Imperial College
London

MSc Individual Project

Imperial College London

Department of Computing

Quantum Autoencoder
for Product State Compression

Supervisors:
Dr. Tobias Haug
Dr. Paul Bilokon

Author:
Adithya Sireesh

Second Marker:
Prof. Herbert Wiklicky

Submitted in partial fulfillment of the requirements for the MSc degree in Advanced Computing of Imperial College London

February 13, 2024

Abstract

A Quantum Autoencoder is an unsupervised variational quantum algorithm that learns to compress quantum n qubit states down to a $k \leq n$ qubits. We propose a variant of the quantum autoencoder, called the **Product State Autoencoder**, that is capable of compressing a set of quantum states to product states. In the context of teleportation of an $n > 1$ qubit state $|\psi\rangle$, using the product state autoencoder provides the benefit of requiring only $\mathcal{O}(\log n)$ repetitions for successful teleportation as compared to the $\mathcal{O}(\exp n)$ repetitions required in the standard teleportation scheme. We further extend the capacity of the product state enhanced teleportation by forcing the trash states qubits of the quantum autoencoder to occupy known basis states that can be communicated classically. Through simulations of training for $n \leq 4$ qubits, we analyse and report the required number of training states and parameters to achieve successful convergence of an n qubit product state autoencoder.

Acknowledgements

Firstly, I would like to thank Dr Tobias Haug for allowing me to work on this project with him. With his extraordinary guidance, support and useful advice I deepened my understanding of the principles of quantum computing and throughout this project became a better researcher.

Secondly, I would like to give my gratitude to Dr Paul Bilokon for his time, regular check-ups as well as opportunity to give a talk at a conference organized by him. Explaining the fundamental concepts of quantum computing enabled me to understands the concepts better myself.

I am also very grateful to my friends and colleagues, discussions with whom inspired me.

Last but not least, I would like to thank my family for their support and kind words.

Summary

Chapter 1

Introduction

The manipulation and transfer of units of information is the basis of modern technology, which has led to the creation of computers and the internet. The recent decades have also seen tremendous progress regarding the control of quantum mechanical systems. This enabled the emergence of quantum information theory which rests on the idea of using quantum mechanical principles for information processing [1]. This concept promises quantum technology that can outperform its classical analogues for important applications such as computing, metrology, and communication. For a quantum device to outperform its classical analogue, a key resource needed is entanglement, which describes a correlation between quantum systems with no classical analogue. Entanglement can be used to perform tasks beyond what classical systems can achieve. One of the first practical applications of entanglement is the transport of quantum information. Bennett et al. [2] coined the term 'Quantum Teleportation' for their theoretically proposed scheme to send an unknown quantum state between two parties that share an entangled pair of qubits. A couple of years later, in the year 1997, two independent research groups [3, 4] experimentally realized the teleportation protocol by using photonics. Since then, the goal has been to improve the robustness and the range of quantum teleportation.

Nowadays, teleporting single qubits is routinely performed with commercial applications. However, the teleportation of higher dimensional quantum systems remains a major practical challenge. We analyse the complexity of the standard multi-qubit teleportation scheme, the sources of error that could creep in during the execution of the protocol, and explore potential enhancements to the protocol.

At the beginning of this thesis, we review the concept of entanglement and introduce some basic ways to characterise entanglement in a quantum system. We also investigate *Variational Quantum Algorithms* [5] with a focus on understanding the *Quantum Autoencoder* [6]. VQAs are constructed in a similar fashion to machine learning algorithms based on neural networks. The VQA solves for an objective that is computed on the quantum hardware and is iteratively optimized using a classical feedback loop. Using the knowledge of entanglement, Quantum Autoencoders, and VQAs, we apply it to disentangling a quantum state i.e. completely removing the entanglement in the state. We design a new type of variational quantum autoencoder, which we call **Product State Autoencoder**, that learns to disentangle quantum states in an unsupervised manner. We then apply our autoencoder to the transport of entangled states through a qubit loss channel as well as in a multi-qubit teleportation scenario. Our new protocol requires exponentially fewer repetitions than the standard protocol. We also explore potential augmentations of quantum autoencoder by using classical information and explore the overall trainability of our VQA.

Part I

Background

Chapter 2

Quantum Computing and Quantum Information Primer

This chapter establishes a common language with the reader i.e. quantum information. We introduce the building blocks of quantum computation, derive the quantum teleportation protocol for quantum communication, and finally end by establishing connections between the notions of entanglement and mixedness in the context of quantum subsystems.

In the year 1980, Paul Benioff proposed a quantum mechanical model of the Turing Machine[7]. A year later, in his keynote speech[8] at the Physics of Computation conference organized by MIT & IBM, Richard Feynman proposed a new type of computer that runs on the principles of quantum mechanics, which would be advantageous for simulating difficult quantum mechanical problems. Both these instances motivated the need to build the foundations and theory behind quantum information processing. In the next few decades, landmark algorithms such as *Shor's Factoring*[9] for prime factorization in $O(logN^3)$ time and $O(logN)$ space (a problem that has no known P-time classical algorithm), *Grover's Search* for searching through an unstructured database in $O(\sqrt{N})$ time have shown that quantum information processing has implications outside of just simulating physics, and has only bolstered the argument that quantum computers could offer advantages to many more problems.

Various models of quantum computation have been proposed, such as gate-based, measurement-based[10], or adiabatic quantum computing[11], each offering its own benefits. This thesis follows the gate-based or circuit model approach.

2.1 Building Blocks of Quantum Information

In classical computation, the abstraction level offered to scientists is high and usually, the most detail a traditional user goes into are concepts of bits and logic gates (NOT, AND, OR etc.). Nonetheless, classical computation enabled the development of game-changing algorithms for various practical problems. This level of abstraction allowed users (scientists) to circumvent understanding of the underlying physics and engineering principles of charge, voltage, switches, transistors etc. that govern the computation [1]. For the quantum mechanical postulates, we have similar abstractions based on linear algebra and probability theory.

2.1.1 Qubit

The most basic unit of quantum computation is a *Quantum Binary Digit* or ***qubit***. Mathematically, a single qubit is represented as a normalized complex 2-D vector. All quantum computation is performed by encoding information into, manipulating, and extracting information from these complex unit vectors. In a more formal quantum mechanical language, qubits are represented using the ket ($|.\rangle$) notation. A single qubit $|\psi\rangle \in \mathbb{C}^2$ can be in the $|0\rangle = \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right)$ state, $|1\rangle = \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right)$ state or a superposition state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \left(\begin{smallmatrix}\alpha\\\beta\end{smallmatrix}\right)$, where $\alpha, \beta \in \mathbb{C}$ and $\|\alpha\|^2 + \|\beta\|^2 = 1$ (normalization condition).[2] It is important to note that unlike in the classical case, where the bits are in a definite state of either 0 or 1, qubits can form *superpositions* with complex amplitudes of the two basis states. The state of a qubit can be visualized as a point on the Bloch Sphere as shown in Fig 2.1 and it is represented by two parameters θ and ϕ as angles with respect to $\hat{z}$ and $\hat{x}$ respectively.

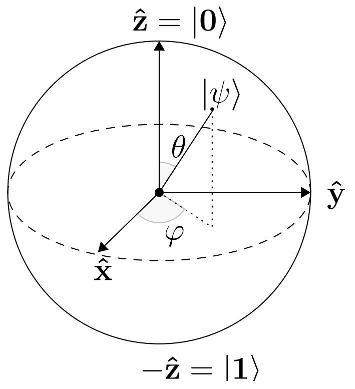

$\hat{\mathbf{z}} = |\mathbf{0}\rangle$
$|\psi\rangle$
θ
$\hat{\mathbf{y}}$
φ
$\hat{\mathbf{x}}$
$-\hat{\mathbf{z}} = |\mathbf{1}\rangle$

Figure 2.1: Bloch Sphere visualization of a single qubit. The +z and -z axes represent the $|0\rangle$ and $|1\rangle$ respectively. Due to the normalization condition, a qubit can be represented just using the parameters θ (angle with $\hat{z}$) and ϕ (angle with $\hat{x}$) as $|\psi\rangle = \cos\theta\,|0\rangle + e^{i\phi}\sin\theta\,|1\rangle$

Example 2.1.1. ***Find the state representation of $\hat{y}$ on the Bloch Sphere***
Answer: *$\hat{y}$ is at a 90 degree angle with both $\hat{x}$ and $\hat{z}$, and thus corresponds to the parameters $\theta = \frac{\pi}{2}$ and $\phi = \frac{\pi}{2}$. Thus we get*

$$\begin{aligned}\hat{y} &= \cos\theta\,|0\rangle + e^{i\phi}\sin\theta\,|1\rangle \\ &= \cos\frac{\pi}{2}\,|0\rangle + e^{i\frac{\pi}{2}}\sin\frac{\pi}{2}\,|1\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\end{aligned}$$

$\hat{y} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ *corresponds to the equal superposition state.*

[1]This statement is generally true for commonly used algorithms, however in the case of FPGAs it is useful to understand the underlying hardware

[2]The general convention is to use $\{|0\rangle, |1\rangle\}$ basis to represent a qubit's state as a superposition of states. However, any orthonormal basis can be used Eg: $\{|+\rangle, |-\rangle\} = \{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\}$

Quantum states of 2 or more qubits (*multi-qubit states*) live in a *tensor product space* ($\otimes$) of the individual qubits' spaces i.e. if $|\psi\rangle$ is an n qubit state, then it is a unit vector $\in \mathbb{C}^{2^n}$.

Example 2.1.2. ***Write out the state of the 2-qubit system*** $|\psi_{AB}\rangle$ ***composed of 2 single qubits states*** $|\psi_A\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$***,*** $|\psi_B\rangle = \frac{1}{\sqrt{2}}|0\rangle + (\frac{1}{2} - i\frac{1}{2})|1\rangle$
Answer*: We can first see that both states,* $|\psi_A\rangle$ *and* $|\psi_B\rangle$*, are unit vectors, so they are valid quantum states. Now, since we know that multi-qubit states live in a tensor product space of the individual qubits' spaces (*$|\psi_{AB}\rangle \in \mathbb{C}^2 \otimes \mathbb{C}^2 = \mathbb{C}^4$*), we get*

$$\begin{aligned}|\psi_{AB}\rangle =&|\psi_A\rangle \otimes |\psi_B\rangle\\ =&(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)) \otimes (\frac{1}{\sqrt{2}}|0\rangle + (\frac{1}{2} - i\frac{1}{2})|1\rangle)\\ =&\frac{1}{2}|0\rangle \otimes |0\rangle + (\frac{1}{2\sqrt{2}} - i\frac{1}{2\sqrt{2}})|0\rangle \otimes |1\rangle + \frac{1}{2}|1\rangle \otimes |0\rangle + \frac{1}{2\sqrt{2}}(1-i)|1\rangle \otimes |1\rangle\end{aligned}$$

and since $|a\rangle \otimes |b\rangle$ *can be rewritten as* $|ab\rangle$*,*

$$=\frac{1}{2}|00\rangle + \frac{1}{2\sqrt{2}}(1-i)|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2\sqrt{2}}(1-i)|11\rangle$$

A general n-qubit state has the following form

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle ; \begin{cases} |i\rangle \text{ is the } 2^n \text{ bit binary representation of i} \\ \sum_i^{n-1} \|\alpha_i\|^2 = 1 \\ \alpha_i \text{ is called the } \textbf{\textit{amplitude}} \text{ associated with basis state } |i\rangle \end{cases} \tag{2.1}$$

2.1.2 Quantum Gates

We have seen how qubits are represented as normalized complex vectors in a tensor product space. To manipulate the states/information stored in these qubits, we require ***quantum gates***. A quantum gate acting on an n-qubit quantum system is a unitary matrix [3] $U \in \mathbb{C}^{2^n} \times \mathbb{C}^{2^n}$. The most commonly seen single qubits gates are the Pauli gates ($\sigma_i; i \in \{x, y, z\}$), Pauli Rotation gates ($R_i(\theta); i \in \{x, y, z\}$), Phase Shift gate ($P(\phi)$) and the Hadamard gate (H):

$$\sigma_x = \left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right), \sigma_y = \left(\begin{smallmatrix}0&-i\\i&0\end{smallmatrix}\right), \sigma_z = \left(\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right), P(\phi) = \left(\begin{smallmatrix}1&0\\0&e^{i\phi}\end{smallmatrix}\right), H = \frac{1}{\sqrt{2}}\left(\begin{smallmatrix}1&1\\1&-1\end{smallmatrix}\right) \tag{2.2}$$

Note: Quantum gates act linearly on qubits that are in a superposition.

Example 2.1.3. ***Find a state we get when we apply the*** σ_z ***gate to the equal superposition state*** $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
Answer*: As mentioned previously, a gate acts linearly on a qubit in a superposition state, hence*

$$\begin{aligned}\sigma_z|+\rangle =&\left(\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right)\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\\ =&\left(\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right)\frac{1}{\sqrt{2}}|0\rangle + \left(\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right)\frac{1}{\sqrt{2}}|1\rangle\end{aligned}$$

[3] Unitary matrices are the only matrices that preserve the normalization condition of quantum states

rewriting the basis states in a vector notation

$$=\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix}$$

and thus

$$=\begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle$$

An example of a 2-qubit gate is the CNOT or CX gate, $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$. If the first qubit (*control*) is in the state $|0\rangle$, then it leaves the second qubit (*target*) unchanged. If the control qubit is in the state $|1\rangle$, then it flips (apply σ_x) to the target qubit.

$$CNOT\,(\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle) \longrightarrow$$
$$\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|11\rangle + \alpha_{11}|10\rangle$$

Note: Every n-qubit gate can be decomposed as a set of single qubit rotation gates ($R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$), phase shift gates $P(\phi)$ and CNOT gates thus making them a ***universal set*** *of gates [12].*

2.1.3 Measurements

Now that we have a way to manipulate qubits, we need a way to extract information from a qubit through a process called ***measurement***. Measurement of a state $|\psi\rangle$ involves projection (***collapse***)[4] of the qubit state vector to a vector in a chosen orthonormal basis $\{|\phi_i\rangle\}_{i=0\ldots2^n-1}$. We can rewrite the state $|\psi\rangle$ in the chosen measurement basis $\{|\phi_i\rangle\}_{i=0\ldots2^n-1}$ as

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \beta_i\,|\phi_i\rangle \tag{2.3}$$

The basis representation of $|\psi\rangle$ can be seen as a discrete probability distribution where $\|\beta_i\|^2$ is the probability of collapse onto the state $|\phi_i\rangle$ when $|\psi\rangle$ is measured (it should now make more sense as to why qubits are normalized). Another, more formal way of representing measurements is through operators $\{M_m\}_{m=1}^n$, with $\sum_m M_m^\dagger M_m = I$, where I is an identity matrix. $P(m) = \langle\psi|\,M_m\,|\psi\rangle$ is the probabilty of the outcome m when measuring $|\psi\rangle$. The general orthonormal bases used are the eigenvectors of the Pauli Operators[13][5].

2.1.4 Quantum Algorithms

Qubits, quantum gates and measurements are the basic building blocks for constructing ***quantum algorithms***. Any quantum algorithm can be seen as taking a state $|\psi\rangle$ and acting

[4]The projection is generally known as ***collapse*** of the wave function in quantum mechanics

[5]Note: After measurement, the state we are left with is one of the basis vectors of the chosen orthonormal basis of measurement ($\{|\phi_i\rangle\}_{i=0\ldots2^n-1}$). We lose the original state. To obtain any form of useful information from the state $|\psi\rangle$, we would need to perform measurement on multiple copies of the $|\psi\rangle$ and aggregate the results

on it (or any of its subsystems) a sequence of k unitary gates $\{U_i[S_i]\}_{i=1..k}$, where U_i is the i^{th} unitary gate acting on a subsystem of qubits S_i. The whole algorithm can be represented as $U_k[S_k]U_{k-1}[S_{k-1}]...U_1[S_1]\,|\psi\rangle$[13]. We extract information from the state $|\psi_{out}\rangle = U_k[S_k]U_{k-1}[S_{k-1}]...U_1[S_1]\,|\psi\rangle$ by using appropriate measurements based on requirements of the algorithm.

