A Study on Expressibility and Entangling Capacity in Parametrized Quantum Circuits

A Dissertation submitted by

Anubhav Trivedi
CS2030

under the guidance of

Prof. Susmita Sur-Kolay, Indian Statistical Institute
Dr. Shesha Raghunathan, IBM Research Labs

in partial fulfilment of the requirements
for the award of the degree of

Master of Technology
in
Computer Science

INDIAN STATISTICAL INSTITUTE

THESIS CERTIFICATE

This is to certify that the thesis titled **A Study on Expressibility and Entangling Capacity in Parametrized Quantum Circuits**, submitted by **Anubhav Trivedi (CS2030)**, to the Indian Statistical Institute, Kolkata for the award of the degree of **Master of Technology in Computer Science**, is a *bona fide* record of the research work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Susmita Sur-Kolay

Prof Susmita Sur-Kolay 20.07.2022
Professor
ACMU
ISI Kolkata
Date: July 20, 2022

Dr Shesha Raghunathan
Senior Engineer
IBM Research Labs
Date: July 20, 2022

ACKNOWLEDGEMENTS

First and foremost, I am really grateful to my Thesis Advisor, Prof Susmita Sur-Kolay and the whole IBM Quantum team without whose support, continuous help and discussions, I would not had been able to do this thesis. All those long hours of discussions about the ideas and results of my experiments were really enjoyable. From the IBM Quantum team, I would especially like to thank Dr Shesha Raghunathan, Dr Anupama Ray and Dr Dhinakaran Vinayaghamurti for always being there whenever I was in need of any kind of help in the duration of this thesis.

I would also like to thank my friend Anurag Pateria, who provided help with understanding concepts which I was having difficulty in understanding. Other than that, I would like to thank my friends and classmates (Nikhil, Suraj, Nitesh et al.) for all our hours of fun and enjoyment. Without their moral support, this MTech would not had been so productive and enjoyable.

I would also like to thank all the professors of ISI who taught us so many subjects of Computer Science. There expertise and wisdom of knowledge that they have shared with us will always remain the base of everything I will do in my entire professional career. I would especially like to thank Prof Ansuman Banerjee for helping with selecting thesis topics and courses and pushing me in the right directions.

Last but not least, I would like to thank my parents for having continuous faith in me that I can do what I dreamt of doing which actually led me to this Institute which I would be forever be thankful to ISI.

ABSTRACT

In Hybrid Quantum Classical (HQC) Algorithms, the Parameterized Quantum Circuits (PQC) play a very important role. However one of the biggest challenges in implementing these HQC Algorithms is choosing an effective circuit which can represent the solution space properly and at the same time is feasible in implementation (having low circuit complexity, depth and number of parameters).

Expressiblity and Entangling Capacity are two such measures which quantifies the extent of solution space that can be covered by a particular PQC. Expressibility quantifies the ability of the PQC to generate pure states that are well representative the Hilbert Space. On the other hand the Entangling Capacity quantifies the ability of the PQC to generate entangled states. The advantage of highly entangled states in PQCs of low depth can be the ability to efficiently represent the solution space for tasks such as ground state preparation, or data classification or capturing non trivial correlation in quantum data.

Both of these metrics can be estimated statistically. In this dissertation, we have taken 19 different Parameterized Quantum Circuits (all of 4 qubits) of different structures having varying circuit complexity, computed Expressibility and entangling capacity. We studied the effect of topology of the machine and the error in the computed Expressibility. We also found the limitation of Entangling Capacity that we can't compute it for Circuits on a Quantum Channel. Finally we considered a Data Science Problem on Fraud Transaction Detection to see how the Quantum Neural Network with all the considered Parameterized Quantum Circuits to see the importance of Expressibility and Entangling Capacity on the training capacity of the circuit.

Contents

List of Figures

ABBREVIATIONS

PQC	Parametrized Quantum Circuit
HQC	Hybrid Quantum Classical
NISQ	Noisy Intermediate Scale Quantum
QNN	Quantum Neural Networks
VQC	Variational Quantum Classifier
VQE	Variational Quantum Eigensolver
Qubit	Quantum Bit
COBYLA	Constrained optimization by linear approximation
ANN	Artificial Neural Network
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative

Chapter 1

Introduction

Quantum Computing is an interdisciplinary field with intersection of Quantum Physics and Computer Science. The computers that we use in our daily lives are based on the classical laws of Physics. But with time, the computer hardware has only became smaller and smaller. If the hardware is so small that it starts showing quantum effects then we need to move to a totally different paradigm of computation called "Quantum Computation". So this is one of the motivation behind studying Quantum Computing. Other motivations are the exponential speed up it may provide as compared to a classical computer. In Quantum Computing, the idea is to leverage the laws of Quantum Physics rather than Classical Physics in order to perform computations.

Monroe et al in 2001 [MTK$^+$00] described trapped ion based technology for scalable quantum computing (Winland, one of the authors of this paper won Nobel Prize in Physics in 2012 partly owing to this paper).After this, interest in Practical Quantum Computation has increased drastically. Now all the big tech giants (IBM, Google etc) have started investing a lot and building their own Quantum Computers.

Now recently, owing to the significant development in Quantum Algorithms and hardware, it is expected that we are approaching the era of "Noisy Intermediate Scale Quantum (NISQ)" in a very near future. In this generation of Quantum Computers, it is expected that we will have support of 50 to 100 qubits and around 10^3 gate operations.

A particular class of Algorithms that maximizes the use of such hardware is HQC (Hybrid Quantum-Classical Algorithms) in which the computational tasks are strategically divided between quantum and classical resources. Few of the examples of such HQC Algorithm are VQE (Variational Quantum Eigen Solver) which is used to computer ground state of Molecular systems, Quantum Machine Learning etc.

This dissertation is focussed on studying and implementing two metrics (Expressibility and Entangling Capacity) which can be used to measure the power of Variational Quantum Circuits (which may be useful in deciding Circuit for a particular HQC Algorithm).

1.1 Noisy Intermediate Scale Quantum (NISQ)

NISQ, the term coined by John Preskil in 2018 [Pre18] refers to the current state of the art in the Quantum Technology in which we have Quantum Computers from 50 to 100 qubits

but these are not advanced enough to reach fault tolerance. These Computers may surpass the Classical Computers in some aspects (like speed) but its not reliable because of the noise we get from the quantum gates.

The term "Intermediate Scale" In NISQ refers to the size of the Quantum Computers present today (50 to few hundred qubits). Although 50 qubits may sound like a small number but it is a huge milestone that have been reached since its beyond any simulation that can be provided by the most powerful classical computer in the world. The term "Noisy" refers to that fact that we only have a very limited and control over these qubits owing to the noise associated with the quantum gates. This noise has placed a lot of limitations in what we can achieve with these Quantum Computers.

Because of the "Noise", these quantum Computer can't actually perform error corrected quantum computations of large scale, but it can provide computations of importance if we combine the use of both Quantum and Classical Computers. Such algorithms, which leverage both Quantum and Classical Computer by strategically dividing the computation tasks between both are called Hybrid Quantum-Classical (HQC) Algorithms.

Now the Parameterized Quantum Circuits (PQCs) or Variation Quantum Circuits or Ansatz (all aliases of PQCs) have a very important role in in performing many HQC Algorithms. However choosing a perfect PQC for a particular task (that properly represents the solution space for the HQC Algorithm and is also having reasonable circuit complexity and cost) is a big challenge as there is no sure shot way to choose that. Finding a perfect PQC is like hyper-parameter tuning in a machine learning algorithm.

Expressibility and the Entangling Capacity are two such metrics which help us in representing the solution space that a PQC can cover by quantifying it. The upcoming sections give a detailed description of these two metrics.

1.2 Hybrid Quantum-Classical (HQC) Algorithms

As explained in the previous section, HQC algorithms are the algorithms which leverage both quantum and classical machine in th computation by strategically dividing the computation tasks between the two.

In this class of algorithms, a quantum state is generally prepared for the quantum circuit and the expectation of the outputs of the quantum circuit is taken (as per the specific algorithm) which is then given to a classical optimizer after which the quantum state of the circuit is changed as per thee need of the optimization technique. Figure 1.1 shows a general schematic diagram for HQC algorithms.

One of the major examples of the HQC Algorithm is VQE (Variational Quantum Eigen-

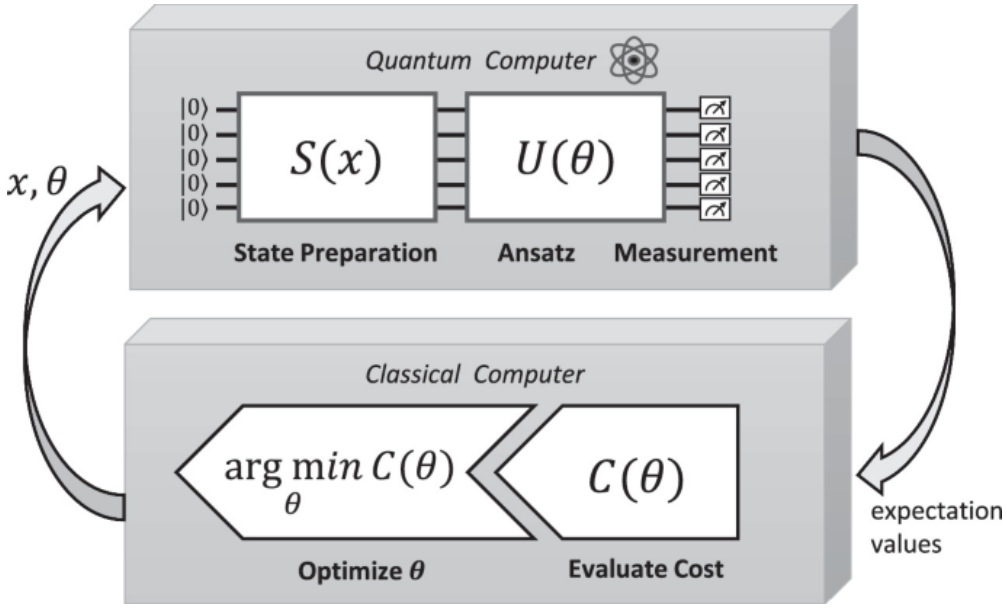

Figure 1.1: A general schematic diagram for Hybrid Quantum- Classical (HQC) Algorithms (courtesy: [WBLV21])

solver) which helps in computing the ground state of molecular systems [MRBAG16] [CRO+19]. In this, a parameterized wave function of the system of our interest is prepared in the quantum computer. This can be done if we tune the Quantum Circuit with the help of the parameterization that is chosen. After that, the expectation value of the energy (Hamiltonian) is measured for that that particular set of parameters of PQC. Then Classical Computer is used for optimization. The feedback is given back to the Quantum Computer on the basis of the Optimization so that we may get a trial state which can be "better learned". This loop of updating the trial state may continue as per the optimization technique till convergence. If we think of it, its just like training a Neural Network. As per the analogy, the parameters of the VQE are the learn-able weights in the neural network (which are actually a part of quantum circuit so quantum circuit is like the neural network). The values go to the classical computer for optimization (again like neural network) and feedback is sent back.

In fact, when PQCs are applied for machine learning tasks, as discussed above it would totally bear resemblance with the Neural Networks. This is why they are also called Quantum Neural Networks (QNNs) [FN20]. These are also a type of HQC Algorithms and can be used for both, classification and regression tasks in supervised learning and as a generative model in an unsupervised learning approach.

Some other examples of HQC Algorithms are Quantum Approximation Optimization Algorithm (QAOA) [FGG14], Quantum Autoencoder (QAE) [ROAG17a] and Quantum Variation Error Corrector (QVECTOR) [JOCAG17].

