ from OR, it cannot be determined if the input was $|01\rangle$, $|10\rangle$, $|11\rangle$. So the input cannot be determined from the output of the OR gate and hence OR is not a reversible gate.

The significance of reversible gates is that erasing information from classical computers causes energy loss and produces heat whereas quantum computing is reversible and does not erase information. Hence quantum computers do not lose energy.

Some significant quantum logic gates are:

3.4.1   Controlled-NOT Gate

Controlled-NOT gate has two inputs and two outputs. As showed in the circuit below, the top input called a control bit determines what the output will be. The bottom output of $|y\rangle$ will be the same as input if $|x\rangle = |0\rangle$. The output will be opposite if $|x\rangle = 1$.

$$
\begin{array}{c}
—|x\rangle——\bullet——|x\rangle—— \\
—|y\rangle——\oplus——|x \oplus y\rangle—
\end{array}
$$
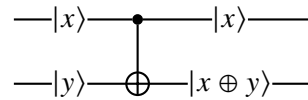
From the circuit, the controlled-NOT gate takes $|x, y\rangle$ to $|x, x \oplus y\rangle$. The $\oplus$ notation indicates the binary exclusive OR operation. The matrix representing the controlled-NOT gate is

$$
\begin{array}{cccc}
 & 00 & 01 & 10 & 11 \\
00 & \begin{pmatrix} 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 1 & 0 \\ 11 & 0 & 0 & 0 & 1 \end{pmatrix}
\end{array}
$$

3.4.2   Toffoli Gate

Toffoli gate has two control bits. It is similar to controlled-NOT gate. The bottom bit flips only if both of the top bits are in state $|1\rangle$. Toffoli gate is also its own inverse. The logic circuit for toffoli gate is given below

$$
\begin{array}{ccc}
\text{---}|x\rangle\text{---}\bullet\text{---}|x\rangle\text{---} \\
\text{---}|y\rangle\text{---}\bullet\text{---}|y\rangle\text{---} \\
\text{---}|z\rangle\text{---}\oplus\text{---}|z \oplus (x \wedge y)\rangle\text{---}
\end{array}
$$
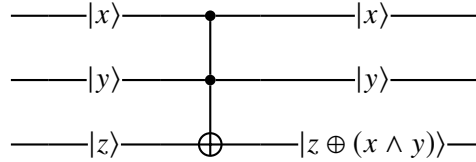
3.4.3   Fredkin Gate

Fredkin gate is another interesting gate. Fredkin gate has three inputs and three outputs which is similar to Toffoli gate. But it has only one control input. If $|x\rangle, |y\rangle$ and $|z\rangle$ are inputs then, $|x\prime\rangle = |x\rangle$. Now, if $|s\rangle = |(y \oplus z) \wedge x\rangle$. Then, $|y\prime\rangle = |y \oplus s\rangle$ and $|z\prime\rangle = |z \oplus s\rangle$.

In brief, $|0, y, z\rangle \longmapsto |0, y, z\rangle$ and $|1, y, z\rangle \longmapsto |1, z, y\rangle$. Likewise Toffoli gate, Fredkin gate is also its own inverse. The logic circuit for Fredkin gate is depicted below.

$$
\begin{array}{ccc}
\text{---}|x\rangle\text{---}\bullet\text{---}|x\rangle\text{---} \\
\text{---}|y\rangle\text{---}\times\text{---}|y\prime\rangle\text{---} \\
\text{---}|z\rangle\text{---}\times\text{---}|z\prime\rangle\text{---}
\end{array}
$$
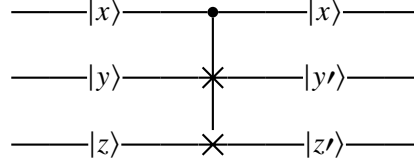
3.4.4   Pauli Gates

There are three Pauli matrices of 2×2 dimension, which are also hermitian and unitary. The matrices are- $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$ and $z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$.

There are also other quantum gates such as **square root of NOT** and is represented by $\sqrt{NOT} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$. But $\sqrt{NOT}$ gate is not its own inverse.

3.5   Representative Quantum Algorithms

3.5.1   Quantum Fourier Transform

The Quantum Fourier Transform which is the quantum analogue of the Discrete Fourier Transform is at the core of numerous quantum algorithms. Discrete Fourier Transform is a square invertible matrix D of dimension N whereas QFT is defined by mapping each computational basis state $|i\rangle$ to a new quantum state $|f_i\rangle = \sqrt{\frac{1}{N}} \sum_{k=0}^{N-1} exp(j\frac{2\pi ik}{N}) |k\rangle$ [16].

3.5.2   Grover's Algorithm

Grover's Algorithm enables to find a special record in a randomly ordered database of N records [17]. While a classical algorithm surely takes $O(N)$ steps to find a specific record in that database, Grover's algorithm $O(\sqrt{N})$ steps to find it.

3.5.3   Shor's Algorithm

Any integer number can be uniquely represented by the product of primes. But finding prime factors in classical computing is not an easy task. In fact, it is one of the hardest problems and so the security of our online transactions are based on the fact that, factoring an integer with thousands of digits is noting but impossible. This belief has been challenged by Peter Shor [15] when he proposed a new quantum method for prime factoring. His solution took $O((n^3 \log n))$ time whereas the most efficient known algorithm take $O(exp(\sqrt[3]{\frac{64}{9}}n(\log n)^2))$ [18] time. This figure 3.3 will show how dramatic the change has been made using the quantum computer.
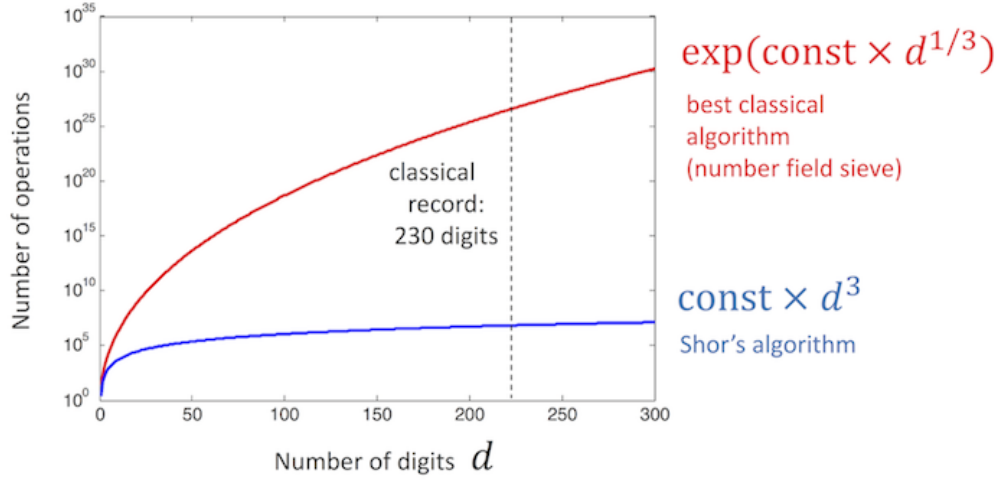
Figure 3.3: Classical vs quantum factoring algorithms [2].

3.6 Quantum Machine Learning Algorithms

3.6.1 The Harrow-Hassidim-Lloyd (HHL) Algorithm

In this section, we discuss about the Harrow-Hassidim-Lloyd (HHL) algorithm, a quantum algorithm for solving the system of linear equations [19]. In this paper, the authors first described why it is necessary to find a quantum solution for solving the system of linear equations efficiently. It is because the classical algorithm takes polynomial time that requires at least N steps to solve the N linear equations. Then the authors described the problem and divided it into sections. The problem is to find $\bar{x}$ from the system $A\bar{x} = \bar{b}$ where A is an $n \times n$ real Hermitian matrix and $\bar{b}$ is a vector. An $n \times n$ matrix is said to be a Hermitian matrix if $A^{\dagger} = (\overline{A})^{T} = \overline{A^{T}} = A$ i.e. $A[j,k] = \overline{A[k,j]}$ [1]. Classically n^2 steps would be necessary for analyzing all the entries of A and n steps would be needed even to write out the solution. HHL assures to solve the system of n equations with complexity $O(\log N)$. The pseudo-code for the algorithm is reproduced in Algorithm 1 from [16].

Algorithm 1 HHL Algorithm

Input: State vector $|b\rangle$, matrix A with oracle access to its elements. Parameters $t_0 = O(\kappa/\epsilon)$, $T = \widetilde{O}(\log(N)s^2 t_0)$, ϵ is desired precision.

$\Lambda_{HHL}(|b\rangle, A, t_0, T, \epsilon)\{$

1. Prepare the input state $|\Psi_0\rangle^C \otimes |b\rangle^I$, where $|\Psi_0\rangle = \sqrt{\frac{2}{T}} \sum_{\tau=0}^{T-1} \sin \frac{\pi(\tau+\frac{1}{2})}{T} |\tau\rangle^C$.

2. Apply the conditional Hamiltonian evolution $\sum_{\tau=0}^{T-1} \langle \tau|\tau\rangle^C \otimes e^{\frac{iA\tau t_0}{T}}$ to the input.

3. Apply the quantum Fourier transform to the register C, denoting the new basis states $|k\rangle$, for $k \in 0, ...T-1$. Define $\widetilde{\lambda} = 2\pi k/t_0$.

4. Append an ancilla register, S, and apply a controlled rotation on S with C as the control, mapping states $\left|\widetilde{\lambda}\right\rangle \longmapsto \left|h(\widetilde{\lambda})\right\rangle$, with $\left|h(\widetilde{\lambda})\right\rangle^a$.

5. Uncompute garbage in the register C.

6. Measure the register S.

7. **if**(result = 'well') **return** register I
   else goto step 1.

$\}$

   Perform $O(\kappa)$ rounds of amplitude amplification on $\Lambda_{HHL}(|b\rangle, A, t_0, T, \epsilon)$.
   Output: State $|\widetilde{x}\rangle$ such that $|| \, |\widetilde{x}\rangle - |x\rangle \, ||_2 \leq \epsilon$.

$^a \left|h(\widetilde{\lambda})\right\rangle$ is a state which comes after the controlled rotation of register S described in [16] in details

First, the matrix A needs to be invertible so that both sides of $A\bar{x} = \bar{b}$ can be multiplied by A^{-1}. The authors mentioned about a crucial factor κ defining the ratio between $A's$ largest and smallest eigenvalues called condition number of A. If this condition number grows, the solution becomes less static as A becomes less invertible. In this situation the authors named the matrix to be "ill-conditioned". But if the complexity of κ becomes poly($\log(n)$), the algorithm achieves an exponential speedup.

The hermitian matrix A needs to be transformed into a unitary operator e^{iAt} for different values of t. For this A needs to be s sparse i.e. s non-zero entries per row so that these entries can be computed in time complexity $O(s)$. Under these circumstances, e^{iAt} can be simulated in time $O(\log(N)s^2 t)$ according to [20]. But if A is not hermitian, a reduction is needed to be applied.

Then b needs to be prepared. The HHL algorithm interprets b as a quantum state $|b\rangle = \sum_{j=1}^{n} b_i |i\rangle$. The authors in s [19] mentioned that the HHL could be a function in a larger quantum algorithm

in which $|b\rangle$ would be produced by some other parts. In the next step, $|b\rangle$ is disintegrated using phase estimation [21–23] into the eigenvector basis. After this, the conditional Hamiltonian simulation [20, 24] is used to apply the unitary operator to $|b\rangle$ for different t.

Coles et al. in [25] noted three sets of qubits named a single ancilla qubit, an n qubit register and a memory of $O(\log(n))$. In quantum world, a qubit can be in one state and in another state simultaneously i.e. a qubit can represent true and false or state $|0\rangle$ and $|1\rangle$ at the same time [1]. The authors in [25] divided the HHL algorithm into three parts: (a) phase estimation using unitary transformation, mapping the eigenvalues into the register in binary form with precision up to n bits, (b) rotation of the ancilla qubits for each eigenvalue, performed through controlled rotation on the ancilla qubit , which yields

$$\sum_{j=1}^{n} \beta_j \left(\sqrt{1 - \frac{C^2}{(\lambda_j)^2}} \, |0\rangle + \frac{C}{\lambda_j} \, |1\rangle \right) |\lambda_j\rangle_{\text{register}} |u_j\rangle_{\text{memory}} \tag{3.4}$$

[25] where C is a normalizing constant, $|u_j\rangle$ and λ_j are the eigenbasis and eigenvalues of A respectively and $|b\rangle = \sum_{j=1}^{n} \beta_j |u_j\rangle$, (c) by reversing the first step i.e. undoing the phase estimation

$$\sum_{j=1}^{n} \beta_j \left(\sqrt{1 - \frac{C^2}{(\lambda_j)^2}} \, |0\rangle + \frac{C}{\lambda_j} \, |1\rangle \right) |u_j\rangle \tag{3.5}$$

[19] is obtained, which further gives

$$|x\rangle \approx \sum_{j=1}^{n} C \left(\frac{\beta_j}{\lambda_j} \right) |u_j\rangle \tag{3.6}$$

[25] by selecting $|1\rangle$ state.