2.1.5 Entanglement and Separability

In **Example 2.1.2**, we have seen that the state $|\psi_{AB}\rangle$ can be written as a tensor product of two qubits i.e. $|\psi_{AB}\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$. This is called a ***separable*** state. However, quantum states need not always be separable.

Example 2.1.4. ***Give an example of a non-separable state***
Answer: *An example of a non-separable state is one of the bell states:*

$$|\phi^+\rangle = \frac{1}{2}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle) = \frac{1}{2}(|00\rangle + |11\rangle) \tag{2.4}$$

If it were separable, we would have

$$|\phi^+\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$$

Now, let

$$\begin{aligned} |\psi_A\rangle &= a_0\,|0\rangle + a_1\,|1\rangle\,;\, \|a_0\|^2 + \|a_1\|^2 = 1 \\ |\psi_B\rangle &= b_0\,|0\rangle + b_1\,|1\rangle\,;\, \|b_0\|^2 + \|b_1\|^2 = 1 \end{aligned}$$

Therefore,

$$\begin{aligned} (a_0\,|0\rangle + a_1\,|1\rangle) \otimes (b_0\,|0\rangle + b_1\,|1\rangle) =& \frac{1}{2}(|00\rangle + |11\rangle) \\ a_0b_0\,|00\rangle + a_0b_1\,|01\rangle + a_1b_0\,|10\rangle + a_1b_1\,|11\rangle =& \frac{1}{2}(|00\rangle + |11\rangle) \end{aligned}$$

From which it follows that

$$a_0b_1 = a_1b_0 = 0$$

In $a_0b_1 = 0$ we can see that a_0 cannot be 0 as this would assign 0 amplitude to the $|00\rangle$ state (which has an amplitude of $\frac{1}{2}$ in $|\phi^+\rangle$). We can argue against the same for $b_1 = 0$ as this would assign an amplitude of 0 to the $|11\rangle$ state (which also has an amplitude of $\frac{1}{2}$ in $|\phi^+\rangle$). Thus we cannot write $|\phi^+\rangle$ as a tensor product of states $|\psi_A\rangle$ and $|\psi_B\rangle$, making it non-separable.

Two quantum subsystems of a composite (larger) system are considered ***entangled*** if they cannot be represented as a tensor product of the individual systems. In **Eg: 2.1.4**, qubit 1 is entangled to qubit 2 as there is no tensor product representation of the entire two qubit system in terms of the individual qubits.

Entanglement is a uniquely quantum mechanical phenomenon and has been shown to be an key resource required for various quantum protocols and algorithms. However, an analogous concept in the classical case is joint distributions and dependent random variables, where the knowledge of one random variable gives partial or complete knowledge of other dependent random variables.

We have looked at one formalism for representing quantum states, the ***state vector*** representation. A different way of representing quantum states is through the ***density matrix*** formalism. Using this formalism, instead of a quantum state being represented as a ket $|\psi\rangle$ (state vector representation), it is written as a density matrix $\rho = |\psi\rangle\langle\psi|$. The quantity $\langle\psi|$ (represented as a row vector) is called the ***bra*** representation of the quantum state, and is the *adjoint* (conjugate transpose) of $|\psi\rangle$. Unitary gates U are applied to ρ by $U\rho U^\dagger$.

Viewing quantum states as density matrices gives us the flexibility of taking into account probabilistic mixtures of quantum states (called ***mixed states***)[6]

$$\rho = \sum_{i=1}^{k} p_i |\psi_i\rangle\langle\psi_i| \,; \text{ where } \sum_{i=1}^{k} p_i = 1 \tag{2.5}$$

The p_i variables in **Eq:2.5** denote the classical probability p_i of state $|\psi_i\rangle$ appearing in the mixture ρ. States whose density matrices can be written as $\rho = |\psi\rangle\langle\psi|$ (where $p_i = 1$ for only one of the states in the mixture) are called ***pure states***. All states we have been considering until now are pure states. Multi-qubit density matrices also live in a tensor product space. If ρ_A is the density matrix/operator of system A and ρ_B is the density matrix of system B, then the combined system has a density matrix $\rho = \rho_A \otimes \rho_B$.

Measurements in the density matrix formalism are defined for a set of measurement operators $\{M_m\}$ and a density matrix ρ as $P(m) = Tr[M_m\rho]$, where $P(m)$ is the probability of obtaining outcome m. As we have seen in ***Fig 2.1***, single-qubit pure states are represented as points on the surface of the Bloch Sphere. Single-qubit mixed states on the other hand correspond to points on the interior of the Bloch sphere with the center denoting the maximally mixed state (**Fig: 2.2**). How mixed or pure a quantum state is can be quantified by computing

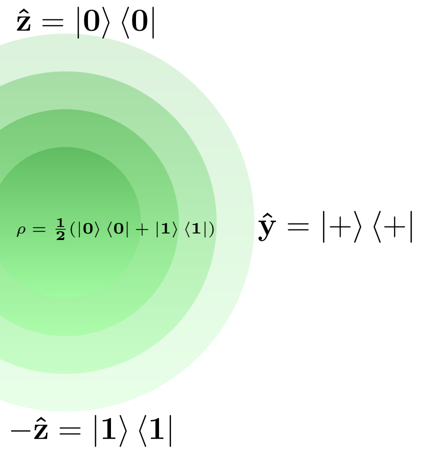

$\hat{\mathbf{z}} = |\mathbf{0}\rangle\langle\mathbf{0}|$
$\rho = \frac{1}{2}(|0\rangle\langle 0| + |1\rangle\langle 1|)$
$\hat{\mathbf{y}} = |+\rangle\langle +|$
$-\hat{\mathbf{z}} = |\mathbf{1}\rangle\langle\mathbf{1}|$

Figure 2.2: Cross-section of Bloch Sphere passing through $\hat{y}\hat{z}$ axis. The circumference represents **pure** states, and as we move towards the center, the 'mixedness' of the states increases **mixed**. Higher intensity of green means the states in that region are more mixed.

the ***purity*** of a quantum state. The purity is defined by $Tr(\rho^2)$, where ρ is the density matrix of the quantum system. An efficient circuit to compute purity will be discussed in the later sections.

[6]The density matrix representation is very useful in describing quantum states in the presence of noise

The next section introduces a real world application of the quantum computing concepts covered in this section, which will also become a core application of the architecture proposed in this thesis.

2.2 Quantum Teleportation Protocol

Quantum teleportation[2] was one of the first protocols devised for quantum communication, and entanglement is one of the key resources used in this protocol to carry out secure communication. The problem setup is as follows:

1. Two parties (Alice and Bob) wish to communicate securely with each other.
2. They both share a pair of qubits entangled in the Bell state $|B_{00}\rangle = |\phi^+_{AB}\rangle = \frac{1}{2}(|0_A 0_B\rangle + |1_A 1_B\rangle)$ (more precisely the EPR pair) which is pre-prepared and one of the qubits is sent to each party beforehand.
3. Alice wishes to send an unknown[7] state $|\psi_C\rangle = \alpha|0\rangle + \beta|1\rangle$ to Bob

The state of the entire system (EPR pair and $|\psi_C\rangle$) can be expressed as a tensor product

$$\begin{aligned}|\psi\rangle &= |\phi^+_{AB}\rangle \otimes |\psi_C\rangle \\ &= \frac{1}{2}(|0_A 0_B\rangle + |1_A 1_B\rangle) \otimes (\alpha|0_C\rangle + \beta|1_C\rangle) \\ &= \frac{1}{2}(\alpha|0_A 0_B 0_C\rangle + \beta|0_A 0_B 1_C\rangle + \alpha|1_A 1_B 0_C\rangle + \beta|1_A 1_B 1_C\rangle)\end{aligned}$$

Rearranging the qubits

$$= \frac{1}{2}(\alpha|0_A 0_C 0_B\rangle + \beta|0_A 1_C 0_B\rangle + \alpha|1_A 0_C 1_B\rangle + \beta|1_A 1_C 1_B\rangle)$$

Rewriting the subsystem AC in the Bell basis

$$\begin{aligned}= \frac{1}{2\sqrt{2}}\Big(&\alpha(|\phi^+_{AC}\rangle + |\phi^-_{AC}\rangle)|0_B\rangle + \beta(|\psi^+_{AC}\rangle + |\psi^-_{AC}\rangle)|0_B\rangle) \\ &+ \alpha(|\psi^+_{AC}\rangle - |\psi^-_{AC}\rangle)|1_B\rangle) + \beta(|\phi^+_{AC}\rangle - |\phi^-_{AC}\rangle)|1_B\rangle\Big)\end{aligned}$$

Grouping the common terms

$$\begin{aligned}= \frac{1}{2\sqrt{2}}\Big(&|\phi^+_{AC}\rangle(\alpha|0_B\rangle + \beta|1_B\rangle) + |\phi^-_{AC}\rangle(\alpha|0_B\rangle - \beta|1_B\rangle) \\ &|\psi^+_{AC}\rangle(\beta|0_B\rangle + \alpha|1_B\rangle) + |\psi^-_{AC}\rangle(\beta|0_B\rangle - \alpha|1_B\rangle)\Big)\end{aligned}$$

Now that the subsystem AC is represented in the Bell basis $(\phi^+, \phi^-, \psi^+, \psi^-)$, it can be clearly seen that measuring subsystem AC collapses subsystem B to one of four possible states $\Big\{\alpha|0_B\rangle + \beta|1_B\rangle, \alpha|0_B\rangle - \beta|1_B\rangle, \beta|0_B\rangle + \alpha|1_B\rangle, \beta|0_B\rangle - \alpha|1_B\rangle\Big\}$ based on the measurement result for subsystem AC. These collapsed states look very similar to the state in the $|\psi_C\rangle$.

[7]the state could be known beforehand, but the most general scenario would be the case when the state to be transferred is unknown

Based on the measurement by Alice on subsystem AC, she would send two classical bits to Bob, which will inform him on what correction to do on his qubit to complete the teleportation of $|\psi_C\rangle$. The correction step is shown in **Table 2.1**

Measurement Result	**Classical bits sent to Bob**	**Correction gates by Bob**
$\left\|\phi^+_{AC}\right\rangle$	00	I
$\left\|\phi^-_{AC}\right\rangle$	01	σ_z
$\left\|\psi^+_{AC}\right\rangle$	10	σ_x
$\left\|\psi^-_{AC}\right\rangle$	11	$\sigma_x\sigma_z$

Table 2.1: Correction protocol that Alice and Bob need to follow after Alice's Bell measurement to complete the teleportation of state $|\psi_C\rangle$

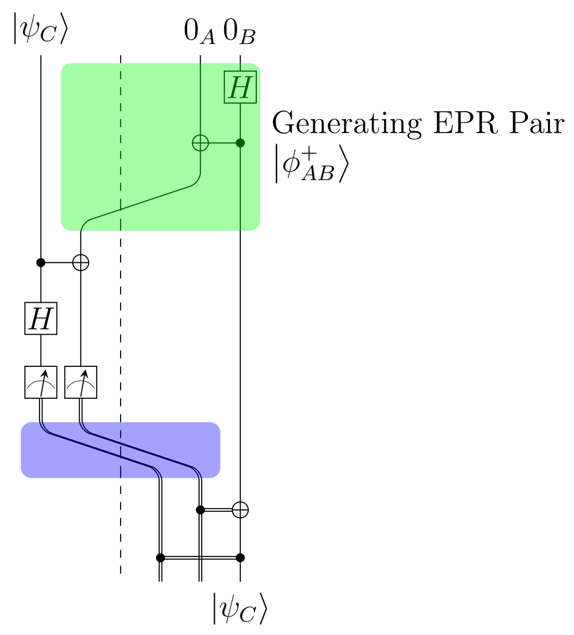


Figure 2.3: Quantum Teleportation Protocol for teleporting a single qubit state $|\psi_C\rangle$ using the entangled pair $|\phi^+_{AB}\rangle$ shared between parties A and B

Note: Quantum teleportation does not create a copy of the state we wish to teleport, nor does it physically send a qubit. As shown in ***Fig: 2.3****, it merely transfers the information of a qubit held by Alice onto a qubit held by party Bob through the entangled Bell pair that they share.*

In the next few sections, we revisit the concept of entanglement and build connections with other properties of quantum states that are not apparent at first.

2.3 Connecting Entanglement with Pure/Mixed states

As seen in the case of quantum teleportation, entanglement is a vital resource in quantum information. In the following section, we further explore this concept by uncovering certain important links with entanglement. In the context of quantum states, entanglement and separability initially seem disconnected from the idea of pure and mixed states. This is true if we look at the entire quantum system. However, when we consider ***subsystems*** of a quantum state/system, we begin to notice that these concepts are in fact related.

Before exploring the links between entanglement and purity, two core concepts need to be introduced. First, the *trace* of a matrix. The trace is defined as the sum of the diagonal elements of the matrix. For any operator, the trace can be computed as:

$$Tr[O] = \sum_{i=0}^{n-1} \langle i| O |i\rangle \, ; \quad \begin{cases} \dim(O) = n, \ \{|i\rangle\} \text{ is any orthonormal basis} \end{cases} \tag{2.6}$$

$\{|i\rangle\}$ is any orthonormal basis as Tr is basis independent. In the case of density matrices, $Tr[\rho] = 1$. The second concept is the partial trace:

$$Tr_B[\rho_{AB}] = \sum_{i=0}^{n-1} \langle i_B| \rho_{AB} |i_B\rangle \quad \begin{cases} \{|i_B\rangle\} \text{ is any orthonormal basis} \\ \text{of subsystem B} \\ Tr_X \text{ take trace of subsystem } X \end{cases} \tag{2.7}$$

For a given pure quantum system ρ_{AB}, if one wanted to construct an operation that gives us the description of only a particular subsystem (Eg subsystem A), then measurement statistics on this resultant state should give us similar results to just performing measurements on just subsystem A in ρ_{AB} i.e. the partial trace of A.

Example 2.3.1. ***Calculate the partial trace and purity over subsystem*** B ***of the density matrix*** $\rho_{AB} = |\phi^+_{AB}\rangle \langle \phi^+_{AB}|$
Answer: *The partial trace over subsystem B is:*

$$\begin{aligned}
\rho_A = Tr_B[\rho_{AB}] =& \sum_{i=0}^{3} (\mathbb{I}_{\mathbb{A}} \otimes \langle i_B|) |\phi^+_{AB}\rangle \langle \phi^+_{AB}| (\mathbb{I}_{\mathbb{A}} \otimes |i_B\rangle) \\
=& \frac{1}{2}(\mathbb{I}_{\mathbb{A}} \otimes \langle 0_B|)(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|)(\mathbb{I}_{\mathbb{A}} \otimes |0_B\rangle) \\
+& \frac{1}{2}(\mathbb{I}_{\mathbb{A}} \otimes \langle 1_B|)(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|)(\mathbb{I}_{\mathbb{A}} \otimes |1_B\rangle) \\
=& \frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|1\rangle\langle 1| \quad \begin{cases} \textit{This is the maximally mixed state} \end{cases} \\
=& \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}
\end{aligned}$$

The purity of this subsystem is

$$Tr[\rho_A^2] = Tr[\begin{pmatrix} \frac{1}{4} & 0 \\ 0 & \frac{1}{4} \end{pmatrix}] = \frac{1}{2}$$

We can clearly see that the more entangled the composite system is, the less pure the reduced density matrix of the subsystem i.e. they are anticorrelated. Thus, the purity of a subsystem can be used as a measure of its entanglement with the rest of the system. The next section presents an efficient approach to compute the purity of a subsystem of interest.

