Mark Fingerhuth

Quantum-enhanced machine learning: Implementing a quantum k-nearest neighbour algorithm

Bachelor Thesis

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

Supervision

Prof. Francesco Petruccione
Dr. Fabrice Birembaut

January 2017

Preface

Blah blah . . .

Contents

Abstract

NEEDS MODIFICATION!

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 implement and simulate two quantum machine learning routines and use them to solve small machine learning problems. That will establish one of the earliest proof-of-concept studies in the field and demonstrate that quantum machine learning is already implementable on small-scale quantum computers. This is vital to show that an extrapolation to larger quantum computing devices will indeed lead to vast speed-ups of current machine learning algorithms.

Nomenclature

Symbols

$\otimes$	Tensor product	
i	Imaginary unit	$i = \sqrt{-1}$
$\dagger$	Hermitian conjugate	Complex conjugate transpose

Indicies

a	Ambient
air	Air

Acronyms and Abbreviations

QML	Quantum machine learning
QC	Quantum computer
Prob	Probability
CNOT	Controlled NOT gate
CCNOT	Controlled controlled NOT gate (Toffoli gate)
CU	Controlled U gate (where U can be any unitary quantum gate)

Chapter 1

Introduction

SOME MORE GENERAL INTRO TEXT HERE?

The ability to understand spoken language, to recognize faces and to distinguish 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 perform these types of tasks, thus enabling computers to find and recognise patterns in data and classify unknown inputs based on previous training with labelled inputs. Such algorithms make the core of e.g. human speech recognition and recommendation engines as used by Amazon.

According to (?, ?), 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 (?, ?). Thus, it becomes increasingly important to find efficient ways of dealing with big data.

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 (qubit) and additionally to being in either state $|0\rangle$ or $|1\rangle$ it can be in their linear superposition:

$$|q\rangle = \alpha |0\rangle + \beta |1\rangle \tag{1.1}$$

where α and β are complex numbers and often referred to as 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 has to sum to unity, the normalization condition $|\alpha|^2 + |\beta|^2 = 1$ must be satisfied at all times.

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. Despite this advantage, the 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 (?, ?). Hence, quantum computation has 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. 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 approach by developing quantum algorithms that tap into the full potential of quantum parallelism.

1.1 Research Question

Matthias Troyer citation about q software engineering needed?

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 small-scale quantum computers?

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

Chapter 2

Theoretical Foundations and Methods

CHECK FOOTNOTE NUMBERS IN WHOLE DOCUMENT!

2.1 Quantum Bits

INTRODUCE CONCEPT OF T1 and T2!

2.1.1 Single Qubit Systems

Classical computers manipulate bits, whereas quantum computer's most fundamental unit is called a quantum bit, often abbreviated as qubit. Bits as well as qubits are binary entities, meaning they can only take the values 0 and 1.

A good example for a physical implementation of a qubit is the single electron of a hydrogen atom sketched in Fig. **??**. Usually, the electron is found in its ground state which one can define as the $|0\rangle$ state of the qubit. Using a laser pulse the electron can be excited into the next highest valence shell which one can define as the $|1\rangle$ state of the qubit. After some time t the electron will decohere to its ground state ($|0\rangle$) which is called the longitudinal coherence or amplitude damping time and is an important parameter for measuring qubit lifetimes.
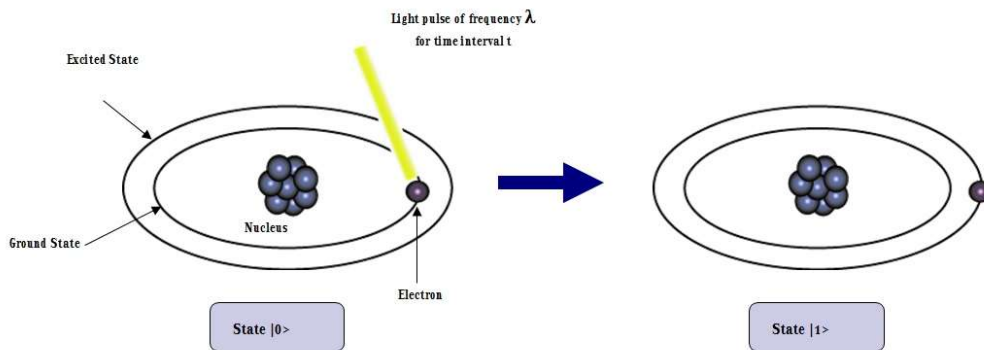

