SIR M. VISVESVARAYA INSTITUTE OF TECHNOLOGY
BANGALORE-562157

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

QUANTUM MACHINE LEARNING

Report

Submitted by

ABHIJITH C.	**1MV14CS004**
RAGHAVA G. DHANYA	**1MV14CS077**
SHASHANK S.	**1MV14CS131**

ABSTRACT

Recently, the increase in computational power and available data, as well as algorithmic advances in machine learning techniques, have led to impressive results in regression, classification, data generation, and reinforcement learning. Despite these successes, the proximity of the physical boundaries of chip manufacturing as well as the growing size of data sets is driving an increasing number of researchers to explore the possibility of harnessing the power of quantum computing to accelerate classical machine learning algorithms

Machine learning tasks often involve problems with manipulation and classification of a large number of vectors in large spaces. Conventional algorithms for solving such problems typically take a polynomial time in the number of vectors and the size of the space. Quantum computers are good for handling high-dimensional vectors in large tensor product spaces

The field of quantum machine learning explores how to design and implement quantum algorithms that makes machine learning faster. Recent work has produced quantum algorithms that could serve as building blocks for machine learning programs, but the hardware and software challenges are still considerable.

Contents

Abstract **i**

1 Introduction **1**

1.1 Deep Learning: The boom 1

1.2 How can Quantum Computers Help? 2

2 Technical Description **3**

2.1 Quantum Enhanced Machine Learning 3

2.1.1 Search: Grover's algorithm 3

2.1.2 Optimization: Quantum Annealing 4

2.1.3 Quantum algorithm for linear systems of equations 5

2.2 Classical learning applied to quantum systems 5

2.3 Fully quantum machine learning 6

3 Applications **7**

4 Conclusion **8**

References **9**

INTRODUCTION

Machine learning is very difficult. It's what mathematicians call an "NP-hard" problem. That's because building a good model is really a creative act. As an analogy, consider what it takes to design a house. You're balancing lots of constraints – budget, usage requirements, space limitations, etc. – but still trying to create the most beautiful house you can. A creative architect will find a great solution. Mathematically speaking, the architect is solving an optimization problem and creativity can be thought of as the ability to come up with a good solution given an objective and constraints. Classical computers aren't well suited to these types of creative problems. Machine learning algorithms are tasked with extracting meaningful information and making predictions about data. In contrast to other techniques, these algorithms construct and/or update their predictive model based on input data. The applications of the field are broad, ranging from spam filtering to image recognition, demonstrating a large market and wide societal impact.

In recent years, there have been a number of advances in the field of quantum information theory showing that particular quantum algorithms can offer a speedup over their classical counterparts. Execution time is just one concern of learning algorithms. Quantum optimizations are capable of finding the global minimum of a non convex objective function in a discrete search space. Storage capacity is also of interest. Quantum associative memories store exponentially more patterns than a classical Hopfield network. In addition to supplying exponential speed-ups in both number of vectors and their dimensions, quantum machine learning allows enhanced privacy: in a quantum clustering algorithm only $O(log(MN))$ calls to the quantum data-base are required to perform cluster assignment, while $O(MN)$ are required to uncover the actual data [2].

Deep Learning: The boom

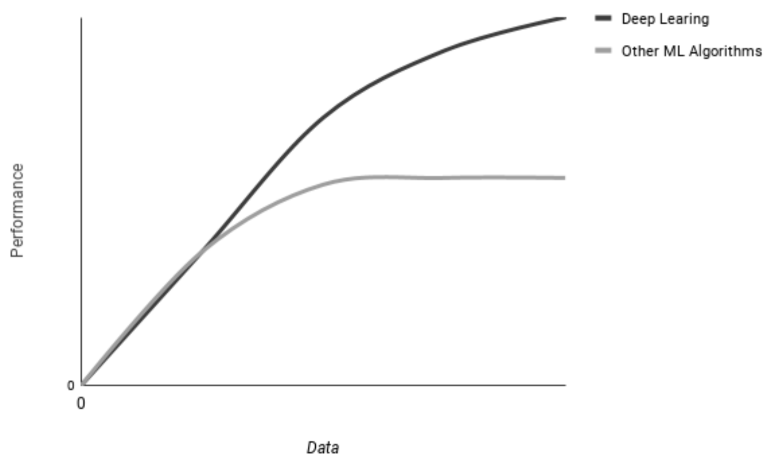

Figure 1.1: Deep learning vs Traditional Machine Learning

Now the main question, why is this field getting traction these days? The first Convolutional Neural Network was developed by Yann LeCun in 1995 [1], why did it take so long for deep learning to take off? One answer, the Scale. Scale drives deep learning progress, scale in both data-set sizes and computational power. But the same scale could act as bottleneck for deep learning.

