Maastricht Science Programme

Research Proposal

Experimental Implementation of Quantum Machine Learning Algorithms

Mark Fingerhuth

September 16, 2016

In partial fulfillment of the requirements for the degree Bachelor of Science (BSc) at Maastricht University.

In collaboration with the Quantum Research Group at the University of KwaZulu-Natal Durban.
Under supervision of Prof. Francesco Petruccione, Quantum Research Group at UKZN, and internal supervision of Dr. Fabrice Birembaut, Maastricht Science Programme, Maastricht University.

Abstract

Quantum machine learning, the intersection of quantum computation and classical machine learning, bears the potential to provide more efficient ways to deal with big data through the use of quantum superpositions, entanglement and the resulting quantum parallelism. The proposed research will attempt to experimentally implement two quantum machine learning routines and use them to solve small machine learning problems. This will establish the fourth proof-of-concept study in the field and demonstrate that quantum machine learning is already implementable on current devices. This is vital in order to show that an extrapolation to larger quantum computing devices will indeed lead to vast speed-ups of current machine learning algorithms.

Contents

1 Introduction

The ability to understand spoken language, to recognize faces and to distinguish different types of fruit comes naturally to humans, even though these processes of pattern recognition and classification are inherently complex. Machine learning (ML), a subtopic of artificial intelligence, is concerned with the development of algorithms that mimic these mechanisms, thereby enabling computers to find and recognise patterns in data and classify unknown inputs based on previous training with labelled inputs. Such algorithms paved the way for e.g. human speech recognition, recommendation engines as used by Amazon and algorithms that can predict heart disease from real-time electrocardiograms (Acharya et al., 2015; Pazzani & Billsus, 2007).

According to IBM (2016), approximately 2.5 quintillion (10^{18}) bytes of digital data are created every day. This growing number implies that every area dealing with data will eventually require advanced algorithms that can make sense of data content, retrieve patterns and reveal correlations. However, most ML algorithms involve the execution of computationally expensive operations and doing so on large data sets inevitably takes a lot of time (Bekkerman, Bilenko, & Langford, 2011). Hence, it becomes increasingly important to find efficient ways of dealing with big data and/or reduce the computational complexity of the algorithms.

A promising solution is the use of quantum computation which has been researched intensively in the last few decades. Quantum computers (QCs) use quantum mechanical systems and their special properties to manipulate and process information in ways that are impossible to implement on classical computers. The quantum equivalent to a classical bit is called a quantum bit (or qubit) and additionally to being in either state ($|0\rangle$ or $|1\rangle$) they can be in a linear superposition of $|0\rangle$ and $|1\rangle$. This peculiar property gives rise to so called quantum parallelism, which enables the execution of certain operations on many quantum states at the same time. However, despite this obvious advantage the real difficulty in quantum computation lies in the retrieval of the computed solution since a measurement of a qubit collapses it into a single classical bit and thereby destroys information about its previous superposition. Several quantum algorithms have been proposed that provide exponential speed-ups when compared to their classical counterparts with Shor's prime factorization algorithm being the most famous (Shor, 1994). As another example, Grover's quantum database search algorithm enables finding a single element in a list of N elements within roughly $\sqrt{N}$ steps instead of a classical minimum of $\frac{N}{2}$ steps (L. K. Grover, 1997). Hence, quantum computation bears the potential to vastly improve computational power, speed up the processing of big data and solve certain problems that are practically unsolvable on classical computers.

Considering these advantages, the combination of quantum computation and classical ML into the new field of quantum machine learning (QML) seems almost natural. However, since most ML algorithms rely on solving some system of linear equations, a corresponding quantum algorithm is required for QML to become achievable. Harrow, Hassidim, and Lloyd (2009) were first to describe a quantum algorithm solving linear equations of the form $A \cdot x = b$ exponentially faster than classical routines. This so called HHL algorithm has since become a subroutine in many QML algorithms. There are currently two main ideas on how to merge quantum computation with ML, namely a) running the classical algorithm on a classical computer and outsourcing only the computationally intensive task to a QC or b) executing the quantum version of the entire algorithm on a QC. Current QML research mostly focusses on the latter by developing quantum algorithms that tab into the full potential of quantum parallelism.