1.3 Motivation of Work and Scope

For all the algorithms mentioned in the previous section, selecting the right PQC such that it covers whole solution space and is also reasonable in cost and complexity is a challenging task. Which bring us to the motivation of this dissertation. Expressibility and entangling capacity [SJAG19] are two such metrics which can be calculated statistically and quantifies the extent a particular PQC can cover in the solution space. In this dissertation, we did an in depth study of these two metrics with extensive experimentation on various situations and directions.

1.4 Contribution of this Dissertation

Our Contributions can be summarized as follows:

- We have Studied the effect of topology of the machine and its noise on the Expressibility of a quantum circuit through extensive experimentation. We Used topologies of two different machines (IBMQ Vigo and IBMQ Mumbai) and ran circuit on simulators based on these topologies. We found that the topology of the machine indeed affects the Expressibility to some extent. But practically the effect of the topology is not significant enough.
- We found that the present method of computing Expressibility of the circuit doesn't take noise into account. Expressibility is the property of the circuit (and not the machine). So to compute Expressibility we must use error free simulators only.
- We found that the Entangling Capacity can't be computed for a Quantum Circuit on a given Noise Channel as the MW Measure (on which the Entangling Capacity is based) is defined only for Pure State and applying the Kraus Operator of these Quantum Channels on Multi-qubit Circuit evolves the Density Matrix to Mixed State.
- We took a Fraud Detection Synthetic Dataset and trained various PQCs (of varying structure, complexity, Expressibility and Entangling Capacity) as the Variational Circuits of Quantum Neural Network and analysed the impact of Expressibility and Entangling Capacity on the Accuracy, Precision and Recall of the trained model. We found that we don't need a maximally entangled of very high expressible circuit. A circuit with entangling capacity of more than0.4 and expressibility with KL Divergence of less than 0.11 is good enough. However this must also be kept in mind that even though these measures can be very useful in deciding PQC for a Quantum Neural Network but these are not full proof.

1.5 Organization

Chapter 1 (this chapter) was the introduction of this dissertation. Going ahead in the chapter 2, we give some preliminary knowledge of Quantum Mechanics and Quantum Computation

which is needed to understand the content of this dissertation. In Chapter 3 and 4, we discuss about our study of expressibility and entanglingeCapacity respectively. In Chapter 5, we discuss our analysis on effect of expressibility and entangling eapacity for deciding PQC for Quantum Neural Network. Finally in Chapter 6, we give the final results and concluding remarks.

Chapter 2

Preliminary Background

Quantum Computing is an interdisciplinary field encompassing of Physics, Mathematics and Computer Science. The background knowledge that could be very basic for people from one field could be totally unfamiliar to people from the other field. In this chapter, we provide a brief and introduction to Quantum Mechanics and Quantum Computing which would be elementary to understand the content of this Dissertation.

2.1 Bits, Qubits and Quantum States

A **bit** (classical bit) is a unit of information for two dimensional Classical System. It can be only in one of the two states (0 and 1).

The state $|0\rangle$ can be represented as $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

The state $|1\rangle$ can be represented as $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

So in the classical world, a bit can either be in state $|0\rangle$ or state $|1\rangle$. For example the electricity would either run or not run. The switch would either be on or of. The proposition be either True or False. And it is sufficient for the classical world.

However, in the Quantum World this Either / Or is not sufficient. In Quantum Realm, a system can be **both** on **and** off at the same time. Or in other words, a quantum system can be in state $|0\rangle$ and $|1\rangle$ at the same time. We say that the quantum system is in **Superposition** of the states.

Analogous to that of the bit, a **Qubit** (Quantum Bit) is a unit of information describing a two dimensional Quantum System.

A qubit can be represented as: $\begin{bmatrix} a \\ b \end{bmatrix}$ where a and b are complex numbers and $|a|^2+|b|^2 = 1$.

a quantum state is generally denoted as $|\psi\rangle$. Let $|\psi\rangle = \begin{bmatrix} a \\ b \end{bmatrix}$.

Now, $|\psi\rangle$ can be written as $|\psi\rangle = \begin{bmatrix} a \\ b \end{bmatrix} = a \begin{bmatrix} 1 \\ 0 \end{bmatrix} + b \begin{bmatrix} 0 \\ 1 \end{bmatrix} = a\,|0\rangle + b\,|1\rangle$

Similarly an n qubit system as 2^n computational basis states and it can be in ***superposition*** (linear combination) of these basis states.

So, the basic difference of Quantum Bit from Classical Bit is that that Quantum Bit can be in a *superposition* of states. Any kind of linear combination of states $\sum_i \alpha_i |\psi_i\rangle$ is a superposition of states $|\psi_i\rangle$ with amplitude α_i. The amplitude has no significance / physical property of its own. However the square of mod of the amplitude of a particular state $|\alpha_i|^2$ would be the probability that the state Quantum system after measurement would be $|\psi_i\rangle$. $\sum_i |\alpha_i|^2 = 1$ is called the *Normalization Condition.*

The Hadamard Basis States $\frac{|0\rangle \mp |1\rangle}{\sqrt{2}}$ is a famous and important example of *superposition.*

If a Qubit is in a state such that it can't be written in form of product of two pure states, we say that the Qubit is in an ***Entangled*** State. A famous and important example of entangled state is the Bell's State $\frac{|00\rangle + |11\rangle}{\sqrt{2}}$.

2.2 Bloch Sphere

Consider a quantum state $|\psi\rangle = c_0 |0\rangle + c_1 |1\rangle$ where $|c_0|^2 + |c_1|^2 = 1$.

Now, writing c_0 and c_1 in Eulars's Form: $c_0 = r_0 e^{i\phi_0}$ and $c_1 = r_1 e^{i\phi_1}$.

$\implies |\psi\rangle = r_0 e^{i\phi_0} |0\rangle + r_1 e^{i\phi_1} |1\rangle$

So, the equivalent expression for the state of the qubit with the amplitude of $|0\rangle$ as real by killing of its phase:

$\implies e^{-i\phi_0} |\psi\rangle = r_0 |0\rangle + r_1 e^{i(\phi_1 - \phi_0)} |1\rangle = r_0 |0\rangle + r_1 e^{i\phi} |1\rangle$ where $\phi = \phi_1 - \phi_0$

Here we know that $r_0^2 + r_1^2 = 1$.

We can substitute $r_0 = cos(\frac{\theta}{2})$ and $r_1 = sin(\frac{\theta}{2})$.

Hence the above expression for the state becomes: $|\psi\rangle = cos(\frac{\theta}{2}) |0\rangle + e^{i\phi} sin(\frac{\theta}{2}) |1\rangle$.

So we have boiled down the expression for the state of qubit to an expression with only two real parameters. Now that we have only two real parameters (that too angles), we can actually map it to a three dimensional sphere of R^3 of radius 1 called the *Bloch Sphere.* However the intuition that we may get from it is limited since there is no simple generalization of the Bloch sphere for multiple qubits.

θ and ϕ in the above expression define a point on a 3 dimensional sphere called *Bloch Sphere.* θ represents the angle that ψ would make with the z axis and ϕ represents the angle that the projection of ψ on the XY plane would make with the x axis. Figure 2.1 illustrates Bloch sphere representation of a Qubit.

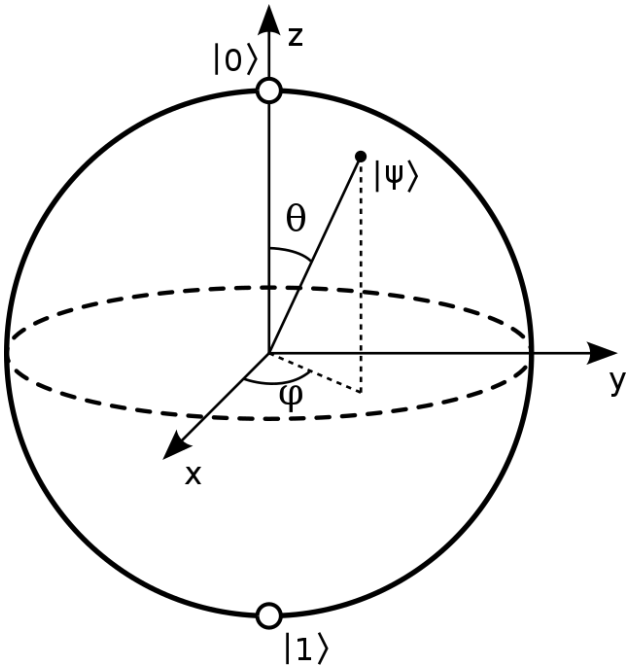

Figure 2.1: Bloch Sphere Representation of a Qubit (courtesy: Wikipedia)

2.3 Postulates of Quantum Mechanics

This Section states the Postulates of Quantum Mechanics which provide a connection between the physical and the mathematics formalism of Quantum Mechanics. These are taken from the book "Quantum Computation and Quantum Information" by Nelson and Chuang [NC11].

Postulate 1: Associated to any isolated physical system is a complex vector space with inner product (that is, Hilbert space) known as the state space of the system. The system is completely described by its state vector, which is a unit vector in the system's state space.

Postulate 2: The evolution of a *close* quantum system is described by a *unitary transformation*. That is, the state $|\psi\rangle$ of the system at time t_1 is related to the state $|\psi'\rangle$ of the system at the time t_2 by a unitary operator U which depends only on the times t_1 and t_2.

$$|\psi'\rangle = U\,|\psi\rangle$$

Postulate 2': The time evolution of the state of a closed quantum system is described by the *Schrödinger equation*,

$$i\hbar\frac{d\,|\psi\rangle}{dt} = H\,|\psi\rangle$$

where $\hbar$ is Planck's constant and H is a fixed Hermitian operator known as the Hamiltonian of the closed system. Since the Hamiltonian is a Hermitian operator it has a spectral

decomposition

$$H = \sum_E E \left|E\right\rangle \left\langle E\right|$$

with eigenvalues E and corresponding normalized eigenvectors $|E\rangle$. The states $|E\rangle$ are conventionally referred to as *energy eigenstates*, or sometimes as *stationary states*, and E is the *energy* of the state $|E\rangle$.

The connection between the Hamiltonian picture of dynamics , Postulate 2', and the unitary operator picture, Postulate 2 can be provided by solving the Schrödinger's equation:

$$\left|\psi(t_2)\right\rangle = exp[\frac{-iH(t_2 - t_1)}{\hbar}] \left|\psi(t_1)\right\rangle = U(t_1, t_2) \left|\psi(t_1)\right\rangle$$

where we define U, a unitary operator as:

$$U(t_1, t_2) \equiv exp[\frac{-iH(t_2 - t_1)}{\hbar}]$$

Postulate 3: Quantum measurements are described by a collection $\{M_m\}$ of *measurement operators*. These are operators acting on the state space of the system being measured. The index m refers to the measurement outcomes that may occur in the experiment. If the state of the quantum system is $|\psi\rangle$ immediately before the measurement then the probability that the result m occurs is given by:

$$p(m) = \left\langle\psi\right| M_m^\dagger M_m \left|\psi\right\rangle$$

, and the state of system after the measurement is

$$\frac{M_m \left|\psi\right\rangle}{\sqrt{\left\langle\psi\right| M_m^\dagger M_m \left|\psi\right\rangle}}$$

. The measurement operator satisfy the *completeness equation*,

$$\sum_m M_m^\dagger M_m = I$$

. The completeness equation expresses the fact that probabilities sum to one:

$$1 = \sum_m p(m) = \sum_m \left\langle\psi\right| M_m^\dagger M_m \left|\psi\right\rangle$$

.

2.4 No Cloning Theorem

The No Cloning Theorem states that it is impossible to make a copy of an arbitrary quantum state without first destroying the original. It can be proved as follows:

Suppose we do have and operation that maps an arbitrary state $|\psi\rangle \rightarrow |\psi\rangle \otimes |\psi\rangle$.