Coles et al. [25] applied the HHL algorithm on a 2×2 system using 4 qubits : one ancilla qubit, one memory, two register qubits . In their experiment, the eigenvalues of A were $\lambda_1 = 1$, $\lambda_2 = 2$ and the eigenvectors were $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$, $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$ or $|-\rangle$, $|+\rangle$ respectively. They used controlled U rotation with $\theta = \pi$ for λ_1 and $\theta = \pi/3$ for λ_2 and $C = 1$ for Eq. 3.4. For b they used three cases: $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$,

$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$. The authors depicted a comparison between the theoretical values and the results that were found in local simulator. They also showed a comparison between the simulator result and the results from ibmqx4 machine with Z measurement on the equivalent composer circuit.

In [6] the author mentioned four caveats of the algorithms. They are :

1. The vector b is required to be loaded into the memory very fast so that a quantum state $|b\rangle = \sum_{i=1}^{n} b_i |i\rangle$ can be prepared.

2. The quantum computer needs to be eligible for applying unitary transformation efficiently. For this purpose A should be s sparse which is discussed before.

3. Another important factor is the matrix A requires to be robustly invertible i.e. "well-conditioned". To be more exact, κ which is the ratio of $\lambda_{\max}$ and $\lambda_{\min}$. If κ grows exponentially, the amount of time needed by HHL also grows linearly which gets rid of exponential speedup.

4. The output of x is a quantum state $|x\rangle$ of $\log_2(n)$ qubits. Learning the value of any specific entry x_i will, by and large, demand recurring the algorithm roughly n times which can terminate the exponential speedup.

These four caveats are so significant that they were further reevaluated in [11]. Additionally, a few more caveats about HHL algorithm were discussed by the author such as the singular values of A locate between $1/\kappa$ and 1 and this caveat derives from quantum phase estimation to estimate the eigenvalues of Hermitian matrix. According to the author, the eigenvalues should be compressed between 0 and 2π. The author also briefly discussed about the solution of the four caveats mentioned earlier. Some quantum machine learning algorithms related to the HHL algorithm were elaborated in this paper.

3.6.2 Quantum Support Vector Machine

Support Vector Machine (SVM), one of the most popular algorithms in modern machine learning, was introduced by Vapnik in 1992 [26]. It is a supervised machine learning algorithm that classifies vectors in a feature space into one of two binary classes, given reasonably sized

datasets. SVM does not work well on enormously large datasets as it becomes computationally expensive.

Given M data points of $(\vec{x_j}, y_j) : j = 1, 2, ..., M$, SVM classifies a vector into one of the two classes, where $\vec{x}$ is an N-dimensional vector in the feature space and $y_j = +1$ or -1 which is the label of the data [25]. Having a weight vector $\vec{w}$, an input vector $\vec{x}$ and a bias weight b, SVM finds the hyperplane with maximum margin. The hyperplane $\vec{w} + \vec{x} = b$ divides the data points into two classes so that $\vec{w} + \vec{x_j} \geq 1$ and $\vec{w} + \vec{x_j} \leq 1$ when $y_j = +1$ and $y_j = -1$ respectively.

Least Squares SVM (LS-SVM) , a variant of SVM formulated in [27] and also mentioned in [7, 25], estimates the hyperplane for obtaining the procedure of SVM by solving the system of linear equation in 3.7:

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & K + \gamma^{-1}1 \end{bmatrix} \begin{bmatrix} b \\ \vec{a} \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{y} \end{bmatrix} \tag{3.7}$$

.

Here K, the $M \times M$ dimensional matrix , is called the kernel matrix, γ is the tuning parameter and $\vec{a}$ forms the vector $\vec{w}$ where $\vec{w} = \sum_{j=1}^{M} a_j \vec{x_j}$. Different types of kernels are in there but linear kernel ($K_{ij} = \vec{x_i} \cdot \vec{x_j}$) is used in Quantum Support Vector Machine [7]. The classical complexity of least square support vector machine is $O(M^2(M + N))$ and the complexity of Quantum Support Vector Machine is $O \log(NM)$. The pseudo-code of quantum support vector machine for solving linear equations is reproduced in 2 from [25].

Algorithm 2 Quantum Support Vector Machine Algorithm

Input:

- Training data set $(\vec{x_j}, y_j) : j = 1, 2, ..., M$.

- A query data $\vec{x}$.

Output: Classification of $\vec{x}$: $+1$ or -1
Procedure:

1. Calculate kernel matrix $K_{ij} = \vec{x_i} \cdot \vec{x_j}$ using quantum inner product algorithm [28].

2. Solve linear equation 3.7 and find $\left| b.\vec{\alpha} \right\rangle$ using a quantum algorithm for solving linear equations [29] (training step).

3. Perform classification of the query data $\vec{x}$ against the training results $\left| b, \vec{\alpha} \right\rangle$ using a quantum algorithm [7].

Chapter 4

Foundation Development

4.1 Literature Review

As integrating machine learning with quantum computing is comparatively a new concept, we are going to present some of the previous works on this field all together. It will help readers to have an overview of the recent works done by researchers on this field. We will also focus on the researches on the basic quantum computing too. We have reviewed the following papers and blogs.

- An introduction to quantum machine learning

- Read the fine print

- Preconditioned quantum linear system algorithm

- Making a Neural Network, Quantum

- Quantum convolutional neural networks

- Generative model benchmarks for superconducting qubits

- A generative modeling approach for benchmarking and training shallow quantum circuits

- Quantum algorithm implementations for beginners

- A survey of quantum learning theory

In this paper [30], the authors gave a systematic overview of quantum machine learning. The authors tried to present the methods of quantum machine learning as well as their computational details in an accessible way. They also discussed the future potentials of the quantum machine learning. As this contribution states, the volume of the globally stored data is growing by around 20% every year. So, the necessity of learning algorithms are increasing. The need of interpreting the data based on experience is rising. The authors focus on the process of exploitation of the potentials of quantum computing in order to optimise classical machine learning algorithms. From this paper, we come to know that, the laws of quantum mechanics restrict us to have an access to the information stored in quantum systems and coming up with quantum algorithms that can outperform their classical counterparts are very difficult. However, the authors mentioned that the toolbox of quantum algorithms is by now fairly established and also contains a number of impressive examples that speed up the best known classical methods. Their work also sited other researchers works. The authors referred to some other contributions where other scientists are trying to develop quantum algorithms that can recognise patterns.

The main focus of this contribution is to recognise patterns with quantum machine learning algorithms. Firstly, the authors described the concepts of classical and quantum learning in brief. Then they divided the paper in seven sections and in each section they described a standard method of machine learning. The methods they described are namely k-nearest neighbour methods, support vector machines, k- means clustering, neural networks, decision trees, Bayesian theory and hidden Markov models.

In the very next section the authors took a quick look into the classical and quantum learning theory. They mentioned the fact that the theory of machine learning is an interdisciplinary subject of both artificial intelligence and statistics. In theory of machine learning the term learning is divided into three categories namely supervised learning, unsupervised learning and reinforcement learning. The authors discussed these three types of learning in short. The authors claimed that, whatever learning method is chosen, optimal machine learning algorithms run with minimum resources and have a minimum error rate related to the task. They think that, challenge is to find the optimal parameter and values that can lead us to a good solution. Otherwise, we can come up with schemes that reduce the complexity of the algorithm. The authors pointed out the fact that, quantum machines can help us on this point.

The next section contains the brief story of quantum machine learning. This term refers to the manipulation of quantum systems to process information. The authors claimed that the characteristics of quantum system which includes the superposition of quantum particles can help us in speeding up the computations in terms of complexity. As our previous sections describe, this paper also has the description of qubits and the superposition of the qubits. This paper has shown some of the mathematical concepts of the quantum system. The authors briefly discussed the unitary quantum gates also in this paragraph. Here, they have shown the basic XOR gate working on the quantum state. In quantum machine learning, quantum algorithms are developed to solve typical problems of machine learning using the efficiency of quantum computing.

In the next section, they discussed the quantum versions of machine learning algorithms. They discussed the previously mentioned algorithms in terms of a quantum perspective. There are many technical details discussed in this paper as well as in the following papers which are reviewed here. Here, for k-nearest neighbours methods, support vector machines and k-means clustering, the authors were mainly concerned to find the efficient calculations of classical distances on a potential quantum computer. But they did not do the same for all other algorithms. Two probabilistic methods, Bayesian theory and hidden Markov models, were used to find an analogy in the formalism of open quantum systems. Again, neural networks and decision trees were used to have first explorations of quantum models.

This paper was our main concern for further progression in this quantum machine learning field as this is kind of an introductory paper. This paper gave an overview of existing ideas and approaches to quantum machine learning. The authors stated that a quantum theory of learning is yet impressive. Although while working on quantum machine learning algorithms, only very few contributions actually answer the question of how the main feature of machine learning, the learning process, can actually be simulated in quantum systems. The authors suggested to investigate the different approaches to quantum computing for this purpose. The authors mentioned the contributors [31], who have already done some of the investigations on the ideas in that directions. Last of all, they are hopeful that even there is still a lot of work to do and quantum machine learning remains a very promising emerging field of research with many potential applications and a great theoretical variety.

Here in this [6] paper, the author has discussed the HHL [19] algorithm. We come to know

from this paper that HHL algorithm can solve the system of linear equations. The goal of HHL is to solve the system,

$$Ax = b \tag{4.1}$$

HHL can solve this equation in logarithmic time, that is in $\log(n)$ time. Here, n depicts the number of unknowns or the number of equations in the linear system. Classically, this complexity seems unreal. As, we need at least n^2 of steps to examine the entries of A and more n steps will be needed to print the solution matrix x. The author summarized the whole concept like this, HHL does not exactly solve the system of linear equations in $\log(n)$ time. It prepares the quantum superposition of the form $|x\rangle$. Here, x is the solution vector to the system of linear equation.

The author made a statement that, we can use HHL as a template for other quantum algorithms. Here, in this [32] Clader et al. claimed that we can use HHL to speed up the calculations of electromagnetic scattering cross-sections, for systems involving smooth geometric figures in 3-dimensional space. In the next section in this paper, the author has shown the mathematical calculation of the fact that HHL algorithm can solve the equations of linear system in logarithmic time. In this thesis work we have already implemented the HHL algorithm on a quantum machine. We have discussed that in later section.

After learning about the quantum HHL algorithm we shifted our focus on a blog post, which contains contents about neural networks in quantum machine learning. This article [33] is all about quantum neural network. In this blog post, the author discussed about the process of implementing the quantum neural network. This will create a good opportunity for computer science. In a figure used in the blog post the author has shown that we can store any configuration of 4 neurons in only 2 qubits. This unlocks our access to a huge number of quantum algorithms that can be implemented to boost processing the network. For this purpose, finding the perfect algorithm to advance the performance is granted as the first task by the author.

The author mentioned about the Hopfield Network which can be used as a addressable memory system. Conventional way of using the Hopfield network is to keep picking neurons at random and updating them by considering the connected neurons, along with their weights. Adding the embedding system into qubits makes the quantum HHL algorithm. This algorithm can process the Hopfield Network. This helps to have an advantage in storage capacity. Researchers have

highlighted in particular the application of the Hopfield network within genetics as a recognizer of infectious diseases also. The author shortly described the next tasks of this field. The author mentioned about the eagerness to uncover improvements to the Hopfield network through quantum mechanics. The speed of reading and writing the data in a quantum machine is still needed to be addressed as the author mentioned. Many experimental teams are working on the technological development like chip design and implementing photonic quantum processors.

Next we are going to review this [34] paper, with the title "Quantum Convolutional Neural Networks". Being motivated by the progress in realization of quantum information processors the authors in this paper introduced and analyzed a novel quantum machine learning (QCNN) model which is basically inspired by convolutional neural networks. They claimed that QCNN circuits combine the multi-scale entanglement re-normalization ansatz and quantum error correction to mimic renormalization-group flow. In this contribution, they proposed a model for QCNN circuit. The input to the circuit model is an quantum state ρ_{in}.