1.1 Motivation

Classical ML is a very practical topic since it can be directly tested, verified and implemented on any commercial classical computer. So far, QML has been of almost entirely theoretical nature since the required computational resources are not in place yet. To yield reliable solutions QML algorithms often require a relatively large number of error-corrected qubits and some sort of quantum data storage such as the proposed quantum random access memory (qRAM) (Giovannetti, Lloyd, & Maccone, 2008). However, to date the maximum number of superconducting qubits reportedly used for calculation is nine, the D-Wave II quantum annealing device delivers 1152 qubits but can only solve a narrow class of problems and a qRAM has not been developed yet (D-Wave, 2015; O'Malley et al., 2016). Furthermore, qubit error-correction is still a very active research field and most of the described preliminary QCs deal with non error-corrected qubits with short lifetimes and are, thus, impractical for large QML implementations.

Until now there has been only three experimental verifications of QML algorithms that provide proof-of-principle. Li, Liu, Xu, and Du (2015) successfully distinguished a handwritten six from a nine using a quantum support vector machine on a four-qubit nuclear magnetic resonance test bench. In addition, Cai et al. (2015) were first to experimentally demonstrate quantum machine learning on a photonic QC and showed that the distance between and the inner product of two vectors can indeed be computed quantum mechanically. Lastly, Ristè et al. (2015) solved a learning parity problem with five superconducting qubits and found that a quantum advantage can already be observed in non error-corrected systems.

Considering the large gap between the number of proposed QML algorithms and experimental realisations of scaled-down QML problems, it remains important to find QML problems which can already be implemented on currently available quantum technology. Hence, the purpose of this study is to provide proof-of-principle implementation of selected QML algorithms on small datasets. This is an important step in the attempt to shift QML from a purely theoretical research area to a more applied field such as classical ML. Furthermore, this can also lead to verification or falsification of the claims and assumptions made in the theoretical field of QML.

1.2 Research Question

In light of the theoretical nature of current QML research and the small number of experimental realizations, this research will address the following question:

How can theoretically proposed quantum machine learning algorithms be implemented on state-of-the-art quantum technology?

The following sections will outline the steps required and the tools used in order to answer this research question.

1.3 Research Objectives

The main objective of this research is to demonstrate that QML algorithms can already be used for solving small problems on currently available quantum technology. Although this seems trivial at first, there are many problems that are often not addressed in the proposals of QML algorithms such as the encoding of data, the influence of quantum noise and the restrictions on the data type (e.g. uniformly distributed or low-rank data sets only). All theses issues have to be addressed when implementing QML algorithms experimentally and thus constitute the major research objectives during this study.

2 Research Methods

The proposed research will be solely based on the two QML algorithms described in Schuld, Sinayskiy, and Petruccione (2014, 2016). Firstly, Schuld et al. (2014) is a quantum version of the distance weighted k-nearest neighbour (kNN) algorithm. For clarification, let us consider a training data set D_T consisting of ten vectors $v_0, v_1, .., v_{10}$ that are each assigned to either class A or B. The training vectors are visualized as yellow and purple circles in Fig. 1. kNN is a non-parametric classifier that given a new unclassified input vector x (red star in Fig. 1) considers the k nearest neighbours (using a predefined measure of distance) and classifies x, based on a majority vote, as either A or B. Thereby, k is a positive integer, usually chosen to be small and its value determines the classification outcome. Namely, in the case $k = 3$ in Fig. 1, vector x would be classified as class B (purple) but in the case $k = 6$ it would be labelled class A (yellow).

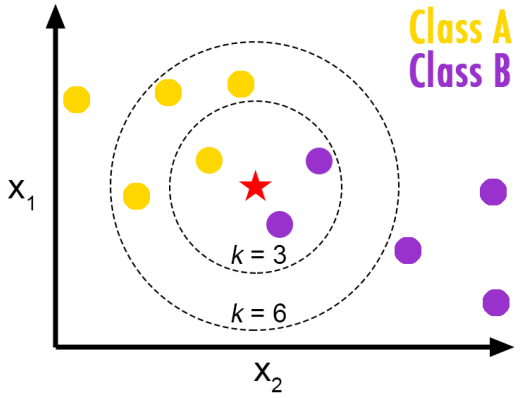

Figure 1: Visualization of a kNN classifier[1]

In the case of $k = all$, input vector x would simply be assigned to the class with the most members. In this case, the training vectors can be given distance-dependent weights (such as $\frac{1}{distance}$) in order to increase the influence of closer vectors over more distant ones. Hereby, the advantages of the quantum version are the parallel computation of the distances between each training vector and the input vector as well as contracting distance computation and distance weighting into one computational step.