Then an arbitrary $|\phi\rangle$ would be mapped as $|\phi\rangle \rightarrow |\phi\rangle \otimes |\phi\rangle$.

Now, since the transformation must be linear,

$$\implies |\psi\rangle + |\phi\rangle \rightarrow |\psi\rangle \otimes |\psi\rangle + |\phi\rangle \otimes |\phi\rangle$$

However

$$|\psi\rangle \otimes |\psi\rangle + |\phi\rangle \otimes |\phi\rangle \neq (|\psi\rangle + |\psi\rangle) \otimes (|\phi\rangle + |\phi\rangle)$$

. Hence we have failed to copy $|\psi\rangle + |\phi\rangle$. ***Proved***

2.5 Quantum Computing and Quantum Circuits

Quantum Computing is the paradigm of computing which follows on the postulates of Quantum Mechanics which means that all the valid quantum operations are actually unitary. The evolution of an isolated quantum system with a finite number of states can be described by a unitary matrix and hence is always reversible. *Reversibility is a necessary condition for quantum computing.*

Quantum Circuits are basically sequence of building blocks that carry out elementary computations consisting of various quantum gates connected in sequence / parallel and can be represented using space-time Diagram. These diagrams are generally drawn in such a way that the time progresses from left to right. Any n - qubit gate or operation can be represented using $2^n \times 2^n$ unitary matrix. A set of quantum gates applied on same set of qubits are applied in series and the overall effect is computed using dot product. Two adjacent qubits in a quantum circuit applied on independent set of qubits are applied in parallel and the overall effect is computed using tensor product of the Unitary Matrices.

2.6 Quantum Gates

In this section, we discuss some important 1 and 2 qubit quantum operations and corresponding quantum gates which are used to build quantum circuits. In quantum circuits, there can't be any feedback mechanism (because of the No Cloning Theorem) hence the quantum circuits are always acyclic graphs. As discussed earlier, since the quantum theory

is unitary, the quantum gates are represented using unitary matrices.

$$U^{\dagger}U = 1$$

2.6.1 One Qubit Gates

One qubit gates are represented by 2×2 unitary matrix. Some of the important one qubit gates are as follows:

- **Global Phase Gate:** The global phase gate, P, is defined as:

  $$P(\theta) = e^{i\theta} I$$

  where I denotes the identity matrix, which indicates that no operation is performed.

  The global phase is actually physically indistinguishable. Therefore its not actually physically implemented. However it can be useful for matching Circuit Identities.

- **Pauli Gates:** The Pauli spin matrices for the x, y and z axes, corresponding to the Pauli Gates σ_x, σ_y and σ_z are respectively:

  $$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

- **Rotation Gates:** Rotation gates represent rotation about the orthogonal set of axis x, y and z.

  $$R_x(\theta) = \begin{bmatrix} cos(\frac{\theta}{2}) & -isin(\frac{\theta}{2}) \\ -isin(\frac{\theta}{2}) & cos(\frac{\theta}{2}) \end{bmatrix}, R_y(\theta) = \begin{bmatrix} cos(\frac{\theta}{2}) & sin(\frac{\theta}{2}) \\ sin(\frac{\theta}{2}) & cos(\frac{\theta}{2}) \end{bmatrix}, R_z(\theta) = \begin{bmatrix} e^{-\frac{\theta}{2}} & 0 \\ 0 & e^{\frac{\theta}{2}} \end{bmatrix}$$

  In general,

  $$R_i(\theta) = R_i(\theta \pm 4\pi) = -R_i(\theta \pm 2\pi), i \in x, y, z$$

- **Hadamard Gate:** Hadamard gate is defined as:

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

2.6.2 Two Qubit Gates

Two qubit gates are represented by 4×4 Unitary Matrix. These gates support interaction between two quantum systems. One of the most useful operations in two qubit gates is *Controlled Operations*. In Controlled Operations, One of the two Qubits act as the *Control Qubit* and the other one would act as *Target Qubit*. In general, In any controlled operation CU or controlled U (say), if the control qubit c is set then the unitary operation U would

be on target qubit t otherwise not.

$$|c\rangle |t\rangle \rightarrow |c\rangle U^c |t\rangle$$

In matrix notation, CU would look like following:

$$CU = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{bmatrix}$$

Here are some of the important two qubit gates that would be used in this dissertation:

- **CNOT Gate:** This is a CU with $U = X$.

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

- **CP Gate:** It refers to controlled phase change operation by angle θ. Its Unitary Matrix is symmetric so there is no distinction between controlled and target qubit.

$$CP(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{bmatrix}$$

- **CZ Gate:** A special type of CP operation with $\theta = \pi$

$$CP(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

- **Controlled Rotation Gates:** These are the controlled operations with U as as one of the Rotation Gates. There can be 3 rotation gates CR_X, CR_Y and CR_Z with R_X, R_Y and R_Z as follows:

$$R_X(\theta) = \begin{bmatrix} cos(\frac{\theta}{2}) & -isin(\frac{\theta}{2}) \\ -isin(\frac{\theta}{2} & cos(\frac{\theta}{2}) \end{bmatrix}, R_Y(\theta) = \begin{bmatrix} cos(\frac{\theta}{2}) & sin(\frac{\theta}{2}) \\ sin(\frac{\theta}{2} & cos(\frac{\theta}{2}) \end{bmatrix}, R_Z(\theta) = \begin{bmatrix} e^{-\frac{\theta}{2}} & 0 \\ 0 & e^{\frac{\theta}{2}} \end{bmatrix}$$

In general,

$$R_i(\theta) = R_i(\theta \pm 4\pi) = -R_i(\theta \pm 2\pi), i \in x, y, z$$

2.7 Some Important Keywords

In this dissertation, we have used some keywords frequently which are commonly used in Quantum Computing but may not be well known to people who have no background of Quantum Computing. We discuss all those keywords here so that we can use them with ease later in the upcoming chapters.

- **Shots:** To estimate any result we get in Quantum Computing, we often have to run the same Quantum Circuit multiple times and then take the expectation of the results of each run of the circuit. Running the circuit one time for the same purpose is called Shot. So we generally decide the number of shots as per our need so that the result we get (the expectation / average of the results in all the shots combined) is close to the actual result that want. As we know, as the sample size increases (here number of shots) in any statistical experiment (here running the circuits multiple times to get the expected result), the sample average tends to be closer to the actual population average / expectation.
- **Layers of Circuit:** In some experiment / problem where we are running a Quantum Circuit, if we concatenate the same circuit with another identical circuit. We say that we have added another layer of the circuit. In the experiments in this we have often experimented on results that we get on increasing the number of layers (or in other words for multiple identical circuits run together when connected sequentially with each other).
- **Topology of a Quantum Computer:** Topology of a quantum computer refers to the way the qubits are connected in it. Topology of IBMQ Vigo and IBMQ Mumbai can be seen in Figure 3.7 and 3.8 respectively.

2.8 Parameterized Quantum Circuits

Parameterized Quantum Circuits (PQC), often also referred as Ansatz or Variational Circuits are Quantum Circuits with changeable parameters. In NISQ Era, the PQCs are an integral part of most of the algorithms to show the Quantum supremacy. PQCs are composed of two types of Quantum Gates viz fixed gates like $CNOT$ (whose operation is fixed and doesn't consist of any changeable parameters) and adjustable gates like rotation gates or controlled rotation gates (whose operation changes with the change of the parameter, which in case of the rotation gate is the angle of rotation). PQCs are capable of very non trivial outputs even for a single layer of the circuit. In fact the main goal of this dissertation is to study two metrics called expressibility and entangling capacity for a PQC which quantifies the PQCs ability to produce such non trivial outputs. Figure 2.2 shows an example of PQC. It is one of the 19 circuits that we considered for our study. In Figure 2.2 the R_X and R_Z gates are the adjustable gates with a changeable parameters (the angles of rotation), changing which will change the output of the circuit, and the $CNOT$ gates are fixed gates. All the other circuits that we considered for out study are shown in Figure 3.2.

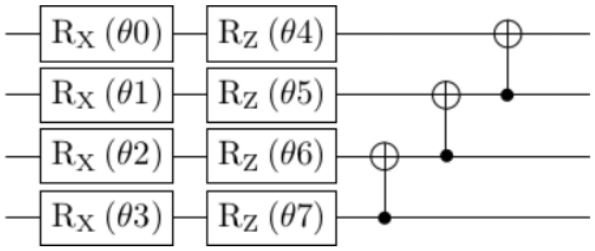

Figure 2.2: An example of PQC

Chapter 3

Expressibility

Choosing an effective PQC that well represents the solution space of our problem is a big challenge. To counter that, [SJAG19] proposed two metrics i.e. expressibility and entangling capacity. In this chapter, we discuss about expressibility in detail and the experiments that we performed on expressibility. In next chapter, we discuss about the entangling capacity.

Expressibility of a circuit in simplest possible words can defined as the ability of the circuit to generate (pure) states that are well representatives of the Hilbert Space. [SJAG19] has given one approach to expressibility. To compute it, we first need the distribution of states of a PQC, which we can get by sampling its parameters. Then we need Uniform Distribution of states, which we can get by using Ensemble of Haar Random states. Then we need to compare the above to distributions (distribution of states of PQC and Haar Distribution). We can compare it by using any distance measure like KL Divergence. However it must be noted that the expressibility would increase with decrease in the KL Divergence and not the other way around. In terms of interpretation / operational meaning of using KL Divergence to define expressibility is that it can be interpreted as the amount of information lost when we are estimating the state fidelities generated by a PQC by comparing it with Haar random state.

The deviation can be quantified as:

$$A = \int_{Haar} (|\psi\rangle \langle\psi|)^{\otimes t} d\psi - \int_{\theta} ((|\phi_\theta\rangle \langle\phi_\theta|)^{\otimes t} d\theta$$

Here, the first integral is taken using Haar Measure over all pure states and the second integral is taken over the states obtained by uniformly sampling the parameters θ of the PQC.

We can get a good intuition of expressibility in the big picture by considering the a single qubit system as we can visualize it with the help of the Bloch Sphere. Figure 3.1 illustrates the quantification of expressibility for single qubit circuits using Bloch Spheres and histograms. Figure 3.1(a) shows the circuit that is being discussed. Figure 3.1(b) shows Bloch Sphere with sampled states plotted in it for the corresponding circuit. Figure 3.1(c) shows the histogram of the estimated fidelity of the PQC with the Haar Distribution. The corresponding KL Divergence is written just above the histogram.

