Quantum Machine Learning

Undergraduate Thesis

Submitted in partial fulfillment of the requirements of
BITS F421T Thesis

By

Harsh Mavani
ID No. 2014B5A30899G

Under the supervision of:

Dr. Radhika Vathsan

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, GOA CAMPUS
October 2018

Declaration of Authorship

I, Harsh Mavani, declare that this Undergraduate Thesis titled, 'Quantum Machine Learning' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Certificate

This is to certify that the thesis entitled, "*Quantum Machine Learning*" and submitted by Harsh Mavani ID No. 2014B5A30899G in partial fulfillment of the requirements of BITS F421T Thesis embodies the work done by him under my supervision.

Supervisor
Dr. Radhika Vathsan
Associate Professor,
BITS-Pilani Goa Campus
Date:

"Twenty years from now, you will be more disappointed by the things that you didn't do than by the ones you did do. So throw off the bowlines. Sail away from the safe harbor. Catch the trade winds in your sails. Explore. Dream. Discover."

Mark Twain

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, GOA CAMPUS

Abstract

Masters of Science (Hons.) Physics

Quantum Machine Learning

by Harsh Mavani

Machine Learning is changing the technological revolution into a smarter one, impacting millions of lives on a daily basis with its advancements in almost every field. There are reasons to believe that Quantum Computing would help in this progress and reveal something the classical age couldn't. This thesis focuses on the algorithmic development part for the same, and also discusses the current state of the art realizations of Quantum Computers ...

Acknowledgements

This project could not have reached its present state of completion without the support of certain people. I thank Prof. Radhika Vathsan for assigning this project to me and for her continuous guidance and support. I also thank Prof. Gaurav Dar and Prof. Ram Prasad Joshi for evaluating and giving feedback on the progress.

Contents

Declaration of Authorship i

Certificate ii

Abstract iv

Acknowledgements v

Contents vi

Abbreviations viii

1 Introduction 1

2 Classical Machine Learning Algorithms 3
- 2.1 Supervised Learning . . . 3
  - 2.1.1 Classification . . . 4
  - 2.1.2 Regression . . . 4
  - 2.1.3 Support Vector Machine . . . 4
  - 2.1.4 k Nearest Neighbor . . . 5
- 2.2 Unsupervised Learning . . . 5
  - 2.2.1 Principal Component Analysis . . . 5
  - 2.2.2 k means Clustering . . . 6
- 2.3 Neural Networks . . . 6
  - 2.3.1 Feedforward NN . . . 7
  - 2.3.2 Hopfield Network and Boltzmann Machine . . . 7

3 Working with Quantum Computers 9
- 3.1 Working with IBM Quantum Computer . . . 9

4 Basic Quantum Algorithms 11
- 4.1 HHL . . . 11
- 4.2 Grover's Algorithm . . . 11
  - 4.2.1 The Oracle . . . 12
  - 4.2.2 Amplitude Amplification . . . 12

5 Quantum ML Algorithms **14**
5.1 Quantum Boltzmann Machine 14

6 Conclusion **17**

A Code for Grover Search Algorithm **18**

Abbreviations

ML	**M**achine **L**learning
NN	**N**eural **N**etwork
kNN	**k** **N**earest **N**eighbors
PCA	**P**rincipal **C**omponent **A**nalysis
SVM	**S**upport **V**ector **M**achine
RBM	**R**estricted **B**oltzmann **M**achine
Qiskit	**Q**uantum **I**nformantion **S**cience **kit**

Chapter 1

Introduction

Machine Learning is at the heart of current technological advancement. From medical sciences to autonomous vehicles, machine learning (ML) plays an important role at the backend of the prediction algorithms. Diving deeper, one can see ML is all about processing data and finding patterns. Long before the age of computing, founding techniques of ML were already developed. From the early ages, astronomers tried finding patterns in data collected from the motion of the stars, sun and moon to explain the retrograde motion of the planets. This led to the development of mathematical techniques like Laplace method for least square fitting, Newton-Gauss's method for solving linear equations, and finding optima through gradient descent by Newton. The digital revolution further promoted the development of methods to recognize patterns in data. Such methods and algorithms are broadly classified in the domain of machine learning.