Figure 2.1: A simple example of a physical qubit using the electron of a hydrogen atom

A classical non-probabilistic bit can only be in one of the two possible states at once. In contrast, qubits obey the laws of quantum mechanics, which gives rise to the powerful property that - besides being a definite 0 or 1 - they can also be in a superposition of the two states. Mathematically this is expressed as a linear combination of the states 0 and 1:

$$|\psi\rangle = \alpha \, |0\rangle + \beta \, |1\rangle \tag{2.1}$$

where α and β are complex coefficients ($\alpha, \beta \in \mathbb{C}$) and are often referred to as phase factors or amplitudes. $|0\rangle$ is the Dirac notation for the qubit being in state 0 and it represents a two-dimensional vector in a complex 2-D vector space (called Hilbert space $\mathcal{H}_2$). $|0\rangle$ and $|1\rangle$ are the computational basis states and they constitute an orthonormal basis of $\mathcal{H}_2$. For the sake of clarity, $|0\rangle$ and $|1\rangle$ can be thought of as the 2-D vectors shown below.

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2.2}$$

Subbing these vectors into Equ. 2.1 yields the vector representation of $|\psi\rangle$:

$$|\psi\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{2.3}$$

However, even though a qubit can be in a superposition of $|0\rangle$ and $|1\rangle$, when measured it will take the value $|0\rangle$ with a probability of

$$Prob(|0\rangle) = |\alpha|^2 \tag{2.4}$$

and $|1\rangle$ with a probability of

$$Prob(|1\rangle) = |\beta|^2 \tag{2.5}$$

The fact that the probability of measuring a particular state is equal the absolute value squared of the respective amplitude is called Born rule (citation). Since the total probability of measuring any value has to be 1, the following normalization condition must be satisfied:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2.6}$$

Therefore, a qubit is inherently probabilistic but when measured it collapses into a single classical bit (0 or 1). It follows that a measurement destroys information about the superposition of the qubit (the values of α and β). This constitutes one of the main difficulties when designing quantum algorithms since only limited information can be obtained about the final states of the qubits in the quantum computer.

Similar to logic gates in a classical computer, a QC manipulates qubit by means of quantum logic gates which will be introduced in detail in Section 2.2. Generally, an arbitrary quantum logic gate U acting on a single qubit state is a linear transformation given by a 2x2 matrix whose action on $|\psi\rangle$ is defined as:

$$U \, |\psi\rangle = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a * \alpha + b * \beta \\ c * \alpha + d * \beta \end{pmatrix} \tag{2.7}$$

---

[1]Reprinted from RF Wireless World, n.d., Retrieved December 23, 2016, from `http://www.rfwireless-world.com/Terminology/Difference-between-Bit-and-Qubit.html`. Copyright 2012 by RF Wireless World. Reprinted with permission.

Matrix U must be unitary which means that a) its determinant is equal to unity and b) its Hermitian conjugate $U^\dagger$ must be equal to its inverse. These properties are expressed in Equ. 2.8 and 2.9 below. All quantum logic gates must be unitary since this preserves the normalization of the qubit state it is acting on.

$$a) \mid det(U) \mid = 1 \tag{2.8}$$
$$b) \quad UU^\dagger = U^\dagger U = \mathbb{1} = UU^{-1} = U^{-1}U \tag{2.9}$$

Using spherical polar coordinates, a single qubit can be visualized on the so-called Bloch sphere by parameterising α and β in Equ. 2.1 as follows:

$$|q\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle \tag{2.10}$$

The Bloch sphere has a radius of 1 and is therefore a unit sphere. The $|0\rangle$ qubit state is defined to lie along the positive z-axis ($\hat{z}$) and the $|1\rangle$ state is defined to lie along the negative z-axis ($-\hat{z}$) as labelled in Fig. 2.2. At this point, it is important to note that these two states are mutually orthogonal in $\mathcal{H}_2$ even though they are not orthogonal on the Bloch sphere.