The first circuit contains only the identity gate which makes it just an idle circuit. Because of this it is not expressible at all. It is not covering any Hilbert Space. It just

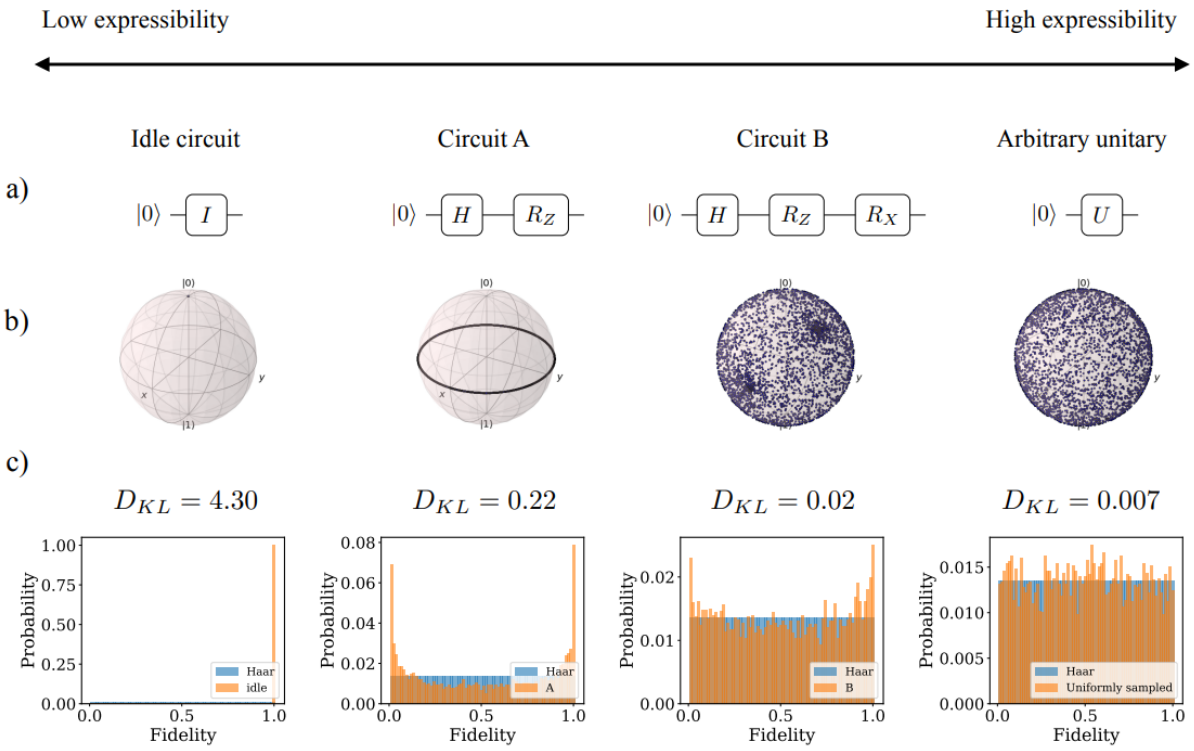

Figure 3.1: expressibility of Single Qubit Circuits. (a) Circuit Diagrams for four types of circuits. (b) Bloch Sphere representation with plotting of the parameterized states on the corresponding circuit. (c) Histogram of the estimated fidelity with the fidelity of Haar distribution using KL Divergence as the distance metric. (courtesy: [SJAG19])

returns what it gets in input. Now, if we add a Hadamard followed by an R_z gate (Circuit A), the states obtained from uniformly sampling θ (about the z axis) would only be limited to the equator of the Bloch Sphere. However since its covering more Hilbert Space than the idle circuit, its expressibility would be more than the idle circuit. As we can clearly see, that the KL Divergence in Circuit is (0.22) is less than the KL Divergence in the Idle Circuit (4.3). Now, if we also add an R_x gate in the Circuit A, we get the circuit B. Now, the rotation about x axis (because of R_x gate) followed by the R_z in the previous will give us a way better coverage of the states in the Hilbert Space as every point of the equator we saw in the Bloch Sphere of Circuit A would then rotate about the x axis. However, it would give us more concentration of states near the ends of the x axis as the point in equator when rotated about x axis would form a small circle of radius close to zero. So for the same reason, the concentration would decrease as we move away from the ends of the x axis and would be minimum when $x = 0$ in the Bloch Sphere as its farthest from the two poles and hence would make the circle of largest radius. Finally the final circuit is for reference of the previous 3 circuits. In that, we are just uniformly sampling single qubit unitary matrices to simulate the most expressible circuit.

3.1 Estimating expressibility

To estimate the expressibility, we need to uniformly sample a sufficient number of fidelity distribution of both the PQC with varying parameters and the haar ensemble. Whatever the size of the sample is, it would be finite. Since we have a finite sample size, hence the probability distribution can be estimated with histogram. We can consider discretization of the histograms (the probability distributions) to estimate the KL Divergence [KL51]. Fidelity Distributions can be estimated by independently sampling pairs of states (i.e. sampling pairs of parameter vectors and obtained parameterized states).

$$Expr = D_{KL}(\hat{P}_{PQC}(F;\theta)||P_{Haar}(F))$$

Here, D_{KL} is the KL Divergence (also often referred as Kullback Leibler Divergence or Relative Entropy) which is basically a measure to quantify the difference between two probability distributions (say P and Q).

If P and Q are discrete probability distributions defined on same probability space χ, KL Divergence from Q to P is defined as:

$$D_{KL}(P||Q) = \sum_{x \in \chi} P(x) log(\frac{P(x)}{Q(x)})$$

3.1.1 Estimating Probability Distribution of Haar Random States

[idZS05] has given the analytical form of PDF (Probability Density Function) for ensemble of Haar random state. We can directly use this PDF to get the Haar distribution:

$$P_{Haar}(F) = (N-1)(1-F)^{N-2}$$

where F is the fidelity and N is the dimension of the Hilbert Space.

The algorithm 1 illustrates how we get discretized histogram of Haar Random States. The code snippet for it can be found in Appendix A.

3.1.2 Estimating Probability Distribution of fidelities from sampling states of Parameterised Quantum Circuits (PQC)

Now, to compute the fidelity in a PQC, we can leverage the fact that *Quantum Computing is Reversible.* In any Quantum Circuit, if a reversed circuit is placed in front of the original circuit, it should give the output same as the input. So to calculate fidelity, for a number

Algorithm 1: Discretized Histogram of Haar Random States

Input: Number of Qubits N and bins length b;
bins: an empty array;
for $i \in [1, 75]$ **do**
    bins.insert(i/b)
P_{Harr}Histogram: empty array to be used to store discretized histogram;
for $i \in [1, 75]$ **do**
    P_{Harr}Histogram.insert($(1 - bins[i])^{N-1} + (1 - bins[i+1])^{N-1}$)
return P_{Harr}Histogram;

of shots, we can keep the count in which the output is same as the input and then divide it by the number of shots.

The algorithm 2 illustrates how we get Estimated Probability Distribution of fidelities from sampling states of Parameterised Quantum Circuits.

Algorithm 2: Discretized Histogram of PQC

Input: Quantum Circuit function qc which takes set of random numbers as parameter, number of shots n, number of parameter sets p;
fidelity: an empty array;
theta: an empty array;
for $i \in [1, p]$ **do**
    $theta \leftarrow generate\_random\_params(500)$;
    $circuit \leftarrow qc(theta)$;
    $job \leftarrow execute(circuit, input =' 0000')$;
    $result \leftarrow job.result()$;
    $count \leftarrow result.get\_counts()$;
    $ratio \leftarrow count['0000']/n$;
    $fidelity.insert(ratio)$
$weights \leftarrow ones\_like(fidelity)/p$;
$P\_histogram \leftarrow histogram(fidelity, bins, weights, range = [0, 1])$;
return P_Histogram;
Function *generate_random_params(x) :* **do**
    for $r \in [1, x]$ **do**
        return $2\pi r$
Function *ones_like(x: array) : array* **do**
    $shape \leftarrow x.shape$;
    ones : array of the above shape;
    return ones

3.2 Experiments and Results

3.2.1 Experimental Setup

We have done our experiments primarily on 19 four qubit circuits of varying complexity, depths and structures with different gates. These circuits are taken from [SJAG19]. The circuits are shown in figure 3.2. These circuits are also actually derived from some past studies and applications. The main motive of this study was to experiment on these two metrics expressibility and entangling capacity on different environments and situations. We first computed expressibility on QASM Simulator with 1000 shots and 2000 parameter samples on each shot. The results were close to those mention in the paper by [SJAG19]. The expressibility results of the paper are illustrated in the Figure 3.3 and our results on QASM Simulator are illustrated in Figure 3.4. In the graphs showing the expressibility values, L refers to the number of layers of the circuits put sequentially.

In this section, we discuss our experiments on expressibility. We computed expressibility in the following three settings:

- On QASM Simulator
- On Simulators with 2 different topologies based on two Actual Quantum Machines (IBMQ Vigo and IBMQ Mumbai).
- On FakeMumbai Simulator (A simulator which tries to completely replicate IBMQ Mumbai with both its noise and topology).

Tools Used: Python programming language and qiskit framework for Quantum Computing Simulations.

The observations and conclusions that we got from our experiments are discussed in the next three subsections.

3.2.2 Expressibility and its Dependency on Expectation of the Fidelities of PQC

Even though our results on 1000 shots and 2000 parameter sample sets were close to those given by [SJAG19], it was a little different for a few circuits as it can be seen in the graphs. But that was only because it was being computed statistically by averaging on (taking expectation on) the different number of shots. Other possibilities are that the the number of shots taken by the authors of that paper could be different. To confirm the hypothesis we increased the number of shots and recomputed expressibility on the circuits with 8000 shots for single layers on the circuits in which the difference was higher. Our results and the

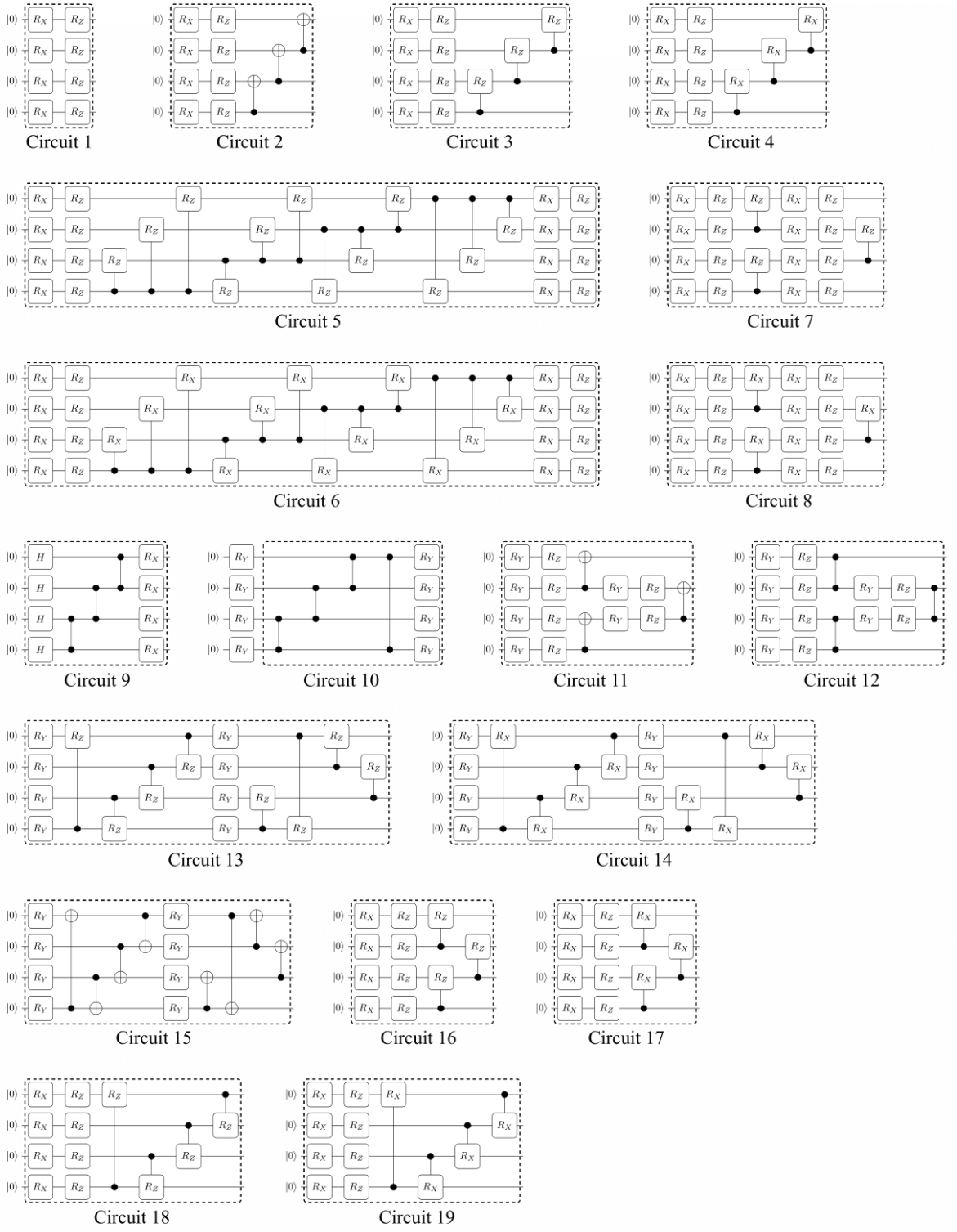

Figure 3.2: Circuit Templates used in this study for experimentation purposes (courtesy: [SJAG19])