The data volumes are exploding: more data has been created in the past two years than in the entire previous history of the human race. Data is growing faster than ever before and by the year 2020, about 1.7 megabytes of new information will be created every second for every human being on the planet. By then, our accumulated digital universe of data will grow from 4.4 zettabyets today to around 44 zettabytes, or 44 trillion gigabytes. The demand for storage has grown more than 50% annually in recent years, a rate faster than the rapidly decreasing unit cost of storage can handle. The growth of data is profound and shows no signs slowing. There is an inconvenient truth in technology: the amount of data keeps growing exponentially, while the increases in the power of computers are slowing down. Computers today can carry out those instructions in nanoseconds, but they still do it one step at a time, which has become a liability known as the von Neumann bottleneck. If individual chips can no longer be made faster, and the amount of work computers have to do continues to grow, the only way to attack ever larger data problems with von Neumann programmable computers will be to build more computers and ever bigger data centers. It seems as though we have hit a wall in growth of computational capacities. How can we meet this demand for data storage and computational power?

One of most promising solution is Quantum computers and Quantum information processing.

How can Quantum Computers Help?

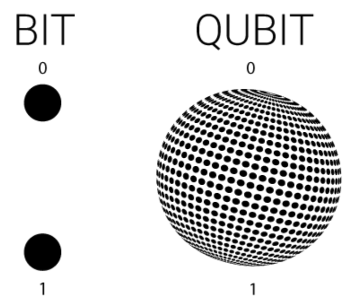

Figure 1.2: Bit vs Qubit

The basic idea is to use the atypical behavior of quantum physics such as quantum tunneling and quantum entanglement to get atoms to make calculations. A dozen atoms in a quantum computer would be more powerful than the world's biggest supercomputer. While regular computers symbolize data in bits, 1s and 0s expressed by flicking tiny switch-like transistors on or off, quantum computers use quantum bits, or qubits, that can essentially be both on and off, enabling them to carry out two or more calculations simultaneously. In principle, quantum computers could prove extraordinarily faster than normal computers for certain problems because they can run through every possible combination at once. In fact, a quantum computer with 300 qubits could run more calculations in an instant than there are atoms in the universe. Also qubits can store more data than classical bits, n qubits can carry about 2^n classical bits of information due to superposition.

Technical Description

Quantum computing can help machine learning in two ways- data and computation. Based on this Quantum Machine Learning can be divided into 3 approaches.

1. Quantum enhanced Machine Learning (classical data with quantum algorithms)
2. Classical learning applied to quantum systems (classical algorithm with quantum data)
3. Fully quantum machine learning (both data and algorithm is quantum based)

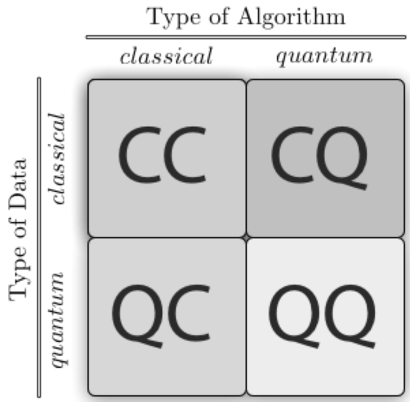

Figure 2.1: Approaches in Quantum Machine Learning

Quantum Enhanced Machine Learning

Quantum Enhanced Machine Learning is the most promising of these methods on early quantum computers. Such algorithms typically require one to encode the given classical dataset into a quantum computer so as to make it accessible for quantum information processing. Machine learning techniques use mathematical algorithms and tools to search for patterns in data. The most important and time taking parts of machine learning algorithms are searching, sampling and optimization. We explain how these can be improved by Quantum Computing in the following sections.

Search: Grover's algorithm

When you compute Grover's Algorithm you are simultaneously testing every input value to see which is the correct input value. Using quantum superposition you can get qubits in a state that represents all possible inputs. Then run that superposition of states through some function to get each input together with its associated output. You are given a list of n elements, and you know that one element satisfies a certain condition, while the other n-1 elements don't. Basically, this is an algorithm for finding a specific element in an unordered list. A classical computer would not be able to exploit any structure in this problem and therefore needs to scan up to $n-1$ elements to find the one needed. Prepare n qubits in a uniform state so that all 2^n numbers are in a uniform superposition, each with a coefficient of $1/\sqrt{n}$ If we would measure

the n qubits now, all 2^n results would be equally likely.
Then run the Grover iteration k times, which consists of two steps (which can be merged into 1 step):

1. Negate the coefficient of the sought element. This is a unitary operation, which leaves all elements not satisfying the condition as they are and only negates the intended one.