Later, experimental considerations have been discussed. This contribution claims that the QCNN architecture they provided can be efficiently implemented on several state of the art experimental platforms. The key ingredients for realizing QCNNs include the efficient preparation of quantum many-body input states, as author stated, the application of two-qubit gates at various length scales, and projective measurements. These capabilities have already been implemented in multiple programmable quantum simulators consisting of $N \geq 50$ qubits, which are actually based on trapped neutral atoms and ions, or superconducting qubits. Authors are expecting that, the limited depth of their circuit which is approximately $log(N)$ may allow its implementation even in the near term experiments with relatively short coherence time.

Next, they discussed about the further scopes for QCNN to work with. As this contribution only presented the QCNN circuit structure for recognizing 1D phases, which is straightforward to generalize the model to higher dimensions. They are hoping that, studying QCNNs in two and higher dimensions could potentially help identify nonlocal order parameters for lesser-understood phases.

Let us move onto another paper [35]. Here, in this paper, the authors discussed about three performance measures for Quantum Machine Learning. They are: Kullbeck-Leiber divergence which compares the target and output distribution , F1 score which measures how well a circuit

each state and qBAS22 which is a modified F1 score. Using these metrics the authors measured how well each circuit can learn the target distribution in the absence of noise. They used different different entangling layers (1-2) with different CNOT gates (2-4) on noiseless qubits and showed the minimum divergence and maximum qBAS scores.

The authors showed that the behaviour of Kullbeck-Leiber divergence is affected with presence of noise. The authors found smallest KL and largest qBAS on IBM Tokyo for entangling layer 2 with 2 CNOT gates. Then the authors showed how QCBM (Quantum Circuit Born Machine) circuit performs on noisy qubits with parameters which are optimized with noisy qubits. Here noise means qubit decoherence, gate infidelity and measurement errors. The authors mentioned that goal of classical machine learning in general is to avoid over-fitting and to tune parameters properly where QML relies on limiting number of training steps and rotational parameters. They discussed about the effect of noise and N_{shots} (sampling) in details. At last they talked about future works for improving noise-resilience of circuit training and other areas for error mitigation for better estimation.

To clarify the topic of quantum machine learning we tried to cover more and more research works. As this topic is an advanced topic we spent more time on reading research papers. Now, we are going to present another research work [36], which discusses the details on NISQ machines. As hybrid quantum-classical algorithms provides a way to use the noisy intermediate-scale(NISQ) quantum computers for practical purposes, the authors tried to expand the portfolio of these techniques and proposed a quantum circuit learning algorithm that can be used to assist the characterization of quantum devices and to train shallow circuits for generative tasks. The authors' approach can learn an optimal preparation for famous Greenberger-Horne-Zeilinger states which is also known as "cat states". In quantum information theory, a Greenberger-Horne-Zeilinger state is a special type of quantum entangled state which involves at least three particles.

In this contribution, the authors claimed that their proposed method can efficiently prepare approximate representation of coherent thermal states, wave functions that encode Boltzmann probabilities in their amplitudes. The authors designed a hybrid quantum-classical framework called data-driven quantum circuit learning (DDQCL). The authors simulated quantum circuits using the QuTiP2 Python library and implemented the constraints dictated by the ion trap experimental setting. They worked with three synthetic data sets: zerotemperature ferromagnet, random

thermal, and bars and stripes (BAS). In all of their numerical experiments the authors used 1000 data points sampled exactly from these distributions.

This time we put focus on the implementation of some quantum algorithms. This paper [25] helped us having the knowledge of implementing the algorithms. Quantum computing uses some quantum-mechanical phenomenons, in particular, superposition, entanglement and quantum tunneling. This article describes the principles of quantum programming with straight forward algebra that makes understanding under laying quantum mechanics. The authors have explained 20 different quantum algorithms. Those algorithms are namely Groverâs Algorithm, Bernstein-Vazirani Algorithm, Shorâs Algorithm, Quantum Minimal Spanning Tree, Quantum Maximum Flow Analysis, Quantum Approximate Optimization Algorithm, Quantum Principal Component Analysis, Quantum Support Vector Machine, Quantum Simulation of the Schrodinger Equation, Quantum Simulation of the Transverse Ising Model, Quantum Partition Function, Quantum Tomography and some more. They have shown, how to implement those algorithms on IBM's quantum computers. They have also discussed about the results of those implementations with respect to differences of the simulator and the actual hardware runs. The authors also experimented with simplest quantum finite automata simulation to study whether quantum error correction (QEC) can improve computation accuracy. They have discovered that, implementation of the QEC does not improve the probability to interpret the final outcome correctly.

In this paper [25] the authors have shown the mathematical implementation of those algorithms too. On a certain point they introduced the readers with the basic gates of the quantum system like Hadamard Gate (or Hadamard Matrix) [37]. The authors classified the quantum algorithms in several groups like number-theory-based, oracle-based, and quantum simulation algorithms.

Now we are going to review this survey of quantum learning theory by Srinivasan Arunachalam and Ronald de Wolf [38]. In this paper the authors survey quantum learning theory and the theoretical aspects of machine learning using quantum computers. They describe the main results known for three models of learning namely exact learning from membership queries, and Probably Approximately Correct (PAC) and agnostic learning from classical or quantum examples. At the beginning part, the authors described the history of Quantum Computing briefly, how this field of knowledge developed and how the Shorâs efficient quantum algorithms for factoring integers and computing discrete logarithms gave momentum to this field.

This contribution focuses on the theoretical aspects of quantum machine learning. They briefly discussed the introduction to quantum information system. Before jumping into learning algorithms they discussed about the basic quantum system including the Qubits, Bra-ket notations, multi-qubit states, quantum states, Hadamard Transformation etc. Then they moved into the query model analysis. After this the authors explored some details about Grover's algorithm and Fourier Sampling.

In the next section, the authors jumped into the Learning models. Here they described the learning models they used in details. They used Exact Learning, PAC Learning and Agnostic Learning. They described the quantum model of those three learning as well as the classical model of them. As the PAC concerns a nice formalism for deciding how much data we need to collect in order for a given classifier to achieve a given probability of correct predictions on a given fraction of future test data.

After describing all of the learning methods they started discussing the results of learning models. Besides, the gave a briefing on the complexity for each learning algorithm in terms of time complexity and memory complexity. Next they discussed the learning ability of quantum states.

In our contribution, we have tried to implement some of those papers' methods and have presented some of the experiments and their results described in the next sections.

4.2 Weekly Homeworks

In this section, all the homeworks that were done as a part of learning are given. The mathematical problem statements are from the book "Quantum Computing for Computer Scientists" [1]. The answers will be provided if asked for.
The following mathematical problems include the very basic concepts that are necessary for understanding quantum computing. There are concepts about complex numbers, complex vector space, difference between classical and quantum computing, some basic quantum theories, architecture etc.

4.2.1 Homework 1

1. Let $r_1 = 2$, $r_2 = 3$ and $V = \begin{bmatrix} 2 \\ -4 \\ 1 \end{bmatrix}$. Verify Property (vi), i.e. calculate $r_1 \cdot (r_2 \cdot V)$ and $(r_1 \times r_2) \cdot V$ and show that they coincide.

2. Let $c_1 = 2i$, $c_2 = 1 + 2i$ and $A = \begin{bmatrix} 1 - i & 3 \\ 2 + 2i & 4 + i \end{bmatrix}$. Verify properties (vi) and $(viii)$ in showing $\mathbb{C}^{2\times 2}$ is a complex vector space.

3. Find the transpose, conjugate, and adjoint of $\begin{bmatrix} 6 - 3i & 2 + 12i & -19i \\ 0 & 5 + 2.1i & 17 \\ 1 & 2 + 5i & 3 - 4.5i \end{bmatrix}$.

4. Show that all real matrices of the form $\begin{bmatrix} x & y \\ -y & x \end{bmatrix}$ comprise a real subspace of R2$\mathbb{R}^{2\times 2}$. Then show that this subspace is isomorphic to $\mathbb{C}$ via the map $f : \mathbb{C} \to \mathbb{R}^{2\times 2}$ that is defined as $f(x + iy) = \begin{bmatrix} x & y \\ -y & x \end{bmatrix}$.

5. Use the python 3.x numpy package to solve the third problem again.

4.2.2 Homework 2

1. Show that the set of vectors $\left\{ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ -4 \\ -4 \end{bmatrix} \right\}$ is not linearly independent.

2. Verify that the preceding three vectors are in fact a basis of $\mathbb{R}^3$.

3. Show that H times itself gives you the identity matrix.

4. Let $V_1 = [2, 1, 3]^T$, $V_2 = [6, 2, 4]^T$ and $V_3 = [0, -1, 2]^T$. Show that the inner product in $\mathbb{R}^3$ respects the addition.

5. Calculate the norm of $[4 + 3i, 6 - 4i, 12 - 7i, 13i]^T$.

6. Write a one page summary of [39].

7. Write a half page summary of this article [34].

4.2.3 Homework 3

1. The following vectors $\left\{ \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} \right\}$ are eigenvectors of the matrix $\begin{bmatrix} 1 & -3 & 3 \\ 3 & -5 & 3 \\ 6 & -6 & 4 \end{bmatrix}$. Find the eigenvalues.

2. Show that the matrix $\begin{bmatrix} 7 & 6+5i \\ 6-5i & -3 \end{bmatrix}$ is hermitian.

3. Prove the same proposition for symmetric real matrices.

4. Prove that the eigenvalues of a symmetric matrix are real.

5. Show that the matrix $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$ is unitary.

6. Show that if U is a unitary matrix and V_1 and V_2 are in $\mathbb{C}^n$, then $d(UV_1, UV_2) = d(V_1, V_2)$ i.e., U preserves distances. (An operator that preserves distances is called an isometry.)

7. Show that I_n and $-1 \cdot I_n$ are both hermitian and unitary.

8. Calculate the tensor product $\begin{bmatrix} 3 \\ 4 \\ 7 \end{bmatrix} \otimes \begin{bmatrix} -1 \\ 2 \end{bmatrix}$.

9. Calculate $\begin{bmatrix} 3+2i & 5-i & 2i \\ 0 & 12 & 6-3i \\ 2 & 4+4i & 9+3i \end{bmatrix} \otimes \begin{bmatrix} 1 & 3+4i & 5-7i \\ 10+2i & 6 & 2+5i \\ 0 & 1 & 2+9i \end{bmatrix}$.

10. Prove that $(A \otimes B)^\dagger = (A^\dagger \otimes B^\dagger)$.

11. Write up to one page review of [35].

12. Write up to one page review of [36].

4.2.4 Homework 4

1. Open an account with IBM Quantum Experience and run the codes in this [40] python notebook from your laptop.

2. Let us assume that the particle is confined to $x_0, x_1, ..., x_5$ and the current state vector is $|\psi\rangle = [2 - i, 2i, 1 - i, 1, -2i, 2]^T$. What is the likelihood of finding the particle at position x_3?

3. Do the vectors $[1 + i, 2i]^T$ and $[2 + 2i, 12i]^T$ represent the same state?

4. Let the spinning electronâs current state be $|\psi\rangle = 3i\langle\uparrow\rangle - 2|\downarrow\rangle$. Find the probability that it will be detected in the up state.

5. Calculate the bra corresponding to the ket $|\psi\rangle = [3 + i, -2i]^T$.

6. Calculate the amplitude of the transition from $\frac{\sqrt{2}}{2}[i, -1]^T$ to $\frac{\sqrt{2}}{2}[1, -i]^T$.

7. Consider a particle in initial spin up. Apply S_x to it and determine the probability that the resulting state is still spin up.

8. Let $\Omega_1 = \begin{bmatrix} 1 & -i \\ i & 1 \end{bmatrix}$ and $\Omega_2 = \begin{bmatrix} 2 & 0 \\ 0 & 4 \end{bmatrix}$. Verify that both are hermitian. Do they commute with respect to multiplication?

9. Explicitly calculate the commutator of the operators of Example 4.2.2.

10. Repeat the steps of the previous example where $|\psi\rangle = \left| \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2} \right\rangle$

11. Write a one page summary of [25].

4.2.5 Homework 5

1. Run the codes in the python notebook from your laptop [41].

2. Normalize $V = \begin{bmatrix} 15 - 3.4i \\ 2.1 - 16i \end{bmatrix}$.

3. Express the three bits 101 or $|1\rangle \otimes |0\rangle \otimes |1\rangle \in \mathbb{C}^1 \otimes \mathbb{C}^2 \otimes \mathbb{C}^3$ as a vector in $(\mathbb{C}^2)^{\otimes 3} = \mathbb{C}^8$. Do the same for 011 and 111.

