NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCES
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

BSc THESIS

The Effects of Quantum Entanglement on Variational Quantum Classifiers

Dimitris C. Chamarias

Supervisors: **Dimitris Syvridis,** Professor
Aikaterini Mandilara, Doctor

ATHENS

MARCH 2024

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Οι Επιδράσεις της Κβαντικής Διεμπλοκής στους Μεταβλητούς Κβαντικούς Ταξινομητές

Δημήτρης Χ. Χαμαριάς

Επιβλέποντες: **Δημήτριος Συβρίδης,** Καθηγητής
Αικατερίνη Μανδηλαρά, Δόκτωρ

ΑΘΗΝΑ

ΜΆΡΤΙΟΣ 2024

BSc THESIS

The Effects of Quantum Entanglement on Variational Quantum Classifiers

Dimitris C. Chamarias
S.N.: 1115201600190

SUPERVISORS: **Dimitris Syvridis,** Professor
Aikaterini Mandilara, Doctor

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Οι Επιδράσεις της Κβαντικής Διεμπλοκής στους Μεταβλητούς Κβαντικούς Ταξινομητές

Δημήτρης Χ. Χαμαριάς
Α.Μ.: 1115201600190

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Δημήτριος Συβρίδης,** Καθηγητής
Αικατερίνη Μανδηλαρά, Δόκτωρ

ABSTRACT

Variational quantum classifiers (VQCs) are a type of machine learning model that leverage the principles of quantum mechanics to perform classification tasks. This thesis seeks to determine if entanglement may be utilized as a freely available resource to improve classification task performance using VQCs. With the ever-increasing interest in quantum computing and its potential applications in machine learning, understanding the role of entanglement in enhancing classification performance becomes imperative. The primary objective of this research is to explore how the presence of entanglement affects the accuracy and generalization capabilities of variational quantum classifiers. To achieve this, we employ the concept of global entanglement, which refers to the average entanglement between multiple qubits within a quantum system. By quantifying the amount of entanglement present in different quantum circuits, we can evaluate its impact on the classifier's performance. Finally, we present a case study to demonstrate the effectiveness of the proposed method. The findings of this research will contribute to the growing body of knowledge in quantum machine learning and ultimately aid in the development of more efficient and powerful quantum algorithms for classification tasks.

SUBJECT AREA: Quantum Computing, Machine Learning

KEYWORDS: quantum circuits, quantum entanglement, machine learning, classification, supervised learning, variational quantum classifier, global entanglement

ΠΕΡΙΛΗΨΗ

Οι μεταβλητοί κβαντικοί ταξινομητές (VQC) είναι ένας είδος μοντέλου μηχανικής μάθησης που αξιοποιεί τις αρχές της κβαντομηχανικής για την εκτέλεση εργασιών ταξινόμησης. Η παρούσα εργασία επιδιώκει να καθορίσει εάν η διεμπλοκή μπορεί να χρησιμοποιηθεί ως ελεύθερα διαθέσιμος πόρος για τη βελτίωση της απόδοσης εργασιών ταξινόμησης με τη χρήση VQCs. Με το συνεχώς αυξανόμενο ενδιαφέρον για την κβαντική πληροφορική και τις πιθανές εφαρμογές της στη μηχανική μάθηση, η κατανόηση του ρόλου της διεμπλοκής στην ενίσχυση της απόδοσης ταξινόμησης καθίσταται επιτακτική. Ο πρωταρχικός στόχος αυτής της έρευνας είναι να διερευνήσει πώς η παρουσία της διεμπλοκής επηρεάζει την ακρίβεια και τις δυνατότητες γενίκευσης των κβαντικών ταξινομητών μεταβλητής. Για να το επιτύχουμε αυτό, χρησιμοποιούμε την έννοια της καθολικής διεμπλοκής, η οποία αναφέρεται στη μέση διεμπλοκή μεταξύ πολλαπλών qubits εντός ενός κβαντικού συστήματος. Με την ποσοτικοποίηση της έκτασης της διεμπλοκής που υπάρχει σε διαφορετικά κβαντικά κυκλώματα, μπορούμε να αξιολογήσουμε την επίδρασή της στην απόδοση του ταξινομητή. Τέλος, παρουσιάζουμε μια μελέτη περίπτωσης για να αποδείξουμε την αποτελεσματικότητα της προτεινόμενης μεθόδου. Τα ευρήματα αυτής της έρευνας θα συμβάλουν στον αυξανόμενο όγκο γνώσεων στην κβαντική μηχανική μάθηση και τελικά θα βοηθήσουν στην ανάπτυξη πιο αποτελεσματικών και ισχυρών κβαντικών αλγορίθμων για εργασίες ταξινόμησης.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Κβαντική Υπολογιστική, Μηχανική Μάθηση

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: κβαντικά κυκλώματα, κβαντική διεμπλοκή, μηχανική μάθηση, κατηγοριοποίηση, επιβλεπόμενη μάθηση, μεταβλητός κβαντικός ταξινομητής, καθολική διεμπλοκή

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Overview

Quantum computers, the cutting-edge technology that harnesses the power of quantum mechanics, have been making waves in the scientific community. With their potential to solve some particular computational problems exponentially faster than classical computers, quantum computers hold the key to revolutionizing various fields, from cryptography to drug discovery. An exciting and active area of research within quantum computing is the development of quantum machine learning, and especially variational quantum classifiers, which are being used to solve classification problems. In this thesis, we will explore the concept of variational quantum classifiers and investigate the role of quantum entanglement in enhancing their performance.

Classification is a crucial aspect of machine learning, where algorithms are trained to classify data into distinct categories based on patterns and characteristics. Traditional machine learning algorithms, such as support vector machines and random forests, have made significant contributions to various fields. However, they may face limitations when dealing with large and complex datasets. This is where quantum computers come into play.

To grasp the significance of quantum computers in machine learning, it is essential to understand their fundamental principles. Traditional computers, known as classical computers, process data in binary form, using bits that represent either a 0 or a 1. In contrast, quantum computers use quantum bits, or qubits, which can represent a 0, a 1, or a superposition of both simultaneously. This unique property of qubits enables quantum computers to perform certain calculations that exponentially increase their computational power.

Variational quantum classifiers (VQCs) are a type of machine learning model that leverage the principles of quantum mechanics to perform classification tasks. Unlike classical machine learning algorithms, VQCs utilize quantum states and quantum gates to represent and manipulate data. These classifiers consist of a parameterized quantum circuit that is optimized to minimize a cost function, enabling them to learn from labeled training data and make predictions on unseen samples.

Quantum entanglement, a fundamental concept in quantum mechanics, allows particles to become correlated in such a way that the state of one particle is dependent on the state of another, regardless of the distance between them. This phenomenon has intrigued scientists for decades and has been harnessed for various applications in quantum computing.

In the context of variational quantum classifiers, quantum entanglement plays a crucial role in their capabilities. By entangling multiple qubits within the classifier's quantum circuit, the model can capture complex relationships between features and exploit quantum parallelism to process information more efficiently. This unique ability of quantum entanglement enables VQCs to potentially outperform classical machine learning algorithms in certain scenarios.

VQCs offer several advantages over their classical counterparts. Firstly, they have the potential to handle high-dimensional data more effectively. Traditional machine learning algorithms often struggle with high-dimensional feature spaces, leading to decreased accur-

acy and increased computational complexity. However, VQCs can leverage quantum superposition and entanglement to represent and process high-dimensional data efficiently, potentially improving classification accuracy. As researchers continue to explore and refine VQC techniques, we can expect further advances in this field, paving the way for new applications and breakthroughs in machine learning and quantum computing.