2. Reflect all quantum states at the arithmetic mean of all quantum states. This also is a unitary operation.

Each iteration amplifies the coefficient of the correct solution while damping the coefficient of all $n-1$ incorrect solutions, however only to a certain point. If you choose the number of iterations k correctly, you have maximized the coefficient for the correct solution. This means that the sought element's probability is now (almost) 1, so if we measure the qubits, we are very likely to get the correct answer. If we don't get it, we can repeat the algorithm from the beginning until we get the right answer. Doing more or less than the optimal k iterations reduces the probability of finding the correct solution - however, the algorithm periodically reaches the maximum coefficient. It can be shown that $k = O(\sqrt{n})$. This is remarkable, because a classical computer needs $O(n)$ steps for solving this problem. This is only a quadratic speedup (rather than an exponential speedup as for other quantum algorithms), but the Grover algorithm has quite a large number of useful applications and is fairly simple. The Grover algorithm can be extended to support a predefined number of $m(0 \leq m \leq n)$ elements which satisfy the condition, instead of 1. m need not be known in advance. Quantum period finding can be used to determine m before starting the extended Grover algorithm.

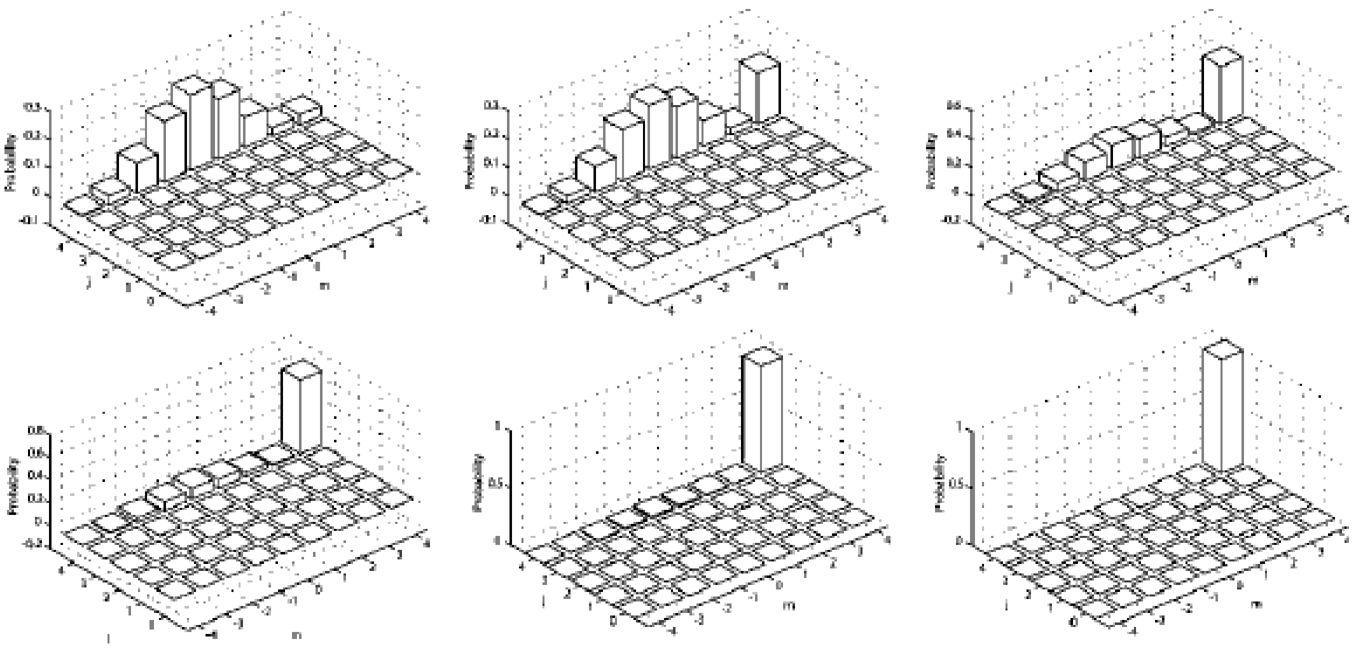

Figure 2.2: Grover's Search

Optimization: Quantum Annealing

Classical computing might use what's called "gradient descent": start at a random spot on the surface, look around for a lower spot to walk down to, and repeat until you can't walk downhill anymore. But all too often that gets you stuck in a "local minimum" – a valley that isn't the very lowest point on the surface. That's where quantum computing comes in. It lets you cheat a little, giving you some chance to "tunnel" through a ridge to see if there's a lower valley hidden beyond it. This gives you a much better shot at finding the true lowest point – the optimal solution. We'll see how that works using Quantum annealing.