Qubit states on the Bloch equator such as the $\hat{x}$ and $\hat{y}$ coordinate axes represent equal superpositions where $|0\rangle$ and $|1\rangle$ both have measurement probabilities equal to 0.5. The $\hat{x}$-axis for example represents the equal superposition $|q\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. As illustrated in Fig. 2.2 any arbitrary 2-D qubit state $|\psi\rangle$ can be decomposed into the polar angles θ and ϕ and visualized as a vector on the Bloch sphere. Such an object is called the Bloch vector of the qubit state $|\psi\rangle$. The Bloch sphere will be the main visualization tool for qubit manipulations in this thesis.
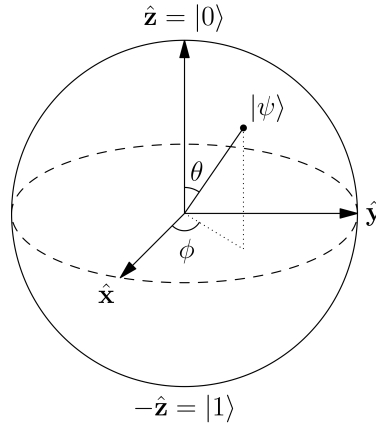

Figure 2.2: An arbitrary two-dimensional qubit $|\psi\rangle$ visualized on the Bloch sphere.[2]

2.1.2 Multiple Qubit Systems

When moving from single to multi qubit systems a new mathematical tool, the so-called tensor product (symbol $\otimes$), is needed. A tensor product of two qubits is written as:

$$|\psi\rangle \otimes |\psi\rangle = |0\rangle \otimes |0\rangle = |00\rangle \tag{2.11}$$

whereby the last expression omits the $\otimes$ symbol which is the shorthand form of a tensor product between two qubits.

In vector notation a tensor product of two vectors (red and green) is defined as shown below:

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 * \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 * \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.12}$$

The last expression in Equ. 2.12 shows that the two-qubit state $|00\rangle$ is no longer two but four-dimensional. Hence, it lives in a four-dimensional Hilbert space $\mathcal{H}_4$. A quantum gate acting on multiple qubits can therefore not have the same dimensions as a single-qubit gate (Equ. 2.7) which demands for a new gate formalism for multi qubit systems.

Consider wanting to apply an arbitrary single-qubit gate U (Equ. 2.7) to the first qubit and leaving the second qubit unchanged, essentially applying the identity matrix $\mathbb{1}$ to it. To do this, one defines the tensor product of two matrices as follows,

$$U \otimes \mathbb{1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} a * \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & b * \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\ c * \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & d * \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} a & 0 & b & 0 \\ 0 & a & 0 & b \\ c & 0 & d & 0 \\ 0 & c & 0 & d \end{pmatrix} \tag{2.13}$$

Thus, the result of the tensor product $U \otimes \mathbb{1}$ is a unitary 4x4 matrix which can now be used to linearly transform the 4x1 vector representing the $|00\rangle$ state in Equ. 2.12:

$$U \otimes \mathbb{1} |00\rangle = \begin{pmatrix} a & 0 & b & 0 \\ 0 & a & 0 & b \\ c & 0 & d & 0 \\ 0 & c & 0 & d \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} a \\ 0 \\ c \\ 0 \end{pmatrix} \tag{2.14}$$

One can also first perform the single qubit operations on the respective qubits followed by the tensor product of the two resulting vectors:

$$(U \otimes \mathbb{1})(|0\rangle \otimes |0\rangle) = U |0\rangle \otimes \mathbb{1} |0\rangle = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} a \\ c \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} a \\ 0 \\ c \\ 0 \end{pmatrix} \tag{2.15}$$

This formalism can be extended to any number of qubits and

Introduce entanglement and non-factorising tensor states here!