2.4 Measuring Purity of Quantum Systems

We have established in the previous section that the purity of a subsystem is a measure of how entangled the subsystem is with the rest of the system. This section aims to introduce an algorithm to measure the purity of an N qubit quantum state ρ. Combining measurements on multiple copies of a state, followed by some post-processing, helps uncover certain properties not accessible from just a single copy of the state [14, 15]. As seen in **Fig: 2.4**, we consider two copies of the state ρ to be subsystems A and B respectively of the larger system $\rho \otimes \rho$ and perform the Bell measurement. This process is repeated N_Q times, and the measurement results $r^j \in \{0,1\}^{2N}$; $j \in \{1, 2, \ldots, N_Q\}$ are recorded. r^j_n, $n = 1, \ldots, N$ is the outcome of measuring the n^{th} qubit of subsystem A and r^j_{N+n} is the outcome of the n^{th} qubit of subsystem B.

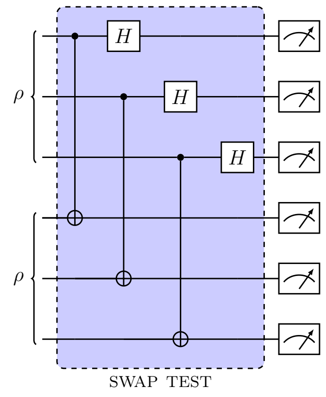


Figure 2.4: Diagram [16] of a circuit to measure purity of a quantum state using the SWAP test [14, 15]

The Bell measurement (SWAP test) on two copies of the same state ρ helps to compute the trace overlap of the two states (i.e. the purity)

$$Tr[\rho^2] = 1 - 2P_{odd} \tag{2.8}$$

where P_{odd} is the probability that $q^j (q^j \in \{0,1\}^N)$ has odd parity, where $q^j_n = r^j_n . r^j_{N+n}$ is the bit-wise AND of the outcomes of subsystems A and B.

Performing the bell measurement on this setting of two copies of a pure state $|\psi\rangle \otimes |\psi\rangle$ gives the outcome r with probability [17]

$$P(r) = \langle\psi| \langle\psi| O_r |\psi\rangle |\psi\rangle = 2^{-N} | \langle\psi| \sigma_r |\psi^*\rangle |^2 \tag{2.9}$$

where $O_r = |\sigma_r\rangle \langle\sigma_r|$ is the projector onto a product of Bell states and $|\psi^*\rangle$ is the complex conjugate of $|\psi\rangle$.

Therefore, for N_Q runs of the circuit with outcomes $\{r^j\}$, we can compute the purity of the state $|\psi\rangle$ as

$$\begin{aligned} Tr[\rho^2] =& 1 - 2P_{odd} && (2.10) \\ =& 1 - 2\sum_r P(r) odd(r) && (2.11) \\ =& 1 - \frac{2}{N_Q}\sum_{j=1}^{N_Q} odd(r^j) \begin{cases} odd(r^j) = 1 \text{ if } q^j \text{ has odd parity} \\ odd(r^j) = 0 \text{ if } q^j \text{ has even parity} \end{cases} && (2.12) \end{aligned}$$

For a pure state $\rho = |\psi\rangle\langle\psi|$ of n qubits, we can compute the purity of a subsystem S of qubits by performing the Bell test on the subsystem S of 2 copies of ρ i.e. we perform the Bell test on the system $\rho_S \otimes \rho_S$

$$\begin{aligned} Tr[\rho_S^2] =& 1 - 2P_{odd} && (2.13) \\ =& 1 - 2\sum_r P(r_S) odd(r_S) \begin{cases} r_S \text{ is the measurement outcome of subsystem } S \end{cases} && (2.14) \\ =& 1 - \frac{2}{N_Q}\sum_{j=1}^{N_Q} odd(r_S^j) \begin{cases} odd(r_S^j) = 1 \text{ if } q_S^j \text{ has odd parity} \\ odd(r_S^j) = 0 \text{ if } q_S^j \text{ has even parity} \end{cases} && (2.15) \end{aligned}$$

Chapter 3

Variational Quantum Algorithms

The third chapter of this thesis explains the problems associated with running theoretically established quantum algorithms on present-day quantum hardware (NISQ hardware), mainly due to noise-related errors. We finally look at the anatomy of a new class of algorithms that can be run on NISQ hardware, Variational Algorithms.

The quantum algorithms such as Shor's Factoring[9] or Grover's Search[18] that have been theoretically proven to give advantages over their classical counterparts cannot be reliably executed on current-day quantum hardware for large problem sizes. Experimentally, Shor's factoring has been used to factor the numbers N=15,20, and 35 [19]. However, it has also been estimated that breaking RSA-2048 bit keys would require 4000-10,000 logical qubits[1] and 100 million gates. This is significantly more than the 127 qubits [20] we have access to on current-day quantum computers called *Noisy Intermediate Scale Quantum (NISQ) computers*.

3.1 Are Noisy Intermediate Scale Quantum Computers useful?

Quantum computers do have the potential to solve certain types of problems in a more efficient way. However, as mentioned at the start of this chapter, Noisy Intermediate Scale Quantum devices (current-day quantum computers) are nowhere near capable of executing a large algorithm like Shor's factoring. The main reason is due the problem of decoherence. We do not currently have a good method to manipulate and stabilize qubits at the same time for large-depth quantum circuits. Each operation on a qubit introduces noise (errors) to the qubit's states; hence, it is beneficial to keep the computations short on NISQ devices.[2].

These restrictions led quantum researchers to come up with a new set of algorithms[3] called ***Variational Quantum Algorithms*** (VQAs) that can be run even on NISQ devices. VQAs may offer potential quantum advantages even in the presence of noise. The key idea of VQA is combining classical and quantum computers in the feedback loop. The quantum computer's

[1]Each Logical qubit is composed of thousands to millions of physical qubits which help with error correction

[2]To counter this, scientists are developing codes and protocols to correct errors introduced during a quantum computation (which are still decades away from being practically useful)

[3]based on the Ritz variational principle

output is classically post-processed and fed back in to the quantum computer. The unitary gates used by variational algorithms are composed of parameters that are to be learned based on the specific problem we are trying to solve. This gives us the freedom of choosing the number of gates for our quantum computation based on hardware restrictions and focus on developing a learning algorithm for the problems.

In Quantum Mechanics, physicists and chemists often find themselves solving problems that involve computing the ground state or ground state energy of a quantum system defined by a Hamiltonian H. If they attempt to test all possible state configurations to observe which one corresponds to the lowest energy, the issue is that there is no clear way to directly search the Hilbert Space $\mathcal{H}$ for the ground state solution. The Ritz method for computing solutions to boundary value problems is one way to tackle such problems. The main observation to make is that for any arbitrarily chosen quantum state $|\psi\rangle$, the energy corresponding to the state follows the inequality

$$\langle\psi| H |\psi\rangle \geq E_0 \tag{3.1}$$

Eq: 3.1 tells us that the energy corresponding to any chosen state of the system is an upper bound to the ground state energy of the quantum system. We need to find a way to pick trial states (sometimes called an ansatz) to compute the energy of the system. This motivates the need to parametrise these quantum states and convert the problem to a problem of optimization of a cost function, where the cost function corresponds to the expectation value/energy $E(\theta)$ of the parametrised state $|\psi(\theta)\rangle$

$$E(\theta) = \langle\psi(\theta)| H |\psi(\theta)\rangle \tag{3.2}$$

Classically solving this problem requires an exponential number (in number of qubit n) to represent the complex amplitudes of $|\psi\rangle$. However, by running a parametrised quantum circuit on real quantum hardware, we can prepare the same state $|\psi\rangle$ with just n qubits. Repeated measurements of the observable of interest can be made to compute the expectation value (energy). This quantum advantage is not just observed for optimizing for the ground state. Any cost function that can be expressed as an observable can be optimized using a parameterised quantum circuit. Some other very important applications of variational quantum algorithms include, but are not limited to, Variational Quantum Eigensolver for finding the lowest energy eigenstate of a Hamiltonian[4] (very useful for quantum chemistry simulations), Quantum Approximate Optimization Algorithm (QAOA)[21] for finding approximate solutions to combinatorial optimizations (generally NP-hard to find an exact solution in the classical case) and Quadratic Unconstrained Binary Optimization (QUBO) problems. The sections below dive deeper into these Variational Quantum Algorithms.

3.2 Anatomy of Variational Quantum Algorithms

A variational quantum algorithm encodes into a parametrised cost function (evaluated on a quantum computer) the task we wish to solve, and employs a classical optimizer to learn the parameters that solve the task. As mentioned previously, quantum circuits could have gates with fixed values (Eg: Hadamard, Pauli-X, Pauli-Y, Pauli-Z, CNOT), or gates that are Parametrised

[4]A Hamiltonian, denoted by a matrix H, describes the energy and evolution of a quantum system. The ground state would correspond to the state with the lowest eigenvalue of the Hamiltonian

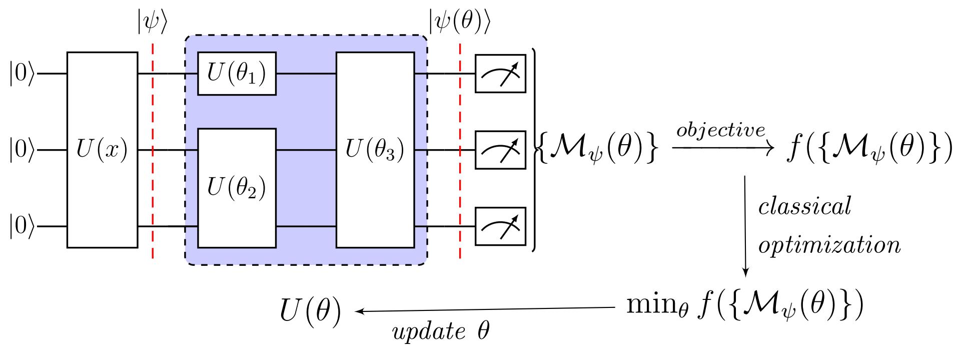


Figure 3.1: NISQ parametrised quantum circuit

(Eg: $R_X(\theta)$, $R_Y(\theta)$, $R_Z(\theta)$). Using parameterised gates, a circuit can be constructed as shown in **Fig: 3.1** The measurements M are then post processed to compute the cost function f into which our task has been encoded. As more parameterised gates (parameters) are added to the circuit, the more expressive the circuit's function space becomes [22]. The parameters θ then need be iteratively updated to optimize the objective. The next section covers a popular optimization routine for parametrised circuits.

3.3 Optimization on Quantum Hardware

When using a quantum computer for information processing, we only have access to the final measurement results. A general method of optimization such as gradient descent requires the output of every parametrised gate in the circuit in order to perform. The VQA computes the expectation value of some observable value. These expectations are a function of the number of shots i.e. number of times the circuit is run and measured. If a method such as finite differences is used to compute the gradients, one needs to be able to resolve small differences in expectation values when the parameters are varied only by a small amount. A more accurate unbiased estimate of the gradients can be computed using the parameter shift rules [23, 24, 25, 26, 27]. These methods compute gradients similar to the finite differences method but in this case we are free to choose large parameter differences, thus reducing the impact of noise on the gradient calculations.

The first versions of the parameter shift rules were only applicable to parameterised unitaries with Pauli generators i.e. the Pauli rotation gates ($R_X(\theta)$, $R_Y(\theta)$, $R_Z(\theta)$). The rules by Mitarai et al, 2018 [23] were extended further by [24] for variational circuits with unitaries formed of hermitian generators with at most 2 unique eigenvalues

$$U(x) = e^{-ixG} \tag{3.3}$$

The proof below has been adapted from [24, 28] with some modifications and explanations for completeness

$$f(\theta) = \langle 0| U(\theta)^\dagger \hat{Q} U(\theta) |0\rangle \tag{3.4}$$

In **Eq 3.4**, $f(\theta)$ is the cost function we compute using our parameterised circuit. We can decompose $U(\theta)$ (into $V\mathcal{G}(\mu)W$) by isolating the parametrised gate ($\mathcal{G}(\mu)$) we want to differentiate and absorb V and W into the $|0\rangle$ state and observable $\hat{Q}$ respectively

$$\begin{aligned} f(\theta) &= \langle\psi|\, \mathcal{G}^\dagger(\mu)\hat{B}\mathcal{G}(\mu)\,|\psi\rangle \\ \partial_\mu f &= \langle\psi|\, \partial\mathcal{G}^\dagger\hat{B}\mathcal{G}\,|\psi\rangle + \langle\psi|\, \mathcal{G}^\dagger\hat{B}\partial\mathcal{G}\,|\psi\rangle \end{aligned} \tag{3.5}$$

For any 2 operators C, D:

$$\begin{aligned} \langle\psi|\, C^\dagger\hat{Q}D\,|\psi\rangle + h.c. =& \frac{1}{2}(\langle\psi|\,(C+D)^\dagger\hat{Q}(C+D)\,|\psi\rangle \\ &- \langle\psi|\,(C-D)^\dagger\hat{Q}(C-D)\,|\psi\rangle) \end{aligned} \tag{Identity 1}$$

Comparing **Identity 1** with **Eq 3.5**, we can see that if $G \pm \partial G_\mu$ is unitary, then we can modify the circuit slightly to compute the derivate w.r.t a parameter μ.

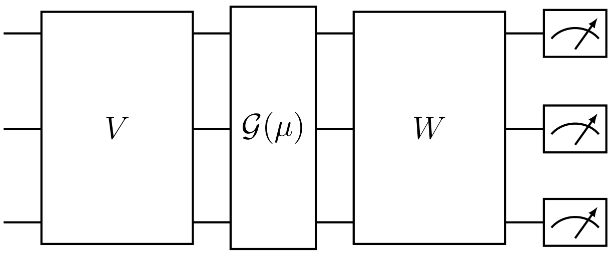


Figure 3.2: The ansatz with the parametrised gate $\mathcal{G}(\mu)$ isolated

Any matrix ($\mathcal{G}(\mu)$) where $\mathcal{G}(\mu) = e^{-i\mu G}$ is Unitary if G is Hermitian. Its derivative is therefore given by **Eq: 3.6**

$$\partial\mathcal{G}_\mu = -iGe^{-i\mu G} \tag{3.6}$$

Using **Eq 3.6** in **Eq 3.5**

$$\partial f_\mu = \langle\psi'|\, B(-iG)\,|\psi'\rangle + h.c. \tag{3.7}$$

Now, using **Identity 1** on **Eq 3.7** with $C = \mathrm{I}$ and $D = -ir^{-1}G$

$$\begin{aligned} \partial f_\mu = & \frac{r}{2}(\langle\psi'|\,(\mathrm{I} - ir^{-1}G)^\dagger B(\mathrm{I} - ir^{-1}G\,|\psi'\rangle - \\ & \langle\psi'|\,(\mathrm{I} + ir^{-1}G)^\dagger B(\mathrm{I} + ir^{-1}G)\,|\psi'\rangle) \end{aligned} \tag{3.8}$$

From Schule et al. [24], we know that if $\mathcal{G}(\mu) = e^{-i\mu G}$ with G having exactly 2 distinct eigenvalues $\pm r$, then $\mathcal{G}(\pm\frac{\pi}{4r}) = \frac{1}{\sqrt{2}}(\mathbb{I} \mp ir^{-1}G)$
The above statement implies that we can compute $\partial_\mu f$ by applying the gate $\mathcal{G}(\pm\frac{\pi}{4r})$ right after the parametrised gate (with parameter μ) we wish to differentiate.
Also, using the fact that if $[A, B] = 0$, then $e^A e^B = e^{A+B}$, we can see that the circuit with the adjacent gates $\mathcal{G}(\mu)$ and $\mathcal{G}(\pm\frac{\pi}{4r})$ has the property that

$$\begin{aligned} \mathcal{G}(\mu)\mathcal{G}(\pm\frac{\pi}{4r}) =& e^{-i\mu G}e^{-i(\pm\frac{\pi}{4r})G} \\ =& e^{-i(\mu\pm\frac{\pi}{4r})G} \\ =& \mathcal{G}(\mu \pm \frac{\pi}{4r}) \end{aligned} \tag{3.9}$$