4. What vector corresponds to the state $3\,|01\rangle + 2\,|11\rangle$?

5. Calculate $AND\,|10\rangle$.

6. Show that this matrix performs the OR operation.

7. Find a matrix that corresponds to NOR.

8. Write the matrix that would correspond to a one-bit adder. A one-bit adder adds the bits x, y, and c (a carry-bit from an earlier adder) and outputs the bits z and c' (a carry-bit for the next adder). There are three inputs and two outputs, so the matrix will be of dimension $2^2 - by - 2^3$.

9. In previous exercise, you determined the matrix that corresponds to a one-bit adder. Check that your results are correct by writing the circuit in terms of classical gates and then converting the circuit to a big matrix.

10. Write a one page summary of [38].

4.2.6 Homework 6

1. Run the codes in this tutorial [42] from your laptop.

2. Run the codes in this tutorial [43] from your laptop.

3. Create a jupyter notebook and implement this project [44].

4. Normalize $V = \begin{bmatrix} 15 - 3.4i \\ 2.1 - 16i \end{bmatrix}$.

5. The NOT gate has no controlling bit, the controlled-NOT gate has one controlling bit, and the Toffoli gate has two controlling bits. Can we go on with this?

6. Show that each of these matrices are unitary.

7. These operations are intimately related to each other. Prove the following relationships between the operations:

   • $X^2 = Y^2 = Z^2 = I,$

- $H = \frac{1}{\sqrt{2}}(X + Z)$,

- $X = HZH$,

- $Z = HXH$,

- $-1Y = HYH$,

- $S = T^2$,

- $-1Y = XYX$.

8. Consider a qubit whose θ is equal to $\frac{\pi}{4}$. Change it to $\frac{\pi}{3}$ and picture the result. Then compute its likelihood of collapsing to the south pole after being observed.

9. Do a similar analysis for the Toffoli gate. Show that the way we set the Toffoli gate to perform the fanout operation does not clone a superposition of states.

10. Write a one page summary of [45].

4.2.7 Homework 7

1. Read this paper [16] and summarize the improvements of the HHL algorithms in three pages.

2. Describe the matrices for the other three functions from $\{0, 1\}$ to $\{0, 1\}$.

3. What is the adjoint of the matrix given in Equation (6.9)? Show that this matrix is its own inverse.

4. Give the unitary matrices that correspond to the other three functions from $\{0, 1\}$ to $\{0, 1\}$. Show that each of the matrices is its own adjoint and hence all are reversible and unitary.

5. Using the matrices calculated in previous exercise, determine the state $|\phi_2\rangle$ for the other three functions.

6. For each of the other three functions from the set $\{0, 1\}$ to the set $\{0, 1\}$, describe what $|\phi_2\rangle$ would be.

7. For each of the other three functions from the set $\{0, 1\}$ to the set $\{0, 1\}$, calculate the value of $|\phi_3\rangle$.

4.2.8 Homework 8

1. Create a LATEX file for the thesis with dummy table of contents.

2. Write a program which reads a system of linear equations from a file and outputs the pdf of the quantum circuit of the HHL algorithm.

3. Let's consider only 8×8 matrices. So, the sparsity, s, can range from 0 to 64. For each value of s, generate 5 random matrices. Now, compute the number of CNOT gates HHL algorithm would need for each cases. So, for each sparsity s, there will be 5 such counts. Now draw a plot where the Y axis will be the number of CNOT gates and the X axis will be s. For each x, plot the average CNOT gate numbers.

4. Count the condition number kappa for each of the matrices above. Draw another plot, where the X axis will now be κ.

Chapter 5

The IBM Quantum Computing Platform

Nowadays anyone can use a Quantum Computer from anywhere in the world. Many famous companies are providing the opportunities to get the quantum experience online. *IBM* is one of them.

Quantum computers can be constructed with trapped ions. IonQ makes platform for trapped ion quantum computers. Google, IBM and some other companies produces the platform for superconducting quantum computers. Superconducting quantum computing is an implementation of quantum computer. As a quantum computer cannot lose energy and information for the sake of reverse computing, it has to be of zero resistant [46]. The IBM quantum computer is made of superconducting material and consists of superconducting circuits.

For all of our experiments, we have utilized the backends administered by IBM. Some of them are original quantum devices and some of them are classical devices which simulate the experiments as a quantum device will do. So, we provide some details about these backends. This section will be convenient for readers to understand the architecture of the devices on which we have run our experiments.

IBM provides nine backends on cloud for any user on the world. Each of them can be accessed from `https://quantum-computing.ibm.com/`. Table 5.1 demonstrates a concise overview of each backends.

Brief description of the 9 backends are given in the following.

Table 5.1: An Overview of IBM Backends.

Gate Name	Opened Since	No. of Qubits	Device Type	Basic Gates
ibmq_16_melbourne v2.0.2	Nov 06, 2018	15	Real Quantum	u1,u2, u3,cx,id
ibmq_london v1.0.0	Sep 13, 2019	5	Real Quantum	u1,u2, u3,cx,id
ibmq_burlington v1.1.4	Sep 13, 2019	5	Real Quantum	u1,u2, u3,cx,id
ibmq_essex v1.0.1	Sep 13, 2019	5	Real Quantum	u1,u2, u3,cx,id
ibmq_ourense v1.0.1	Jul 03, 2019	5	Real Quantum	u1,u2, u3,cx,id
ibmq_vigo v1.0.2	Jul 03, 2019	5	Real Quantum	u1,u2, u3,cx,id
ibmq_5_yorktown - ibmqx2 v2.0.5	Jan 24, 2017	5	Real Quantum	u1,u2, u3,cx,id
ibmq_armonk v1.1.0	Oct 16, 2019	1	Real Quantum	id,u1,u2, u3
ibmq_qasm_simulator v0.1.547	May 02, 2019	32	Quantum Simulator	u1,u2,u3,cx,cz, id,x,y,z,h,s,sdg, t,tdg, ccx,swap, unitary, initialize,kraus

5.1   ibmq_16_melbourne

This is a real quantum hardware. It has got 15 qubits. It supports the basics gates- id, u1, u2, u3 and cx. Single-qubit U2 error rate is in between $3.644e-4$ and $2.706e-3$. CNOT error rate is in between $1.683e-2$ and $8.174e-2$. Figure 5.1 shows the hardware architecture of this machine using qubits and connections.
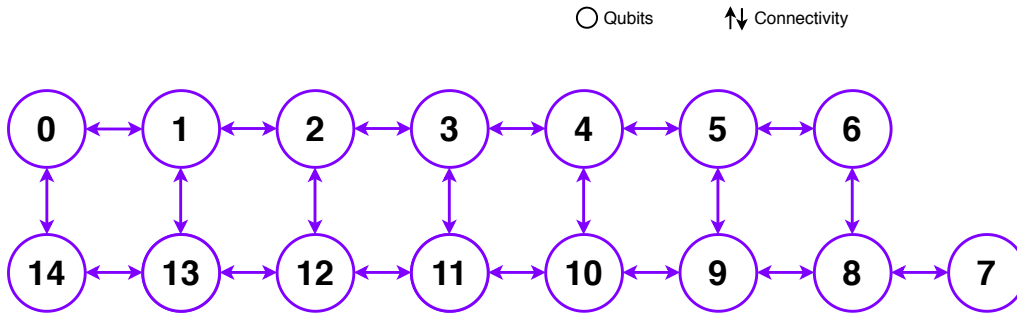

Figure 5.1: Hardware Architecture of ibmq_16_melbourne.

5.2   ibmq_london

This is again a real quantum hardware. It has taken only 5 qubits. It reinforces the basics gates- u1, u2, u3, cx and id. Single-qubit U2 error rate is in between $2.864e-4$ and $5.819e-4$.
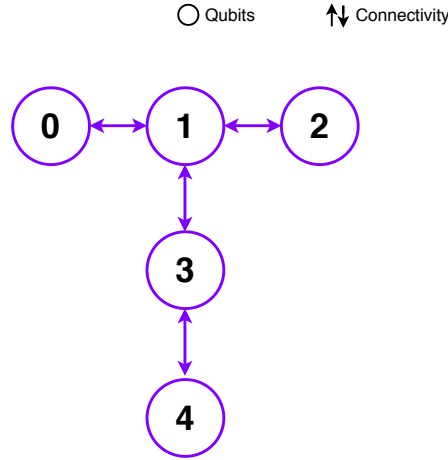
Figure 5.2: Hardware Architecture of ibmq_london, ibmq_burlington, ibmq_essex, ibmq_ourense, ibmq_vigo. They look all the same but their error spectrum is different.

CNOT error estimate is in between $9.056e - 3$ and $3.028e - 2$. The architecture design is depicted in figure 5.2.

5.3 ibmq_burlington

This real quantum hardware holds 5 qubits. It supports the basics gates- u1, u2, u3, cx and id. Single-qubit U2 error rate is in between $4.028e - 4$ and $7.133e - 4$. CNOT error percentage is in between $1.081e - 2$ and $2.210e - 2$. The design of its architecture is as same as 5.2 with a different error rate.

5.4 ibmq_essex

This real quantum hardware consists of 5 qubits. It reinforces the basics gates- u1, u2, u3, cx and id. Single-qubit U2 error rate is in between $3.405e - 4$ and $8.875e - 4$. CNOT error score is in between $9.416e - 3$ and $1.533e - 2$. Figure 5.2 is the design of this machine but with a different error rate.
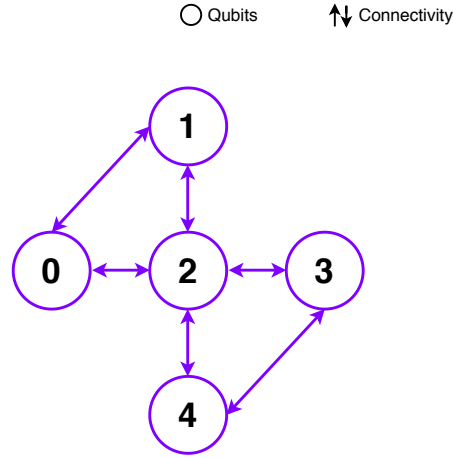
Figure 5.3: Hardware Architecture of ibmq_5_yorktown - ibmqx2.

5.5 ibmq_ourense

This is still a real quantum hardware. It has got only 5 qubits. It supports the basics gates- u1, u2, u3, cx and id. Single-qubit U2 error rate is in between $1.988e - 4$ and $8.408e - 4$. CNOT error rate is in between $6.319e - 3$ and $1.766e - 2$. Figure 5.2 is also the design of ibmq_ourense.

5.6 ibmq_vigo

This real quantum hardware consists of 5 qubits. It supports the basics gates- u1, u2, u3, cx and id. Single-qubit U2 error rate is in between $3.660e - 4$ and $1.062e - 3$. CNOT error rate is in between $7.586e - 3$ and $1.414e - 2$. The architechture of ibmq_vigo is also same as four other machines shown in 5.2 and with a different level of spectrum.

5.7 ibmq_5_yorktown - ibmqx2

This is also a real quantum hardware. It has got only 5 qubits. It supports the basics gates- u1, u2, u3, cx and id. Single-qubit U2 error rate is in between $4.802e - 4$ and $5.927e - 4$. CNOT error rate is in between $1.365e - 2$ and $1.736e - 2$. Figure 5.3 shows the architecture of the hardware.
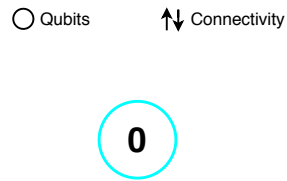
Figure 5.4: Hardware Architecture of ibmq_armonk.

5.8 ibmq_armonk

This is a real quantum hardware which has a single qubit. The design of this machine is shown in figure 5.4. It supports the basics gates- id, u1, u2 and u3. Single-qubit U2 error rate is in between $6.985e - 4$ and $6.985e - 4$. CNOT error rate is in between $1.000e + 0$ and $-1.000e + 0$.

5.9 ibmq_qasm_simulator

This device is a quantum simulator. It can simulate 32 qubits. It supports the basic gates- u1, u2, u3, cx, cz, id, x, y, z, h, s, sdg, t, tdg, ccx, swap, unitary, initialize and kraus.

Chapter 6

Design of the experiments

In this chapter, the procedure of our experiments is described. For this contribution we used IBM's quantum composer, publicly accessible at `https://quantum-computing.ibm.com/composer` to do some experiments. This composer gives us the opportunity to write programs on quantum machines.

To run those experiments we need to use Qiskit [47] which is an open-source framework. It is basically used for quantum computing. It gives apparatuses for making and controlling quantum projects and running them on model quantum gadgets and test systems. It pursues the circuit model for widespread quantum calculation and can be utilized for any quantum equipment that pursues this model.