The hope that quantum computers would be able to reveal something more than the classical computers in the field of machine learning comes form the following hypothesis: Quantum mechanics can produce data atypical in nature. Classical systems can both produce and identify patterns in data. Now if quantum processors can produce data that is computationally difficult for a classical processor to compute, then quantum systems might be able to recognize patterns which are difficult for classical processors to recognize.

To claim that quantum computers would be able to perform better than classical computers requires either mathematical proofs or physical realization of such a device with a solid statistical evidence of quantum supremacy. Such a potential is termed as quantum speedup. The two measures of quantum speedup are query complexity and gate complexity. Query complexity is

the measure of number of queries made from a classical system to a quantum system. Gate complexity is the measure of number of elementary gates required to obtain the output.

This aim of this thesis is to understand the developments made till now in the field of quantum machine learning. The focus is also on the state of the art quantum computers and ways to use them for QML. The thesis is divided into 4 parts. The first part discusses the algorithms of classical machine learning which would form the base for future discussions on quantum algorithm. The second part discusses the developments in the field of quantum computers and a few demonstrations. The third part discusses the quantum algorithms and important information related to quantum computing which are required for the QML algorithms. The fourth part discusses the quantum machine learning algorithms.

Chapter 2

Classical Machine Learning Algorithms

Machine Learning is a broad topic which consists of different categories of algorithms. Each category has its own area of application, advantages and disadvantages. The categories are: Supervised learning, clustering, unsupervised learning, reinforcement learning and Neural networks. The categories might be overlapping, for instance neural network can come under supervised as well as unsupervised learning algorithms, but the developments in that area are massive to be considered as a separate topic itself. The topics are briefly discussed below:

2.1 Supervised Learning

Supervised learning is a class learning algorithms where the training data is given as input with its corresponding output. In such a scenario, the training algorithm learns from the training data set and tries to correctly predict the output on the test data set. The output could be a simple yes or no classification or it could be predicting price for a certain commodity at a specified time. Based on the type of output the problem is termed as classification or regression problem.

2.1.1 Classification

To take an example, consider an email spam classifier. For a certain set of input mails the user classifies whether the email is spam or not. Hence the training algorithm is fed emails as input and spam or not spam as corresponding output. After some training, the algorithm is used to predict and classify spam emails. Such problems are termed as classification problem.

2.1.2 Regression

Take another example where the price of a house is given based on the its square feet area. Here the area is considered as input and price as output. After few training examples, the algorithm is used to predict the price of a house based on the given input area. Such problems where the output takes a range of values rather than fixed value from the output set, is termed as a regression problem.

2.1.3 Support Vector Machine

One way to perform classification is through using a technique called Support Vector Machine (SVM). Examples are represented as points in n-dimensional space where n represents the number of features. SVM tries to divide the output groups by a hyper-plane of (p-1) dimension (where p is the dimension of the data points) and classify the new example in one of the groups. The hyper-plane is chosen in such a way such that the distance between the margin and the two classes is maximum. If a hyper-plane exists where the distance between the nearest point of the two classes and the margin is maximum, it is termed as a maximum margin hyper-plane.

Few problems faced with a SVM is choosing the dimension of the hyper-plane. Although it can be any dimensional value, its preferable to not make it too high because it then becomes computationally expensive. Another problem with high dimensional value is of over-fitting. In this case the hyper-plane tries to accommodate all the examples belonging to a particular class despite the fact that few examples could be erroneous. This leaves no room for error correction while training and hence it becomes sensitive to error. On the other hand if the dimension of the plane is chosen very small then one might face the problem of under-fitting. In this case the classifier cannot classify the data properly because the hyper-plane cannot separate the two classes properly.

2.1.4 k Nearest Neighbor

k nearest neighbor (kNN) is another type of classification algorithm. In this type of setting, there is no as such training required by the algorithm. All the training examples with their labels are mapped in an n dimensional vector space. When a new unlabelled test example is to be classified, the algorithm considers the nearest k examples, calculates the majority and classifies accordingly. A variation of this algorithm to improve the accuracy is to normalize the example by the distance from the test input. Hence the training input farther away from the test input gets less weight-age and the one nearer get more. The only tricky part of this algorithm is to correctly choose the value of k. If the value is very less, the algorithm becomes prone to error and noise where as if the value is very large, the classification accuracy decreases because of low variance.