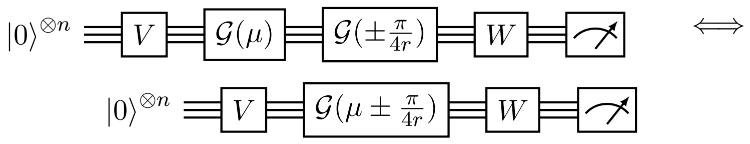


Figure 3.3: Parameter Shift rule equivalence with shift $s = \frac{\pi}{4r}$

Therefore, we can finally write the gradient of the cost function $\partial f(\mu)$ (wrt μ) in terms of evaluations of the cost function at 2 shifted values

$$\begin{aligned}\partial_\mu f =& r(f(\mu + s) - f(\mu - s)) \\ =& r(f(\mu + \frac{\pi}{4r}) - f(\mu - \frac{\pi}{4r}))\end{aligned} \tag{3.10}$$

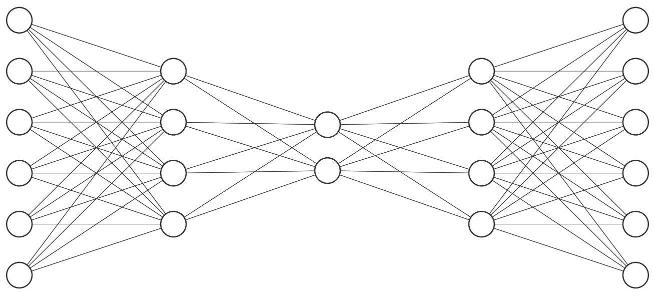
Chapter 4

Quantum Autoencoder and its Applications

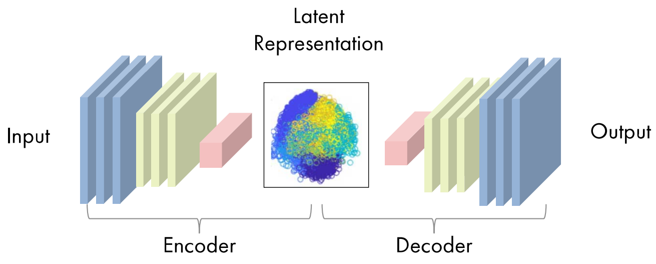
This chapter covers one type of variational algorithm known as the quantum autoencoder. We draw parallels with the classical autoencoder, and also mention a few applications of this autoencoder to various problems in quantum computing.

4.1 Classical Autoencoder

The autoencoder is a very important algorithm in machine learning. It has applications in compression, dimensionality reduction, image generation, anomaly detection and image denoising. A general autoencoder uses an unsupervised learning approach to learn the latent space repre-



(a)



(b)

Figure 4.1: (a) Architecture of a classical autoencoder taking as input 6 dimensional data, and encoding it into a 2 dimensional latent space, (b) convolutional autoencoder (reproduced from [29])

sentation[1] of a dataset. This is made possible with the help of an **Encoder** based on the actual task at hand, the autoencoder takes as input X_{train} and learns a set of sparse features from which X_{train} can be reconstructed easily. The algorithm thus learns to exploit the sparsity of the dataset as the datapoints in X_{train} could potentially reside some a lower dimensional space (manifold).

For example, if we have a dataset of 64×64 RGB images of living rooms, each image in the dataset is naively described by $64 \times 64 \times 3 = 12,288$ parameters. However, due to certain commonalities in these images, they can be described by drastically fewer parameters such as the type, number and position of the furniture. The autoencoder algorithm takes a set of datapoints, and applies a set of non linear operations using a neural network to learn an encoding which becomes some latent space representation of the data. A corresponding decoder takes as input this latent space representation to reconstruct the data (or some desired version of the data). The algorithm is then iteratively run to minimize this reconstruction error which is the mean squared error of datapoints and the network's output. The corresponding objective is:

$$\min_{\theta_E,\theta_D} \mathcal{L}(\theta_E,\theta_D;X) = \min_{\theta_E,\theta_D} \mathbb{E}_X\left[(x - D_{\theta_D}(E_{\theta_E}(x))^2\right] \begin{cases} E,\theta_E: & \text{Encoder, Encoder parameters} \\ D,\theta_D: & \text{Decoder, Decoder parameters} \end{cases} \tag{4.1}$$

$$= \frac{1}{N}\sum_{i=1}^{N}(x^{(i)} - D_{\theta_D}(E_{\theta_E}(x^{(i)}))^2 \tag{4.2}$$

4.2 Quantum Autoencoder

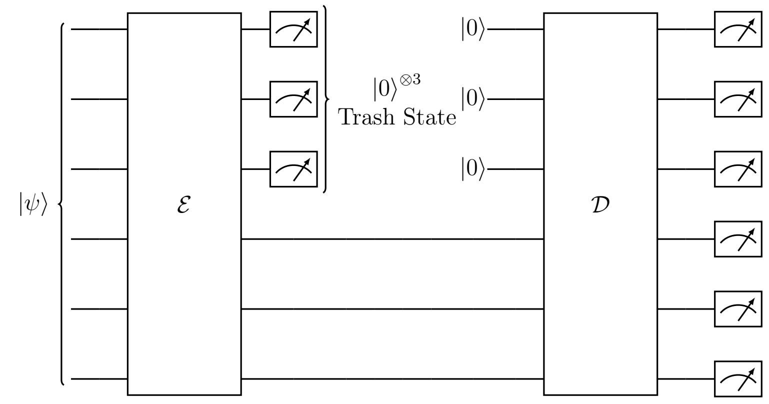


Figure 4.2: Quantum Autoencoder architecture

Analogous to the classical autoencoder, the quantum autoencoder[6] is an algorithm that learns latent representations of quantum states. The assumption is that the data in our quantum states could be living in some lower dimensional Hilbert space that could take fewer qubits to represent. As before, the architecture is composed of an encoder and a decoder, where the encoder is composed of a set of parametrised quantum gates $U(\theta)$, and its main aim is to take

[1]generally much lower dimensionality than the number of features

as input a set of n qubit quantum states $\{|\psi\rangle^{(i)}\}$; $i = 1, \ldots, N$ (we denote $\rho^{(i)} = |\psi\rangle^{(i)\,(i)}\langle\psi|$), and encode them into $k \leq n$ qubits. In **Fig 4.2**, the encoder takes 6 qubit states as inputs $\{|\psi\rangle^{(i)}\}$, and reduces them to 3 qubit states $\{\rho_{enc}^{(i)}\}$ as outputs. We want $\{\rho_{enc}^{(i)}(\theta)\}$ to be a set of pure states. Since quantum computations are reversible, the original set of states can be reconstructed applying $\mathcal{D} = \mathcal{E}^{\dagger}$. Therefore, the key difference in the quantum autoencoder case is that only the parameters of the Encoder $\mathcal{E}$ need to be learned. There are multiple ways to achieve this, however, a popular and simple approach is to train the encoder to push each training state $|\psi\rangle^{(i)}$'s subsystem $\{\rho_{tra}^{(i)}(\theta)\}$ that is not used for encoding (***trash state qubits***) to some reference state $|a\rangle$. Generally, the reference state is chosen to be some easy to prepare state such as $|0\rangle^{\otimes^{n-k}}$ where $n - k = 3$ as shown in **Fig: 4.2**). The equation pertaining to the above optimization is:

$$\max_{\theta} \mathcal{L}(\theta; \{|\psi\rangle^{(i)}\}) = \max_{\theta} \sum_{i} \langle a|\, \rho_{tra}^{(i)}(\theta)\, |a\rangle \begin{cases} \text{This can be computed easily} \\ \text{by using the SWAP test[30]} \end{cases} \tag{4.3}$$

The term inside the summation is the fidelity of the encoder's trash state with the reference state. If the reference state is picked to be $|a\rangle = |0\rangle^{\otimes^{n}}$ as in the case of **Fig: 4.2**, then we can just compute the sum of expectation values of the σ_z operator on each qubit of the **trash system** and maximize this quantity as the $|0\rangle^{\otimes^{n}}$ state gives the maximum expecatation value under this measurement

$$\max_{\theta} \mathcal{L}(\theta; \{|\psi\rangle^{(i)}\}) = \max_{\theta} \sum_{i} \frac{1}{3} \sum_{j=1}^{3} Tr[\sigma_z^j\, \rho^{(i)}] \tag{4.4}$$

4.3 Quantum Autoencoder Applications

Quantum Autoencoders have been applied to problems such as quantum error correction, denoising quantum data, compression, anomaly detection to name a few.

For efficient compression of data, a quantum autoencoder model in the context of quantum simulation produced good compressed representations for ground states of the Hubbard model and molecular Hamiltonians [6].

Quantum Autoencoders have also been used to successfully denoise Greenberger-Horne-Zeilinger (GHZ) states subject to spin-flip errors and random unitary noise [31].

In high energy physics, quantum autoencoders based on variational quantum circuits have been used to study the problem of anomaly detection at the Large Hadron Collider (LHC) [32].

Part II

Methods

Chapter 5

Product State Autoencoder

The following chapter introduces the rationale behind the Product State Autoencoder Architecture and a proposed protocol for its use in the problem of Multi Qubit Teleportation.

Building on the topics introduced in the background, we tackle the problem of converting a given dataset of pure n qubit quantum states $\{|\psi\rangle^{(i)}\}$ into a set of ***product states*** i.e. we are trying to learn a unitary $U(\theta)$

$$U(\theta) : \mathbb{C}^{2^n} \longrightarrow \mathbb{C}^{2^n} \tag{5.1}$$

$$U(\theta)\,|\psi\rangle^{(i)} = |\phi_1\rangle^{(i)} \otimes |\phi_2\rangle^{(i)} \ldots \otimes |\phi_k\rangle^{(i)}\,|0\rangle^{\otimes^{n-k}} \tag{5.2}$$

$$= \otimes^{j=1,\ldots,k}\,|\phi_j\rangle^{(i)}\,|0\rangle^{\otimes^{n-k}} \quad \Big\{\, |\phi_j\rangle^{(i)} \in \mathbb{C}^2; \tag{5.3}$$

5.1 Problem Discussion

The variational algorithm that solves **Eq: 5.3** needs to solve two problems at once i.e. It needs to encode a given set of quantum states to a set of lower-dimensional quantum states, and the encoded quantum states need to be product states.

The main aim of the above-mentioned problem is to find a mapping from our dataset space $\mathbb{D}$ to a space $\mathbb{P}$ where there is no entanglement between the various subsystems in $\mathbb{P}$. Since this problem deals with entanglement, the purity can be used as an easy-to-calculate measure of entanglement between subsystems (as mentioned in **Section 2.4**), and this problem can be converted to that of finding a unitary U that maps our dataset $\{|\psi\rangle^{(i)}\}$ to a space $\mathbb{P}$ where the purity of all subsystems for states in $\mathbb{P}$ is maximum. From here on, we refer to solving this problem as ***disentangling*** a set of quantum states.

5.2 Architecture Overview and Optimization

As mentioned previously, disentangling a single n qubit state $|\psi\rangle$ into a k qubit state involves learning a parametrised unitary $U(\theta) : |\psi\rangle \longrightarrow |\phi_1\rangle \otimes |\phi_2\rangle \ldots \otimes |\phi_k\rangle\,|0\rangle^{\otimes^{n-k}}$. The problem associated with **Eq: 5.3** motivates the use of some combination of the quantum autoencoder and purity circuit to achieve the goal of disentangling a given set of quantum states while encoding the dataset into a smaller dimensional Hilbert Space $\mathcal{H}$. If we have N states in our dataset, then we want to maximize

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\left(\text{fidelity}(|0\rangle^{\otimes n-k}, \rho_{tra}^{(i)}) + \frac{1}{n-k}\sum_{j=k+1}^{n}\text{purity}(\rho_j^{(i)})\right) \tag{5.4}$$

A variational algorithm can be employed to solve this problem. The cost function in **Eq: 5.4** has two parts. During the training of the circuit, the required measurements and post-processing should allow for the calculation of the purity of each compressed qubits as well as the closeness (fidelity) of the trash system with the reference state $|0\rangle^{\otimes n-k}$. In **Fig: 5.1**, the circuit is

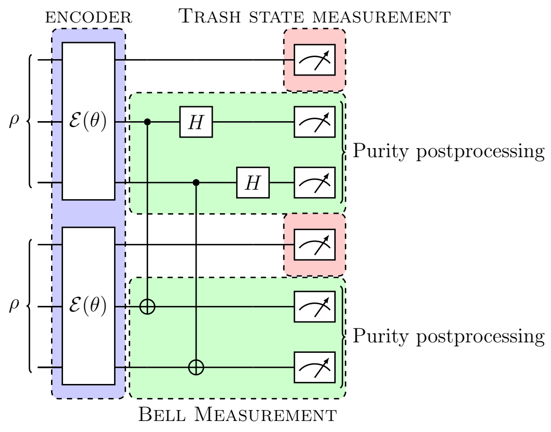


Figure 5.1: Product state autoencoder training architecture

constructed to disentangle a set of $n = 3$ qubit pure states $\{\rho^{(i)}\} = |\psi\rangle^{(i)}\,{}^{(i)}\langle\psi|$; $i = 1, \ldots, N$ to $k = 2$ qubits state. If the circuit was run N_Q times for each state in our dataset, and the measurements r are collected, then using **Eq: 5.4** for the example in **Fig: 5.1**, the cost function is:

$$\mathcal{L}(\theta, \{\rho^{(i)}\}) = \frac{1}{N}\sum_{i=1}^{N}\left(\left|\langle 0.\overset{n-k}{\cdots}.0|\,\rho_{tra}^{(i)}\,|0.\overset{n-k}{\cdots}.0\rangle\right|^2 + \sum_{j=k+1}^{n} P(r_j^{s^{(i)}})\,\text{isOdd}(r_j^{s^{(i)}})\right) \tag{5.5}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{k}\sum_{j=1}^{k} Tr[\sigma_z^i\,\rho^{(i)}] + \frac{1}{n-k}\sum_{j=k+1}^{n} Tr[(\rho_j^{(i)})^2]\right) \tag{5.6}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{k}\sum_{j=1}^{k} Tr[\sigma_z^i\,\rho^{(i)}] + \frac{1}{n-k}\sum_{j=k+1}^{n} Tr[(\rho_j^{(i)})^2]\right)\Big\{ \text{ using } \textbf{Eq: 2.15} \tag{5.7}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{k}\sum_{j=1}^{k}\left(\frac{2}{N_Q}\Big[\sum_{s=1}^{N_Q}\text{isZero}(r_j^{s^{(i)}})\Big] - 1\right)\right) + \left(\frac{1}{n-k}\sum_{j=k+1}^{n}\left(1 - \frac{2}{N_Q}\sum_{s=1}^{N_Q}\text{isOdd}(r_j^{s^{(i)}})\right)\right) \tag{5.8}$$