2.2   Quantum Logic Gates

In order to perform quantum computations, tools, analogous to the classical logic gates, are needed for qubit manipulation. Quantum logic gates are square matrices that can be visualized as rotations on the Bloch sphere. The following subsections will introduce the major single and multi qubit logic gates.

2.2.1   Single Qubit Gates

Qubit Flip (X) Gate

The quantum equivalent of the classical NOT logic gate is called X gate and is given by the 1st Pauli matrix:

$$\sigma_1 = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.16}$$

The action of a quantum gate on a qubit state can be analysed using the gate's matrix and the qubit's vector representation. Applying some straightforward linear algebra yields:

$$X \otimes |\psi\rangle = X \otimes (\alpha |0\rangle + \beta |1\rangle) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix} = \beta |0\rangle + \alpha |1\rangle \tag{2.17}$$

Thus, applying the X gate to qubit state $|\psi\rangle$ swaps the amplitudes of $|0\rangle$ and $|1\rangle$. More specifically, applying X to the $|0\rangle$ state results in the $|1\rangle$ state,

$$X \otimes |0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{2.18}$$

In terms of the example with the electron of a hydrogen atom shown in Fig. **??** consider the qubit initially being in state $|0\rangle$. Applying an X gate will flip the qubit into the $|1\rangle$ state which can be implemented in the lab by exciting the electron into the next highest electron valence shell using a controlled laser pulse.

The $|0\rangle$ state is recovered when applying X again to the $|1\rangle$ state,

$$X \otimes |1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \tag{2.19}$$

On the Bloch sphere, the X gate corresponds to a anti-clockwise π rotation around the x-axis as shown in Fig. 2.3.

From Equ. 2.17, 2.18 and 2.19 it follows that X is its own inverse as well as its own Hermitian conjugate:

$$XX = XX^{\dagger} = \mathbb{1} \tag{2.20}$$
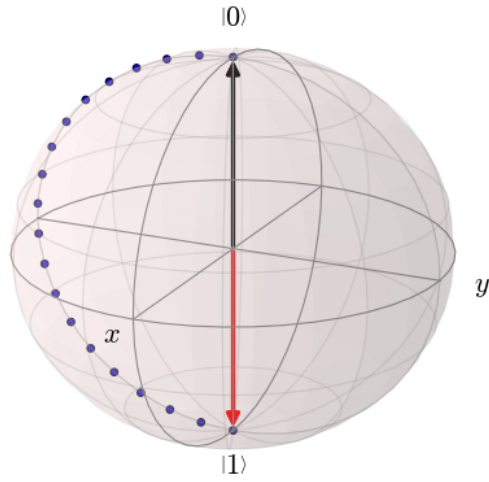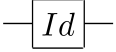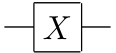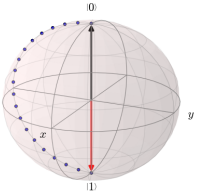
$$X = X^{-1} = X^{\dagger} \tag{2.21}$$
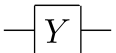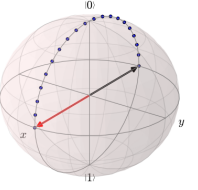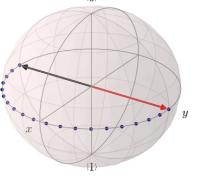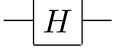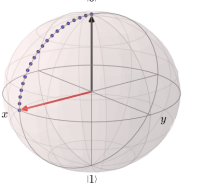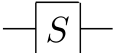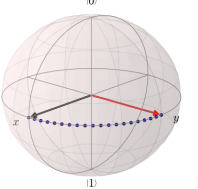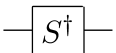
Figure 2.3: Visualization of the X gate on the Bloch sphere. Initial state (black) transformed to final state (red).