Despite these promising prospects, there are still challenges that need to be overcome before quantum computers become widely accessible. One major hurdle is the issue of qubit stability. Quantum computers rely on qubits, the basic units of quantum information, to perform calculations. However, qubits are extremely delicate and prone to errors caused by environmental disturbances. Developing error-correcting techniques and improving qubit stability are critical for the practical implementation of quantum computers.

In addition, the resources needed for quantum computers are substantial, encompassing both software and hardware. Sophisticated manufacturing methods and experience are needed to build a quantum computer that can solve complicated problems. It's also a constant struggle to create software tools and quantum algorithms that take full advantage of the capabilities of quantum computers. Quantum computing has the potential to revolutionize machine learning and open the door to ground-breaking solutions for a wide range of computational issues as we continue to push the frontiers of computing.

1.2 Structure and Objective

In an effort to guarantee readability and accessibility for readers who might not be familiar with quantum computing beforehand, this thesis is structured in a certain way. Background information is provided in the first two chapters. Beginning with the fundamental mathematical underpinnings of quantum computing in the first chapter, even readers with no prior experience with it can participate fully in the following chapters.

The theoretical underpinnings and characteristics of machine learning are covered in detail in the second chapter. By concentrating on variational quantum circuits and emphasizing the methods employed in the suggested implementation, the third chapter acts as a transition between the first two.

In-depth examination of the suggested VQC design and interpretations of the outcomes are covered in the fourth chapter. The fifth and last chapter, which comes after the simulation chapter, offers results from the investigation and suggests possible directions for further research and development. In the end, an appendix with details on the code implementation is also included.

This thesis' primary goal is to conduct a comprehensive investigation of whether entanglement can be used as readily available resource to enhance the performance of a classification task by means of a VQC.

2. QUANTUM COMPUTING

In this section, we will delve into the fundamental principles and mathematical formalism of quantum computation that will be needed in order to understand the results of this thesis. Firstly, we will explore the underlying concepts of quantum mechanics, like superposition and entanglement, which form the basis of quantum computation. Subsequently, the quantum circuit model will be introduced, which helps us visualize complex quantum computations. Finally, we will look at the applications of the density operator for systems with multiple qubits. We assume that the reader has a basic understanding of complex numbers, probability theory, and linear algebra. The sources used for this chapter are [1] [2] [3] unless otherwise explicitly cited.

2.1 Foundations of Quantum Computation

2.1.1 The Qubit

One of the fundamental building blocks of quantum computers is the qubit (short for quantum bit), which is the quantum mechanical analogue of a classical bit. Information is represented in bits in classical computing, with each bit having a possible value of either zero or one. In quantum computing, qubits are used to store and encode information. A qubit is a two-level quantum system where the two basis qubit states are usually written as $|0\rangle$ and $|1\rangle$. A qubit can be in any of the following states: $|0\rangle$, $|1\rangle$ or, in contrast to a classical bit, a superposition, a linear combination of both states.

In general, a two-dimensional quantum state (that is, a qubit), can be written in the following form:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

where α and β are probability amplitudes with complex values. The two basis states are written in Dirac or bra-ket notation and represent the column vectors $\begin{pmatrix}1\\0\end{pmatrix}$ and $\begin{pmatrix}0\\1\end{pmatrix}$ respectively. The "$|\cdot\rangle$" symbol is called a ket, and its dual partner, "$\langle\cdot|$", is called a bra. It is the Hermitian conjugate of its corresponding ket and represents a row vector, for example:

$$\langle\psi| = \begin{pmatrix}a^* & b^*\end{pmatrix}$$

After measuring a qubit, the only possible states that it can be in are the basis states. In our case, the states are $|0\rangle$ and $|1\rangle$. The choice to which of those two states the qubit is going to "collapse" is decided according to the Born rule, where the probability of finding the $|\psi\rangle$ in state $|0\rangle$ is equal to $|\alpha|^2$ and for state $|1\rangle$ it is equal to $|\beta|^2$. Since the squares of the coefficients of $|\psi\rangle$ represent the probability of the only two possible outcomes of a measurement, they must comply with the following condition:

$$|\alpha|^2 + |\beta|^2 = 1$$

If this condition is not met, we say that our state is not normalized. Another way of probing whether a state is not normalized, is to take the square root of the inner product of the state with itself, otherwise called the norm of the vector, and compare the result with the unity.

If it is equal to one, then the condition is met and the state is normalized. Consequently, the condition can also be written as:

$$||\psi|| = \sqrt{\langle\psi|\psi\rangle} = 1$$

We can compute the normalized version of our state, if it does not meet the previous criteria, by dividing it by its norm such that:

$$|\tilde{\psi}\rangle = \frac{|\psi\rangle}{||\psi||}$$

Lastly, a vector space coupled with an inner product is referred to as a Hilbert space. Thus, a qubit is a vector in a two-dimensional complex Hilbert space, or $\mathbb{C}^2$. A state vector can also be written, up to a global phase, in the following form:

$$|\psi\rangle = cos(\theta/2)\,|0\rangle + e^{i\phi}sin(\theta/2)\,|1\rangle$$

This form will help us visualize a qubit through, what is called, a Bloch sphere. Any two-level quantum state can be expressed as a point on the surface of the Bloch sphere with radius $r = 1$ and therefore can be described by two angles θ and ϕ. As long as the vector is normalized, the two parameters, θ and ϕ, are adequate to characterize a state. On the Bloch sphere, the standard basis states are situated at opposite locations. When $\theta = 0$, $|\psi\rangle = |0\rangle$ and when $\theta = \pi$, $|\psi\rangle = |1\rangle$.

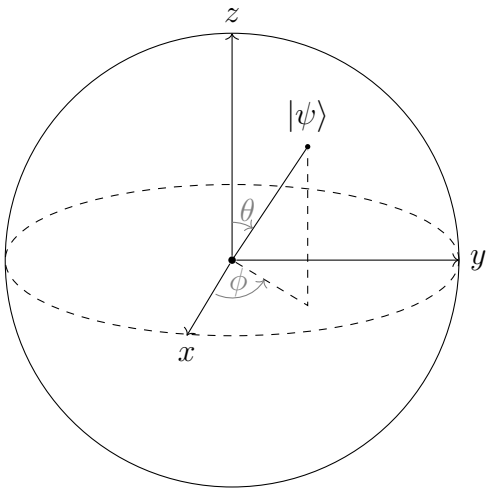

Figure 2.1: Bloch sphere representation of a qubit

2.1.2 Manipulating Qubits

Similarly to classical bits, we can transform qubits from one state to another. The tools employed for this task are called operators. Let A be an operator that acts on a state ψ. The result will be a new state:

$$\hat{A}\,|\psi\rangle = |\phi\rangle$$

The same holds true not only for ”ket”s, but also for ”bra”s:

$$\langle\psi|\,\hat{A} = \langle\phi|$$

Two of the most basic and simple operators are the identity operator, which leaves the state as it is, and the zero operator which transforms the state to the zero vector:

$$\hat{I}\left|\psi\right\rangle = \left|\psi\right\rangle$$

$$\hat{0}\left|\psi\right\rangle = 0$$

A class of operators that prove to be very significant in quantum computing are called Pauli operators. Included in these four operators is the identity one, which we saw previously. The first of the three remaining Pauli operators is usually referred to as the NOT operator, because it flips the basis states and is denoted as σ_1, σ_x, or X:

$$\hat{X}\left|0\right\rangle = \left|1\right\rangle,\ \hat{X}\left|1\right\rangle = \left|0\right\rangle$$

Next, we have the operator σ_2, σ_y, or Y that acts on the basis states as follows:

$$\hat{Y}\left|0\right\rangle = -i\left|1\right\rangle,\ \hat{Y}\left|1\right\rangle = i\left|0\right\rangle$$

And finally, we have σ_3, σ_z, or Z which acts as:

$$\hat{Z}\left|0\right\rangle = \left|0\right\rangle,\ \hat{Z}\left|1\right\rangle = -\left|1\right\rangle$$

Apart from its action on a basis state, an operator can be represented in other ways as well. One of these is through an outer product. An outer product between a ket and a bra is written as $\left|\psi\right\rangle\left\langle\phi\right|$. Then, if we apply this product to an arbitrary state, we have:

$$\left(\left|\psi\right\rangle\left\langle\phi\right|\right)\left|\chi\right\rangle = \left|\psi\right\rangle\left\langle\phi|\chi\right\rangle = \left\langle\phi|\chi\right\rangle\left|\psi\right\rangle$$

We can see that our original state $\left|\chi\right\rangle$ has been transformed to a new state that is proportional to $\left|\psi\right\rangle$, as the inner product $\left\langle\phi|\chi\right\rangle$ is just a complex value.