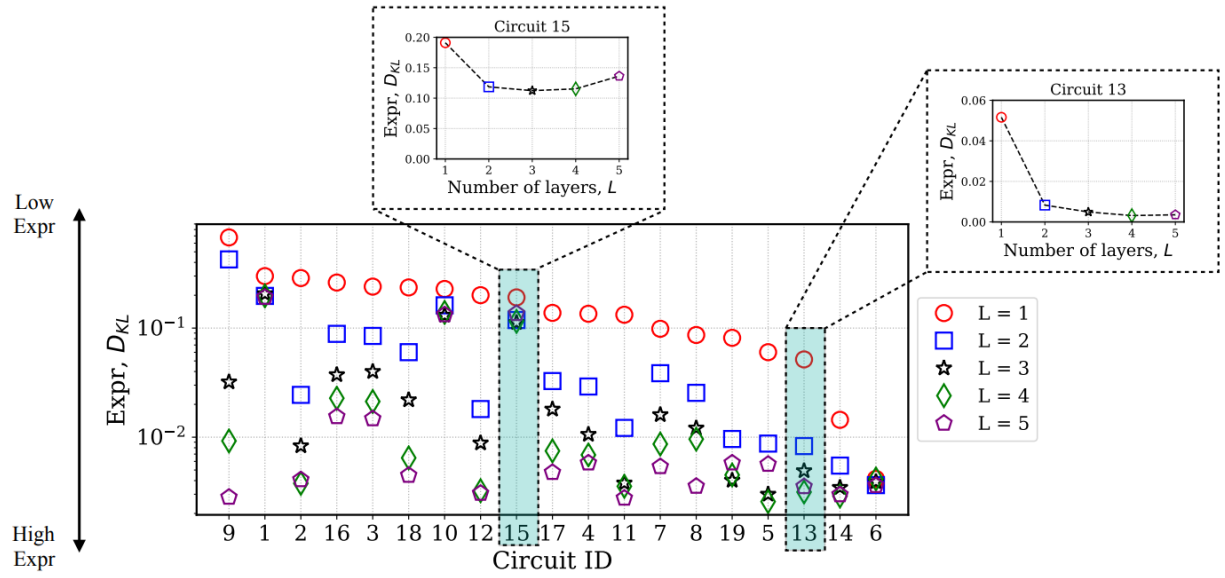

Figure 3.3: expressibility of different circuits as given in the paper [SJAG19]

papers results converged after that. The comparison of our results (on QASM Simulator) and those mentioned in the paper for a single layer on all the 19 circuits are illustrated in the Figure 3.5 for 1000 shots and Figure 3.6 for 8000 shots (on increasing the shots).

Conclusion: Since expressibility is calculated statistically using expectation on probability distributions, we may get minute differences when we compute the same thing at 2 different times. However if we increase the number of shots, the results would converge to what it should be.

3.2.3 Effect of Topology of the Machine

The other research question we had was "Does the Topology of the machine affect the expressibility in any way?". We ran the same experiments on simulators with machines of two different topologies. The two topologies that we used were of IBMQ-Vigo (5 qubit machine, topology illustrated in Figure3.7) and IBMQ-Mumbai (27 qubit machine, topology illustrated in Figure 3.8). The results were close, but there are some interesting irregularities there. Since expressibility is a Statistical Property, its values slightly changes every time it runs but it of course has to stay close to its expectation. So for IBMQ-Vigo topology, the difference of expressibility in all the circuits (as computed using QASM Simulator VS IBMQ-Vigo topology based simulator) was less than 0.03 (illustrated in Figure 3.9). Initially we had a difference greater than 0.03 for few other circuits too, but after multiple independent runs the result changes. So only circuit 11 (for single layer) was giving an expressibility difference more than 0.03 which is interesting since it remained the same even on multiple independent computations. Similar anomalies can be seen with IBMQ-Mumbai bases simulator too. However the circuit showing this kind of anomaly in that was circuit 19 (for single layer)

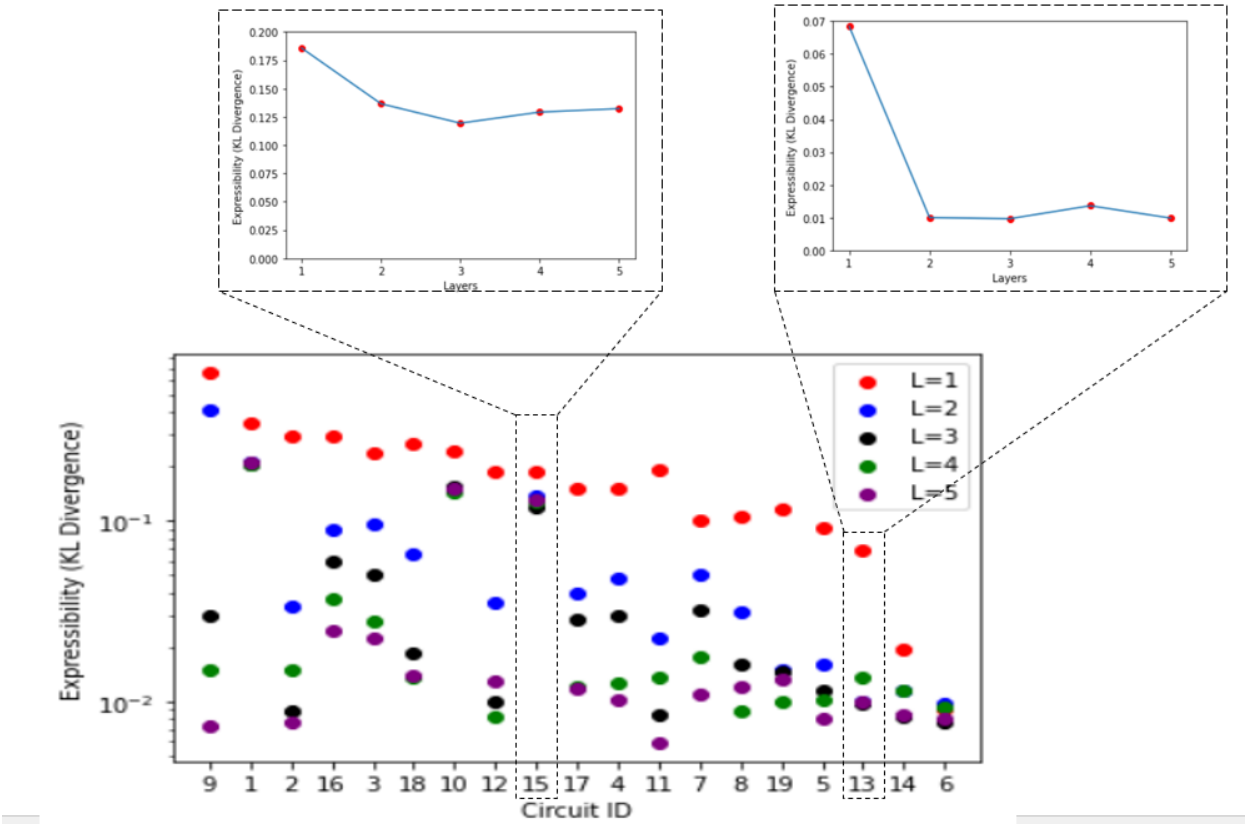

Figure 3.4: expressibility of different circuits as ran on QASM simulator.

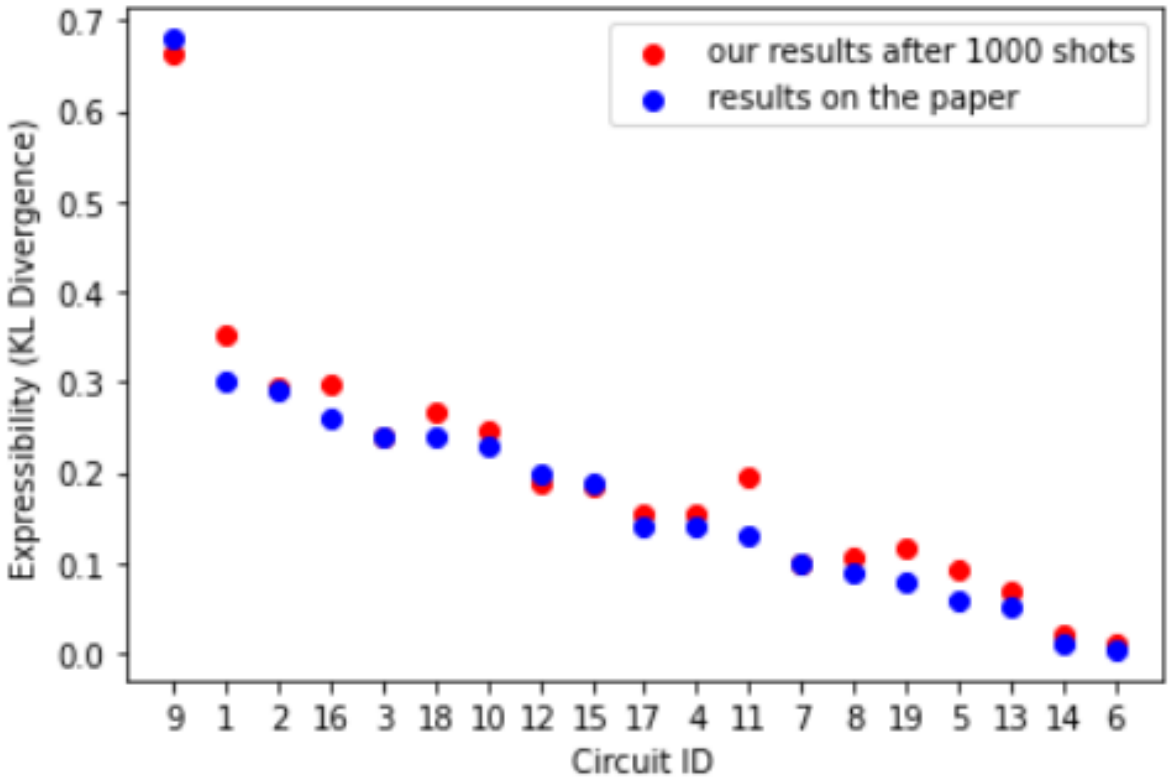

Figure 3.5: Expressibility: our expressibility results on single layer vs [SJAG19] paper with 1000 shots.