We have used Jupyter Notebook as our IDE. Anyone can use the online editor that IBM provides. We have also run our codes in the Google's colaboratory for some cases.

6.1 The Harrow-Hassidim-Lloyd Algorithm

Our goal of this contribution is to find out how the HHL algorithm performs on various matrices which we have found is not done earlier by anyone. To conduct this experiment, we used the platform, Google's Colaboratory and Python to be our programming language. There are a few libraries available on Python programming language which help conduct quantum experiments. Qiskit is a python library which is used for IBM's Quantum Computer backend and available online. All of our codes for this experiment will be found at this github repository. The Qiskit

official repository [48] has been of great assistance to us.

We have planned to observe the characteristics of the HHL algorithm from two different perspectives. One perspective is to show how the size of matrices affects the performance of the HHL algorithm. To do so, we have taken three matrices of $2 \times 2, 4 \times 4$ and 8×8 dimensions. Both diagonal and non-diagonal matrices have been experimented. We have also considered the sparsity of a matrix as a parameter to measure this algorithm. We aimed to understand how HHL works when the sparsity of a matrix increases or decreases.

It is ensured that the matrix A is a hermitian matrix. Later, we have checked if the matrix is diagonal or not. Our assumption was this algorithm can deal better with the system with diagonal matrices than with non-diagonal matrices. We have tested different matrices with different sparsity. Again we have taken three $2 \times 2, 4 \times 4$ and 8×8 matrices with sparsity 0.5 and 1. The readers should be introduced to some particular metrics which are crucial to understand this experiment.

6.1.1 Probability

Due to the probabilistic nature of the quantum computers, they cannot provide with the exact results. They run the same experiments for some number of times and show the probability of acquiring a particular result. This means they produce the probability of a result being true. We phrase the number of experiments as "shots" and the term "probability" refers to the highest probability of a result to be true.

6.1.2 Fidelity

"Fidelity" was first induced by Richard Jozsa [49]. Fidelity illustrates how much we can rely on the result. Using Qiskit we can achieve the result of the classical computer as well as the quantum computer simulation. Comparing the quantum computerâs simulation with classical ones can produce the fidelity value. This value depicts the perfection of the proceeding. The more accurate result the quantum simulation provides, the closer the fidelity value becomes to 1. Otherwise, the fidelity value declines to 0. We can represent the results as quantum states and measure fidelity of those quantum states. State fidelity is the measure of how close two quantum states are to each other. If we simplify this, we can say that fidelity is the distance between two quantum states. It can be stated as the cosine of the smallest angle between two quantum states and also established

as cosine similarity. This article [50] proves that fidelity can be used as a standard parameter to distinguish two quantum states or to measure the similarity between two states. This metric has turned into one of the most universally used quantities to quantify the degree of similarity between quantum states. So we regard fidelity to be a decisive parameter in our experiments too.

6.1.3 Sparsity

The "sparsity" of a matrix refers to the density of the generated matrix. Sparsity equal to 1 means a full matrix with no element equals to 0. Sparsity 0 means a matrix with no non-zero items i.e. each entry of the matrix will be zero. We assumed that the lesser the sparsity of the matrix is, the better the HHL performs. That means with the lower sparsity matrix HHL performs better. The later section presents the comparisons necessary for this argument.

6.1.4 Circuit

Different circuits combining different types of gates are necessary for solving distinct problems using quantum machines. The circuit information consists of information about the depth and the width of the circuit. The width of a circuit also means the number of qubits used in a circuit. Circuit depth refers to the longest path required from the input to the output. Having a smaller quantum circuit requires less computational power. Figure 6.1 depicts the circuit description necessary for solving a $2 * 2$ matrix using the HHL algorithm. We will show a comparative study based on the depth and width of the related quantum circuits in the next chapter.

Figure 6.1: Circuit for Solving HHL Algortihm on a $2 * 2$ Matrix.

6.1.5   Random Sparse Hermitian Matrix

A matrix added to its hermitian is also a hermitian matrix [1]. So ascertaining our final matrix B is hermitian, where A is an arbitrarily generated sparse matrix, we used this equation

$$B = A + A^{\dagger} \tag{6.1}$$

This makes the final matrix B a hermitian matrix. We generated the random sparse matrix by using the SciPy library of Python.

6.2   Quantum Support Vector Machine

6.2.1   Dataset

In this part of our experiment, we have taken two beginner's datasets used for machine learning purposes. These are Iris dataset and the Breast Cancer dataset those can be easily accessed by the machine learning package Scikit-learn and are publicly available to use from `https://scikit-learn.org/stable/datasets/index.html`.

6.2.1.1   Breast Cancer Dataset

The Breast Cancer dataset [13] is for the binary classification system. The target classes of this dataset are- Malignant and Benign. There are 569 data-points in this dataset and each of them have 30 features. We have separated the dataset into train and test set. The train set is kept small as we could not use so many data-points on the quantum support vector machine algorithm. We used 500 data-points for test set and only 69 points for training.

6.2.1.2   Iris Dataset

The Iris dataset [14] is for a multivariate classification problem. Iris dataset contains only 150 data points and each of them has 4 features. We have used 80% of our data as the training data. Figure 6.3 depicts the features of the Iris dataset.
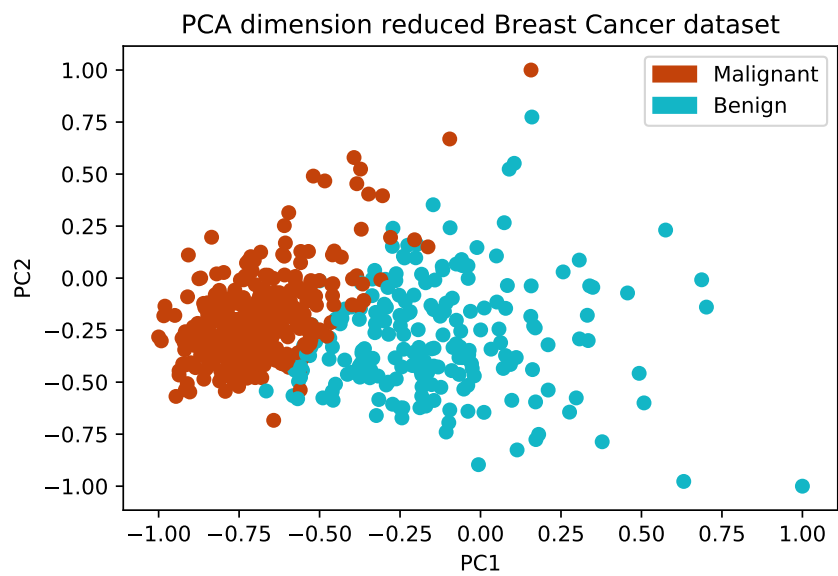
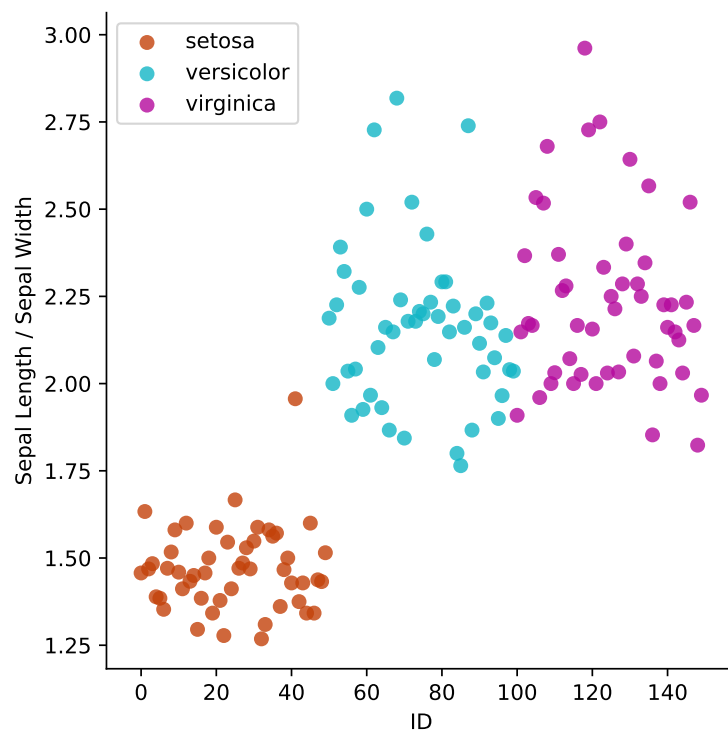Figure 6.2: Visualization of the Breast Cancer dataset.

Figure 6.3: Visualization of the features of the Iris dataset.

6.2.2 Preprocessing

The datasets are preprocessed as they contain too many features and are not perfectly adjusted. For both the dataset we have applied Principal Component Analysis to diminish the dimensionality of the datasets. As more dimensionality induces more processing power, we have taken the decision to stand with the smaller number of features. For both datasets, we have lowered the dimensionality to 2. Again, we have standardized the dataset around 0 with unit variance. We have then scaled all features in between -1 to $+1$ so that the computation becomes simpler.

6.3   Implementation

6.3.1   Support Vector Machine

For both datasets, Support Vector Machine has been used as the operating engine for the classification problem. Support Vector Machine is used as a linear classifier that divides the hyperplane with maximum possible distance. We have selected linear kernel for SVM and $C = 1$.

6.3.2   Quantum Support Vector Machine

The Quantum part of this experiment on both datasets has been conducted in the QASM Simulator with 32 qubits and depth = 2. The shot numbers are $1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024$. For feature mapping, second order expansion provided by Qiskit Aqua library is used.

6.4   Performance Evaluation

For performance evaluation we have used four different metrics or parameters. They are accuracy, sensitivity, specificity and F1 score. The definitions of these metrics are given in the

Table 6.1: Confusion Matrix for Binary Classification Problem.

		Predictions	
		Positive	Negative
Real	Positive	True Positive	False Negative
Labels	Negative	False Positive	True Negative

following:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.2}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{6.3}$$

$$Specificity = \frac{TN}{TN + FP} \tag{6.4}$$

$$F1 - score = \frac{2TP}{2TP + FP + FN} \tag{6.5}$$

where TP, TN, FP, FN refers to true positive, true negative, false positive and false negative respectively. These values come from a confusion matrix described in table 6.1.

Chapter 7

Results and Discussion

7.1   The Harrow-Hassidim-Lloyd Algorithm

This section illustrates the results of the conducted experiments. As the preceding chapter articulates, our approach is to figure out the algorithm based on two distinct norms which we introduced earlier.

Our initial point of concern has been whether a matrix is diagonal or not. We have assumed that a diagonal matrix may be simpler to analyze than a non-diagonal matrix. Statistics that we have found from our experiments reinforce our hypothesis. After experimenting we have discovered that the fidelity of matrices of various dimensions does not vary on the matrix being diagonal or not. This means it can solve the system of the linear equation almost flawlessly. But the actual difference comes to the sight when we become concern about the probability and the circuit information. Probability dramatically falls if a matrix is non diagonal. Figure 7.1 explains the probability comparison. The dropped probability depicts that additional unnecessary solutions have been generated by the quantum machine and therefore the probability of the actual solution is declined. So, it leads to the wastage of computational power.

The circuit information also shows the evidence of misuse of computing power. As the size of the matrix enhances, the size of the circuit increases as well. We somewhat expect this increment. But surprisingly, the circuit size also expands when the HHL algorithm deals with the non-diagonal matrices.