2.2 Unsupervised Learning

Unsupervised learning algorithms are a set of algorithms for which the data is provided in the form of only input and no labelling is done. This is one step closer to true AI. The algorithm looks at the data and tries to find patterns or groups on its own. Unlike supervised learning, there is nothing to tell the algorithm whether the output is correct or not. This kind of learning is generally used when the data is cheap to obtain and the output is very expensive or is not available. An example of unsupervised learning would be to detect anomalous instances in a time series.

2.2.1 Principal Component Analysis

It is very crucial to be able to visualize the data in order to understand what it means. At times the data consists of many feature dimension and only the important features needs to be taken into account. Principal Component Analysis (PCA) is used in such scenarios. Lets consider that the data is provided in the form of vectors v_j which is a d dimensional vector such that $d = 2^n = N$. The covariance matrix of the data is $C = \sum_j v_j v_j^T$. The covariance matrix represents the correlations between the different components of the data. The PCA starts by diagonalizing the covariance matrix: $C = \sum_k e_k c_k c_k^\dagger$ where c_k are the eigenvectors of C and e_k are the corresponding eigenvalues. The eigenvectors for which the corresponding eigenvalues

are large are know as the principal components of C. Each principal component represents an underlying common trend or form of correlation in the data, and decomposing a data vector v in terms of principal components, $v = \sum_k v_k c_k$ allows to compress the data and also predict the future behaviour.

2.2.2 k means Clustering

One way to achieve unsupervised learning is through k means clustering algorithm. The algorithm starts off with k randomly placed points in the current example input space. All the points from the input are mapped to one of the points which is closest amongst the k points. After all the points are mapped, each point from the k points set is moved to the mean position of all the input points mapped to it. This procedure is repeated till the position of the k points do not change significantly. One way to decide the value of number of clusters is to iterate from 1 till the point when the value of cost function does not change much. A cost function calculates the total absolute difference in the distance of a point from its assigned cluster mean value. The optimum number of clusters (k value) is the one which is at the elbow point when the cost function is plotted against the total number of clusters.

2.3 Neural Networks

Neural Networks are a class of ML algorithms which are inspired by the functioning of brain. Just like brain, the most elementary structure of a NN is a neuron. A neuron can take multiple inputs, and outputs a single value. Each input is multiplied with a weight value and fed to the activation function which then calculates the output value.

$z = \sum w.x + b$

$a = f(z)$

where, b is the bias, a is the activation value and f is the activation function

The activation function is chosen such that the output lies between 0 and 1. There are two types of commonly used activation function. The first one is just a threshold function. Based on the input the threshold is selected such that, below that value the neuron output 0 and above that

it outputs 1. Although this does not allow smooth learning of the weights and bias parameters. The second and most frequently used activation function is the sigmoid function.

$\sigma(z) = 1/(1 + \exp(-z))$

For z tending to $-\infty$ the activation function outputs value 0 and for z tending to $+\infty$ the activation function outputs value 1.

In a typical NN setting, there are multiple neurons in a layer and there are multiple such layers. Based on the type of connections each neuron has with other neurons, the NN can be classified into different categories. Some of them are mentioned below.

2.3.1 Feedforward NN

This is the simplest NN setting where all the neurons from one layer are connected to all the neurons in the next layer without having interconnections in the same layer.

2.3.2 Hopfield Network and Boltzmann Machine

A hopfield network is a form of recurrent NN where each node is connected to all the other nodes except itself, i.e. it does not form a loop. This means that there are $K(K-1)$ interconnections, where K is the number of nodes. Each neuron only takes value -1 or 1. One of the assumptions of Hopfield network is that the weights interconnecting the nodes are symmetric, $w_{ij} = w_{ji}$. The activation of a neuron s_i follows the following rule,

$$s_i = \begin{cases} +1 & \text{if } \sum_j w_{ij}.s_j \geq \theta_i \\ \text{-1} & \text{otherwise} \end{cases}$$

A hopfield network is generally associated with a scalar value known as its energy. The energy function resembles the Ising model which makes it easier to exploit by some models of quantum computing.