An additional way by which we can represent an operator is through a matrix. In an n-dimensional vector space, operators are represented by $n \times n$ matrices. For the purposes of this thesis, we are interested in operators that act on qubits, which "live" in a two-dimensional Hilbert space $\mathbb{C}^2$, so that means that the operators we are going to be looking at are 2×2 matrices.

To find each element of the matrix representation of an operator A, in the standard basis, we use the expression:

$$\hat{A} = \begin{pmatrix} \langle 0|A|0\rangle & \langle 0|A|1\rangle \\ \langle 1|A|0\rangle & \langle 1|A|1\rangle \end{pmatrix}$$

Doing the calculations for the Pauli matrices, we end up with the following:

$$\hat{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},\ \hat{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},\ \hat{Y} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix},\ \hat{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

In quantum theory and hence in quantum computation, two distinct kinds of operators are essential: Hermitian and unitary operators. An operator A is Hermitian if:

$$\hat{A} = \hat{A}^\dagger$$

$\hat{A}^\dagger$ is called the Hermitian adjoint and it is equal to the transpose of the matrix, but with each of its elements replaced by their complex conjugate. For an operator to be Hermitian, it

means that its diagonal matrix elements have real values. Another property of a Hermitian operator is that its eigenvalues are also real. Because of this characteristic, Hermitian operators in quantum physics represent physical observables.

For an operator to be unitary, its inverse needs to be equal to its Hermitian adjoint. The letter U is commonly used to represent unitary operators. So, by definition:

$$\hat{U}\hat{U}^\dagger = \hat{U}^\dagger\hat{U} = \hat{I}$$

Unitary operators are significant because they describe how a quantum state changes over time. The Pauli operators are both Hermitian and unitary.

Next, we will discuss the spectral decomposition theorem, which states that for every normal operator acting on a vector space there is a diagonal matrix representation with regard to some orthonormal basis of that space. An operator A is normal if it satisfies the following:

$$\hat{A}\hat{A}^\dagger = \hat{A}^\dagger\hat{A}$$

As a result of the theorem, we can express the operator in the form:

$$\hat{A} = \sum_i \lambda_i \left|u_i\right\rangle \left\langle u_i\right|$$

where λ_i are the eigenvalues of the operator and $|u_i\rangle$ is a basis.

Operators can also be utilized as function inputs. Using the Taylor series expansion, we get:

$$f(\hat{A}) = \sum_{n=0}^{\infty} a_i A^n$$

Thanks to the spectral decomposition theorem we can simplify formulas for functions of operators. If an operator A is normal and given its spectral decomposition, the previous formula transforms to:

$$f(\hat{A}) = \sum_i f(\lambda_i) \left|u_i\right\rangle \left\langle u_i\right|$$

Finally, due to the probabilistic nature of quantum mechanics, when measuring an observable, which is represented by an operator, instead of measuring once, we prepare a quantum state $|\psi\rangle$ several times, and then we average the measurement findings. This mean, also known as the expectation value of the operator, is written as:

$$\langle\hat{A}\rangle = \langle\psi|\hat{A}|\psi\rangle$$

2.1.3 Multiple Qubits

So far, we have only examined single qubits, their features and how to use them for simple calculations. Understanding how quantum mechanics operates for systems made up of several qubits interacting with one another is essential if we are to research more complex and potentially beneficial quantum computations. In order to accomplish that, we must be able to generate a Hilbert space consisting of each qubit's own independent Hilbert space. The mathematical equipment needed to carry out this task is known as the Kronecker or tensor product.

Let H_1 and H_2 be two Hilbert spaces of N_1 and N_2 dimensions respectively. By using the tensor product of those two spaces, we can construct a new composite Hilbert space H, of $N_1 N_2$ dimensions, such that:

$$H = H_1 \otimes H_2$$

In a similar fashion, we can use the tensor product to construct a state that belongs to H. If $|\phi\rangle \in H_1$ and $|\chi\rangle \in H_2$ are two state vectors members of the Hilbert spaces that were utilized to build H, then the following holds:

$$|\psi\rangle = |\phi\rangle \otimes |\chi\rangle$$

Given that spaces made up of two-dimensional subspaces are of interest to us, the subsequent steps are taken to calculate the product:

$$|\psi\rangle = |\phi\rangle \otimes |\chi\rangle = \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix}$$

Again, by employing the tensor product, we can construct a basis for our Hilbert space H. If $|u_i\rangle$ and $|v_i\rangle$ are the basis vectors of H_1 and H_2, then our new basis is the following:

$$|w_i\rangle = |u_i\rangle \otimes |v_i\rangle$$

Most of the time the "$\otimes$" symbol is omitted, so, for example, the tensor product $|\phi\rangle \otimes |\chi\rangle$ is written as $|\phi\rangle\,|\chi\rangle$, or even more concisely as $|\phi\chi\rangle$.

Additionally, we can also make operators that act on our composite system. Let $|\phi\rangle \in H_1$ and $|\chi\rangle \in H_2$, as well as operator A acting on $|\phi\rangle$ and operator B acting on $|\chi\rangle$. We can create an operator that is the tensor product of these two operators and that acts on our state $|\psi\rangle \in H$, as follows:

$$(A \otimes B)\,|\psi\rangle = (A \otimes B)(|\phi\rangle \otimes |\chi\rangle) = (A\,|\phi\rangle) \otimes (B\,|\chi\rangle)$$

Assuming that the operators A and B act on a two-dimensional Hilbert space, we can compute the matrix representation of their tensor product:

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B \\ a_{21}B & a_{22}B \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

2.1.4 Quantum Entanglement

It is crucial to keep in mind that, for example, a two-qubit composite system's state is not always able to be expressed in terms of a tensor product. The state can be expressed in product form if the two qubits are prepared separately and maintained in isolation, constituting a closed system for each one. Writing the state in the product form might not be feasible if the qubits are permitted to interact, as this would result in a closed system that consists of both qubits. In this situation, we refer to the qubits as being entangled, and they are a vital component of quantum computation.

Entanglement is a phenomenon that occurs not only in two-qubit but in multiple-qubits systems as well.

2.2 Quantum Circuit Model

In classical computers, the fundamental actions that can be performed on a bit of information are moving it from one location in memory to another, or we can process and transform that bit using what are known as logic gates. These logic gates can be combined to form digital circuits that enable us to carry out even more intricate calculations. The same is true for quantum computers, where any state transformations that act on qubits are called quantum gates, and a series of these gates constitute a quantum circuit.