Figure 7.2 exhibits the circuit's width which means the number of qubits both for diagonal and
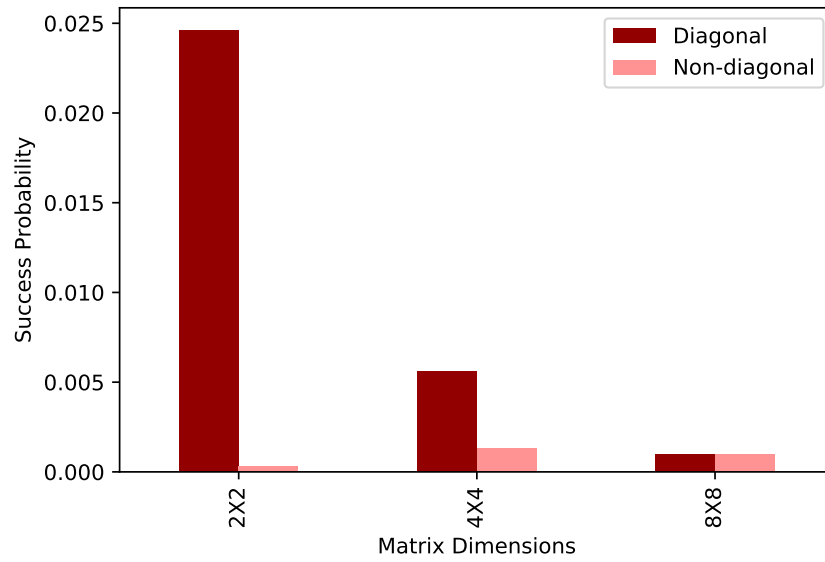
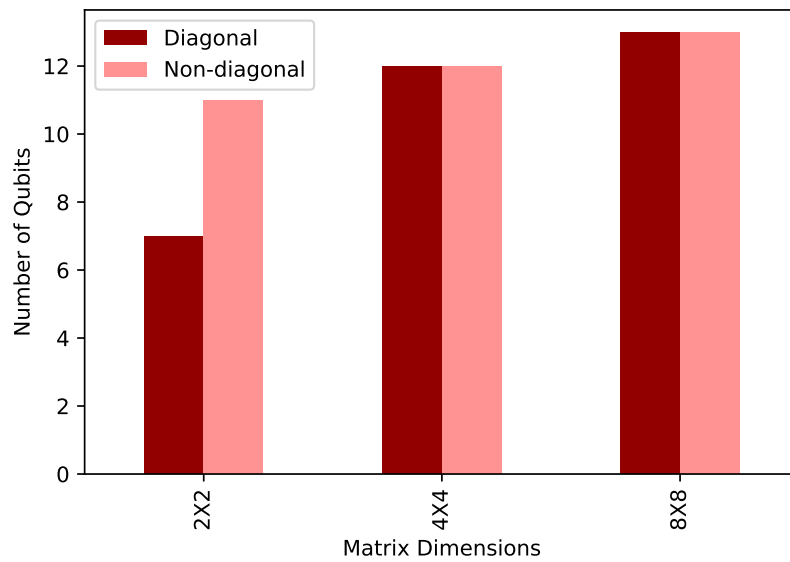Figure 7.1: Success probabilities of diagonal and non-diagonal matrices.

Figure 7.2: Number of qubits needed for diagonal and non-diagonal matrices.
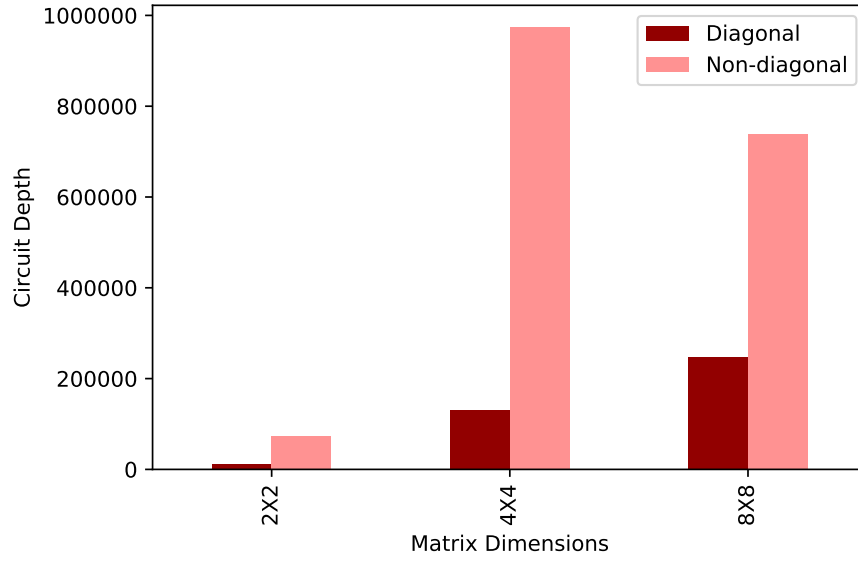
Figure 7.3: Quantum gate depth comparison of diagonal and non-diagonal matrices.

non-diagonal matrices. For 4×4 and 8×8 matrices the width of the circuit does not increase. But when we plot the depth of the circuit, the contrast becomes clear. Figure 7.3 shows the difference.

From this discussion, it assures our conjecture that the system of the linear equation with non-diagonal matrices take much higher computational power and considerable effort to be solved. Quantum machines require more circuit width and depth to execute the computation. Also, it provokes some invalid outputs to diminish the probability of actual output even though the fidelity remains approximately constant. Table 7.1 shows the accurate comparison between diagonal and non-diagonal matrices.

Up to this point, the result is satisfactory in spite of taking extended computing capability because it does not influence the fidelity. The output is approximately identical as the classical output. The table 7.1 demonstrates that the fidelity is relatively uninterrupted which is over 99% in every instance. But the fidelity dramatically drops when the experiment is based on arbitrary sparse hermitian matrix. When we observe the dimensions of the matrices, we can see that the fidelity shrinks as the dimensions increase. We can solve the matrix of size 2×2 with fidelity 0.422602. But it falls down to 0.000011 when the size becomes 4×4. This fidelity tends to zero which implies we cannot rely on this outcome and so is of no benefit.

The figure 7.4 illustrates how the fidelity is damaged when we want to solve the system of the

Table 7.1: Overall Comparison between Diagonal and Non-diagonal Matrices.

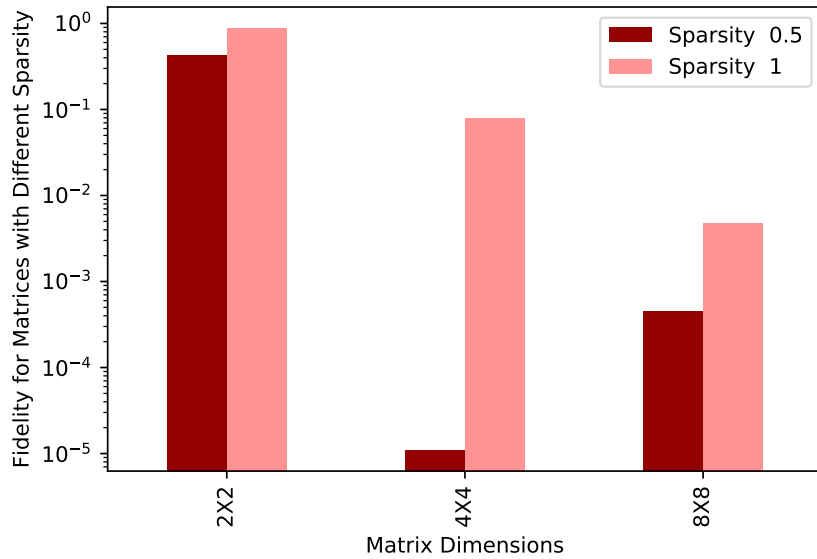| Dimension of the Matrix	Parameters	Diagonal Matrix	Non-diagonal Matrix
2 x 2	Probability	0.024630	0.000316
	Fidelity	0.999389	0.999996
	Circuit Width	7	11
	Circuit Depth	12256	73313
4 x 4	Probability	0.005583	0.001285
	Fidelity	1.0	0.999994
	Circuit Width	12	12
	Circuit Depth	130723	973521
8 x 8	Probability	0.000982	0.001002
	Fidelity	0.999992	0.999997
	Circuit Width	13	13
	Circuit Depth	248342	738340

Figure 7.4: Diminishing Fidelity with Random Sparse Hermitian Matrix.

Table 7.2: Overall Comparison between Sparsity 0.5 and Sparsity 1 matrices.

Dimension of the Matrix	Parameters	Sparsity = 0.5	Sparsity = 1
2 x 2	Probability	0.099122	0.028234
	Fidelity	0.422602	0.883097
	Circuit Width	7	7
	Circuit Depth	30254	30254
4 x 4	Probability	0.135214	0.146572
	Fidelity	0.000011	0.077870
	Circuit Width	8	8
	Circuit Depth	165262	165262
8 x 8	Probability	0.087196	0.152713
	Fidelity	0.000453	0.004737
	Circuit Width	9	9
	Circuit Depth	801281	801281

linear equation with random sparse hermitian matrix with various densities. This figure shows the fidelity value in logarithmic scale.

From table 7.2 we see that the circuit size deepens as the matrix grows bigger. We anticipated this to a small extent. But here, we can observe that the sparsity has no effect on the circuit size. With both of the sparsity 0.5 and sparsity 1, the circuit size remains the same.

There is another feature the HHL algorithm shows. The fidelity of the solution diminishes as the sparsity of the matrix increases. To investigate this characteristic, we have chosen matrices of $0.5, 0.6, 0.7, 0.8, 0.9$ and 1.0 sparsity. To make our result more acceptable, we have taken 3 random sparse matrix for each of these sparsities and solved the system of linear equation. The figure 7.5 demonstrates how fidelity decreases with density. This causes this algorithm more susceptible.

After all the experiments are done, it is time to sum up all the results. As this [51] contribution summarizes, HHL algorithm is for linear systems of equations when the input $n \times n$ matrix is s-sparse and well-conditioned. And most notably, the solutions are not exact as we only obtain an approximate solution for this problem. From our experiments we have identified that this HHL
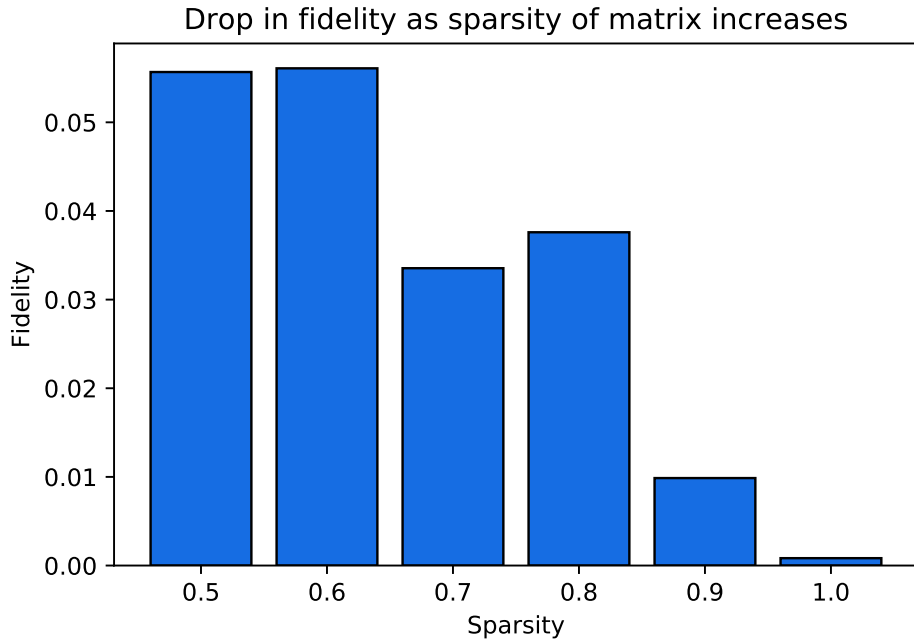
Figure 7.5: Drop in Fidelity with respect to Sparsity.

algorithm bears some further limitations too. One of the most remarkable disadvantage of this algorithm is it requires more computation capability when a matrix from a system of the linear equation is non-diagonal.

The most significant disadvantage of HHL algorithm is it cannot solve the random sparse hermitian matrix. What it produces as result for random sparse hermitian matrix is not accurate at all. Though we hypothesized that this algorithm may perform better with the matrices with lower sparsity, we have now come to find that it indeed does not work on random sparse hermitian matrix. It produces faulty outputs when the experiments are carried out with random sparse hermitian matrices. The positive side is that it does not consume extra computational power to solve the matrices with higher sparsity as we have found the circuit size has remained the same.

In spite of being a remarkably powerful algorithm in the field of quantum computing, the HHL algorithm possesses some limitations. In this contribution, we have investigated to find out some of these limitations. Our aim was to experience the behavior of the HHl algorithm when it encounters some unusual situations. This contribution presents the urgency of tuning the HHL algorithm up and design an updated or new algorithm to cope with these critical cases.

7.2 Quantum Support Vector Machine

The performance of the quantum computer has not been up to the mark. The classical interpretation of the Support Vector Machine algorithm has outperformed the quantum one. But despite possessing plenty of obstacles, the quantum variant of the support vector machine has not performed bad. In this section, a comparative investigation between the classical and quantum variant of the SVM algorithm on two distinctive datasets mentioned in the previous section is displayed. At first, the performance comparison is depicted when we have conducted the experiment on the quantum computer just for once. Afterwards the information about how the performance steps up with the quantity of experiments increases is shown.