$E = -\sum_{i,j} w_{ij} s_i s_j - \sum_i s_i$

In a hopfield network all the neurons are visible. But if few neurons were to be made hidden and the update rule to be stochastic, that would make it a Boltzmann Machine. Further, if the hidden neurons were allowed to be connected only with visible neurons and vice-versa, that would

make it a Restricted Boltzmann Machine (RBM). The reason for introducing the constraints for RBM is that training the Boltzmann machine is very computationally expensive.

Chapter 3

Working with Quantum Computers

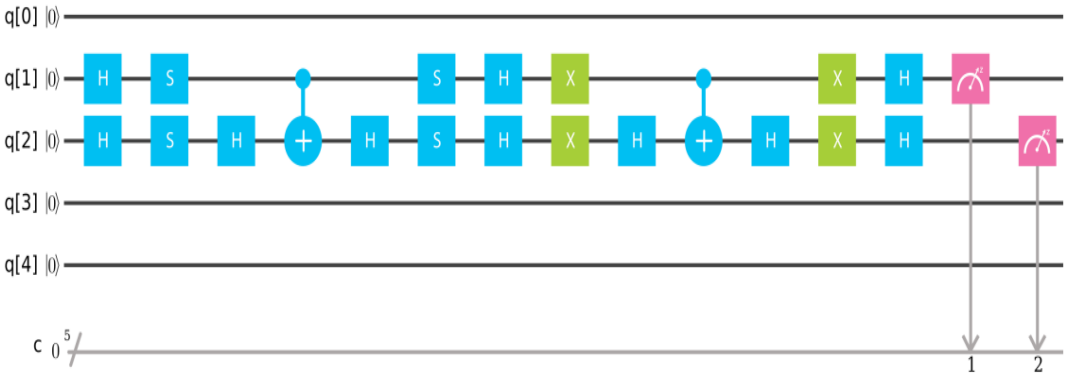
Currently there exists a few quantum computers(QC) which can be used for experimental purposes. Few of them like IBM's 5q quantum computer or Riggeti's quantum computer are available on cloud with open access to all. Furthermore many simulators have been created by various groups which can be used for learning purposes. An opensource framework named Quantum Information Science Kit (Qiskit) has been developed together by many organizations. IBM's quantum computer uses the Qiskit framework for developing quantum circuits. Running an IBM QC can be done through the graphic user interface developed by them or by writing a circuit code which is similar to a Hardware Descriptive language (HDL). The GUI is easy to use where one can pick and drop the gates to create a quantum circuit. Microsoft has developed a language named Q# which can be run on Visual Studio after installing a few packages. Although Q# also uses the Qiskit framework at its backend. Similarly Rigetti has develop Quil (Quantum Instruction Language). Quil helps run instructions on any implementation of a quantum abstract machine, such as a QVM (Quantum virtual machine) or a real QPU (Quantum processing unit). Pyquil is a library developed by Rigetti to help run the Quil code using the Python language. D wave has developed a Quantum Annealer which is a commercial product now.

Below are a few implementations on IBM 5Q quantum computer.