The action, circuit & matrix representation and Bloch sphere visualization for the most important single-qubit quantum logic gates are summarised in Table 2.1 below.

Table 2.1: Table of major single-qubit quantum logic gates.

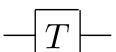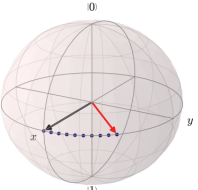| Gate	Name	Circuit representation	Matrix	Description	Rotation	Bloch sphere
$\mathbb{1}$	Identity	$-\boxed{Id}-$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	Idle or waiting gate	-	-
X	Qubit flip	$-\boxed{X}-$	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	Swaps amplitudes of $|0\rangle$ and $|1\rangle$	π rotation around $\hat{x}$	
Y	Qubit & phase flip	$-\boxed{Y}-$	$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$	Swaps amplitudes and introduces phase	π rotation around $\hat{y}$	
Z	Phase flip	$-\boxed{Z}-$	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	Adds a negative sign to the $|1\rangle$ state	π rotation around $\hat{z}$	
H	Hadamard	$-\boxed{H}-$	$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$	Creates equal superpositions	$\frac{\pi}{2}$ rotation around $\hat{y}$ and π rotation around $\hat{x}$	
S	$\frac{\pi}{2}$ rotation gate	$-\boxed{S}-$	$\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$	$\sqrt{Z}$	$\frac{\pi}{2}$ rotation around $\hat{z}$	
$S^\dagger$	$-\frac{\pi}{2}$ rotation gate	$-\boxed{S^\dagger}-$	$\begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix}$	Adjoint of S	$-\frac{\pi}{2}$ rotation around $\hat{z}$	
T	$\frac{\pi}{4}$ rotation gate	$-\boxed{T}-$	$\begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{pmatrix}$	$\sqrt{S}$	$\frac{\pi}{4}$ rotation around $\hat{z}$	
$T^\dagger$	$-\frac{\pi}{4}$ rotation gate	$-\boxed{T^\dagger}-$	$\begin{pmatrix} 1 & 0 \\ 0 & e^{-\frac{i\pi}{4}} \end{pmatrix}$	Adjoint of T	$-\frac{\pi}{4}$ rotation around $\hat{z}$	
ZM	Z-basis measurement	$-\boxed{\text{measure}}=$	-	Measurement in standard basis	Collapses the state	-

2.2.2   Multiple Qubit Gates

Controlled NOT Gate

The most important two-qubit quantum gate is the controlled NOT or CNOT gate given by the following 4x4 matrix:

$$CNOT = \begin{pmatrix} \mathbb{1} & 0 \\ 0 & X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.22}$$

The CNOT gate takes two qubits, control and target qubit, as input. If and only if the control qubit is in the $|1\rangle$ state, the NOT (X) gate is applied to the target qubit. In equations, the CNOT will always be followed by parantheses containing the control qubit followed by the target qubit (e.g. CNOT(0,1)). The input-output relation for the CNOT gate is given in the truth table 2.2 below.

Table 2.2: CNOT truth table with first qubit as control, second qubit as target.

Input	Output
00	00
01	01
10	11
11	10

To demonstrate the usefulness of the CNOT gate consider starting with two unentangled qubits both in the $|0\rangle$ state,

$$|\phi\rangle = |0\rangle \otimes |0\rangle = |00\rangle \tag{2.23}$$

Applying the H gate onto the first qubit yields the following (still unentangled) state:

$$(H \otimes \mathbb{1}) |\phi\rangle = (H \otimes \mathbb{1}) |00\rangle = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |10\rangle \tag{2.24}$$
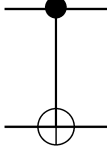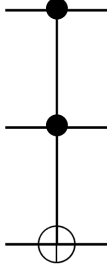
Now consider applying the CNOT to this state whereby the control qubit is coloured red and the target qubit green.

$$CNOT(0, 1) \otimes (\frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |10\rangle) = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} (\mathbb{1} \otimes X) |10\rangle = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle \tag{2.25}$$

The last expression in Equ. 2.25 is one of the famous Bell states which are four maximally entangled states. Thus, this example demonstrates how the CNOT gate is crucial for the generation of entangled states since it applies the X gate to a qubit depending on the state of a second qubit.