Next, Schuld et al. (2016) details a quantum algorithm based on linear least-squares regression. In contrast to the discrete classes (A or B) used in kNN, this algorithm focusses on continuous outputs and the learning task is to find the best linear fit by reducing the least-squares error to all training data points. A simple example is that in a certain country, given ten data points that link the size of farm land (in m^2) to the total prize (in \$), predict the total prize for a new input size of farm land. Predicting such continuous class labels is important in many fields ranging from medicine to finance. The classical algorithm involves the computation of the (pseudo)inverse of a matrix which becomes computationally intensive for large data sets. In contrast, the runtime of the quantum algorithm is independent of the size of the training data set and mainly depends on the dimension of the training vectors only.

The first step towards implementation of the outlined algorithms will be the identification of one or several small ML problems that can be executed on maximally nine qubits. For example, this might include the characterization of colours or differentiation of digits or letters. It thereby plays an important role if the respective ML problem can be approached using a very small dataset such as the average pixel brightness or the ratio of pixels above and below the image bisector. Ideally, the data should be representable as a 2-D vector such that it requires only a few qubits to encode the information quantum mechanically.

[1]Reprinted from GitHub, Burton de Wilde, Retrieved September 13, 2016, from `http://bdewilde.github.io/blog/blogger/2012/10/26/classification-of-hand-written-digits-3/`. Copyright 2012 by Burton de Wilde. Reprinted with permission.

There are two types of tools available for the implementation of the QML algorithms. Firstly, since current state-of-the-art quantum technology uses maximally nine qubits, a classical computer can still be used to simulate the behaviour of the QC. For example, such a software architecture is provided by Microsoft Research which has released the quantum simulation toolsuite Liqui|⟩ based on the programming language F#. There are many more QC simulators available and the selection depends on the chosen QML problem and corresponding dataset (Quantiki, 2016).

Secondly, earlier this year technology company IBM has enabled public access to their experimental quantum processor containing five non error-corrected superconducting qubits. Instead of only simulating on classical hardware, this opens up the possibility of executing the QML algorithm on actual quantum hardware. Whether it is possible to make use of IBM's QC is highly dependent on the data set chosen. Furthermore, it still remains unclear if the algorithm in Schuld et al. (2016) can be executed using five qubits only.

For the subsequent discussion of quantum data encoding it is important to mathematically express the superposition of a qubit $|q\rangle$ as a linear combination of the qubit states $|0\rangle$ and $|1\rangle$ as follows:

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

where α and β are complex coefficients and are often referred to as phase factors or amplitudes. When measuring qubit $|q\rangle$ it will take the value $|0\rangle$ with a probability of $|\alpha|^2$ and $|1\rangle$ with a probability of $|\beta|^2$. Since the total probability of measuring any value has to be 1, the normalization condition $|\alpha|^2 + |\beta|^2 = 1$ must be satisfied at all times.

Both algorithms assume that the classical data is readily available in the form of quantum states. Hence, since it is a non-trivial and still researched topic, the first challenge will be translating the classical data into such states. The quantum version of the distance weighted kNN requires a binary data string of length n (e.g. 0110) to be encoded one to one into n qubits (e.g. $|0110\rangle$). This is referred to as *qubit encoding* and will be done using the algorithm outlined in Ventura and Martinez (1999). The quantum linear regression algorithm is based on so called *amplitude encoding*, where the classical data is written into the amplitudes (α and β in Equ. 1) of quantum states which is considerably more difficult to achieve than qubit encoding. It is still a very active field of research and so far it can only be done with relatively uniform data sets. For this purpose, the algorithm proposed in L. Grover and Rudolph (2002) will be used and special attention will be paid to choosing a suitable dataset.