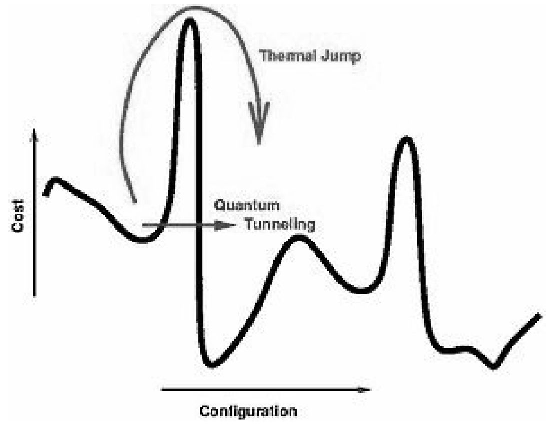

Figure 2.3: Quantum annealing

Quantum annealing can be compared to simulated annealing, whose "temperature" parameter plays a similar role to Quantum Annealing's tunneling field strength. In simulated annealing, the temperature determines the probability of moving to a state of higher "energy" from a single current state. In quantum annealing, the strength of transverse field determines the quantum-mechanical probability to change the amplitudes of all states in parallel. The tunneling field is basically a energy term that does not commute with the classical potential energy part. Analytical and numerical evidence suggests that quantum annealing outperforms simulated annealing under certain conditions. Quantum annealing is also used in sampling from high-dimensional probability distributions.

Quantum algorithm for linear systems of equations

Solving a system of linear equations can be thought of as a Matrix inversion problem.The best known classical algorithm to solve matrix inversion takes $O(n^2 log(n))$(best proven lower bound) time. Using Quantum algorithms we can perform matrix inversion in logarithmic time complexity of N. This is because using superposition we can find the eigenvectors and eigenvalues of the matrix by eigen decomposition of a matrix in a logarithmic time. This algorithm can be used to improve Least Square fitting , Principal component analysis and support vector machines which is a large margin optimized linear or non-linear classifier. Due to the prevalence of linear systems in virtually all areas of science and engineering, the quantum algorithm for linear systems of equations has the potential for widespread applicability.

Other than these basic algorithms that are used in almost all Machine Learning systems there are more complex ones which are applicable for specific systems such ANN, HMM etc. There are Quantum algorithms which can provide exponential speedup for these systems.

Classical learning applied to quantum systems

This approach uses classical learning techniques to process large amounts of experimental quantum data in order to characterize an unknown quantum system.

The ability to experimentally control and prepare increasingly complex quantum systems brings with it a growing need to turn large and noisy data sets into meaningful information. This is a problem that has already been studied extensively in the classical setting, and consequently, many existing machine learning techniques can be naturally adapted to more efficiently address experimentally relevant problems. For example, Bayesian methods and concepts of algorithmic learning can be fruitfully applied to tackle quantum state classification, Hamiltonian learning and the characterization of an unknown unitary transformation. It has many applications such as

- Identifying an accurate model for the dynamics of a quantum system
- Extracting information on unknown states
- Learning unknown unitary transformations and measurements

As Classical learning on classical data helps us understand the data and the system that produced that data, Classical learning on Quantum data helps us understand Quantum data and the Quantum system that produced that data. This is relatively small and current research field.

Fully quantum machine learning

In the most general case of quantum machine learning, both the learning device and the system under study, as well as their interaction, are fully quantum. This goes beyond the early quantum computers and requires a much sophisticated Quantum Computer.

One class of problem that can benefit from the fully quantum approach is that of 'learning' unknown quantum states, processes or measurements, in the sense that one can subsequently reproduce them on another quantum system. For example, one may wish to learn a measurement that discriminates between two coherent states, given not a classical description of the states to be discriminated, but instead a set of example quantum systems prepared in these states. The naive approach would be to first extract a classical description of the states and then implement an ideal discriminating measurement based on this information. This would only require classical learning. However, one can show that a fully quantum approach is strictly superior in this case. (This also relates to work on quantum pattern matching.) The problem of learning unitary transformations can be approached in a similar way. A fully quantum clustering algorithm is also available - A Quantum Nearest-Centroid Algorithm for k-Means Clustering k-means clustering is a popular machine learning algorithm that structures an unlabelled dataset into k classes. k-means clustering is an NP-hard problem. The complexity arguments on the dependence of m were rigorously confirmed using the QPCA construction for a support vector machine (SVM) algorithm. This can roughly be thought of as a k-means clustering problem with $k = 2$. A speedup is obtained due to the classical computation of the required inner products being $O(nm)$.

APPLICATIONS

CONCLUSION

Head of Department

Project Guide **Project Coordinator**

REFERENCES

[1] Yann LeCun and Yoshua Bengio. The handbook of brain theory and neural networks. chapter Convolutional Networks for Images, Speech, and Time Series, pages 255–258. MIT Press, Cambridge, MA, USA, 1998.

[2] S. Lloyd, M. Mohseni, and P. Rebentrost. Quantum algorithms for supervised and unsupervised machine learning. *ArXiv e-prints*, July 2013.