The three most important multi qubit quantum gates CNOT, Toffoli and nCNOT are characterised in Table 2.3.

Table 2.3: Table of major multi-qubit quantum logic gates where c_j stands for the j$^{\text{th}}$ control and $\mathbb{1}_k$ for the kxk identity matrix.

Gate	Name	Circuit representation	Matrix	Description
CNOT	Controlled NOT		$\begin{pmatrix} \mathbb{1} & 0 \\ 0 & X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$	CNOT(c_1, target)
Toffoli	Controlled controlled NOT		$\begin{pmatrix} \mathbb{1}_6 & 0 \\ 0 & X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$	CCNOT(c_1, c_2, target)
nCNOT	n-controlled NOT	-	$\begin{pmatrix} \mathbb{1}_{2^n-2} & 0 \\ 0 & X \end{pmatrix}$	nCNOT(c_1,..,c_n, target)

2.3   Classical Machine Learning

Introduce the three different types of ML!

2.3.1   k-nearest Neighbour Algorithm

In this thesis a quantum version of the distance weighted k-nearest neighbour (kNN) algorithm as described in (?, ?) will be implemented. Understanding the QML algorithm requires knowledge of the classical version of the distance weighted kNN algorithm which will be introduced in this subsection.

Imagine working for a search engine company and you are given the task of classifying unknown pictures of fruits as either apples or bananas. To train your classification algorithm you are given 5 pictures of apples and 5 pictures of bananas - this will be called the training data set D_T. Mathematically, the training data set D_T consists of ten vectors $v_0, v_1, .., v_{10}$ that are each assigned to either class A (apple) or B (banana). The training vectors are visualized as yellow and purple circles in Fig. 2.5.

kNN is a non-parametric classifier that given a new unclassified input vector x (red star in Fig. 2.5) 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. 2.5, vector x will be classified as class B (purple) but in the case $k = 6$ it will be labelled class A (yellow).

---

[3]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.

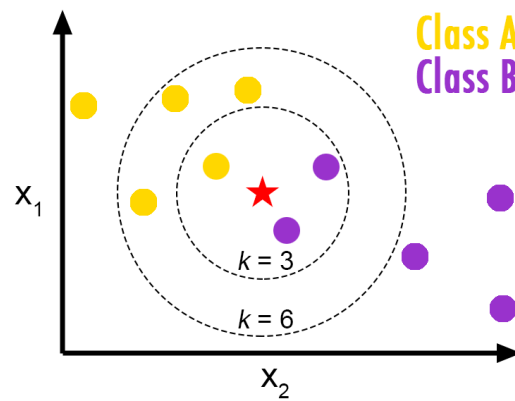Figure 2.4: Pictures of apple and banana from training data set.

Figure 2.5: Visualization of a kNN classifier[3]

In the case of $k = 10$, x would simply be assigned to the class with the most members. In this case, the training vectors should be given distance-dependent weights (such as $\frac{1}{distance}$) to increase the influence of closer vectors over more distant ones.

2.4 Methods (own chapter?)

2.4.1 Programming environment

Python Liquid

2.4.2 IBMs QC

IBM QASM

Chapter 3

Quantum-enhanced Machine Learning

3.0.1 Quantum k-nearest Neighbour Algorithm

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.

3.0.2 Quantum State Preparation Algorithms

Grover & Rudolph

Diffusion matrix from quantum random walks

Trugenberger et al.

Schack paper Maria mentioned but which I haven't used and looked at.

References

Nielsen, M. A., & Chuang, I. L. (2010). *Quantum computation and quantum information.* Cambridge university press.

Shende, V. V., & Markov, I. L. (2008). On the cnot-cost of toffoli gates. *arXiv preprint arXiv:0803.2316*.