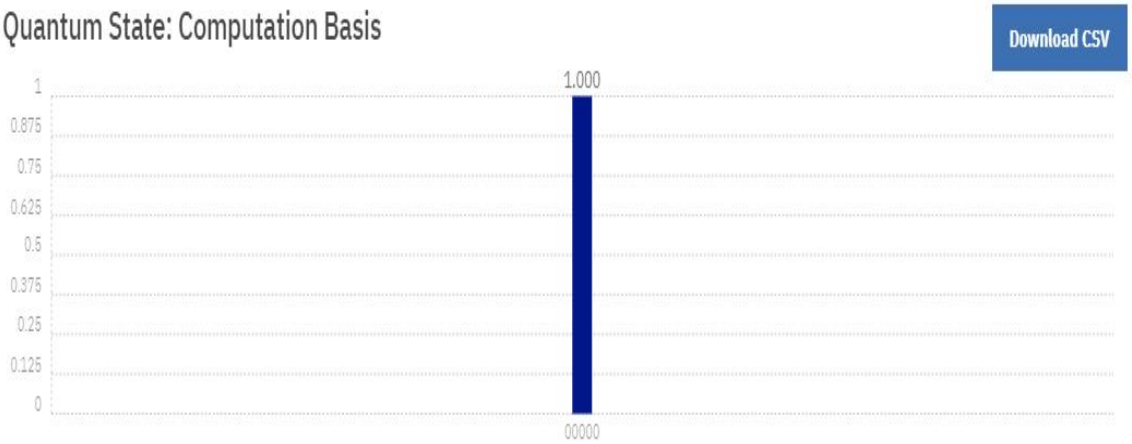
3.1 Working with IBM Quantum Computer

Below shown is a circuit for Grover's Algorithm where N = 2^2. The first part creates a uniform superposition using the Hadamard gate. The second part tags the state with U_f and the last

part performs U_s.



The equivalent code for it is given in Appendix A. The result for the above circuit is shown below.



Chapter 4

Basic Quantum Algorithms

This chapter covers the basic algorithms which either form the foundation for later algorithms or are used in the background.

4.1 HHL

Abbreviated after the authors (Harrow, Hassidim and Lloyd), this algorithm harnesses the power of quantum computer to gain an exponential speedup for solving the linear equation. The method is used when the solution is not explicitly required but is needed to find an expectation value of some operator. For some cases, the authors show that the time complexity dependency scales logarithmic in N and polynomial on the condition number and the precision. The dependence on N is exponentially better where as the dependence on the condition number is comparable and on error is worse. A condition number kappa is the ratio between the largest and the smallest eigenvalue of A where Ax = b.

4.2 Grover's Algorithm

Grover's algorithm is a quantum algorithm which can quadratically speed up the search process of an element in an unstructured database. For a classical system this would take time O(N) (N is the size of the database) since the complete database needs to be traversed. On the other hand a quantum system can search the element in root(N) time using the grover's algorithm. Also the search algorithm is not dependent on the internal structure of the list, and hence its a

generic algorithm. The use of grover's algorithm extends beyond the search problem and is used in variety of other algorithms to gain a quadratic run time improvement. This is done by a trick called Amplitude Amplification.

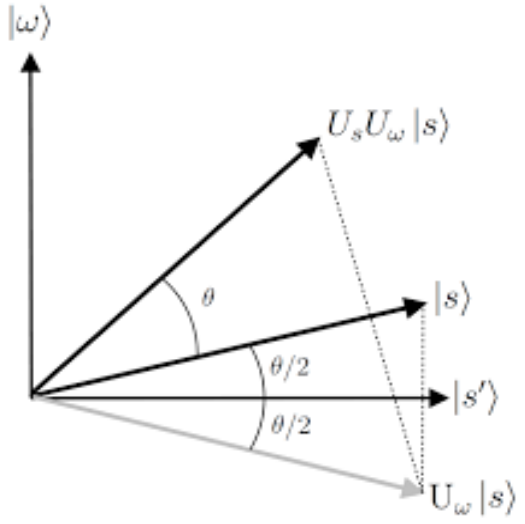
4.2.1 The Oracle

To distinguish between the element to be searched and the other elements, define a function f(x) such that f(x) = 1 when its the element of interest, else f(x) = 0. Although the items needs to be provided in superposition with this function. A unitary matrix called an oracle is defined such that: U$|x\rangle = (-1)^{f(x)} |x\rangle$

The above equation implies that whenever the element is the one of interest, the oracle flips it about the origin, else it is left unchanged.