Quantum gates are unitary operations on qubits; hence, gates and unitary operators are equivalent notions, and we will be alternating between these two terms from here on. Operators can be represented by matrices, thus for a quantum gate we need its appropriate unitary matrix. A gate with n inputs and outputs requires a $2^n \times 2^n$ matrix. So, for example, a single-qubit gate is represented as a 2×2 matrix and quantum gate that acts on two qubits is a 4×4 matrix.

The simplest single-qubit gate is one we are already familiar with. The quantum NOT gate, which flips the basis states, is just the X Pauli operator. By writing the matrix in the standard basis, we have:

$$\hat{U}_{NOT} = \hat{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

The other three Pauli operators, I, Y, Z, can also be used as a gate, since they are unitary. We have already seen how they operate on a qubit in Section 2.1.2.

A very important gate for quantum computing is the Hadamard gate, which puts both standard basis states in an even superposition. The Hadamard gate operates on the basis states in the following manner:

$$\hat{H}\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$$

$$\hat{H}\,|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle$$

Given the standard basis, we can write the Hadamard gate in the following matrix form:

$$\hat{H} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Let's now examine how to use exponentiation to generate more single-qubit gates. Given a matrix U that is unitary and Hermitian, we can easily prove with the help of the Taylor expansion that:

$$e^{-i\theta\hat{U}} = cos(\theta)\hat{U} - isin(\theta)\hat{U}$$

What that means is that by exponentiating a matrix we can construct a new gate. By exponentiating the Paul matrices, for example, we can create gates that rotate a qubit along the x, y and z axes on the Bloch sphere. These matrices are the following:

$$\hat{R}_x(\theta) = e^{-i\theta X/2} = \begin{pmatrix} cos(\theta/2) & -isin(\theta/2) \\ -isin(\theta/2) & cos(\theta/2) \end{pmatrix}$$

$$\hat{R}_y(\theta) = e^{-i\theta Y/2} = \begin{pmatrix} cos(\theta/2) & -sin(\theta/2) \\ sin(\theta/2) & cos(\theta/2) \end{pmatrix}$$

$$\hat{R}_z(\theta) = e^{-i\theta Z/2} = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}$$

Using the rotation matrices above, we can reduce any single-qubit gate to a sequence of rotations along the z and y axes, called a Z-Y decomposition. The z and y axes are not special, and we could have chosen any other two non-parallel axes on the Bloch sphere. Given an operator U, real numbers a, b, c, d exist such that:

$$\hat{U} = e^{ia}\hat{R}_z(b)\hat{R}_y(c)\hat{R}_z(d)$$

A circuit diagram can be used to illustrate how quantum gates operate. The input and output of each unitary operator or gate are represented by lines, often known as ”wires”, and the gate itself is represented by a rectangle. A sample graphical representation is depicted in Figure 2.2. We begin on the left side where we see the state of each qubit before the application of any gate. Then, each gate is applied one at a time, from left to right, until we arrive at the right side where our output is. In our example, in the beginning, a Hadamard and a Pauli-X gate is applied to the first and third qubit respectively. After that, a two-qubit U gate is applied to the first and second qubit, followed by a rotation on the z axis for the first qubit. At last, we perform a measurement on the first and third qubit, which is depicted as the meter in the rectangle.

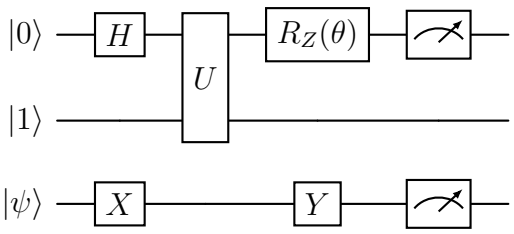

Figure 2.2: An example of a quantum circuit diagram

Now, we shall examine two-qubit gates, just like the one we saw in Figure 2.2. Gates with more than one qubit as input are also known as controlled gates, because they are comprised of control and target qubits, in no particular order, and in our case for two-qubit gates, one control and one target qubit. The target qubit experiences no changes if the control qubit is zero; however, if the control qubit is equal to one, a unitary operation is applied to the target. One may compute the matrix representation of a two-qubit gate using the following 4×4 matrix:

$$\hat{U} = \begin{pmatrix} \langle 00|\hat{U}|00\rangle & \langle 00|\hat{U}|01\rangle & \langle 00|\hat{U}|10\rangle & \langle 00|\hat{U}|11\rangle \\ \langle 01|\hat{U}|00\rangle & \langle 01|\hat{U}|01\rangle & \langle 01|\hat{U}|10\rangle & \langle 01|\hat{U}|11\rangle \\ \langle 10|\hat{U}|00\rangle & \langle 10|\hat{U}|01\rangle & \langle 10|\hat{U}|10\rangle & \langle 10|\hat{U}|11\rangle \\ \langle 11|\hat{U}|00\rangle & \langle 11|\hat{U}|01\rangle & \langle 11|\hat{U}|10\rangle & \langle 11|\hat{U}|11\rangle \end{pmatrix}$$

The two-qubit gate that we need to know for the objectives of this thesis is the $controlled-NOT$ or $CNOT$ gate. If the control qubit is $|0\rangle$, then the target qubit stays as is, but if the control qubit is $|1\rangle$ then the target flips its value, just as if a gate NOT had been applied to it. The actions of a $CNOT$ gate to all possible input states are as follows:

$$CNOT\,|00\rangle = |00\rangle$$

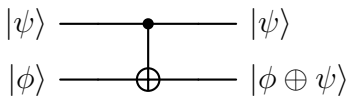

Figure 2.3: Circuit diagram representation of a CNOT gate

$$CNOT\,|01\rangle = |01\rangle$$
$$CNOT\,|10\rangle = |11\rangle$$
$$CNOT\,|11\rangle = |10\rangle$$

Knowing the matrix representation of a two-qubit gate and the actions of a $CNOT$ gate on the two qubits, it follows that the $CNOT$ matrix is equal to:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

2.3 Density Operator

We have always assumed in the previous sections that a system's state has an exact state vector. That is, if an orthonormal basis $|u_i\rangle$ exists, we can write our state vector in that basis as:

$$|\psi\rangle = c_1\,|u_1\rangle + c_2\,|u_2\rangle + ... + c_n\,|u_n\rangle$$

The likelihood that the system will be in state $|u_i\rangle$ at measurement is thus provided by $|c_i|^2$, as determined by the Born rule. The term used to describe such a state is "pure".

Often, we find that we are only interested in or have access to a small portion of a broader system. One way to illustrate this is by considering an EPR pair example. Let's assume the system in question is equal to the Bell state:

$$|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

where Alice is in possession of one qubit and Bob is in possession of the other and they separate to completely opposite directions. That state contains information about the whole entangled system, but Alice has no way of describing her own qubit since it is not in the usual form of $a\,|0\rangle + b\,|1\rangle$. The density operator formulation is a valuable tool for characterizing the state of a composite system's subsystem, which is referred to as a mixed state.

When considering a pure state $|\psi\rangle$, the definition of the density operator is:

$$\rho = |\psi\rangle\,\langle\psi|$$

Returning to our example, the density operator for the composite system is:

$$\rho = |\beta_{00}\rangle\,\langle\beta_{00}| = \frac{1}{2}(|00\rangle\,\langle 00| + |00\rangle\,\langle 11| + |11\rangle\,\langle 00| + |11\rangle\,\langle 11|)$$

Using the density operator, we can define a reduced density operator for Alice, which is defined through the partial trace operation:

$$\rho_A = Tr_B(\rho)$$

The partial trace over Bob's system acts on the basis states in the following manner:

$$Tr_B(|a_1\rangle \langle a_2| \otimes |b_1\rangle \langle b_2|) = |a_1\rangle \langle a_2| Tr(|b_1\rangle \langle b_2|)$$

Using the property above, we calculate Alice's reduced density operator, by tracing out Bob's system:

$$\rho_A = Tr_B(\rho) = \frac{1}{2}(|0\rangle \langle 0| + |1\rangle \langle 1|)$$

All the pertinent information about Alice's system is contained in the partial trace $Tr_B(\rho)$, and the same is true for Bob's system, in $Tr_A(\rho)$. But there is not enough information in these local descriptions to recreate the system's overall state.