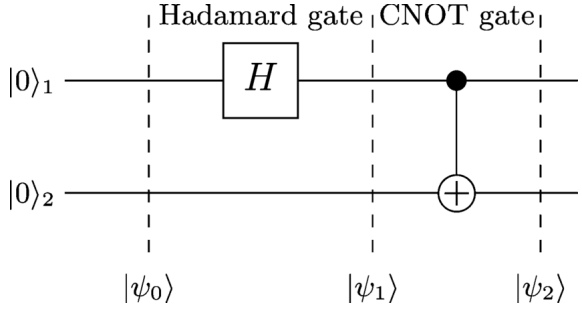

Figure 2: Basic quantum circuit with a single-qubit Hadamard gate (creates a superposition as shown in Equ. 1 with $\alpha = \beta = \frac{1}{\sqrt{2}}$) and a two-qubit CNOT gate (controlled bit flip)[2]

Next, two separate quantum circuits consisting of quantum logic gates will be designed that accurately represent the two QML algorithms as outlined in the paper of Schuld et al. (2014, 2016). Each quantum circuit consist of single- and multi-qubit gates of which a simple example is shown in Fig. 2. Each of those circuits is then combined with its respective quantum data encoding circuit.

Lastly, the computed solution needs to be retrieved by measuring specific qubits. By repeating the execution of the algorithms, a probability distribution over the qubit measurements is obtained that represents the solution to the given problem.

[2]Reprinted from Botsinis, Ng, and Hanzo (2013), Retrieved September 13, 2016, from `https://www.researchgate.net/publication/236883187_Quantum_Search_Algorithms_Quantum_Wireless_and_a_Low-Complexity_Maximum_Likelihood_Iterative_Quantum_Multi-User_Detector_Design`. Copyright 2013 by Botsinis et al. (2013). Reprinted with permission.

To summarize, the steps needed for the successful implementation of a QML algorithm are given below.

1. Find a small implementable ML problem
2. Generate or find a suitable dataset
3. Encode the classical data into quantum states
4. Design a quantum circuit representing the QML algorithm
5. Execute the entire quantum circuit multiple times
6. Retrieve the solution from the resulting probability distribution

3 Timeline

The projected timeline for the proposed bachelor thesis research is given below.

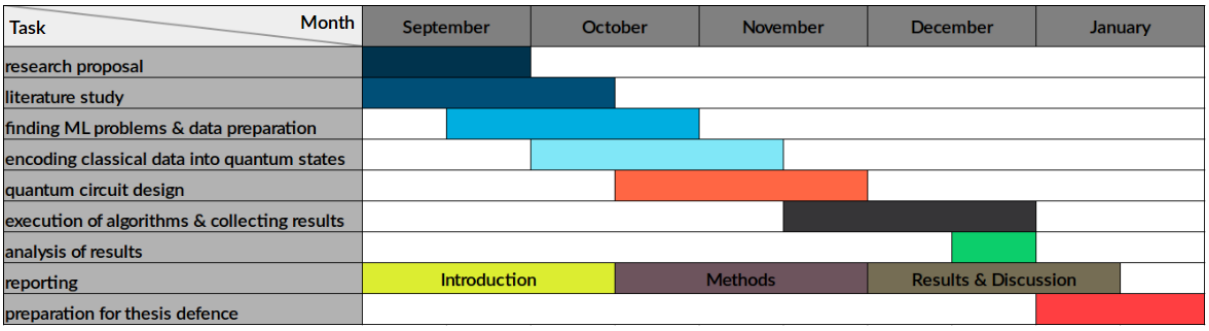

Figure 3: Projected timeline for proposed research

4 Research Impact

The proposed research is part of the recent effort to move QML from theory to praxis since so far there have been only three experimental QML verifications. Ideally, the research outcome will demonstrate quantum advantages over the classical algorithms providing supporting evidence for the claim that QML can indeed be used to solve ML problems. Additionally, successful proof-of-principle studies are crucial for further research to be funded and supported since it shows that an upscaling of quantum computational power will eventually lead to at best exponential speed-ups compared to classical ML and hence has the potential to revolutionize the handling of big data.