7.2.1 Classical vs Quantum SVM

On both of the datasets, the classical algorithm performed better. The figure 7.6 illustrates the difference. For the Breast Cancer dataet, the Quantum SVM gives us only 55% accuracy after we take simply one shot. The classical subpart, for this dataset, presents a remarkable 92% of accuracy. Again, on the Iris Dataset, QSVM gives a worse output. It can achieve only 23% of accuracy after one shot. Certainly, this is not a progress. It is far away from the state-of-the-art performance. This result is some sort of random predictions. The classical one can produce a stunning 98% accuracy without any kind of kernel tricks. So, there are many opportunities for upgrading the performance.

7.2.2 QSVM with Shots Increased

The number of shots is the total of conducted experiments on a quantum system. We have observed that conducting the experiments several times on the quantum machine provides us a stronger performance for both of the datasets. Though for a specific period it declined for both of the datasets, but overall performance is increased. The dilemma is that taking more shots can bring about more processing power, which is not anticipated. So, there is always a compensation between the efficiency and the number of shots.

The results of the increased shots have been demonstrated in figure 7.7 and 7.8. Though this accuracy has some ups and downs, the performance has increased noticeably. From this experiment,
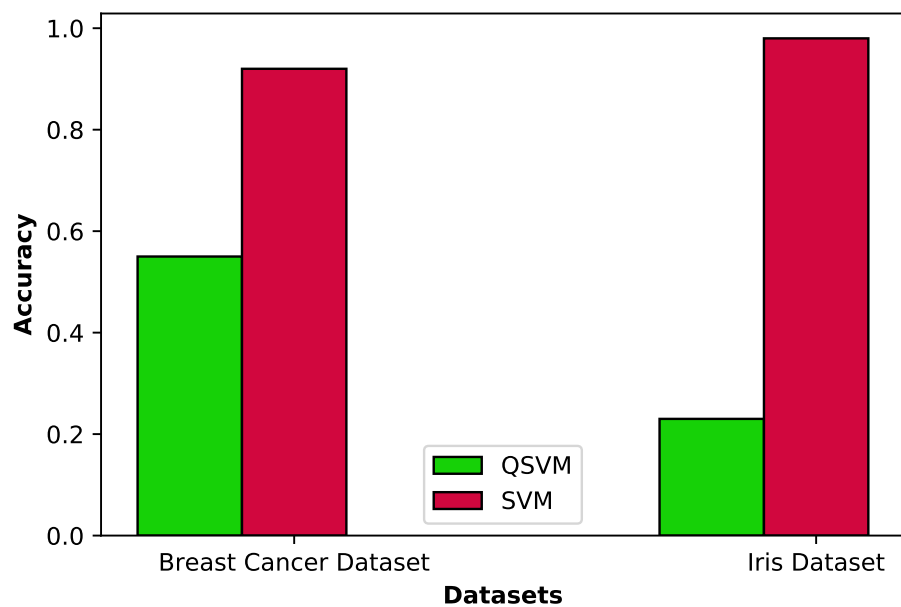
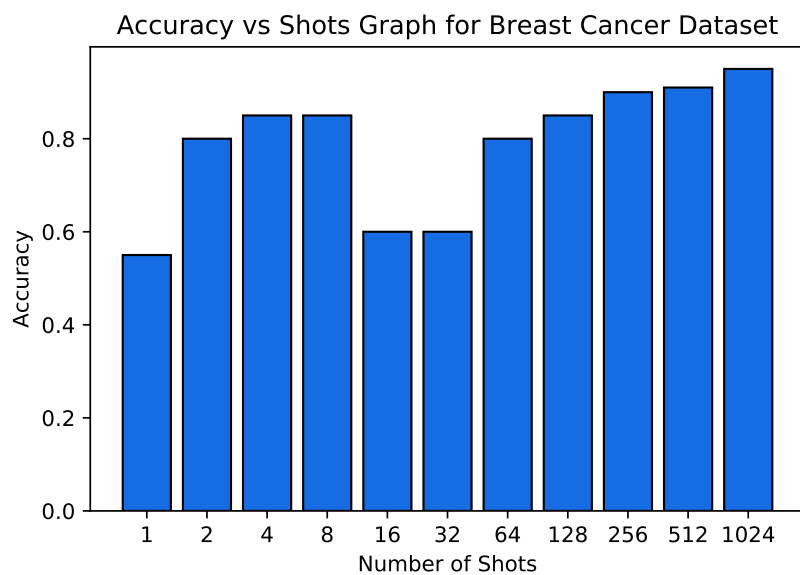Figure 7.6: Accuracy Comparison for Classical and Quantum SVM for both datasets.

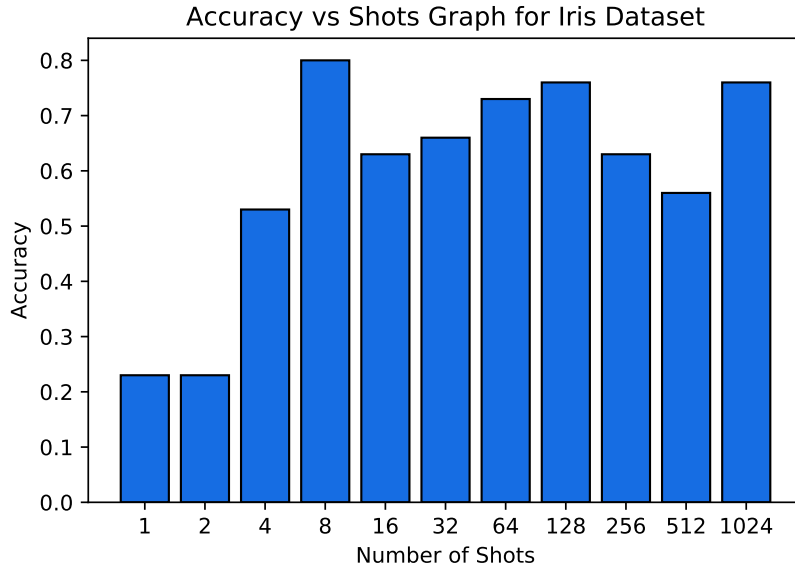Figure 7.7: Accuracy vs shots on Breast Cancer Dataset.

Figure 7.8: Accuracy vs shots on Iris Dataset.

we can come to a settlement that, QSVM shows better performance on binary classification problem than on multi-label classification problem. With sufficient quantity of shots, the performance can be as satisfying as classical computer for binary classification problem. But there are many fields left for multi-label classification where further consideration is demanded.

The confusion matrices shown in 7.9 and 7.10 give the idea of exact results that have been found through the experiments. To analyze our results we have generated 4 confusion matrices for both of the datasets. On both of them, we have applied SVM and QSVM algorithm. To obtain the finest result, the number of shots on which the QSVM performed best are chosen, which are 1024 or the Breast Cancer dataset and 8 for Iris dataset. As from the figure 7.7 and figure 7.8 shows the best number of shots for these two datasets.

Let us go into a bit deep into the confusion matrix for the Breast Cancer dataset. We have taken 198 samples for evaluation purpose. SVM kernel has produced 94.9% of accuracy. So, most of the cases have been recognized as True Positives and True Negatives. 69 samples have been designated as Benign class and 119 samples have been detected as Malignant class. For a total of 188 samples out of 198 has been perfectly classified. Remaining 10 specimens have been labeled incorrectly. On the other hand, the Quantum Support Vector Machine kernel produces 86.9% of accuracy. Total 172 samples have been classified accurately. Among them 80 samples have been

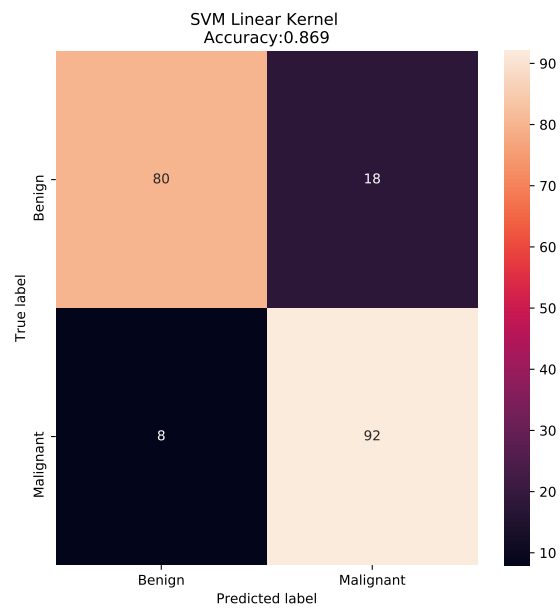chosen as Benign class and remaining 92 have been classified as Malignant class.
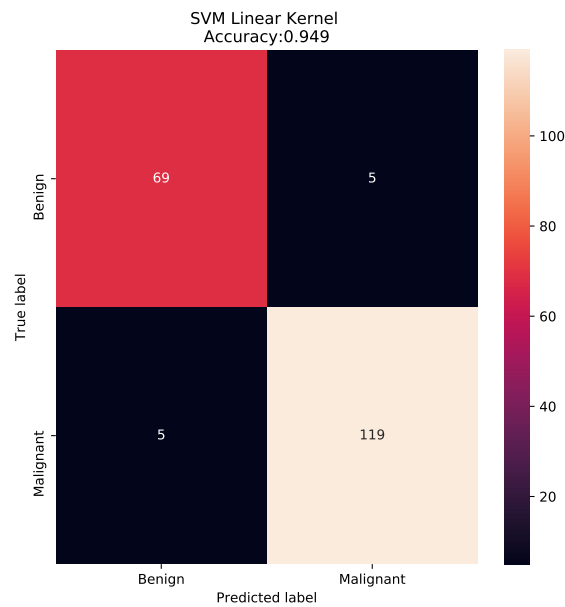
(a) Confusion Matrix of SVM on Breast Cancer Dataset.
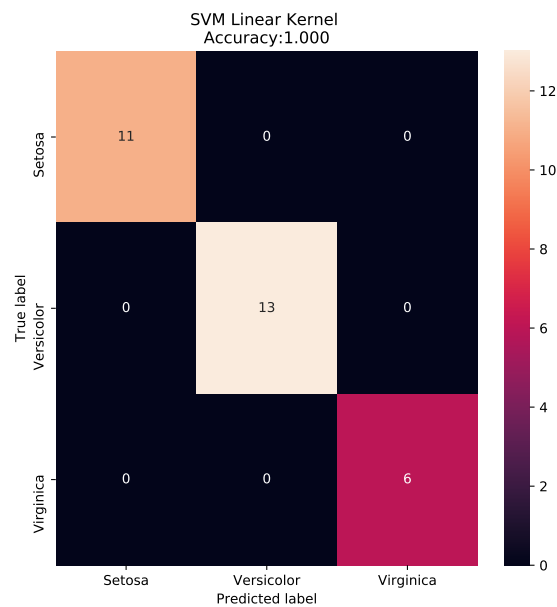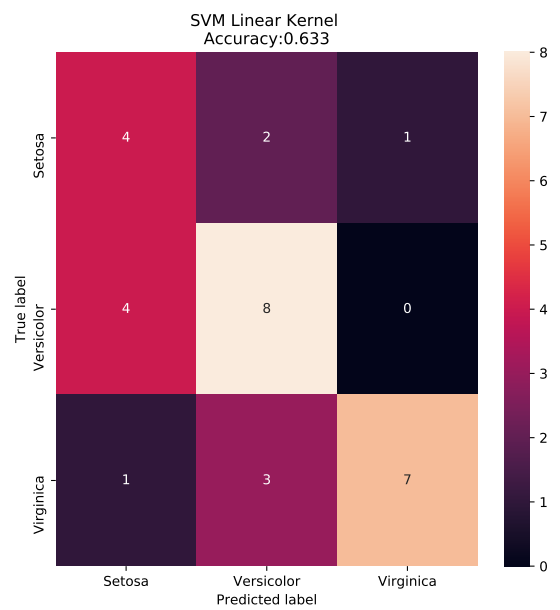

(b) Confusion Matrix of SVM on Iris Dataset.

Figure 7.9: Confusion Matrices from Classical SVM experiments.

(a) Confusion Matrix of QSVM on Breast Cancer Dataset.

(b) Confusion Matrix of QSVM on Iris Dataset.

Figure 7.10: Confusion Matrices from Quantum SVM experiments.

On the Iris dataset, the QSVM kernel has not performed that much satisfactory. This dataset

Table 7.3: Comparison of Performance of Prediction between SVM and QSVM on Two Datasets.