One scenario where density operators are also useful is when we want to quantify the degree of entanglement between two qubits. Figuring out the concurrence of a state is one method of measuring entanglement. Concurrence between two qubits is defined as [4]:

$$C(\rho) = max(0, \lambda_1 - \lambda_2 - \lambda_3 - \lambda_4)$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are the eigenvalues,in decreasing order, of the following matrix R:

$$R = \sqrt{\sqrt{\rho}\tilde{\rho}\sqrt{\rho}}$$

$$\tilde{\rho} = (\sigma_y \otimes \sigma_y)\rho^*(\sigma_y \otimes \sigma_y)$$

The value of concurrence ranges from zero (no entanglement at all) to one (maximally entangled state).

We can generalize the concurrence measure for systems with multiple qubits. Suppose that we are interested in measuring the entanglement of a subsystem of m qubits, that is part of a larger composite system of n qubits. Let $\mathcal{M}$ denote this collection of qubits. Then, we can rewrite concurrence as [5]:

$$C_{\mathcal{M}}(\rho) = \sqrt{2(1 - Tr\rho_{\mathcal{M}}^2)}$$

where $\rho_{\mathcal{M}}$ is the reduced density matrix of the subsystem $\mathcal{M}$.

3. MACHINE LEARNING

In this part, we shall explore the basic ideas behind machine learning as a field, in order to later synthesize them with our understanding of quantum computation, so we can analyze the topic of our thesis, which is VQCs. First, we start with a brief introduction to machine learning. Subsequently, the problem of classification is analyzed and how we can solve it using gradient descent on a loss function.

3.1 Introduction

Generally speaking, there are two primary types of machine learning: unsupervised and supervised learning [6] [7].

The goal of unsupervised learning is to find structure and patterns in unlabeled data. ”Unsupervised” refers to the fact that these algorithms find patterns in data that are concealed and do not require human ”supervision” or assistance. Unsupervised learning methods are typically employed for three tasks: clustering, anomaly detection and dimensionality reduction.

The main characteristic of supervised learning as a machine learning technique is the usage of labeled datasets. The model can learn over time due to the training dataset, which contains inputs paired with the correct outputs. Through, what is called, a loss function, the algorithm gauges its accuracy and makes adjustments until the error is suitably reduced. Regression and classification are the two main subcategories of tasks in supervised learning. We consider the classification task that is relevant for this thesis.

3.2 Classification

Classification algorithms aim to precisely classify test data into distinct groups, for example, to find out whether a dog or a cat is depicted in an image or if an email is spam or not. Classification may be separated into to two categories: binary and multi-class classification. As the names suggest, in binary classification there are only two categories for the model to choose from, whereas in multi-class classification there are more than two.

To be more specific, given an input domain X, an output domain Y and a dataset D of pairs $(x^m, y^m) \in X \times Y, m = 1, ..., M$, in addition to a new unclassified input $x \in X$, the goal is to estimate the matching output $y \in Y$. In order to solve this problem, we create a model family $\{f_\theta\}$, which may be represented mathematically as a collection of functions f that translate inputs from X to outputs from Y. These models are based on a set of parameters θ that identify a specific model within the family. Then, the optimal model, together with its distinct parameter set of θ, which replicates the data, is chosen from this set of functions.

A loss or cost function that calculates the difference between the model’s prediction $f_\theta(x)$ for an input x and the intended output y defines the concept of ”best model”. The model that minimizes the expected loss, that is, the average loss over all possible data, is the

best. If $L(f_\theta(x), y)$ is our loss function, then the best parameters are [6] [7]:

$$\theta^* = \min_\theta \frac{1}{M} \sum_{i=1}^{M} L(f_\theta(x_i), y_i)$$

Machine learning poses a problem in that the optimal model must be chosen with limited access to data - a finite number of instances are provided. In other words, finding a model that can generalize from the small sample size to the entirety of the data domain.

3.2.1 Loss Function

Among the several loss functions available, the most straightforward one for classification tasks is the one that depends on a model's accuracy, or the percentage of samples that have been properly categorized. If accuracy is defined as [6]:

$$Accuracy = \frac{C}{T}$$

where C are the number of correctly categorized examples and T is the total number of examples, then our loss function will be equal to [6]:

$$L(f_\theta(x), y) = 1 - \frac{C_\theta}{T}$$

Nevertheless, the majority of training methods depend on a continuous-valued loss, which is essential for calculating gradients, a topic that we will cover in Section 3.2.2 on optimization. The squared Euclidean distance between the prediction and the target, referred to as the mean squared error loss, is one of the most widely used continuous loss functions [6] [7]:

$$L(f_\theta(x), y) = (f_\theta(x) - y)^2$$

The most often used loss function for classification, and the one we will use in this thesis, is the cross-entropy loss function [6]:

$$L(f_\theta(x), y) = -\sum_{d=1}^{D} y_d log p_d$$

where D is the number of classes, y_d is a binary value that indicates whether the true of class of x is class i and p_d is the probability of x being of class i.

3.2.2 Optimization

Optimizers are algorithms that are used in machine learning to adjust a model's parameters in an iterative fashion as a means to minimize the loss function.

If C is our cost function and θ its parameters, then the parameters are updated iteratively by gradient descent using the following rule [6] [7]:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \nabla C(\theta^{(t)})$$

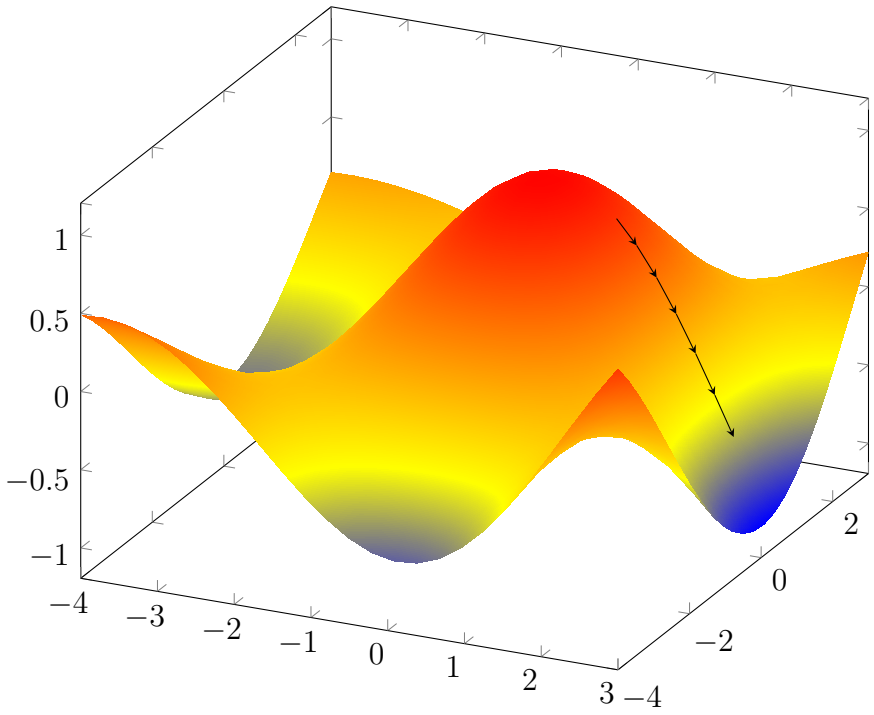

Figure 3.1: 3D plot of gradient descent on a function

where η is a hyperparameter known as the learning rate and t is an integer that indicates the number of iterations completed thus far. A hyperparameter is a parameter that is chosen before the execution of the training process. The learning rate, basically, dictates the pace at which a minimum of the function, local or global, is reached. Its value is usually a small number that is updated and assessed in accordance with the cost function's behavior. Greater learning rates lead to bigger leaps, but there's a chance they'll exceed the minimum, but on the other hand, lower learning rates take longer to converge to a minimum. In the landscape of the cost function, the gradient points in the direction of ascent, so following its negative means heading in the direction of valleys, until the gradient is zero, as shown in Figure 3.1.