4.2.2 Amplitude Amplification

The algorithm starts in a uniform superposition state $|s\rangle$. At this point if the state is measured, it would collapse to one of the state $|x\rangle$ with an equal probability of 1/N. Therefore the probability of getting the element of interest is 1/N. By using the Amplitude Amplification procedure, the probability of the element of interest is increased whereas the probability of rest of the elements is reduced. This is done by first reflecting the uniform superposition state $|s\rangle$ about the state which excludes the element of interest i.e. $|s'\rangle$, by using the oracle U_f defined above. The graphical representation shows this procedure, where the y axis is the element of interest and the x axis is the uniform superposition state excluding the element of interest.



Now reflect this new state about the original uniform superposition state $|s\rangle$ by applying Us, where $U_s = 2|s\rangle\langle s| - 1$. This leads the final state closer to the element of interest state. Hence after applying this above procedure for at max root(N) times, the measurement on the resultant state produces the element of interest.

Chapter 5

Quantum ML Algorithms

5.1 Quantum Boltzmann Machine

The boltzmann machine is a stochastic recurrent hopfield network which has hidden nodes in addition to the hopfield network structure. The nodes $z_a \in$ {-1,+1}. To distinguish between the hidden and visible nodes, the notation $z_a = (z_v, z_t)$ is used where v represents the visible variables and i represents the hidden variable. The vector notation is used to represent the states of the variables $\mathbf{z = (v,h)}$. The energy function is given by:

$E_z = -\sum_{a,b} w_{ab} z_a z_b - \sum_a b_a z_a$

The tuning of parameters b_a and w_{ab} is done during training. In equilibrium, the probability of observing a state $\mathbf{v}$ is given by:

$P_v = Z^{-1} \sum_{\mathbf{h}} exp^{-E_z}, Z = \sum_{\mathbf{z}} \exp^{-E_z}$

The goal is to train the parameters $\theta \in \{b_a, w_{ab}\}$ such that $P_{\mathbf{v}}$ beecomes as close as possible to $P_{\mathbf{v}^d ata}$ which is defined by the training set. We define the loss function by the negative of log-likelihood:

$L = -\sum_{\mathbf{v}} P^d_{\mathbf{v}} atalog P_{\mathbf{v}}$

substituting the above defined probability distribution gives:

$L = -\sum_{\mathbf{v}} P^d_{\mathbf{v}} ata \frac{\sum_{\mathbf{h}} \exp^{-E_{\mathbf{z}}}}{\sum_{\mathbf{z}'} \exp^{-E_{\mathbf{z}'}}}$

The training is done using the gradient descent method

$\delta\theta = -\eta\partial_\theta L$

where η is the learning parameter. Solving the equation for b_a and w_{ab} gives:

$\delta b_a = \eta\big(\overline{\langle z_\mathbf{a}\rangle_\mathbf{v}} - \langle z_\mathbf{a}\rangle\big)$

$\delta w_{ab} = \eta\big(\overline{\langle z_\mathbf{a} z_\mathbf{b}\rangle_\mathbf{v}} - \langle z_\mathbf{a} z_\mathbf{b}\rangle\big)$

The above two equations show the update rule for parameters using gradient descent.

For quantum boltzmann machine, the classical bits are replaced with qubits. Since quantum mechanics works with operators and quantum circuits needs to be reversible, the above equations and states are coverted into martix representation as follows. The dimensionality is equal to the number of possible states which is 2^N. Therefore instead of considering an energy function, a $2^N \times 2^N$ diagonal matrix is considered known as the Hamiltonian function:

$H = -\sum_a b_a\sigma_a^z - \sum_{a,b} w_{ab}\sigma_a^z\sigma_b^z$

where σ_a^z is a replacement of z_a such that:

$\sigma_a^z = I \otimes I \otimes ...(a-1)times...I \otimes \sigma_z \otimes I \otimes ...(N-a)times... \otimes I$

the partition function is given by $Z = Tr[\exp^{-H}]$. The density matrix is defined as $\rho = Z^{-1}\exp^{-H}$

For a given state$|\mathbf{v}\rangle$ of the visible variables the marginal probabilities can be obtained by tracing over the hidden variables:

$P_\mathbf{v} = Tr[\Lambda_\mathbf{v}\rho]$

where $\Lambda_\mathbf{v}$ limits the trace only to diagonal terms that correspond to the visible variables being in the state $\mathbf{v}$. Thus $\Lambda_\mathbf{v}$ is a diagonal matrix with diagonal elements being either 1, when the visibles are in state $\mathbf{v}$, or 0 otherwise.