In order to perform optimization on this objective, we need to compute gradients of the parameters of the encoder $\mathcal{E}(\theta)$. New parameter shift rules need to be derived as the structure of our variational circuit is slightly different (we are using 2 copies of parametrised unitaries on two copies of a state). The derivation of the shift rules will be done for a single state n qubit $|\psi\rangle$

in the dataset with the task of compressing to a k qubit state and a reference state $\left|0.\overset{n-k}{\ldots}.0\right\rangle$ (the gradients for N states would then just be the average of the gradients for each individual state). The measurement statics for each iteration is computed by running the circuit N_Q times per iteration and the corresponding measurements r^j; $j = 1, \ldots, N_Q$ are collected for postprocessing. The proof below is adapted from the shift rule for Bell magic [15], with modifications specific to our problem:

$$
\begin{aligned}
\mathcal{L}(\theta, |\psi\rangle) =& \left(\frac{1}{k}\sum_{s=1}^{k}(2P(r_s = 0) - 1)\right) + \left(\frac{1}{n-k}\sum_{s=k+1}^{n} P(r_{s,s+n})\,\text{isOdd}(r_s.r_{s+n})\right)\\
\partial_{\theta_i}\mathcal{L}(\theta, |\psi\rangle) =& \left(\frac{1}{k}\sum_{s=1}^{k}(2\partial_{\theta_i}P(r_s = 0) - 1)\right) + \left(\frac{1}{n-k}\sum_{s=k+1}^{n} \partial_{\theta_i}P(r_{s,s+n})\,\text{isOdd}(r_s.r_{s+n})\right)
\end{aligned}
$$

We have to estimate 2 partial derivatives for the product state autoencoder training: $\partial_{\theta_i}P(r_s = 0)$ and $\partial_{\theta_i}P(r_{s,s+n})$, where $P(r_{s,s+n}) = \langle\psi(\theta)|\langle\psi(\theta)|\,O_{s,s+n}\,|\psi(\theta)\rangle|\psi(\theta)\rangle$ is the chance of measuring the Bell state $\sigma_{s,s+n}$ for the combined subsystem of the s and $(s+n)^{th}$ qubits. Using a parametrised circuit with d layers $\mathcal{E}(\theta)\otimes\mathcal{E}(\theta)$ producing a state $|\psi(\theta)\rangle|\psi(\theta)\rangle$, we can express $|\psi(\theta)\rangle = \prod_{i=m+1}^{d} V_i(\theta_i)W_i\,|\psi\rangle$ with fixed entangling gates W_i adn parametrised rotations $V_i(\theta_i) = e^{-i\frac{\theta_i}{2}\sigma_i}$ for some Pauli string σ_i. The derivative of this parametrised state is:

$$
\begin{aligned}
\partial_{\theta_i}|\psi(\theta)\rangle =& \prod_{i=m+1}^{d}[V_i(\theta_i)W_i](-i\frac{1}{2}\sigma_i)\prod_{i=1}^{m}[V_i(\theta_i)W_i]\,|\psi\rangle\\
=& U_i(-i\frac{1}{2}\sigma_i)\,|\omega_i\rangle \begin{cases}|\omega_i\rangle = \prod_{i=1}^{m} V_i(\theta_i)W_i\,|\psi\rangle\\ U_i = \prod_{i=m+1}^{d} V_i(\theta_i)W_i\end{cases}
\end{aligned}
$$

The derivative of $P(r_{s,s+n})$ using the product rule is:

$$
\partial_{\theta_i}P(r_{s,s+n}) = 2\,\langle\omega_i|\langle\omega_i|\,(U_i\otimes U_i)^{\dagger}O_{s,s+n}(U_i\otimes U_i)[(-i\frac{1}{2}\sigma_i)\otimes I]\,|\omega_i\rangle|\omega_i\rangle + h.c. \tag{5.9}
$$

Taking $O^{'}_{s,s+n} = (U_i\otimes U_i)^{\dagger}O_{s,s+n}(U_i\otimes U_i)$, and introducing an arbitrary factor v>0

$$
\begin{aligned}
\partial_{\theta_i}P(r_{s,s+n}) =& 2v\,\langle\omega_i|\langle\omega_i|\,O^{'}_{s,s+n}[(-i\frac{1}{2}\sigma_i)\otimes I]\,|\omega_i\rangle|\omega_i\rangle + h.c. \qquad (5.10)\\
=& v\,\langle\omega_i|\langle\omega_i|\,[(I - i\frac{1}{2v}\sigma_i)^{\dagger}\otimes I]O^{'}_{s,s+n}[(I - i\frac{1}{2v}\sigma_i)\otimes I]\,|\omega_i\rangle|\omega_i\rangle -\\
& v\,\langle\omega_i|\langle\omega_i|\,[(I + i\frac{1}{2v}\sigma_i)^{\dagger}\otimes I]O^{'}_{s,s+n}[(I + i\frac{1}{2v}\sigma_i)\otimes I]\,|\omega_i\rangle|\omega_i\rangle
\end{aligned}
$$

A rotation generated by a Pauli strings σ can be expressed as [24]

$$
e^{-i\frac{\pi}{4v}\frac{1}{2}\sigma} = \frac{1}{\sqrt{2}}(I - i\frac{1}{2v}\sigma) \tag{5.11}
$$

Substituting 5.11 in 5.10 yields

$$\begin{aligned}
&\partial_{\theta_i} P(r_{s,s+n}) = \\
&\qquad 2v \langle\omega_i| \langle\omega_i| [e^{-i\frac{\pi}{4v}\frac{1}{2}\sigma} \otimes I]^\dagger O'_{s,s+n} [e^{-i\frac{\pi}{4v}\frac{1}{2}\sigma} \otimes I] |\omega_i\rangle |\omega_i\rangle - \\
&\qquad 2v \langle\omega_i| \langle\omega_i| [e^{i\frac{\pi}{4v}\frac{1}{2}\sigma} \otimes I]^\dagger O'_{s,s+n} [e^{i\frac{\pi}{4v}\frac{1}{2}\sigma} \otimes I] |\omega_i\rangle |\omega_i\rangle \\
\implies &\partial_{\theta_i} P(r_{s,s+n}) = \\
&\qquad 2v \left\langle \psi(\theta + \frac{\pi}{4v} e_i) \right| \langle\psi(\theta)| O_{s,s+n} \left| \psi(\theta + \frac{\pi}{4v} e_i) \right\rangle |\psi(\theta)\rangle - \\
&\qquad 2v \left\langle \psi(\theta - \frac{\pi}{4v} e_i) \right| \langle\psi(\theta)| O_{s,s+n} \left| \psi(\theta - \frac{\pi}{4v} e_i) \right\rangle |\psi(\theta)\rangle \quad \begin{cases} e_i \text{ is a unit vector} \\ \text{to update } \theta_i \end{cases}
\end{aligned}$$

Using the same derivation, we get the other partial derivative $\partial_{\theta_i} P(r_s = 0)$:

$$\begin{aligned}
&\partial_{\theta_i} P(r_s = 0) = \\
&\qquad 2v \left\langle \psi(\theta + \frac{\pi}{4v} e_i) \right| \langle\psi(\theta)| O^{c_s} \left| \psi(\theta + \frac{\pi}{4v} e_i) \right\rangle |\psi(\theta)\rangle - \\
&\qquad 2v \left\langle \psi(\theta - \frac{\pi}{4v} e_i) \right| \langle\psi(\theta)| O^{c_s} \left| \psi(\theta - \frac{\pi}{4v} e_i) \right\rangle |\psi(\theta)\rangle \begin{cases} O^{c_s} = |0_s\rangle\langle 0_s| - |1_s\rangle\langle 1_s| \text{ is the} \\ \sigma_z \text{ expectation of qubit } s \end{cases}
\end{aligned}$$

Therefore, θ_i is shifted by $\theta \pm \frac{\pi}{4v} e_i$ in only the first encoder, and $\mathcal{L}$ is computed. Their difference is scaled by $2v$ to obtain $\partial\mathcal{L}_{\theta_i}$. This process is repeated for each parameter of $\mathcal{E}(\theta)$.
Using the gradients $\{\partial_\theta \mathcal{L}\}$, we can employ any gradient ascent algorithm to update the parameters θ, and iteratively maximize the objective function. Also, when the dataset has N states, the parameters shift rules are repeated for each state, and the gradients are averaged to obtain the final gradient for each iteration.

5.3 Dataset Discussion

Now that we have a way to maximize the subsystem purities for a given dataset of quantum states, the natural question to ask is **'Is it possible to encode any random set of quantum states into a set of product states?'**.

Theorem 5.3.1. *There doesn't always exist a unitary U that perfectly disentangles a set of quantum states $\{|\psi\rangle^{(i)}\}_{i=1}^N$*
Proof: *A unitary is a linear operator that preserves the norm of the state. The space of product states is a lower dimensional subspace. Therefore, not every set of states can be disentangled.*

We notice that the most general set of states $\{|\psi\rangle^{(i)}\}$ that can be disentangled are of the form presented in **Eq: 5.12**, as we know that there is at least one unitary $U^\dagger$ that would transform our dataset states into a set of product states.

$$|\psi\rangle^{(i)} = U \otimes_{j=1}^{k} |\phi_j\rangle^{(i)} |0\rangle^{\otimes n-k} \quad \begin{cases} U \text{ is an } n \text{ qubit Unitary} \\ |\phi_j\rangle^{(i)} \text{ is any arbitrary single qubit states} \end{cases} \tag{5.12}$$

Chapter 6

Product State Autoencoder Applications to Quantum Communication

*In **Chapter 6**, we revisit the Teleportation protocol in the case of multiple qubits and explore the application of the Product State Autoencoder model introduced **Chapter 5** to Multi Qubit Teleportation. We finally propose an extension to the Product State Autoencoder Architecture to extend its encoding capacity with the help of classical information.*

6.1 Transport of entangled states with qubit loss

Sending quantum information either through a physical quantum channel or teleportation has a certain probability of failure q. One main factor is due to noise present in the quantum channel. In the case of teleportation, failure to teleport a qubit could be related to the inability of linear optical methods to distinguish between 2 of the 4 bell states [33, 34]. This is an important step in order to send the correction bits to Bob in the teleportation protocol so that Bob can apply local transformations on his qubit to retrieve the exact quantum information that Alice wanted to send.

In the case of sending a single qubit state $|\psi\rangle$, if the protocol has a failure probability $P_f = q$, then the expected number of copies required for the protocol to succeed at least once is $L = \frac{1}{1-q}$. Therefore, for a reasonable P_f, the sender would not require many copies of $|\psi\rangle$ to successfully send the state.

In the case when the state to be sent is an n-qubit state with failure probability $P_f = q$ of each sending qubit, the protocol is only considered successful if all the qubits in a copy of the state were successfully sent. Failure to teleport even a single qubit would lead loss of entanglement information in the quantum state, thus requiring the sender to restart the protocol with a completely new copy of $|\psi\rangle$. Using a single copy of the state, the probability of the protocol succeeding is $(1-q)^n$. As n increases, the probability of success of the protocol decreases. If we had L copies of the n qubit state, and the protocol fails for all L copies, then this happens with a probability of

$$\epsilon = \left(1 - (1-q)^n\right)^L . \tag{6.1}$$

If we want the failure probability to be less than some value ϵ, then:

$$
\begin{aligned}
&(1-(1-q)^n)^L \leq \epsilon && (6.2)\\
&L\log(1-(1-q)^n) \leq \log\epsilon && (6.3)\\
&L \geq \frac{\log\epsilon}{\log(1-(1-q)^n)} && (6.4)\\
&\quad\approx \frac{\log\epsilon}{-(1-q)^n} && (6.5)\\
\implies &L \gtrsim -(1-q)^{-n}\log\epsilon && (6.6)
\end{aligned}
$$

where we assumed $(1-q)^n \ll 1$ in the approximation. This tells us that for a fixed ϵ, q the number of copies required for successful transport scales exponentially with the number of qubits. A potential solution to this problem is proposed in the next section.

6.2 Proposed Solution

To address the problem introduced in the previous section, the proposal is to use the Product State Autoencoder. The Product State Autoencoder compresses the incoming n-qubit states into a product state of k qubits. The compressed state is highly suited for transfer via a lossy channel, either via teleportation or physical transport. As mentioned previously, while teleportation for individual, single qubit states is well established, teleportation becomes exponentially more demanding for higher-dimensional states. Qubits can be teleported with a Bell pair and linear optics with a failure probability of $q = \frac{1}{2}$. With increasing Hilbert space, the resources needed increase substantially.

Physically transporting an entangled n qubit state via a lossy channel fails when even just one qubit is lost. Thus, here we have the failure probability $\epsilon = (1-(1-q)^n)^L$.

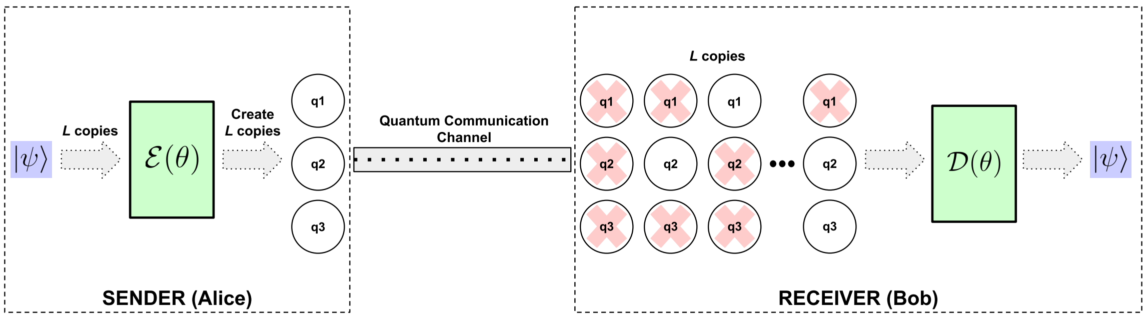


Figure 6.1: Applying the product state autoencoder to the teleportation/quantum channel transportation of a 3 qubit state. L copies of the state are used, and each qubit is teleported individually. The red x's denote that the qubit was not sent successfully.

With the compressed product state, we can achieve much better transfer rates. We assume we compress our state into k unentangled qubits, which we denote by q_1,q_2,...,q_k. We then transfer the state via a lossy channel with a failure probability q for each qubit. For example, for teleportation with photons we have $q \geq \frac{1}{2}$. We repeat the transfer for L copies the compressed state. From the successfully transferred qubits over the L copies, we select one qubit for each of the k qubits of the state. As the transferred qubits are not entangled and belong to a product states, we can freely pick out qubits from any copy without incurring any error. The decoding unitary is then applied on the selected k qubits to reconstruct the state.

For $k = 1$, the probability of successfully transferring at least one qubit is $1 - q^L$. For k qubits, the failure probability ϵ of transferring at least one qubit for each qubit of the state is

$$\epsilon = 1 - (1 - q^L)^k \tag{6.7}$$

Now, we choose the parameters such that $\epsilon \ll 1$ and $q^L \ll 1$. In this limit, we find the simplified relation

$$\frac{\log(\frac{\epsilon}{k})}{\log(q)} \lesssim L \tag{6.8}$$

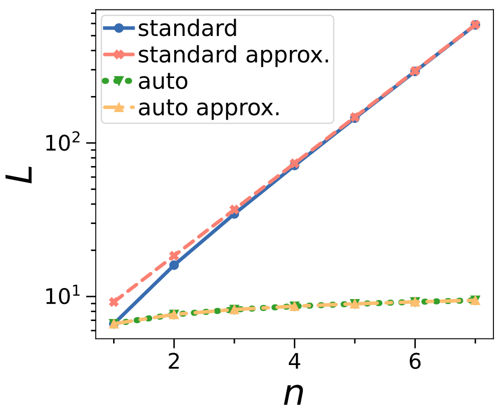


Figure 6.2: A comparison between the scaling of the number of copies L of n qubit states using the standard multiqubit teleportation protocol (standard) and our proposed product autoencoder solution (auto) for $q = 1, \epsilon = 0.01$, and $k = n$. We also show the approximations computed in **Eqs: 6.6, 6.8** (standard approx and auto approx)

In particular, L increases only logarithmically with the number of encoded qubits k. On the receiver side, the state can be reconstructed by composing the successfully transferred qubits and applying the decompression unitary $\mathcal{D}(\theta) = \mathcal{E}^\dagger(\theta)$. The success of the transfer of each qubit can be verified by measuring an unrelated degree of freedom of the system.