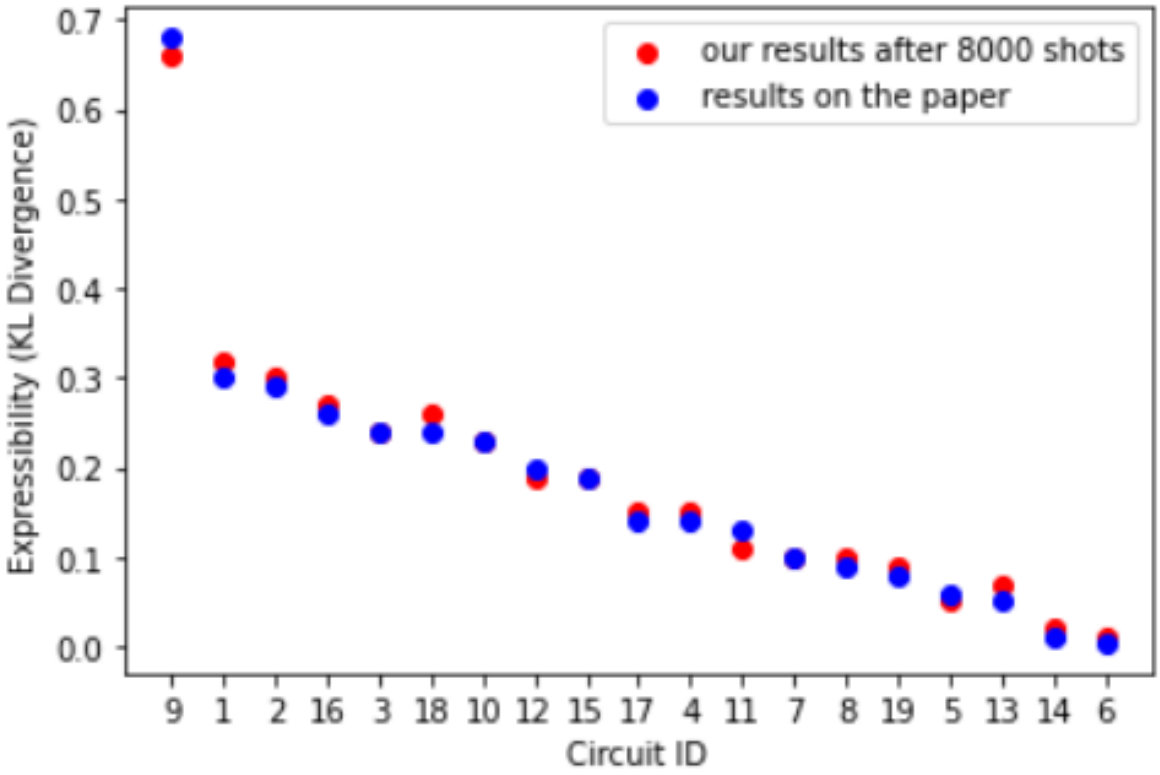

Figure 3.6: Expressibility: our expressibility results on single layer vs [SJAG19] paper with 8000 shots.

and circuit 9 (for double layer) rather than circuit 11 (illustrated in Figure 3.10).

Conclusion: Topology of the machine that we are using to compute expressibility indeed has some effect on the expressibility. The results however are close enough for our purposes. Even if there is a change because of topology, it was observed that it is on some particular circuit and for a single layer only. This explanation of it could be that as we increase the number of layers of the circuit, its expressibility also increase (as the circuit gets more complex) which decreases its chances of getting affected due to external factors like topology which may have a very minute effect on it. The exact reason of it is not known to us yet. Finding that out is a future work that can be studied.

An Observation: For experimentation, we swapped all the R_x gates in Circuit 19 with R_y gates. The intuition was that it should change nothing as R_x and R_y are practically symmetrical in the Bloch Sphere. The expressibility values actually didn't change and remained the same but the on running in the IBMQ-Vigo topology based simulator, the original circuit continued giving a difference of more than 0.03 and but the modified circuit was within that. Practically, replacing x by y should not change anything in circuit 19, but as it turns out, we can see the change in a simulator of a particular topology.

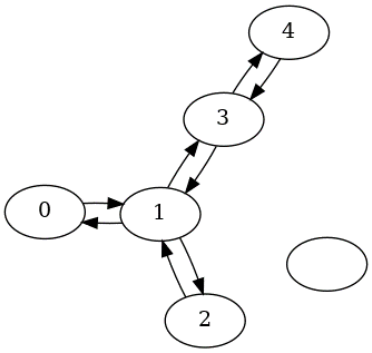

Figure 3.7: Topology of IBMQ-Vigo (5 qubit machine).

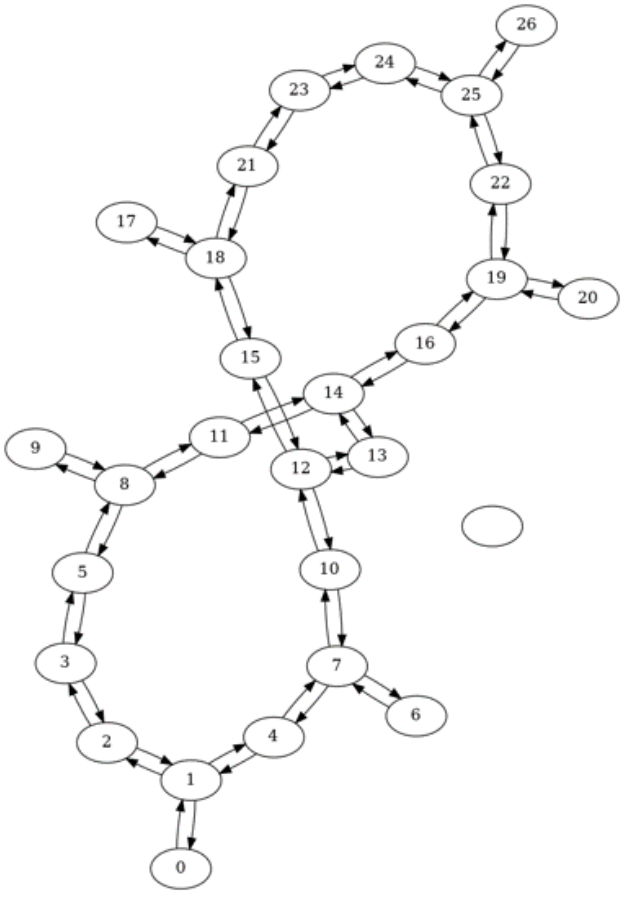

Figure 3.8: Topology of IBMQ-Mumbai (27 qubit machine).

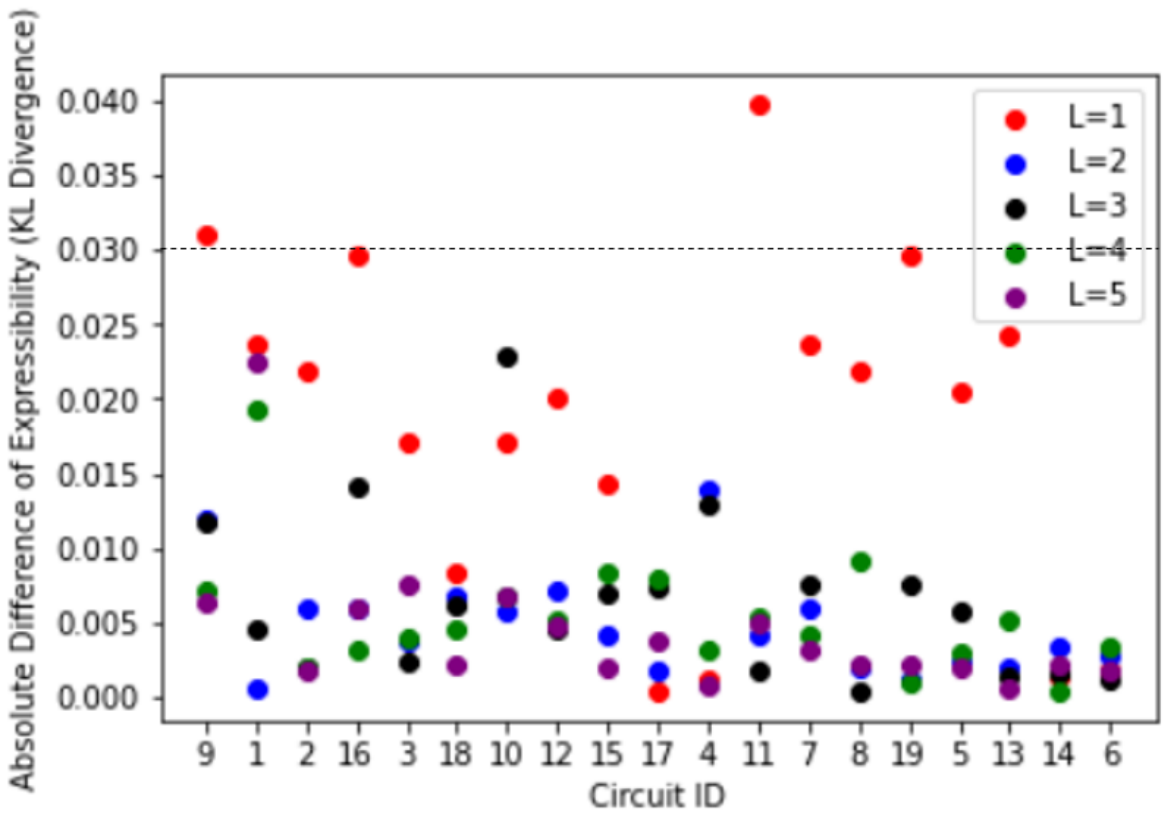

Figure 3.9: Absolute values of difference of expressibility as computed by QASM Simulator VS IBMQ-Vigo Topology based simulator. As we can clearly see, for circuit 11 (single layer), the difference is more than 0.03

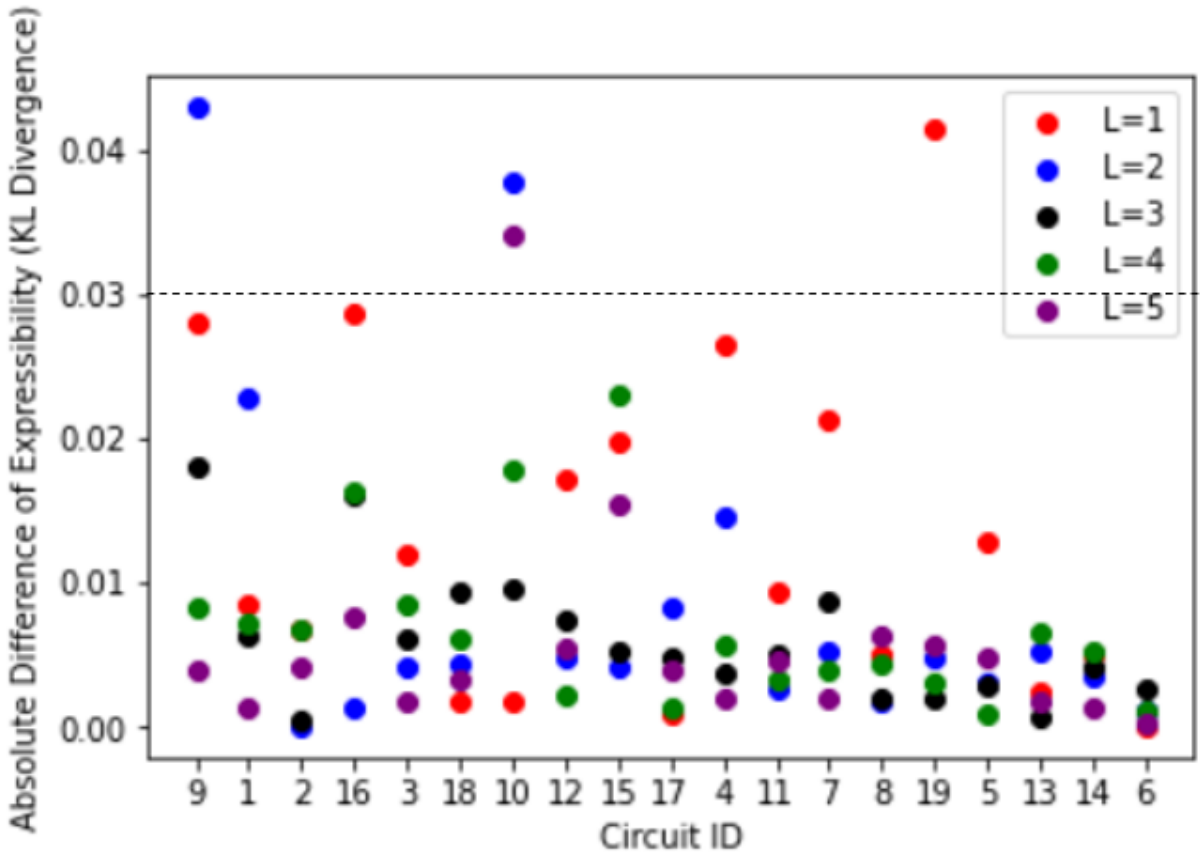

Figure 3.10: Absolute values of difference of expressibility as computed by QASM Simulator VS IBMQ-Mumbai Topology based simulator.As we can clearly see, for circuit 19 (single layer) and circuit 9 (double layer), the difference is more than 0.03 .