Dataset	Classifier	Name of classes	Accuracy	Sensitivity	Specificity	F1 Score
Breast Cancer	SVM	Benign	0.9495	0.9324	0.9597	0.9324
	QSVM	and Malignant	0.8687	0.8162	0.9224	0.8623
Iris	SVM	Setosa	1.000	1.000	1.000	1.000
		Versicolor		1.000	1.000	1.000
		Virginica		1.000	1.000	1.000
	QSVM	Setosa	0.6333	0.667	0.571	0.640
		Versicolor		0.375	0.666	0.500
		Virginica		0.900	0.636	0.783

includes only 150 samples. This can be an explanation of poor performance. We have taken 30 of them for testing purpose. The SVM kernel has performed adequately on this dataset. SVM can produce 100% of accuracy on these 30 samples. So all of them as classified as True Positives and True Negatives. Again, the QSVM kernel has not performed that much good on this dataset. Total 19 samples out of 30 has been classified precisely. This demonstrates that there are many other opportunities of investigation in multi-label classification problem.

Table 7.3 presents different features of the confusion matrices such as- Accuracy, F1 Score, Sensitivity, Specificity and compares them.

Chapter 8

Future Work

As we pointed out in the earlier chapters, Quantum Machine Learning is a rapid-growing field. We have tried to work with a small part of this gigantic field. This field is still recent. There are lots of opportunities for the researchers in this field. In our investigations, we have tried to explore two of the most promising algorithms in quantum machine learning sector.

We have studied the HHL algorithm in the perspective of a matrix being diagonal or non-diagonal, having different sparsity and of different shapes. There are other parameters for this algorithm too, which can affect the fidelity and the probability of the solutions. We believe that the researchers will concentrate on these issues and try to figure out valuable insights.

QSVM section also has considerable research to do. We have worked with two standard datasets. But real world datasets can be more diverse and challenging to fit in. To create real impact on the world, we need to work with real life datasets. We are looking forward to continue our work too.

We have so far focused on the number of shots to increase the accuracy. But there are other issues which can be focused as well. We used QSVM as our model. Our machine learning and deep learning models are waiting to be implemented and benchmarked on some quantum devices. Moreover, we have implemented the linear kernel of the SVM algorithm. We hope many researchers will also work with other kernel tricks.

In a nutshell, there are so many windows yet to be opened. We hope our work can initiate anyone's journey to the world of quantum computing and help others to find something valuable for humankind.

Chapter 9

Conclusion

In this paper, We have presented two quantum subroutines of machine learning algorithms for the purpose of solving system of linear equations and for supervised classification of data.

The Harrow-Hassidim-Lloyd algorithm is designed to find out the solution for a system of the linear equations on a quantum device. In our contribution, we wanted to investigate how effectively this algorithm performs when the environment is unusual. We tried to study and explain the background of the HHL algorithm in brief before moving forward to further contribution and we have been able to detect some limitations. We have experimented with diagonal and non-diagonal matrices of of 2×2, 4×4 and 8×8 dimensions. Further, we have also analyzed the algorithm with arbitrary sparse hermitian matrices of different densities such as 0.5 and 1. We started our experiment by mentioning two hypotheses. We found one of them to be true. We suspected that the HHL algorithm would perform better on diagonal matrix than on non-diagonal matrix. After experimenting we have found that this algorithm demands more processing power to process the non-diagonal matrices than the diagonal ones. To solve the system with non-diagonal matrices, it has produced deeper and wider circuits and has taken more processing time. We also presumed that this algorithm would perform better on lower density random hermitian matrices than higher density ones but this assumption was not accurate in this case. Because this algorithm failed to perform well in both situations. We have discovered that this algorithm produces invalid results when we provide random hermitian matrices. Those outputs are not trustworthy at all as the differences with the real outputs are immense. There are some positive outcomes too. This algorithm performs almost accurately with diagonal and non-diagonal matrices. Another positive

thing is it does not require bigger circuit to solve higher density matrices. Our article shows the necessity of tuning this HHL algorithm or finding a new algorithm which can survive these critical cases.

As quantum computing is comparatively a modern research field, many noble contributions are yet to be done. Enthusiasts can explore how the HHL algorithm performs when we provide larger matrices and with different densities. We hope that this contribution may open a new window to explore HHL and such algorithms.

Quantum Suppot Vector Machine is also a promising algorithm in the field of Quantum Machine Learning. We have experimented with two datasets named Breast Cancer dataset and Iris Dataset. QSVM with linear kernel and SVM with linear kernel have been run on both of these datasets. We have pre-processed the data, applied Principal Component Analysis on the datasets, normalized them around 0 and scaled them between -1 and $+1$. Though QSVM has not performed as well as SVM performed on both of the datasets, there are so many possibilities to increase the efficiency of this algorithm. Our research has been a small contribution to the vast field of quantum machine learning. We hope that more research work will be conducted on this field and they will produce more valuable knowledge.

References

[1] N. S. Yanofsky and M. A. Mannucci, *Quantum computing for computer scientists*. Cambridge University Press, 2008.

[2] I. Research and the IBM QX team., "Shorâs algorithm" https://quantumexperience. ng.bluemix.net/proxy/tutorial/full-user-guide/004-Quantum_Algorithms/110-Shor's_ algorithm.html, 2017, [Online; accessed 14-July-2019].

[3] S. Hassan, "Quantum Computing â The Future" https://medium.com/predict/ quantum-computing-the-future-992c493ae1a1, 2018, [Online; accessed 12-July-2019].

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

[5] V. Dunjko and H. J. Briegel, "Machine learning\& artificial intelligence in the quantum domain," *arXiv preprint arXiv:1709.02779*, 2017.

[6] S. Aaronson, "Read the fine print," *Nature Physics*, vol. 11, no. 4, p. 291, 2015.

[7] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Physical review letters*, vol. 113, no. 13, p. 130503, 2014.

[8] P. Rebentrost, T. R. Bromley, C. Weedbrook, and S. Lloyd, "Quantum hopfield neural network," *Physical Review A*, vol. 98, no. 4, p. 042308, 2018.

[9] M. Schuld, I. Sinayskiy, and F. Petruccione, "Prediction by linear regression on a quantum computer," *Physical Review A*, vol. 94, no. 2, p. 022342, 2016.

[10] N. Wiebe, D. Braun, and S. Lloyd, "Quantum algorithm for data fitting," *Physical review letters*, vol. 109, no. 5, p. 050505, 2012.

[11] C. Shao, "Reconsider hhl algorithm and its related quantum machine learning algorithms," *arXiv preprint arXiv:1803.01486*, 2018.

[12] A. Y. Kitaev, "Quantum measurements and the abelian stabilizer problem," *arXiv preprint quant-ph/9511026*, 1995.

[13] W. H. Wolberg, W. N. Street, and O. L. Mangasarian, "Breast cancer wisconsin (diagnostic) data set," *UCI Machine Learning Repository [http://archive. ics. uci. edu/ml/]*, 1992.

[14] R. A. Fisher, "Uci machine learning repository: Iris data set," 1936.

[15] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994, pp. 124–134.

[16] D. Dervovic, M. Herbster, P. Mountney, S. Severini, N. Usher, and L. Wossnig, "Quantum linear systems algorithms: a primer," *arXiv preprint arXiv:1802.08227*, 2018.

[17] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[18] C. Pomerance, "A tale of two sieves," *Biscuits of Number Theory*, vol. 85, p. 175, 2008.

[19] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical review letters*, vol. 103, no. 15, p. 150502, 2009.

[20] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, "Efficient quantum algorithms for simulating sparse hamiltonians," *Communications in Mathematical Physics*, vol. 270, no. 2, pp. 359–371, 2007.

[21] A. Luis and J. Peřina, "Optimum phase-shift estimation and the quantum description of the phase difference," *Physical review A*, vol. 54, no. 5, p. 4564, 1996.

[22] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, "Quantum algorithms revisited," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1969, pp. 339–354, 1998.

[23] V. Bužek, R. Derka, and S. Massar, "Optimal quantum clocks," *Physical review letters*, vol. 82, no. 10, p. 2207, 1999.

[24] A. M. Childs, "On the relationship between continuous-and discrete-time quantum walk," *Communications in Mathematical Physics*, vol. 294, no. 2, pp. 581–603, 2010.

[25] P. J. Coles, S. Eidenbenz, S. Pakin, A. Adedoyin, J. Ambrosiano, P. Anisimov, W. Casper, G. Chennupati, C. Coffrin, H. Djidjev *et al.*, "Quantum algorithm implementations for beginners," *arXiv preprint arXiv:1804.03719*, 2018.

[26] S. Marsland, *Machine learning: an algorithmic perspective*. Chapman and Hall/CRC, 2014.

[27] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[28] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum algorithms for supervised and unsupervised machine learning," *arXiv preprint arXiv:1307.0411*, 2013.

[29] V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum random access memory," *Physical review letters*, vol. 100, no. 16, p. 160501, 2008.

[30] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.

[31] A. Bisio, G. Chiribella, G. M. DâAriano, S. Facchini, and P. Perinotti, "Optimal quantum learning of a unitary transformation," *Physical Review A*, vol. 81, no. 3, p. 032324, 2010.

[32] B. D. Clader, B. C. Jacobs, and C. R. Sprouse, "Preconditioned quantum linear system algorithm," *Physical review letters*, vol. 110, no. 25, p. 250504, 2013.

[33] T. Bromley, "Making a Neural Network, Quantum" https://medium.com/xanaduai/making-a-neural-network-quantum-34069e284bcf, 2018, [Online; accessed 12-July-2019].

[34] I. Cong, S. Choi, and M. D. Lukin, "Quantum convolutional neural networks," *arXiv preprint arXiv:1810.03787*, 2018.

[35] K. E. Hamilton, E. F. Dumitrescu, and R. C. Pooser, "Generative model benchmarks for superconducting qubits," *Physical Review A*, vol. 99, no. 6, p. 062323, 2019.

[36] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, and A. Perdomo-Ortiz, "A generative modeling approach for benchmarking and training shallow quantum circuits," *npj Quantum Information*, vol. 5, no. 1, p. 45, 2019.

[37] Wikipedia, "Hadamard Matrix" https://en.wikipedia.org/wiki/Hadamard_matrix, 2019, [Online; accessed 12-July-2019].

[38] S. Arunachalam and R. de Wolf, "Guest column: A survey of quantum learning theory," *ACM SIGACT News*, vol. 48, no. 2, pp. 41–67, 2017.

[39] T. Bromley, "Making a Neural Network, Quantum," https://medium.com/xanaduai/making-a-neural-network-quantum-34069e284bcf, 2018, [Online; accessed 08-April-2019].

[40] Qiskit, "Getting Started with qiskit," https://nbviewer.jupyter.org/github/Qiskit/qiskit-tutorials/blob/master/qiskit/basics/1_getting_started_with_qiskit.ipynb, [Online; accessed 20-May-2019].

[41] ——, "linear systems of equations," https://github.com/Qiskit/qiskit-iqx-tutorials/blob/master/qiskit/aqua/linear_systems_of_equations.ipynb, [Online; accessed 30-May-2019].

[42] M. Schuld, "Building the worldâs smallest quantum classifier," https://medium.com/qiskit/building-the-worlds-smallest-quantum-classifier-7da7cd845b84, 2018, [Online; accessed 25-July-2019].

[43] T. Bassan, "Quantum Machine Learning," https://hackernoon.com/quantum-machine-learning-d0037f59f31a, 2018, [Online; accessed 25-July-2019].

[44] C. P. dos Santos Gonçalves, "Quantum neural machine learning: Theory and experiments," in *Artificial Intelligence-Applications in Medicine and Biology*. IntechOpen, 2019.

[45] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, "Quantum machine learning: a classical perspective," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2209, p. 20170551, 2018.

[46] J. Hui, "QC â How to build a Quantum Computer with Superconducting Circuit?" https://medium.com/@jonathan_hui/ qc-how-to-build-a-quantum-computer-with-superconducting-circuit-4c30b1b296cd, 2019, [Online; accessed 8-February-2020].

[47] Wikipedia, "Qiskit" https://en.wikipedia.org/wiki/Qiskit, 2019, [Online; accessed 13-July-2019].

[48] Qiskit, "Qiskit Aqua: Solving linear systems of equations with the HHL algorithm," https://github.com/Qiskit/qiskit-tutorials/blob/master/qiskit/advanced/aqua/linear_ systems_of_equations.ipynb, 2019, [Online; accessed 19-August-2019].

[49] R. Jozsa, "Fidelity for mixed quantum states," *Journal of modern optics*, vol. 41, no. 12, pp. 2315–2323, 1994.

[50] M. Cerezo, "Quantum Fidelity, or, how to compare quantum states," https://entangledphysics. com/2019/06/24/quantum-fidelity-or-how-to-compare-quantum-states/, 2019, [Online; accessed 19-August-2019].

[51] V. T. Xuan, "Cr04 report: Solving linear equations on a quantum computer," 2016.