The gradient descent method is applied in three different ways [6] [7]. The first is called batch gradient descent (BGD). After evaluating every training sample, a procedure known as a training epoch, batch gradient descent adds up the errors for every point in a training set and updates the model. Despite the fact that batching reduces computing costs, processing large training datasets may still take a while.

The second variation of the gradient descent algorithm is called stochastic gradient descent (SGD). In SGD, every example in the dataset undergoes a training epoch, and each training example's parameters are updated one at a time. Compared to BGD, these frequent updates may result in losses in computing efficiency although they can provide greater detail.

And last, we have mini-batch gradient descent where ideas from both BGD and SGD are combined. It divides the training dataset into smaller batches, and performs updates on each of those batches. This method finds a compromise between batch gradient descent's computing efficiency and stochastic gradient descent's speed.

4. VARIATIONAL QUANTUM CLASSIFIER

4.1 Introduction

A tremendous amount of effort has gone into creating the earliest iterations of what might eventually become a complete quantum computer, known as Noisy Intermediate-Scale Quantum (NISQ) devices. These devices lack error correction and can only yield approximations of computing results since they currently consist of around 50–100 qubits, not all of which interact with each other. In theory, quantum devices in the NISQ era might be used to explore the benefits of quantum computing; nevertheless, quantum algorithmic design is significantly impacted by the requirement to restrict algorithms to a small number of qubits and gates. In order to address the aforementioned problem, a brand-new category of algorithms known as variational quantum algorithms was created [8].

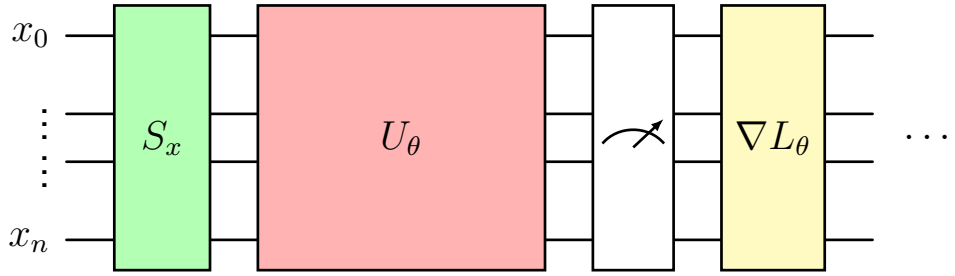

Figure 4.1: General structure of a variational quantum circuit

These algorithms are not specified by a predetermined sequence of gates, but rather by an ansatz, a pattern that determines which gates are applied to which qubits. The ansatz template is modular and may be applied to varying quantities of qubits by repeating it in layers. Certain gates in the ansatz, typically Pauli rotations, depend on freely adjustable parameters, which is where the term ”variational” originates. An optimization procedure that is classical in nature and optimizes a cost function obtained from the problem at hand will select the parameters.

Figure 4.1 illustrates a variational quantum circuit in the form of a diagram. We begin with an input vector x with n features. First, we need to prepare our classical input so it can be processed by the quantum circuit. This procedure is called state preparation, which is represented by the S_x block. Then, we apply unitary transformations to our input, using a mixture of parameterized and non-parameterized gates, which is the U_θ block. After that, our system is ready for measurement. Using the output accordingly and passing it to a classical optimizer, we can calculate the new parameters. Finally, we adjust the parameters of the U_θ block and this process is repeated until the optimal solution is reached.

Variational quantum circuits can be thought of as machine learning models [6] [8]. Applying a quantum circuit $U(x, \theta)$ that depends on both the input x and the parameters θ to the initial state, allows us to view a variational quantum circuit as a machine learning model, where the output of that model is nothing more than a function of the expectation value of an observable O. In the case of a classification task, the expectation value $\langle O \rangle \in \mathbb{R}$, can be interpreted as the probability of our input being in a certain class or category, and the difference between the true label and the expectation value is then used to define the cost function [8] [9].

4.2 Input Encoding

The choice of how to represent data, whether it be single inputs or large datasets, by quantum states must be made if we wish to utilize a quantum computer to learn from classical data. In this section, we will explain various pertinent methods for encoding or embedding information into an n-qubit system's quantum state. There are primarily four ways to encode classical data in a quantum state [6]: basis encoding, amplitude encoding, time-evolution encoding and Hamiltonian encoding. We will concentrate on two of this methods for this thesis: amplitude encoding and time-evolution encoding.

Suppose that we want to encode a single input with N features, so $x \in \mathbb{C}^N$, into the amplitudes of a state. First of all, since every qubit has two amplitudes, in a n-qubit system the amplitudes become 2^n, which means that the number of features must be $N = 2^n$. If that is not the case, then our state needs to be padded with some value. Next, a state would need to be normalized before it could be regarded as valid. Therefore, the second step we need to take is to normalize our input so that $\sum_{i=0}^{2^n-1} |x_i|^2 = 1$. Once we have completed all of the above steps, we can finally encode our input in the following manner [6]:

$$|\psi_x\rangle = \sum_{j=0}^{2^n-1} x_j \, |j\rangle$$

On to the second encoding, time-evolution encoding, according to which, a Hamiltonian H relates a scalar value $x \in \mathbb{R}$ with the time t in the unitary evolution of the quantum state [6]:

$$U(x) = e^{-ixH}$$

The most widely used subcategory of the time-evolution encoding is referred to as rotation encoding or angle encoding, with the Pauli rotation gates being utilized as the unitary transformation. A gate is applied to each qubit; one qubit for each feature. Suppose that our input has N features, just like in the previous example, but $x \in \mathbb{R}^N$ this time. First, we are going to normalize our input so that each feature is in the interval $[0, 2\pi]$. If we wish to apply a Pauli Y rotation gate to our input, the unitary transformation becomes [6]:

$$U(x) = R_y(x_1) \otimes \cdots \otimes R_y(x_N)$$

If all of the N qubits of our system are initialized to $|0\rangle$, then the final state is encoded as [10]:

$$|\psi_x\rangle = \begin{pmatrix} cos(x_0/2) \\ sin(x_0/2) \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} cos(x_n/2) \\ sin(x_n/2) \end{pmatrix}$$

4.3 Model Training

Determining the values of θ that minimize a cost function that depends on input data is the process of training a variational quantum model. As we saw in Section 3.2.2, for this to be achieved, we need to compute the gradient of the cost function C_θ with respect to its

parameters $\theta = \{\theta_1, \ldots, \theta_K\}$ [6], which by definition is equal to:

$$\nabla C(\theta) = \begin{pmatrix} \frac{\partial C(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial C(\theta)}{\partial \theta_K} \end{pmatrix}$$

Each partial derivative with respect to a parameter θ_i is:

$$\frac{\partial C(\theta)}{\partial \theta_i} = \lim_{h \to 0} \frac{C(\theta_1, \ldots, \theta_i + h, \ldots, \theta_K) - C(\theta)}{h}$$

We can approximate these derivatives by choosing an infinitesimal value that is close to zero $\Delta\theta$ in place of h:

$$\frac{\partial C(\theta)}{\partial \theta_i} \approx \frac{C(\theta_1, \ldots, \theta_i + \Delta\theta, \ldots, \theta_K) - C(\theta)}{\Delta\theta}$$

4.4 Measuring Entanglement

In Section 2.3, we talked about how we can quantify the amount of entanglement of a single qubit that is a component of a larger composite system. The purpose of this thesis is to investigate, using a VQC, whether entanglement is associated with improved performance in a classification problem. It is therefore necessary to quantify the total quantum circuit's entanglement. We use a metric known as global entanglement to do that.