The scaling of the minimum number of copies L of states (against the number of qubits n) required to maintain the probability of failure under a certain ϵ is shown in **Fig: 6.2** for both the standard multiqubit teleportation protocol and our proposed solution (for $k = n$). We see that the number of copies of L required when using the product state autoencoder scales logarithmically in n as compared to the exponential scaling seen in the standard multiqubit teleportation protocol. We also notice that for larger n's, the approximations computed in **Eqs: 6.6, 6.8** tends to the exact formulas.

6.3 Extending PSA Capacity with Classical Information

In the standard autoencoder architecture, the encoded state has only k qubits, while the other $n-k$ qubits are trained to become a fixed trash state which is measured and discarded. Commonly, one chooses an easily preparable state such as a computational basis such as $|0\rangle^{\otimes^{n-k}}$) for the trash state. To decode the encoded state, the chosen trash state is prepared, and the decoding unitary is applied on the trash state and the encoded state.

However, one could allow for multiple trash states to extend the capacity of the autoencoder. We propose to allow all 2^{n-k} computational basis states as trash states. After the encoding, the state must have the form $|\psi_{\text{enc}}\rangle \otimes |m\rangle$, where $|\psi_{\text{enc}}\rangle$ is the encoded state and $|m\rangle$ is one of the 2^{n-k} computational basis states for the trash. The trash states are measured and the measurement result is transferred as classical information of $n-k$ bits to the decoder. Then, the decoder prepares the measured trash state and applies the decoding unitary. This step can enhance the total amount of states that can be encoded in the autoencoder by a factor of 2^{n-k}.

To train this extended autoencoder, we change the trash training term of the standard product autoencoer (5.5) to maximize the square of the expectation value of σ_z^j. This forces the trash state to be one of the computational basis states.

Part III

Experiments, Results and Evaluation

Chapter 7

Product State Autoencoder Experiments

*In **Chapters 7** and **8**, we show simulations of the product state encoder, and analyse the resources required to train it successfully.*

7.1 Technical Setup

The main python packages used in this thesis were Qutip [35] for toy data preparation, Pennylane [36] for the quantum circuit simulations and PyTorch for circuit training and optimization [37]. The simulation experiments were configured to run in parallel on Imperial College London - Department of Computing's High Throughput Computing system configured with HTCondor. Access to these systems, allowed for parallel runs of up to 274 quantum circuit different simulations at a time, thus drastically reducing the time required to test and develop the product state autoencoder.

7.2 Dataset preparation

The quantum states with which the product state autoencoder was trained were synthetically generated ***Haar random*** states. For example, if we want our product state autoencoder to learn n qubit states and produce k qubit encoded product states, then we synthetically create the dataset by first generating a Haar random unitary U. Next, to generate each quantum state $|\psi\rangle^{(i)}$ in our dataset, create k Haar random single qubit states $\{|\phi_j\rangle^{(i)}\}_{j=1..k}$ and a $|0\rangle^{\otimes N-K}$ state and apply the unitary U as follows:

$$|\psi\rangle^{(i)} = U \otimes_{j=1}^{k} |\phi_j\rangle^{(i)} |0\rangle^{\otimes n-k} \tag{7.1}$$

Using the Haar random metric to generate states and unitaries gives a uniform sample of states from the 2^n dimensional Hilbert Space $\mathcal{H}$ of the n qubits states (sampling from a Euclidean space would not give a uniform sample of states for our dataset). *Qutip* provides inbuilt methods to generate Haar random states. A comparison of the

Training datasets of various sizes ($d_{size} \in \{1, 2, 4, 8, 16, 32, 48\}$) were created for all pairs of n and k with $n \in \{2, 3, 4\}$ and $k \in \{1, 2, \dots, n\}$. A test set of size 48 quantum states was generated and used to check the performance of each trained product state autoencoder.

7.3 Training

The experiments were not run on real quantum hardware as we did not have access to an actual quantum computer. They were simulated using circuits created with Pennylane [36]. The benefit of using classical simulations for small problems is that any sort of training can be done in an ideal environment, and an efficient learning algorithm can be used with backpropagation. The structure of the variational circuit (encoder $\mathcal{E}(\theta)$) used to train the product autoencoder is shown in **Fig 7.1**.

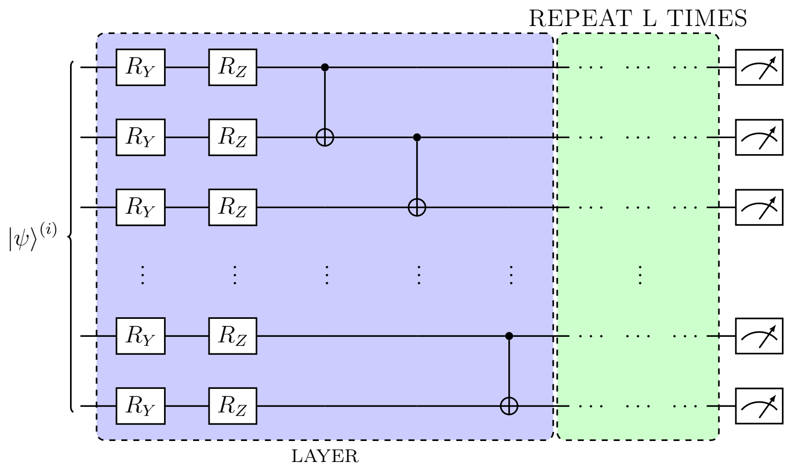


Figure 7.1: structure of the ansatz used for training of the product state autoencoder. The ansatz is composed as L repeated layers where in a layer each qubit has R_Y, R_Z parametrised gates followed by a nearest neighbour chain of $CNOT$ gates

For each (n, k) pair, a repetition is a newly generated Haar random unitary U and test dataset $\{|\psi\rangle_{test}^{(i)}\}$ with 48 samples. Each repetition was trained over varying number of layers and training dataset sizes.

Since we run simulations, we have a lot more flexibility to manipulate the states acted on by the encoder. For every input state $|\psi\rangle^{(i)}$, the output of the circuit $\mathcal{E}(\theta)$ is a parametrised state $|\psi(\theta)\rangle^{(i)}$. The objective is to learn a set of parameters θ such that the outputs become $\{|\psi(\theta)\rangle^{(i)}\} = \{\otimes_{j=1}^{K} |\phi_j'\rangle^{(i)} |0\rangle^{\otimes^{N-K}}\}$ where $|\phi_j'\rangle^{(i)}$ are single qubit states . During training, each training state is passed through the circuit, and the output states are collected (Note: this is possible because we are running classical simulations). For each output state, we compute the average purity of each of the first k qubits.

$$\text{purity}_{enc}(|\psi(\theta)\rangle^{(i)}) = \frac{1}{k}\sum_{j=1}^{k} Tr[(\rho_j^{(i)}(\theta))^2] \tag{7.2}$$

For each output state, we also compute $P_{tra}(|\psi(\theta)\rangle^{(i)}) = P(\rho_{tra}^{(i)}(\theta) = |0.\overset{n-k}{\ldots}.0\rangle\langle 0.\overset{n-k}{\ldots}.0|)$ - the total probability of all basis states with $|0\rangle^{\otimes^{n-k}}$ in the final $n-k$ qubits of the output state i.e. how close are is are the last $n-k$ qubits to the 0 state. This is quite easy to do as we have the output state vector (and in turn the probabilities associated with each basis state in the superposition). Thus, the cost that needs to be minimized is

$$\mathcal{L}(\{|\psi\rangle^{(i)}\}, \theta) = -\sum_i \text{purity}_{enc}(|\psi(\theta)\rangle^{(i)}) + \big(2P_{tra}(|\psi(\theta)\rangle^{(i)}) - 1\big) \tag{7.3}$$

Minimization of this cost function results in maximizing (for all output states $|\psi(\theta)\rangle^{(i)}$) the purity of the first k qubit subsystems, while simultaneously maximizing the probability of the last $n-k$ qubits of $|\psi(\theta)\rangle^{(i)}$ being in the $\big|0.\overset{n-k}{\ldots}.0\big\rangle$ state.

The new purity function was written using Pennylane, and PyTorch (instead of using Qutip's method) in order for the computations to be compatible with PyTorch's Automatic Differentiation. The training set loss $\mathcal{L}_{train}$, test set loss $\mathcal{L}_{test}$, gradients $\partial\theta$ of each parameter (computed with backpropagation) and the subsystem purities of each state $Tr[(\rho_j^{(i)}(\theta))^2]$ were also stored per iteration. A learning rate of 0.01 was used with the *Adam Optimizer*, for gradient descent. Finally, we also measure the fidelity. The fidelity F of a state is the metric that is normally used to determine how close a state ρ is to another state σ.

$$F(\rho, \sigma) = Tr[\rho\sigma] = F(\sigma, \rho) \tag{7.4}$$

We want the fidelity between an input state $|\psi\rangle$ and its corresponding reconstructed state $\mathcal{D}(|\psi(\theta)\rangle)$ to be close as possible to 1. The reconstructed state depends on which copies and qubits have been successfully transported. In the best case, we have at least one copy of the encoded state where all qubits have been successfully transported. In this case, this reconstructed state will always be equal to the input state, independent of the encoding unitary due to unitarity. In contrast, the worst case is where each qubit that was successfully transported belongs to a different copy. For example, for two transported copies of a 2 qubit state, only the first qubit of the first copy, and the second qubit of the second qubit were succesfully transported. If the encoding did not remove entanglement completely, the qubit losses will lead to a loss of quantum information and result in a mixed state. Thus, the fidelity between between input and reconstructed state is reduced. We now compute the fidelity of the worst case scenario F_{wc}. We assume an n qubit state encoded using a unitary $U(\theta)$ ($U^\dagger$ is the decoder) to an $k = n$ qubit state $|\psi(\theta)\rangle$ and the encoded state $|\psi(\theta)\rangle\langle\psi(\theta)| = \rho$. Further, in the worst case the state of the jth transported qubit is given by the partial trace over the encoded state except the jth qubit. Then, the fidelity is given by

$$F_{wc} = F(U\rho_1 \otimes \rho_2 \ldots \otimes \rho_n U^\dagger, U\rho U^\dagger) \tag{7.5}$$

Since Fidelity is invariant under unitary transformation we can compute the fidelity as

$$F_{wc} = F(\rho_1 \otimes \rho_2 \ldots \otimes \rho_n, \rho) \tag{7.6}$$

7.4 Results and Evaluation

As mentioned previously, the experiments were run for 10 repetitions for various (n, k) pairs, by varying over the number of layers and training dataset sizes. Each run was trained for 600 epochs with batch gradient descent using the Adam optimizer. We set the learning rate lr to 0.01, and trained on the Adam optimizer's default PyTorch parameters (betas=(0.9, 0.999), eps=10^{-8}, weight$_{decay}$=0)

7.4.1 Convergence Analysis

The test set errors for the product state autoencoder is shown below for the case of $n = 4, k = 4$.

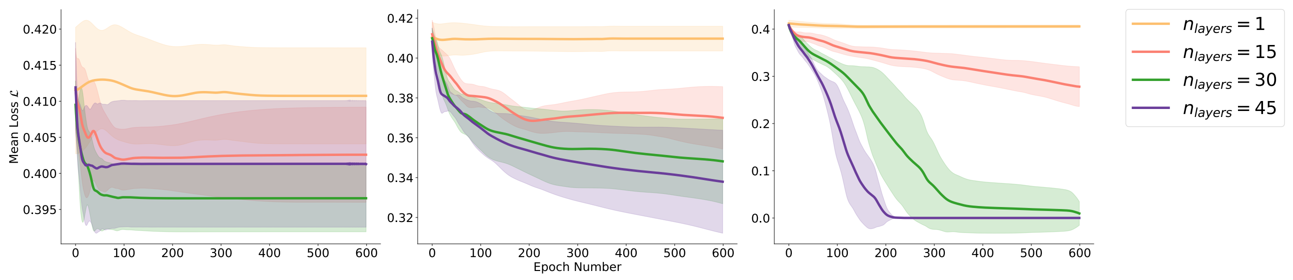


Figure 7.2: Mean and standard deviation of $\mathcal{L}_{text}$ for 48 test quantum state. $n = 4, k = 4$ and training dataset sizes $d_{size} = 4$ (left), $d_{size} = 16$ (middle) and $d_{size} = 48$ (right).

We can clearly see from the plots in **Fig: 7.2** that convergence improves as we increase the number of training datapoints. Fewer training datapoints would lead to solutions that may perfectly fit the training data, however the circuit would not have enough information to grasp the structure of the entire dataspace The variational circuit also needs to be overparameterised in order to converge to a solution as this would give the circuit access to a larger part of the Hilbert Space $\mathcal{H}$. We see in the case of the $n_{layers} = 30, 45$ layers, the circuit is ***overparametrised*** and is able to arrive at a solution, as compared to the ***underparametrised*** $n_{layers} = 1$ and $n_{layers} = 15$. For circuit of with n_{layers} layers, the number of parameters the circuit contains (based on the *ansatz* structure we chose for this problem) is given by:

$$M = 2n \times n_{layers} \tag{7.7}$$

In general, the number of free parameters required to completely describe any n qubit unitary is of the order $M = 4^n$. Therefore, to learn a $n = 4$ qubit unitary in the case of the product state autoencoder, **Eq: 7.7** gives $n_{layers} = \frac{4^n}{2n} = 32$. Which is in line with our observations in **Fig: 7.2**.

However, we make an interesting observation for the cases where $k < n$. In **Fig: 7.3** we show the training runs for $n = 4, k = 2$. We notice that the training not only converges for fewer layers $n_{layers} = 15$, but for fewer datapoints $d_{size} = 16$. This is due to the the constraint on the size of the encoding number of qubits k.

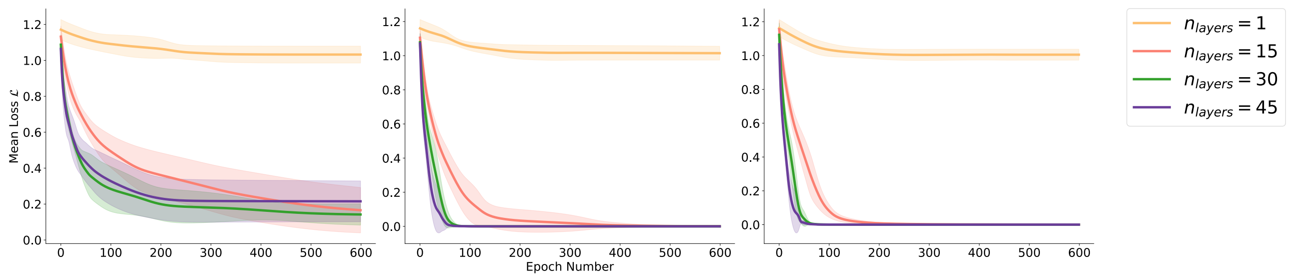


Figure 7.3: Mean and standard deviation of 48 test quantum state for $n = 4, k = 2$ and training dataset sizes $d_{size} = 4$ (left), $d_{size} = 16$ (middle) and $d_{size} = 48$ (right).

We also notice a trend in the scaling for the size of the training dataset to achieve an ideal performance on the test dataset. If the number of datapoints in the train dataset is low, then the problem becomes ***underconstrained*** and the model may not be able to fully uncover the product state structure of the underlying data. As more data is added to the training set, we reach a point when the problem becomes ***overconstrained***. At this point, the model converges to the global minima.

7.4.2 Error Analysis

Further analysis was done on the errors achieved for various combinations of training dataset sizes and layer sizes. The minimum errors $\mathcal{L}$ were computed for each repetition, and the median of these errors were extracted for further analysis.