$\Lambda_\mathbf{v} = |\mathbf{v}\rangle\langle\mathbf{v}| \otimes I_\mathbf{h}$

A transverse field is introduced to the Ising Hamiltonian by introducing a non diagonal matrices:

$\sigma_a^x = I \otimes I \otimes ...(a-1)times...I \otimes \sigma_x \otimes I \otimes ...(N-a)times... \otimes I$

Therefore, $H = -\sum_a \Gamma_a\sigma_a^x - \sum_a b_a\sigma_a^z - \sum_{a,b} w_{ab}\sigma_a^z\sigma_b^z$

The desity matrix now also has off diagonal elements. In each measurement the states of the qubits are read out in the σ_z-basis and the outcome will be a classical value ± 1. Because of the statistical nature of quantum mechanics, after each measurement a classical output $\mathbf{v}$ will appear for the visible variables with the probability $P_{\mathbf{v}}$.

Training is achieved by minimizing the negative log-likelihood given by:

$L = -\sum_{\mathbf{v}} P_{\mathbf{v}}^{data} log \frac{Tr[\Lambda_{\mathbf{v}} \exp^{-H}]}{Tr[\exp^{-H}]}$

The gradient of L is given by:

$\partial_\theta L = \sum_{\mathbf{v}} P_{\mathbf{v}}^{data} \big(\frac{Tr[\Lambda_{\mathbf{v}} \partial_\theta \exp^{-H}]}{Tr[\Lambda_{\mathbf{v}} \exp^{-H}]} - \frac{Tr[\partial_\theta \exp^{-H}]}{Tr[\exp^{-H}]} \big)$

Solving it using bound-based method gives:

$\delta b_a = \eta \big(\overline{\langle \sigma_{\mathbf{a}}^z \rangle_{\mathbf{v}}} - \langle \sigma_{\mathbf{a}}^z \rangle \big)$

$\delta w_{ab} = \eta \big(\overline{\langle \sigma_{\mathbf{a}}^z \sigma_{\mathbf{b}}^z \rangle_{\mathbf{v}}} - \langle \sigma_{\mathbf{a}}^z \sigma_{\mathbf{b}}^z \rangle \big)$

The above can be obtained by sampling from a Boltzmann distribution with hamiltonians H and $H_{\mathbf{v}}$. Γ_a can be trained in the following ways:

$\delta \Gamma_a = \eta \big(\overline{\langle \sigma_{\mathbf{a}}^x \rangle_{\mathbf{v}}} - \langle \sigma_{\mathbf{a}}^x \rangle \big)$

The problem with using the above equation is that $\langle \sigma_{\mathbf{a}}^x \rangle$ cannot be calculated in the $\sigma_{\mathbf{a}}^z$ basis. Therefore, measurement in the $\sigma_{\mathbf{a}}^x$ basis is needed to estimate $\langle \sigma_{\mathbf{a}}^x \rangle$.

Chapter 6

Conclusion

This thesis summarizes the current progress made in the field of Quantum Machine Learning.Machine learning plays an important role in the current technological advancement. The thesis discusses about the most commonly used machine learning techniques and algorithms. Chapter 3 discusses about the state of the art quantum computers. Grovers algorithm has been demonstrated on the IBM Quantum computer. The next chapter lays down the foundation for the QML algorithms. The final chapter describes some of the algorithms developed in the field. The future prospects of this topic could be to try and implement the algorithms on the IBM quantum computer.

Appendix A

Code for Grover Search Algorithm

```
include "qelib1.inc";

qreg q[5];

creg c[5];

h q[1];

h q[2];

s q[1];

s q[2];

h q[2];

cx q[1],q[2];

h q[2];

s q[1];

s q[2];

h q[1];

h q[2];

x q[1];
```

```
x q[2];

h q[2];

cx q[1],q[2];

h q[2];

x q[1];

x q[2];

h q[1];

h q[2];

measure q[1] -> c[1];

measure q[2] -> c[2];
```