For a system of n qubits, the global entanglement measure is defined as [11] [12]:

$$Q(\psi) = \frac{1}{n} \sum_{i=1}^{n} C_i^2(\psi)$$

where C_i is the concurrence of the i-th qubit, which we discussed in Section 2.3. The value of the global entanglement measure ranges from zero to one, similarly to concurrence, which allows us to compare the entanglement of circuits with different numbers of qubits.

5. SIMULATION AND RESULTS

5.1 Setup

The VQC was implemented as a quantum circuit in the Python programming language with the help of the PyTorch and NumPy libraries. More information can be found in the Appendix A.

5.1.1 Datasets and Metrics

The datasets used to evaluate the effects of entanglement on VQCs are the IRIS dataset and a generated concentric circles dataset. The IRIS dataset is possibly one of the most well-known datasets in the machine learning space. It contains measurements of sepal length, sepal width, petal length, and petal width for three different species of iris flowers, which means that each input vector has 4 features with 3 classes to be predicted by the model. As for the concentric circles dataset, each data point has 2 features with 2 classes available for prediction. Therefore, the performance of the models will be tested on both binary and multi-class classification tasks.

In order to assess the impact of entanglement on the classifier, we will juxtapose the global entanglement with the accuracy of the model on the previously mentioned datasets.

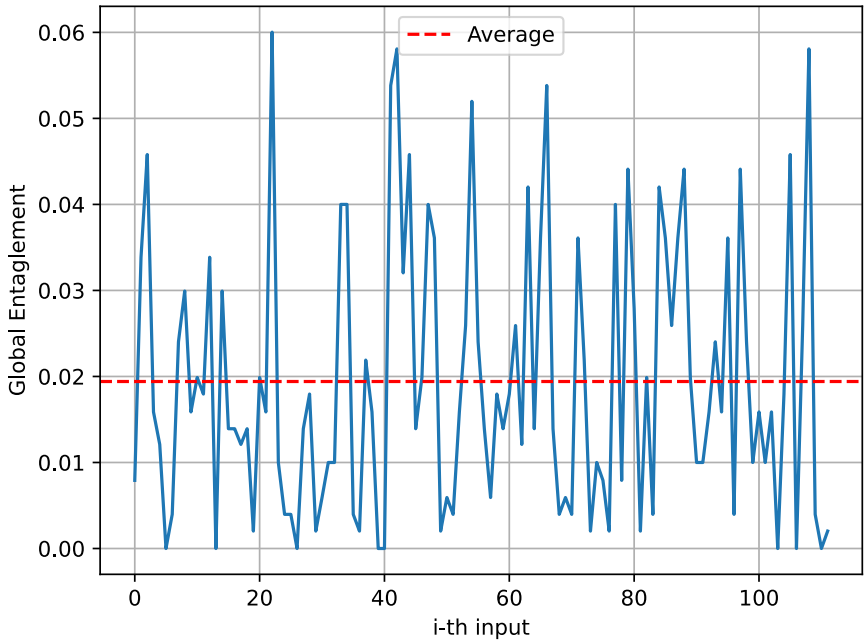

Figure 5.1: Global Entanglement of classifier for IRIS dataset

As shown in Figure 5.1, the global entanglement measure is dependent upon the input of the classifier and fluctuates significantly for each data point of a dataset (for the depiction, the IRIS dataset was used as an example). As a result, to calculate the total entanglement of our circuit, we simply take the average of all the global entanglement measures for the dataset and use it as our metric.

5.1.2 Ansatz Design

Various combinations of encodings, as well as, rotation and entangling gates were investigated for the overall purposes of this thesis. The encodings employed in the simulation are the amplitude and rotation encodings, which were covered in Section 4.2. For single-qubit gates, the Pauli Y and Z rotation gates were used, with the addition of the Hadamard gate that was used for superposition and entanglement. The only two-qubit gate utilized was the CNOT gate.

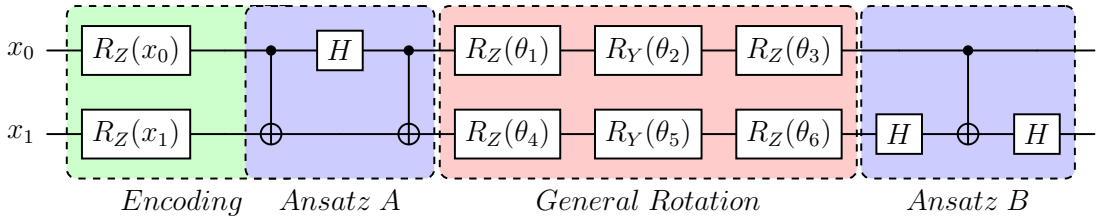

Figure 5.2: General structure of the ansatz patterns used

In general, the classifier is configured as follows: After encoding the input using the established methods, the first ansatz uses a combination of single and two-qubit gates to calibrate the model's entanglement. Next comes the variational portion of the classifier, where we rotate each qubit in a certain direction using a set of trainable parameters θ, which will be optimized iteratively. Then, the second ansatz follows, which is comparable to the first. Finally, we measure our system to get the prediction. For a visual representation, see Figure 5.2 and for one of the circuits used in the simulation, see Figure 5.3

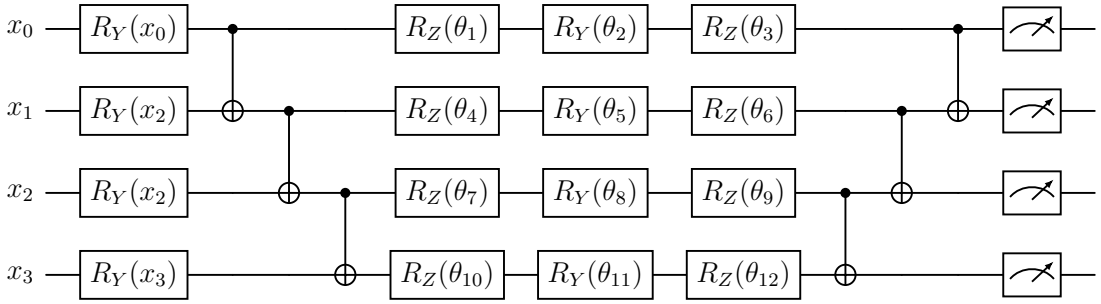

Figure 5.3: Example of an ansatz pattern

5.2 Results

We begin with the results for the IRIS dataset. Figure 5.4 shows a clear trend of decreasing accuracy, although the correlation seems to be weak. When entanglement is zero our model performs poorly. The same can be said about increased levels of entanglement. The most striking result to emerge from Figure 5.4 is that, when entanglement is just above zero, more specifically about 0.02, the accuracy of the model shoots up to almost 90%. Amplitude encoding was the encoding used for this specific model; however, given the other models with the same encoding fared badly, it is extremely improbable that the rise was caused by the specific encoding.