5 Conclusion

Recently, many inventive and powerful improvements to classical ML algorithms were published in the field of QML. Such algorithms are required in order to cope with and make sense of the enormous amounts of data generated in contemporary society. However, the required computational resources are not available yet and proof-of-concept studies are needed to further stress the value of theoretical developments in QML. The proposed research will identify small ML problems and suitable datasets that can already be solved using QML algorithms on state-of-the-art quantum computing technology. Investigated will be two algorithms given in Schuld et al. (2014, 2016) which focus on supervised pattern classification based on linear regression and kNN. The key steps will be encoding the classical data into quantum states and designing quantum circuits representing the algorithms. Possible tools for the research include IBM's five-qubit quantum processor or QC simulator toolsuites that are executed on a classical computer.

6 References

Acharya, U. R., Fujita, H., Sudarshan, V. K., Sree, V. S., Eugene, L. W. J., Ghista, D. N., & San Tan, R. (2015). An integrated index for detection of sudden cardiac death using discrete wavelet transform and nonlinear features. *Knowledge-Based Systems*, *83*, 149–158.

Bekkerman, R., Bilenko, M., & Langford, J. (2011). *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press.

Botsinis, P., Ng, S. X., & Hanzo, L. (2013). Quantum search algorithms, quantum wireless, and a low-complexity maximum likelihood iterative quantum multi-user detector design. *IEEE access*, *1*, 94–122.

Cai, X. D., Wu, D., Su, Z. E., Chen, M. C., Wang, X. L., Li, L., ... Pan, J. W. (2015). Entanglement-based machine learning on a quantum computer. *Physical Review Letters*, *114*(11), 1–5. doi: 10.1103/PhysRevLett.114.110504

D-Wave. (2015). *Breaking through 1000 qubits.* `http://www.dwavesys.com/blog/2015/06/breaking-through-1000-qubits`. (Accessed: 2016-09-09)

Giovannetti, V., Lloyd, S., & Maccone, L. (2008). Quantum random access memory. *Physical review letters*, *100*(16), 160501.

Grover, L., & Rudolph, T. (2002). Creating superpositions that correspond to efficiently integrable probability distributions. *arXiv preprint quant-ph/0208112*.

Grover, L. K. (1997, Jul). Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.*, *79*, 325–328. Retrieved from `http://link.aps.org/doi/10.1103/PhysRevLett.79.325` doi: 10.1103/PhysRevLett.79.325

Harrow, A. W., Hassidim, A., & Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Physical Review Letters*, *103*(15), 1–4. doi: 10.1103/PhysRevLett.103.150502

IBM. (2016). *What is big data?* `https://www-01.ibm.com/software/data/bigdata/what-is-big-data.html`. (Accessed: 2016-09-08)

Li, Z., Liu, X., Xu, N., & Du, J. (2015). Experimental realization of a quantum support vector machine. *Physical Review Letters*, *114*(14), 1–5. doi: 10.1103/PhysRevLett.114.140504

O'Malley, P. J. J., Babbush, R., Kivlichan, I. D., Romero, J., McClean, J. R., Barends, R., ... Martinis, J. M. (2016, Jul). Scalable quantum simulation of molecular energies. *Phys. Rev. X*, *6*, 031007. Retrieved from `http://link.aps.org/doi/10.1103/PhysRevX.6.031007` doi: 10.1103/PhysRevX.6.031007

Pazzani, M. J., & Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web* (pp. 325–341). Springer.

Quantiki. (2016). *List of quantum computing simulators.* `https://quantiki.org/wiki/list-qc-simulators`. (Accessed: 2016-09-12)

Ristè, D., da Silva, M. P., Ryan, C. A., Cross, A. W., Smolin, J. A., Gambetta, J. M., ... Johnson, B. R. (2015). Demonstration of quantum advantage in machine learning. *arXiv:1512.06069*, 1–12. Retrieved from `http://arxiv.org/abs/1512.06069`

Schuld, M., Sinayskiy, I., & Petruccione, F. (2014). Quantum computing for pattern classification. , 14. Retrieved from `http://arxiv.org/abs/1412.3646` doi: 10.1007/978-3-319-13560-1

Schuld, M., Sinayskiy, I., & Petruccione, F. (2016). Pattern classification with linear regression on a quantum computer. , 5. Retrieved from `http://arxiv.org/abs/1601.07823`

Shor, P. W. (1994). Polynomial-time algorithm for prime factorization and discrete logarithms on a quantum computer: Proc. , 124.

Ventura, D., & Martinez, T. (1999). Initializing the amplitude distribution of a quantum state. *Foundations of Physics Letters*, *12*(6), 547–559.