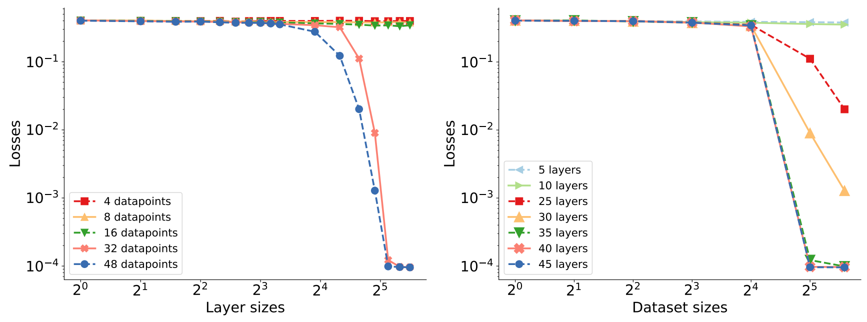


Figure 7.4: Median of errors over 10 repetitions for $n = 4, k = 4$. The graph on the left shows the errors achieved for different datasizes over increasing layer sizes. The graph on the right shows the errors achieved for different layers sizes over increasing layer sizes.

In **Fig: 7.4** (right), an interesting characteristic of the problem is observed for increasing n_{layers}. When $n_{layers} < 35$, it is very obvious that the problem is not able to converge to the minimum. However, when n_{layers} increases to 35, we notice a transition in the region $16 \leq d_{size} \leq 32$. In our case, while varying the n_{layers}, the problem reaches a point where the circuit is moves from being underparametrised to overparametrised. This transition leads to a rapid decrease in the **energy** (loss $\mathcal{L}_{test}$). A similar transition in **Fig: 7.4** (left) for $d_{size} \geq 32$ in the region $20 \leq n_{layers} \leq 35$. This further corroborates our findings in the **Subsection: 7.4.1** that circuit needs to be overparametrised in order to achieve converge to a global minimum. The median losses for the other n, k qubit pairs are shown in **Figs: 7.5, 7.6, 7.7, 7.8, 7.9, 7.10, 7.11**

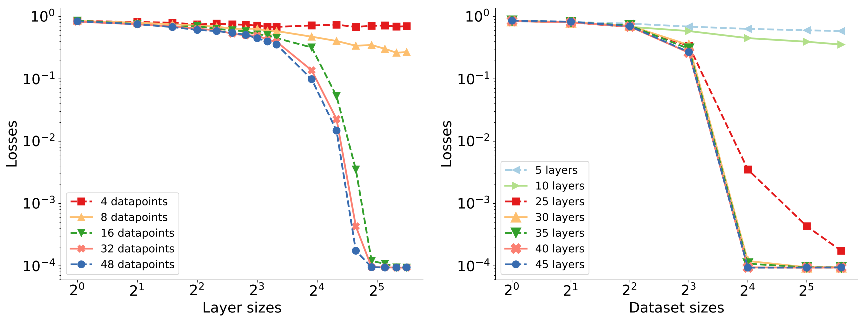


Figure 7.5: Median of errors over 10 repetitions for $n = 4, k = 3$

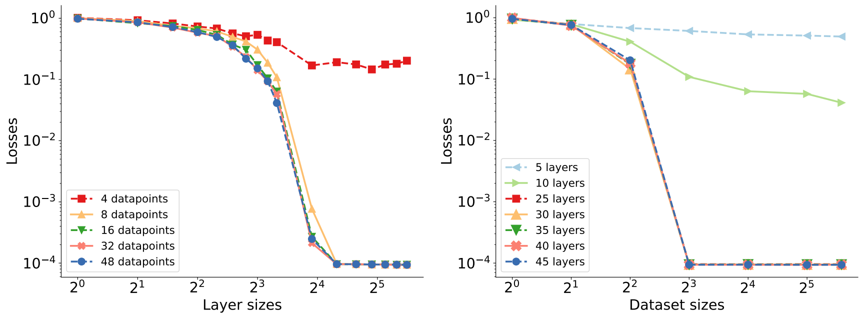


Figure 7.6: Median of errors over 10 repetitions for $n = 4, k = 2$

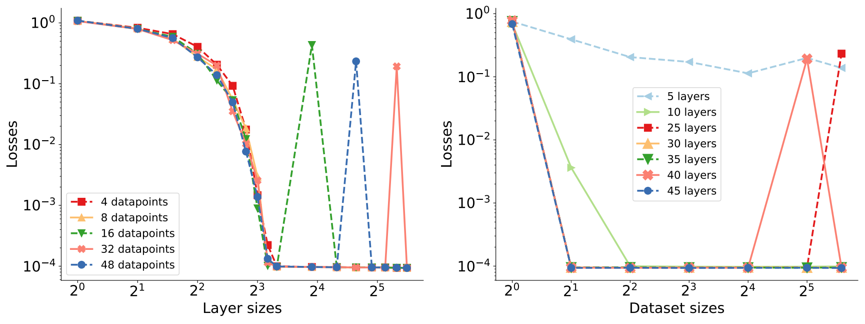


Figure 7.7: Median of errors over 10 repetitions for $n = 4, k = 1$

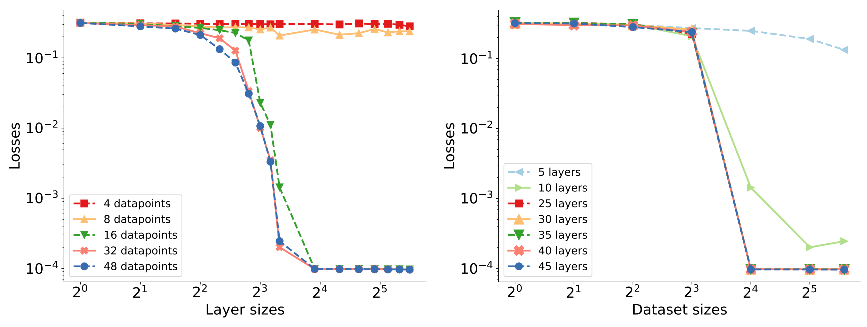


Figure 7.8: Median of errors over 10 repetitions for $n = 3, k = 3$

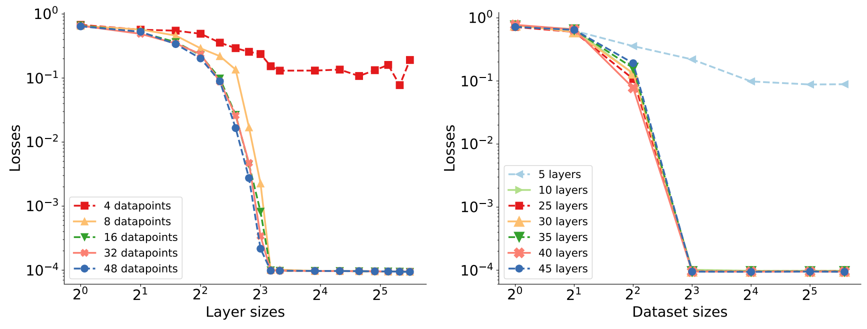


Figure 7.9: Median of errors over 10 repetitions for $n = 3, k = 2$

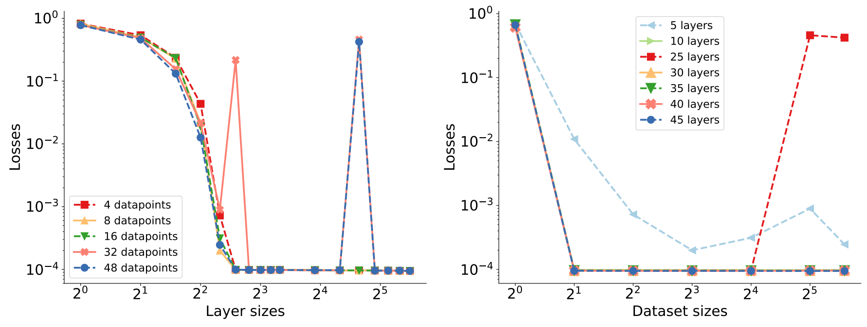


Figure 7.10: Median of errors over 10 repetitions for $n = 3, k = 1$

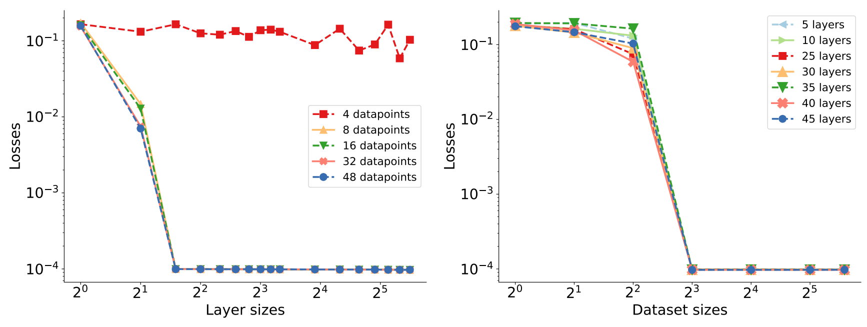


Figure 7.11: Median of errors over 10 repetitions for $n = 2, k = 2$

Another interesting point to note in the case **Figs: 7.7, 7.10**, where $k = 1$, is that the convergence of the model is affected for large layer sizes and large number of datapoints. The model seems to get stuck in a local minima. Exploration with higher learning rates and a learning rate scheduler may help with escaping the local minima. Also, the batch was set to the whole training set. It may be more beneficial to use a smaller batch size, when $d_{size} > 8$ for example.

7.5 Scaling Analysis

We plot the minimum number of parameters required for a different n, k to achieve a loss below a threshold value of 0.01,

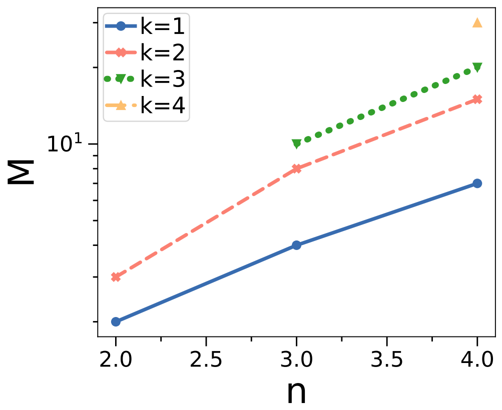


Figure 7.12: minimum number of parameters required to achieve loss below 0.01

In **Fig: 7.12**, we notice that for a fixed k, the number of parameters required scales exponentially in n. The number of free parameters to describe any mapping from an n qubit Hilbert Space, to a k qubit Hilbert Space is of the order $M = \mathcal{O}(2^{n+k})$. This can be easily verified as only 2^k columns of the generating unitary $U \otimes_{j=1}^{K} |\phi_j\rangle^{(i)} |0\rangle^{\otimes^{n-k}} = |\psi\rangle^{(i)}$ are responsible for the creation of the dataset. The number of parameters required to describe this specific unitary is of the order $M = 2^n \times 2^k = 2^{n+k}$.

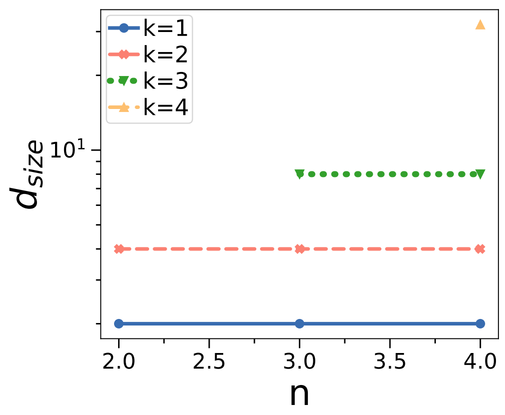


Figure 7.13: minimum number of training datapoints d_{size} required to achieve loss below 0.01

In **Fig: 7.13**, we plot the minimum number of training datapoints to achieve a loss below a threshold of 0.01 over different combinations of n, k. As the graphs show, the minimum number of datapoints remains the same for varying n. For a fixed n, the d_{size} scales exponentially in k, showing that the number of datapoints needed is only a function of k.

Chapter 8

Capacity Extended Product State Autoencoder Experiments

For the same number of qubits, the **capacity extended product state autoencoder** reuses the same architecture as the **product state autoencoder**, with one key difference: there is no longer a fixed reference state. By forcing the trash state of different input states to one of the computational basis states, we can increase the amount of information that can be encoded into the outputs states of the encoder. These computational basis trash states can then be measured and sent classically. The sections below will cover the dataset preparation, training simulations and result analysis of the capacity extended product state autoencoder.

8.1 Dataset preparation

The quantum states with which the capacity extended product state autoencoder was trained were synthetically generated haar random states as in the case of the normal product state autoencoder. The main difference is that the trash state is a randomly generated computational basis state. Therefore the dataset is of the form

$$|\psi\rangle^{(i)} = U \otimes_{j=1}^{k} |\phi_j\rangle^{(i)} |c\rangle^{(i)} \quad \begin{cases} |c\rangle^{(i)} \text{ is any } (n-k) \text{ qubit computational basis state} \end{cases} \tag{8.1}$$

Training datasets of various sizes ($s \in \{1, 2, 4, 8, 16, 32, 48\}$) were created for all pairs of n and k with $n \in \{2, 3, 4\}$ and $k \in \{1, 2.., n-1\}$. A test set of size 48 quantum states was generated and used to check the performance of each trained product state autoencoder.

8.2 Training

As in the case of the product state encoder's training (**Chapter 7**), the capacity extended product state encoder simulations are trained in a similar manner with a slight modification to the final cost function. For every input state $|\psi\rangle^{(i)}$, the output of the encoder is a parametrised state $|\psi(\theta)\rangle^{(i)}$. However, instead of computing $P_{tra}(|\psi(\theta)\rangle^{(i)}) = P(\rho_{tra}^{(i)}(\theta) = |0.\overset{n-k}{\cdots}.0\rangle \langle 0.\overset{n-k}{\cdots}.0|)$

$$P_{\text{basis}}(|\psi(\theta)\rangle^{(i)}) = \sum_{j=k+1}^{n} \left({}^{(i)} \langle \psi(\theta)| \sigma_z^j |\psi(\theta)\rangle^{(i)} \right)^2 \tag{8.2}$$

8.3 Results and Evaluation

8.3.1 Convergence Analysis

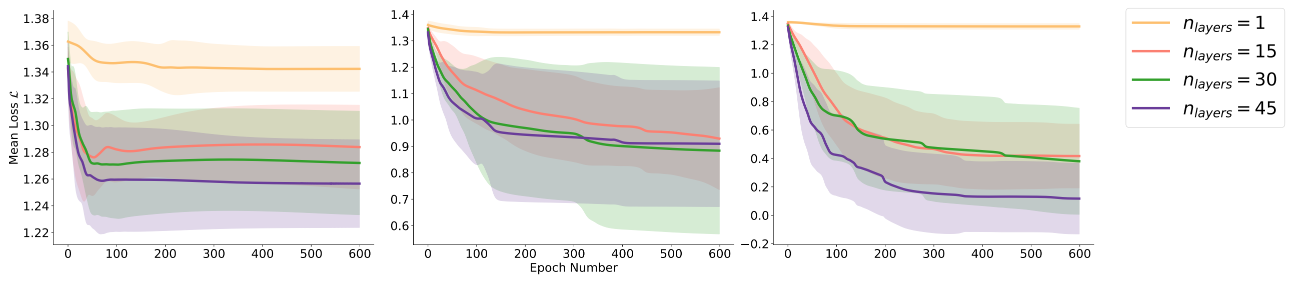


Figure 8.1: Mean and standard deviation of 48 test quantum state for $n = 4, k = 3$ (and $4 - 3 = 1$ a single qubit is used to increase capacity by represent 2 bits of classical information) and training dataset sizes $d_{size} = 4$ (left), $d_{size} = 16$ (middle) and $d_{size} = 48$ (right)

The model convergence in the capacity extended case for $n = 4, k = 3$ (**Fig: 8.1**) is not as smooth as the same problem for the normal product autoencoder. The large variance is most likely attributed to the distribution of the samples with different computational basis states ($|0\rangle$ and $|1\rangle$ for the case where $k = n - 1$). Even though the computational basis states were uniformly sampled for the dataset creation, it does not guarantee a balanced allocation as our dataset sizes are quite small.