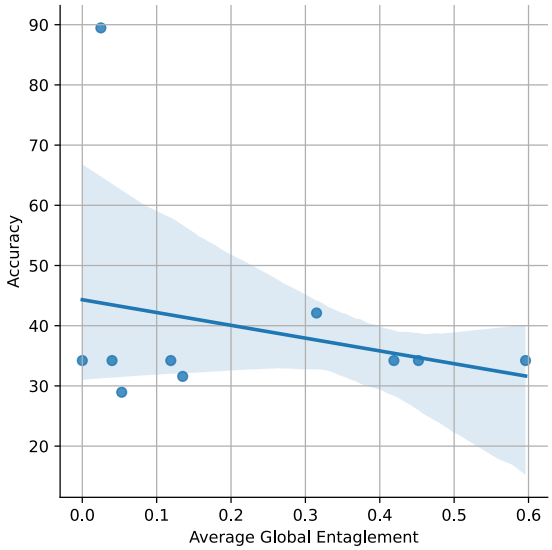

Figure 5.4: Relationship between global entanglement and accuracy for the IRIS dataset

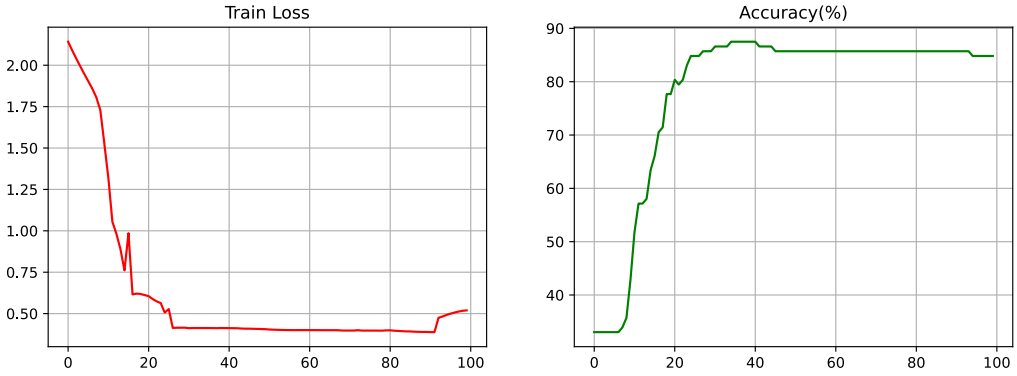

Figure 5.5: Learning curve with best results for IRIS dataset

The same slight downward trend is seen for the concentric circles dataset, as illustrated in Figure 5.6. Compared to the IRIS dataset, here, the results are a bit more balanced. The model that performed the best had approximately 0.45 entanglement reaching 65% accuracy, which is not ideal, with the second best model having zero entanglement but being not far behind, scoring 62%.

One interesting aspect that emerged from the analysis, that both plots seem to have in common, is that there is an outlier in the data, performing much better than the others. A conclusion we can draw from this fact is that, even though improved performance cannot simply be ascribed to the effects of entanglement, there exists a ”sweet spot” of entanglement, which is different for each dataset. When a certain level is reached, the model can improve drastically.

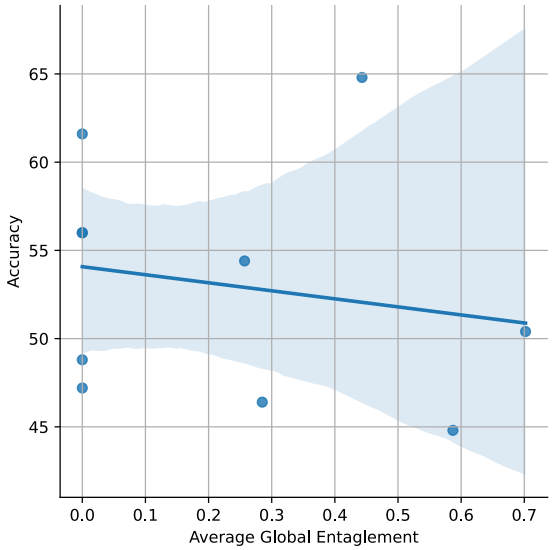

Figure 5.6: Relationship between global entanglement and accuracy for the Concentric Circles dataset

Having said that, extreme caution must be exercised in interpreting these data, as the combinations of different encodings, gates and ansatz patterns were limited. But all things considered, these findings seem to point in the direction that entanglement is not a fool-proof way to boost a VQC's performance.

6. CONCLUSIONS AND FUTURE WORK

In conclusion, this thesis explored the effects of entanglement on VQCs. Through a comprehensive analysis of the experimental results, it has been determined that there appears to be no significant correlation between the extent of entanglement and the accuracy of the classifier. These findings challenge the notion that entanglement plays a crucial role in enhancing classifier performance.

However, it is important to note that this conclusion is based on the specific datasets and parameters utilized in this study. To establish a more definitive understanding, future research should focus on employing larger and more diverse datasets. Additionally, investigating the impact of varying entanglement measures and exploring alternative quantum architectures may provide valuable insights.

The implications of this research extend beyond the scope of quantum machine learning, as it prompts further investigation into the fundamental role of entanglement in quantum information processing. It is pertinent for researchers to collaborate and share their findings to collectively advance the field. The results presented in this thesis contribute to the existing body of knowledge surrounding variational quantum classifiers and pave the way for future investigations in this domain.

ABBREVIATIONS - ACRONYMS

VQC	Variational Quantum Classifier
CNOT	Controlled-NOT
BGD	Batch Gradient Descent
SGD	Stochastic Gradient Descent
NISQ	Noisy Intermediate-Scale Quantum

APPENDIX A. CODE IMPLEMENTATION

The full code implementation, as well as all the simulation results, can be found in the GitHub repository:

`https://github.com/DImiTrisXam/bsc_thesis`

BIBLIOGRAPHY

[1] David McMahon. *Quantum computing Explained*. John Wiley & Sons, 2007.

[2] Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An Introduction to Quantum Computing*. OUP Oxford, 2007.

[3] Eleanor G Rieffel and Wolfgang H Polak. *Quantum computing: A Gentle Introduction*. MIT press, 2011.

[4] William K. Wootters. Entanglement of formation of an arbitrary state of two qubits. *Phys. Rev. Lett.*, 80:2245–2248, Mar 1998.

[5] Vineeth S. Bhaskara and Prasanta K. Panigrahi. Generalized concurrence measure for faithful quantification of multiparticle pure state entanglement using lagrange's identity and wedge product. *Quantum Information Processing*, 16(5):118, Mar 2017.

[6] Maria Schuld and Francesco Petruccione. *Machine Learning with Quantum Computers*. Springer International Publishing, Cham, 2021.

[7] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[8] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, Sep 2021.

[9] Maria Schuld, Alex Bocharov, Krysta M. Svore, and Nathan Wiebe. Circuit-centric quantum classifiers. *Phys. Rev. A*, 101:032308, Mar 2020.

[10] Manuela Weigold, Johanna Barzen, Frank Leymann, and Marie Salm. Expanding data encoding patterns for quantum algorithms. In *2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)*, pages 95–101, 2021.

[11] Gavin K. Brennen. An observable measure of entanglement for pure states of multi-qubit systems. *Quantum Info. Comput.*, 3(6):619–626, Nov 2003.

[12] Peter J. Love, Alec Maassen van den Brink, A. Yu. Smirnov, M. H. S. Amin, M. Grajcar, E. Il'ichev, A. Izmalkov, and A. M. Zagoskin. A characterization of global entanglement. *Quantum Information Processing*, 6(3):187–195, Jun 2007.