8.3.2 Error Analysis

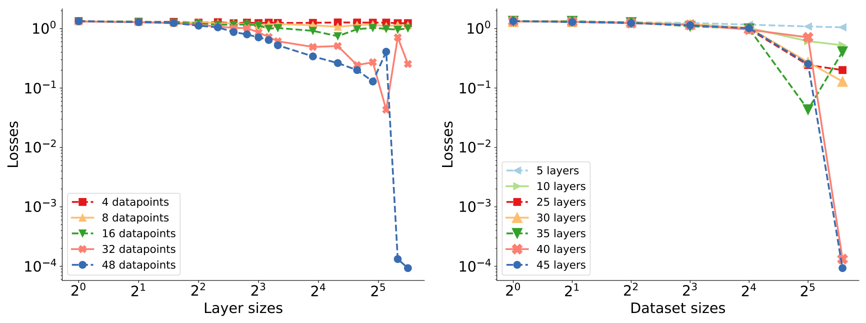


Figure 8.2: Median of errors over 10 repetitions for $n = 4, k = 3$ for capacity extended autoencoder. The graph on the left shows the errors achieved for different datasizes over increasing layer sizes. The graph on the right shows the errors achieved for different layers sizes over increasing layer sizes.

From **Fig: 8.2**, it comes as no surprise that we need atleast twice the number of datapoints in the capacity extended encoder as compared to the same $n = 4, k = 3$ case in the normal product state autoencoder (with no capacity extended **7.5**). The single qubit used for increasing the capacity of the network add a new constraint to the problem.

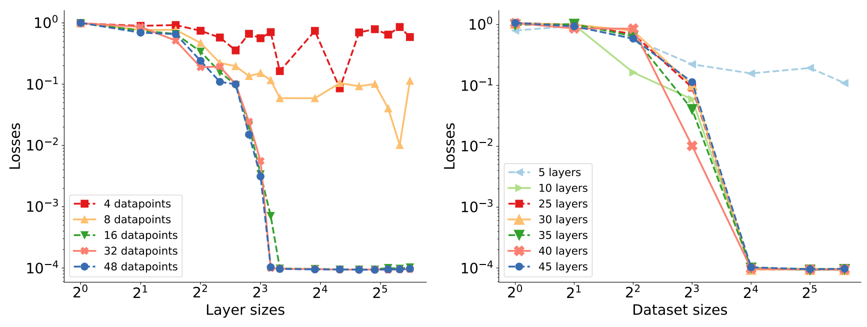


Figure 8.3: Median of errors over 10 repetitions for $n = 3, k = 2$ for capacity extended autoencoder

In the case of $n = 3, k = n - 1 = 2$, we again see that twice the d_{size} is required as compared to the same setting in the normal product state autoencoder. For cases where $k < n - 1$, we notice that convergence is not achieved for the median runs. We believe that the reason could be due the imbalance in the basis states as seen in **Figs: 8.4, 8.5**

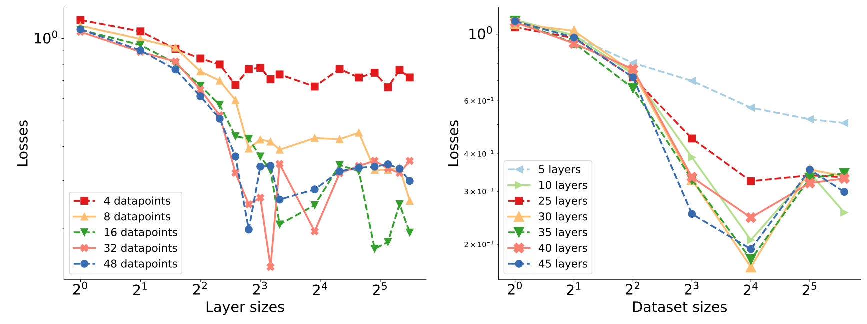


Figure 8.4: Median of errors over 10 repetitions for $n = 3, k = 1$ for capacity extended autoencoder

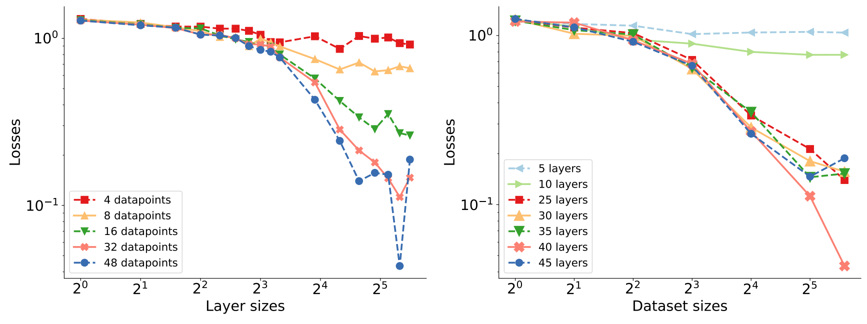


Figure 8.5: Median of errors over 10 repetitions for $n = 4, k = 1$ for capacity extended autoencoder

Part IV

Technical Contributions, Conclusion and Future Work

Chapter 9

Technical Contributions

In this work, we managed to

- use the purity of quantum states in a quantum autoencoder to learn to disentangle a set of quantum states,
- propose a new protocol to perform multi-qubit teleportation using the above method of disentangling quantum states and prove that the protocol works through numerical simulations,
- derive new parameter shift gradient rules for parametrized quantum circuits when using purity as a cost function,
- propose an extension of the autoencoder based on measured classical information to enhance its capacity,
- adapt the product state autoencoder experiment code to utilize and run on a high-throughput computing (HTC) cluster with up to 274 machines in parallel.

Chapter 10

Conclusion

This thesis demonstrates a new type of variational quantum autoencoder that compresses entangled states into unentangled product states. We apply our **Product State Autoencoder** to the problem of multi-qubit quantum teleportation. For a direct approach to quantum teleportation, one requires at least $\mathcal{O}(\exp(n))$ copies of a state to teleport an entangled n-qubit state. In contrast, our method requires only $\mathcal{O}(\log(n))$ copies.

First, we covered the basic building blocks governing quantum information and quantum computation. We designed our algorithm with the limitations of currently available quantum computers in mind. These Noisy Intermediate Scale Quantum (NISQ) computers allow only for quantum computations involving a low number of gates. Variational Quantum Algorithms (VQAs) have been proposed to design algorithms that can function even with these restrictions. These algorithms rely on a hybrid quantum-classical feedback loop to iteratively minimize a cost function. The output of the quantum computer is processed on a classical computer and then fed back into the quantum computer.

We adapt the variational quantum autoencoder, which is used to compress quantum states to lower dimensions, to compress entangled quantum states into unentangled product states. This algorithm, which we call the **Product State Autoencoder**, can be used to encode n qubit states into a $k \leq n$ qubit product state. In **Chapter 6**, we show that teleporting these k qubit product states and then reconstructing them on the receiver's end requires only $\mathcal{O}(\log(k))$ copies of the state, in contrast to $\mathcal{O}(\exp(n))$ copies required for a direct n qubit teleportation protocol. We then proposed an extension to this product state autoencoder architecture, called the **Capacity Extended Product State Autoencoder**. This model makes use of the $n-k$ qubits (trash state qubits not used in the k qubit encoding) to encode one of the 2^{n-k} computational basis states, thus increasing capacity of the autoencoder. This state is then measured and the corresponding binary measurement information is sent classically to the receiver in the teleportation protocol.

We ran simulations of both the product state autoencoder and capacity extended product state autoencoder for n qubit $n = 2, 3, 4$ input states k qubit $k = 1, \ldots, n$ compressed states. These experiments were run for 10 repetitions on varying training dataset sizes $1 \leq d_{size} \leq 48$ and the parametrised encoder's layers $1 \leq n_{layers} \leq 45$. Analysis on the product state autoencoder's experiments for any (n, k) pairs showed that for large enough d_{size}, a transition is observed when varying n_{layers}. This is due to the shift from an underparametrised circuit to an overparametrised one. A transition is also observed when we vary the d_{size} for a sufficiently overparametrised circuit. This occurs because the mapping inferred from the training dataset's structure, transitions from an underconstrained problem to an overconstrained one above a certain d_{size}.

Chapter 11

Future Work

This work shows the benefits of using the Product State Autoencoder for the transport and teleportation of multi qubit entangled states. However, we believe this algorithm could also be used as a subroutine in other protocols and algorithms. For example, product states are known to be more robust to noise, where our algorithm could be applied to enhance robustness of quantum memory.

Our numerical simulations in **Chapters 7** and **8** explore the trainability of the product state autoencoder and the the minimum number of layers n_{layers} and training dataset d_{size} required for various configurations of number of qubits n and number of encoding qubits k. The scaling is shown to be exponential in $n + k$ for both the n_{layers} and d_{size}. However, there may be more constrained datasets that could show a more efficient scaling for the parameters d_{size}, n_{layers}. For example, if the structure of the dataset generating process is known beforehand, then the same structure could be replicated in the Product State Encoder, and trained. Further analysis on this could uncover a more efficient scaling for the product encoder's parameters based on the problem.

In this thesis, we did not put a heavy emphasis on hyperparameter tuning for the optimizer. However, in certain small training instances (results not discussed in this thesis), starting with a higher learning rate and setting a learning rate scheduler did show faster convergence to a global minima. Nonetheless, this needs to be investigated further, and could lead to some interesting findings. The same can be said about the structure of the ansatz used. Ansatz selection is an open problem in for the field of variational algorithms, and can be explored further in the context of the product state autoencoder.

The Product State Autoencoder was trained using classical simulations, and in an ideal scenario. We did not have access to an actual quantum computer to test our proposal. Therefore, one of the next steps should be to run the algorithm on quantum hardware. The noise induced gradients, both due to decoherence (minimal) and number of shots might even help with the training and escaping local minima (we stumbled upon this local minima problem while training capacity extended autoencoder). The noise can also have a negative impact on the training and performance of the autoencoder. An interesting future direction is to study the resilience of autoencoders to experimental noise.

Bibliography

[1] Nielsen MA, Chuang I. Quantum computation and quantum information. American Association of Physics Teachers; 2002.

[2] Bennett CH, Brassard G, Crépeau C, Jozsa R, Peres A, Wootters WK. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. Physical review letters. 1993;70(13):1895.

[3] Bouwmeester D, Pan JW, Mattle K, Eibl M, Weinfurter H, Zeilinger A. Experimental quantum teleportation. Nature. 1997;390(6660):575-9.

[4] Boschi D, Branca S, De Martini F, Hardy L, Popescu S. Experimental realization of teleporting an unknown pure quantum state via dual classical and Einstein-Podolsky-Rosen channels. Physical Review Letters. 1998;80(6):1121.

[5] Bharti K, Cervera-Lierta A, Kyaw TH, Haug T, Alperin-Lea S, Anand A, et al. Noisy intermediate-scale quantum algorithms. Reviews of Modern Physics. 2022;94(1):015004.

[6] Romero J, Olson JP, Aspuru-Guzik A. Quantum autoencoders for efficient compression of quantum data. Quantum Science and Technology. 2017;2(4):045001.

[7] Benioff P. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. Journal of statistical physics. 1980;22(5):563-91.

[8] Feynman RP. Simulating physics with computers. In: Feynman and computation. CRC Press; 2018. p. 133-53.

[9] Shor PW, et al.. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. Los Alamos Physics Preprint Archive; 1995.

[10] Briegel HJ, Browne DE, Dür W, Raussendorf R, Van den Nest M. Measurement-based quantum computation. Nature Physics. 2009;5(1):19-26.

[11] Farhi E, Goldstone J, Gutmann S, Lapan J, Lundgren A, Preda D. A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. Science. 2001;292(5516):472-5.

[12] Williams CP. Quantum gates. In: Explorations in Quantum Computing. Springer; 2011. p. 51-122.

[13] Romero Fontalvo J. Variational Quantum Information Processing; 2019.

[14] Garcia-Escartin JC, Chamorro-Posada P. Swap test and Hong-Ou-Mandel effect are equivalent. Physical Review A. 2013;87(5):052330.

[15] Haug T, Kim M. Scalable measures of magic for quantum computers. arXiv:220410061. 2022.

[16] Kay A. Tutorial on the quantikz package. arXiv preprint arXiv:180903842. 2018.

[17] Montanaro A. Learning stabilizer states by Bell sampling. arXiv:170704012. 2017.

[18] Grover LK. A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing; 1996. p. 212-9.

[19] Martin-Lopez E, Laing A, Lawson T, Alvarez R, Zhou XQ, O'brien JL. Experimental realization of Shor's quantum factoring algorithm using qubit recycling. Nature photonics. 2012;6(11):773-6.

[20] Ball P, et al. First quantum computer to pack 100 qubits enters crowded race. Nature. 2021;599(7886):542-2.

[21] Farhi E, Goldstone J, Gutmann S. A quantum approximate optimization algorithm. arXiv preprint arXiv:14114028. 2014.

[22] Haug T, Bharti K, Kim MS. Capacity and Quantum Geometry of Parametrized Quantum Circuits. PRX Quantum. 2021 Oct;2:040309.

[23] Mitarai K, Negoro M, Kitagawa M, Fujii K. Quantum circuit learning. Physical Review A. 2018;98(3):032309.

[24] Schuld M, Bergholm V, Gogolin C, Izaac J, Killoran N. Evaluating analytic gradients on quantum hardware. Physical Review A. 2019;99(3):032331.

[25] Wierichs D, Izaac J, Wang C, Lin CYY. General parameter-shift rules for quantum gradients. Quantum. 2022;6:677.

[26] Kyriienko O, Elfving VE. Generalized quantum circuit differentiation rules. Physical Review A. 2021;104(5):052417.

[27] Izmaylov AF, Lang RA, Yen TC. Analytic gradients in variational quantum algorithms: Algebraic extensions of the parameter-shift rule to general unitary transformations. Physical Review A. 2021;104(6):062443.

[28] Sireesh A. Approximate Parameter Shift Rules: Independent Study Option, Imperial College London. GitHub; 2022. `https://github.com/AdithyaSireesh/approx-parameter-shift-rules`.

[29] MathWorks. Autoencoders; 2022. `https://uk.mathworks.com/discovery/autoencoder.html/`.

[30] Barenco A, Berthiaume A, Deutsch D, Ekert A, Jozsa R, Macchiavello C. Stabilization of quantum computations by symmetrization. SIAM Journal on Computing. 1997;26(5):1541-57.

[31] Bondarenko D, Feldmann P. Quantum autoencoders to denoise quantum data. Physical review letters. 2020;124(13):130502.

[32] Ngairangbam VS, Spannowsky M, Takeuchi M. Anomaly detection in high-energy physics using a quantum autoencoder. Physical Review D. 2022;105(9):095004.

[33] Michler M, Mattle K, Weinfurter H, Zeilinger A. Interferometric Bell-state analysis. Physical Review A. 1996;53(3):R1209.

[34] Calsamiglia J, Lütkenhaus N. Maximum efficiency of a linear-optical Bell-state analyzer. Applied Physics B. 2001;72(1):67-71.

[35] Johansson JR, Paul D. Nation, and Franco Nori. Qutip: An open-source python framework for the dynamics of open quantum systems. Computer Physics Communications. 2012;183(8):1760-72.

[36] Bergholm V, Izaac J, Schuld M, Gogolin C, Alam MS, Ahmed S, et al. Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv preprint arXiv:181104968. 2018.

[37] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E, Garnett R, editors. Advances in Neural Information Processing Systems 32. Curran Associates, Inc.; 2019. p. 8024-35. Available from: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.