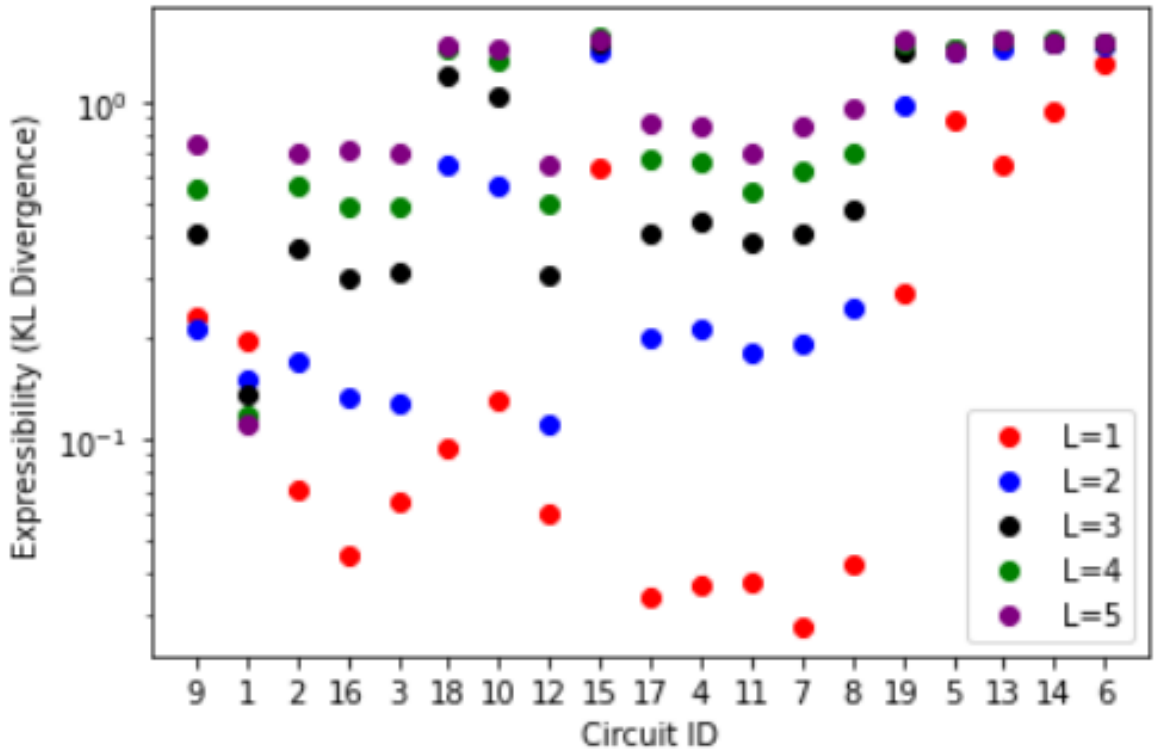

Figure 3.11: Expressibility of different circuits as ran on FakeMumbai simulator.

3.2.4 The present way of computing expressibility doesn't take noise into account

Expressibility is theoretical metric used to quantify the ability of the circuit to generate (pure) states that are well representatives of the Hilbert Space. So it doesn't take noise of the machine in account. Even an actual quantum computer can be capable of good expressibility but when we try to compute expressibility by this traditional method on that, then it would give garbage results as our definition of expressibility (the way we are computing it, doesn't take the noise of the machine into account). We tried running our experiments on real quantum computer, but because of very long queues it was taking too much time. So we rather computed expressibility on FakeMumbai based simulator available on qiskit (which tries to completely simulate IBMQ-Mumbai including both, its noise and its topology). The results that we got totally different (garbage) as compared to the actual results. The results are illustrated in Figure 3.11.

Conclusion: As it can be observed from the Figure 3.11, the computed expressibility for single layer is more than that of multiple layers. This is probably because as the depth of the circuit increases (with increase in number of layers), the effect of noise in the circuits also increase. because of that, we are getting less counts of correct results on output when putting a reversed circuit in front of the original circuit. So the expressibility values we get are garbage.

Chapter 4

Entangling Capacity

In previous chapter, we discussed in detail about the expressibility of a PQC which represents the ability of the circuit to generate (pure) states that are well representatives of the Hilbert Space. However, in quantum computing the actual phenomena that is different from classical computing is *Entanglement* which expressibility doesn't take into account at all. So expressibility can be considered more as the quantity which describes how much classical solution space (pure states) the PQC is covering. To quantify the extent of entanglement (which is actually a quantum phenomena rather than what is computed by expressibility), entangling capacity is used. Quantifying entanglement can very useful in many tasks which behave more like Quantum Phenomena (for example Ground State Preparation or capturing trivial correlations in quantum data [SBSW20] [KMT$^+$17]). It is also found useful in VQE [KMT$^+$17] and Quantum Machine Learning [SBSW20][HCT$^+$19].

[NDD$^+$03] has given many ways to quantify entanglement (basically compute entangling capacity) however MW Measure is most scalable and easy to compute among them. In our study, we focus only on the Meyer Wallach (MW) Measure [MW02] as the way to quantify entanglement.

4.1 Meyer Wallach Measure

Definition 1: For an n qubit system, consider a linear mapping $i_j(b)$ acting on computational basis:

$$i_j \left|b_1 \dots b_n\right\rangle = \delta_{bb_j} \left|b_1 \dots \hat{b}_j \dots b_n\right\rangle$$

here $b_j \in \{0, 1\}$ and $\hat{}$ denotes absence of j^{th} qubit. Meyer Wallach Measure Q is defined as:

$$Q(\left|\psi\right\rangle) = \frac{4}{n} \sum_{j=1}^{n} D(i_j(0) \left|\psi\right\rangle, i_j(1) \left|\psi\right\rangle)$$

where the distance D is defined as:

$$D(\left|u\right\rangle, \left|v\right\rangle) = \frac{1}{2} \sum_{i,j} |u_i v_j - u_j v_i|^2$$

here $\left|u\right\rangle = \sum u_i \left|i\right\rangle$ and $\left|v\right\rangle = \sum v_i \left|i\right\rangle$.

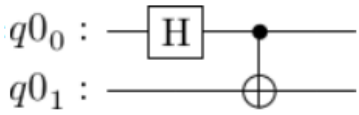

Figure 4.1: EPR Circuit with quantum state $|\psi\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$

Example 1. Consider the EPR Circuit shown in Figure 4.1 (which is fully entangled). So by the definition of the Meyer Wallach Measure, it should be 1 for this circuit. with quantum state as follows:

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

Here,

$$i_1(0)\,|00\rangle = |00\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} = i_2(0)\,|00\rangle$$

$$i_1(1)\,|00\rangle = \begin{pmatrix}0\\0\\0\\0\end{pmatrix} = i_2(1)\,|00\rangle$$

$$i_1(0)\,|11\rangle = \begin{pmatrix}0\\0\\0\\0\end{pmatrix} = i_2(0)\,|11\rangle$$

$$i_1(1)\,|11\rangle = |11\rangle = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = i_2(1)\,|11\rangle$$

$$\implies i_1(0)\,|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\0\end{pmatrix}, i_1(0)\,|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\0\end{pmatrix},$$

$$i_1(0)\,|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\0\end{pmatrix}, i_1(0)\,|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

$$\implies D(i_j(0)\left|\psi\right\rangle, i_j(1)\left|\psi\right\rangle) = (\frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}})^2 = \frac{1}{4}$$

$$\implies Q(|\psi\rangle) = \frac{4}{2}(\frac{1}{4} + \frac{1}{4}) = 1$$

As it can be seen from the above example that the Meyer Wallach Measure of a fully entangled circuit is 1.

Definition 2: The MW Measure Q for a circuit is the average linear entropy of all single qubit reduced state. The formal proof of this was given by [**?**]. Linear entropy of a single qubit reduced state refers to the linear entropy of a quantum state while keeping all other states inactive / constant. It is $2(1 - tr\{\rho^2\})$ if ρ is the density matrix of the single qubit considered. So average linear entropy of all single qubit reduced states refers to the average of linear entropy of all single qubits.

Example 2. Consider the EPR Circuit (Figure 4.1) again and see what we get from the above formula. The quantum state is as follows:

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

A refers to qubit $q0_0$ and B refers to qubit $q0_1$ in Figure 4.1.

$$\begin{aligned}\rho_{AB} &= \frac{1}{2}(|00\rangle + |11\rangle)(\langle 00| + \langle 11| \\ &= \frac{1}{2}(|00\rangle\langle 00| + |11\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 11|)\end{aligned}$$

$$\begin{aligned}\rho_A &= \frac{1}{2}\sum_{i=0}^{1}\langle i|_B(|00\rangle\langle 00| + |11\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 11|)_{AB}|i\rangle_B \\ &= \frac{1}{2}(|0\rangle\langle 0| + |1\rangle\langle 1|)_A \\ &= \frac{1}{2}\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} = \frac{I}{2} \implies \rho_A^2 = \frac{1}{4}I\end{aligned}$$

$\implies tr(\rho_A^2) = \frac{1}{2}$ and similarly, $tr(\rho_B^2) = \frac{1}{2} \implies Q(|\psi\rangle) = 2(\frac{\frac{1}{2}+\frac{1}{2}}{2}) = 1$

Example 3. Lets consider another fully separable (completely un-entangled) circuit (Figure 4.2) and compute the MW Measure the same way. We should get 0 as the result. The quantum state is as follows:

$$|\psi\rangle = |{+}{+}\rangle = \frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle + |1\rangle) = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

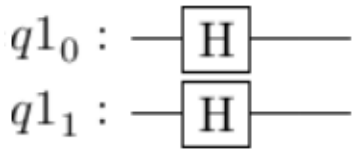

Figure 4.2: A 2 qubit circuit with no entanglement and quantum state $|{+}{+}\rangle$

A refers to qubit $q1_0$ and B refers to qubit $q1_1$ in Figure 4.2.

$$\rho_{AB} = \frac{1}{4}(|00\rangle\langle 00| + |01\rangle\langle 00| + |10\rangle\langle 00| + |11\rangle\langle 00| + \cdots + |00\rangle\langle 11| + |01\rangle\langle 11| + |10\rangle\langle 11| + |11\rangle\langle 11|)$$

$$\rho_A = \sum_{i=0}^{1}\langle i|\rho_{AB}|i\rangle = \tfrac{1}{2}\times 2(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| + |1\rangle\langle 1|) = \tfrac{1}{2}\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix}$$

$$\implies \rho_A^2 = \tfrac{1}{4}\begin{pmatrix}2 & 2\\ 2 & 2\end{pmatrix} = \tfrac{1}{2}\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix} = \rho_A$$

$$\implies tr(\rho_{A^2}) = 1 = tr(\rho_B^2)$$

$$\implies Q(|\psi\rangle) = 2\times\tfrac{(1-1)+(1-1)}{2} = 0$$

Example 4. Now that we have seen both the extremes viz fully entangled circuit (MW Measure = 1) and a fully separable Circuit (MW Measure = 0), lets consider another circuit of 3 qubits which is neither fully entangled nor fully separable (Figure 4.3). The MW Measure of this circuit should be something between 0 and 1. The quantum state of this circuit would be as follows:

A refers to qubit $q2_0$, B refers to qubit $q2_1$ and C refers to qubit $q2_2$ in Figure 4.3.

$$|\psi\rangle = \frac{1}{2}(|00\rangle + |11\rangle)(|0\rangle + |1\rangle) = \frac{1}{2}(|000\rangle + |110\rangle + |001\rangle|111\rangle) =$$

$\rho_{ABC} = (|\psi\rangle\langle\psi|)$
$= \frac{1}{4}(|000\rangle\langle 000| + |000\rangle\langle 001| + |000\rangle\langle 110| + |000\rangle\langle 111| + |001\rangle\langle 000| + |001\rangle\langle 001| + |001\rangle\langle 110| + |001\rangle\langle 111| + |110\rangle\langle 000| + |110\rangle\langle 001| + |110\rangle\langle 110| + |110\rangle\langle 111| + |111\rangle\langle 000| + |111\rangle\langle 001| + |111\rangle\langle 110| + |111\rangle\langle 111|)$

$$\rho_A = \sum_{i=0}^{3}\langle i|_{BC}\,\rho_{ABC}\,|i\rangle_{BC} = \tfrac{1}{4}(|0\rangle\langle 0| + |0\rangle\langle 0| + |1\rangle\langle 1| + |1\rangle\langle 1|) = \tfrac{I}{2}$$

$$tr(\rho_A^2) = \tfrac{1}{2} = tr(\rho_B^2)$$

$$\rho_C = \sum_{i=0}^{3}\langle i|_{AB}\,\rho ABC\,|i\rangle_A\,B = \tfrac{1}{4}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| + |1\rangle\langle 1|) = \tfrac{1}{4}\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix} = \rho_C^2$$

$$\implies tr(\rho_C^2) = 1$$

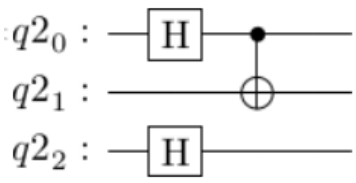

Figure 4.3: A three qubit circuit with MW Measure between 0 and 1 (neither fully entangled, nor fully separable).

$Q(|\psi\rangle) = 2 \times (\frac{(1-\frac{1}{2})+(1-\frac{1}{2})+(1-1)}{3}) = \frac{2}{3}$

4.2 Estimating Entangling Capacity

Entangling Capacity can be estimated by sampling the circuit parameters and computing the average of the MW Measure of the output states.

$$Ent = \frac{1}{|S|} \sum_{\theta_i \in S} Q(|\psi_\theta\rangle)$$

where $S = \{\theta_i\}$ is the set of sampled circuit parameters.

This will give entangling capacity as 0 for the PQCs giving only product states in output and will give entangling capacity close to 1 for a PQC that give that produce highly entangled states in output. The point is that the entangling capacity will always lie between 0 and 1 with boundary values for extreme cases.

4.3 Effect of Different Types of Noises on MW Measure

We know from the definition of the MW Measure that it primarily depends on the quantum state (and hence the density matrix) of the quantum circuit. In this section, we discuss how different kinds of noise affect the density matrix of the circuit which in turn affect the entangling capacity.

4.3.1 Depolarizing Noise

The depolarizing noise consist of one of the three kinds of errors (Pauli Errors):

- **Bit Flip Error:** $\begin{matrix}|0\rangle \to |1\rangle \\ |1\rangle \to |0\rangle\end{matrix}$ or $|\psi\rangle \to \sigma_x$ such that $\sigma_x = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$ is Pauli X Operator

- **Phase Flip Error:** $\begin{matrix}|0\rangle \to |0\rangle \\ |1\rangle \to -|1\rangle\end{matrix}$ or $|\psi\rangle \to \sigma_z$ such that $\sigma_z = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$ is Pauli Z Operator
- **Both:** $\begin{matrix}|0\rangle \to i|1\rangle \\ |1\rangle \to -i|0\rangle\end{matrix}$ or $|\psi\rangle \to \sigma_y$ such that $\sigma_y = \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}$ is Pauli Y Operator

We describe a depolarizing channel using a parameter p which is the probability that the noise occurs. It is assumed that if an error occur, it should occur with equal probability divided among the above three errors.

The depolarizing channel is generally mapped from A to A by using isometric mapping A to AE where E is a four dimensional environment acting on A as:

$$U_{A\to AE} : |\psi_A\rangle \to \sqrt{1-p}\,|\psi\rangle_A \otimes |I\rangle_E + \sqrt{\tfrac{p}{3}}(\sigma_x\,|\psi\rangle_A \otimes |x\rangle_E + \sigma_y|\psi\rangle_A \otimes |y\rangle_E + \sigma_z\,|\psi\rangle_A \otimes |z\rangle_E)$$

Kraus Operator: These are also known as Error Operators or Noise Operators and represents the noise effects of environment. According to Kraus Theorem, $\sum_i K_i K_i^\dagger = 1$ where K_i are the Kraus Operators.

The Kraus operators of Depolarizing Channel are as follows:

$$M_0 = \sqrt{1-p}I, M_1 = \sqrt{\frac{p}{3}}\sigma_x, M_2 = \sqrt{\frac{p}{3}}\sigma_y, M_3 = \sqrt{\frac{p}{3}}\sigma_z$$

where I is the identity matrix and $\sigma_x, \sigma_y, \sigma_z$ are Pauli Operators.

Clearly, $\sum_i M_i M_i^\dagger = ((1-p) + \frac{3p}{3}) = 1$

Density matrix ρ of a qubit in depolarizing channel evolves as follows:

$$\rho \to \rho' = (1-p)\rho + \frac{p}{3}(\sigma_x\rho\sigma_x + \sigma_y\rho\sigma_y + \sigma_z\rho\sigma_z)$$

4.3.2 Amplitude Damping Noise

The amplitude damping channel represents model of a excited state decay of a two level atom. Lets denote the atomic ground state by $|0\rangle_A$ and the excited state as $|1\rangle_A$. Here the "environment" (an electromagnetic field), initially vacuum (denote by $|0\rangle_E$). Now, if the atom is in excited state ($|1\rangle_A$) then with probability p, it will decay to ground state ($|0\rangle_A$), hence emitting a photon and transitioning the environment from state $|0\rangle_E$ (no photon) to state $|1\rangle_E$. This evolution can be represented by the following unitary transformation:

$$\begin{aligned}&|0\rangle_A \otimes |0\rangle_E \to |0\rangle_A \otimes |0\rangle_E \\ &|1\rangle_A \otimes |0\rangle_E \to \sqrt{1-p}\,|1\rangle_A \otimes |0\rangle_E + \sqrt{p}\,|0\rangle_A \otimes |1\rangle_E\end{aligned}$$

Kraus Operator: The Kraus Operators of the Amplitude Damping Channel are as follows:

$$M_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{pmatrix}, M_1 = \begin{pmatrix} 0 & 0 \\ \sqrt{p} & 0 \end{pmatrix}$$

Clearly $\sum_i M_i M_i^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & 1-p \end{pmatrix} + \begin{pmatrix} 0 & p \\ 0 & 0 \end{pmatrix} = 1$

For 1 qubit with density matrix $\rho = \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix}$, the evolution in amplitude damping channel with transition probability p will be as follows:

$$\begin{aligned} \rho \to \rho' &= M_0 \rho M_0^\dagger + M_1 \rho M_1^\dagger \\ &= \begin{pmatrix} \rho_{00} & \rho_{01} \\ \sqrt{1-p}\rho_{10} & \sqrt{1-p}\rho_{11} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{pmatrix} + \begin{pmatrix} \sqrt{p}\rho_{10} & \sqrt{p}\rho_{11} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ \sqrt{p} & 0 \end{pmatrix} \\ &= \begin{pmatrix} \rho_{00} & \sqrt{1-p}\rho_{01} \\ \sqrt{1-p}\rho_{10} & (1-p)\rho_{11} \end{pmatrix} + \begin{pmatrix} p\rho_{11} & 0 \\ 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} \rho_{00} + p\rho_{11} & \sqrt{1-p}\rho_{01} \\ \sqrt{1-p}\rho_{10} & (1-p)\rho_{11} \end{pmatrix} \end{aligned}$$

4.3.3 Effect of Kraus Operator on a Multi-Qubit System

For a 4-qubit circuit, the Kraus Operator would evolve its density matrix ρ as follows:

$$\rho \to \rho' = \sum_{i,j,k,l \in [0,n]} (M_i \otimes M_j \otimes M_k \otimes M_l) \rho (M_i^\otimes M_j \otimes M_k \otimes M_k)$$

for a Quantum Channel with n Kraus Operators. For Depolarization Channel, $n = 4$ and for Amplitude Damping Channel, $n = 2$.

A general quantum circuit is unitary. However, noise in the system, in general, evolves it into a mixed state. However, MW Measure is defined only for pure states, and hence is not applicable for a noisy quantum system which leads to a mixed state.

Other entanglement measures have been defined which are applicable to mixed states as well [LS98]. One such measure is Lewenstein-Sanpera entanglement measure where a general density matrix ρ is decomposed into a convex combination of a maximally entangled state ρ_e and a separable state ρ_s as follows:

$$\rho = \lambda \rho_s + (1-\lambda)\rho_e$$

where $\lambda \in \{0, 1\}$. Under such a decomposition, $1 - \lambda$ is a measure of entanglement of

the system. However, most of the general density matrices cannot be represented as such. For example, a density matrix evolved under depolarization or amplitude damping noise cannot be represented in such a form. For such scenarios, a *good enough approximation* is proposed where ρ_s, ρ_e and λ are chosen so as that their convex combination is *close enough* to the original state ρ. However, the authors themselves state that this is a computationally expensive problem, and hence is not suitable for most practical scenarios. Furthermore, in this paper [LS98], the authors defined this decomposition only for two qubit states, and do not provide any means for its extension to multi-qubit systems.

Therefore, study of entanglement measure of noisy circuits remain a problem of interest to be pursued as a future research.

4.4 Experiments and Results

4.4.1 Entangling Capacity Depends only on the quantum state

Unlike expessibility, we only need the quantum state (by computing its density matrix) of a particular PQC and not the distribution of its parameters (which we compute by running the circuits for multiple shots) for computing entangling capacity. Hence neither topology nor noise of the machine has any effect on the way we compute it. Qiskit can provide us with the density matrix for any PQC and we can calculate its entangling capacity without actually running those circuits, just by sampling a set of circuit parameters and estimating the entangling capacity for it.

We computed entangling capacity for the same 19 circuits discussed in the previous chapter (in Figure 3.2) by taking average / expectation of the MW Measure on a sample size of 2000 for the circuit parameters. The results for the extreme circuits (0 and 1) for circuit 1 and circuit 9 was same as given by [SJAG19]. However all results on all the other circuits were different then what was mentioned in that paper even after using the same formulae given in that. 4.6.

4.4.2 An explanation for difference in results with the reference Paper [SJAG19]

One possible explanation to that could be that the number of parameter sets used by them could be smaller. We observed that the Entangling Capacity of Circuit 1 and Circuit 9 were immune to the number of the parameter sets used to take the expectation of the MW Measure of them. Even for a single parameter set, the value of the MW Measure remained the same. Hence as we can observe from Figure 4.6 that the entangling capacity as computed

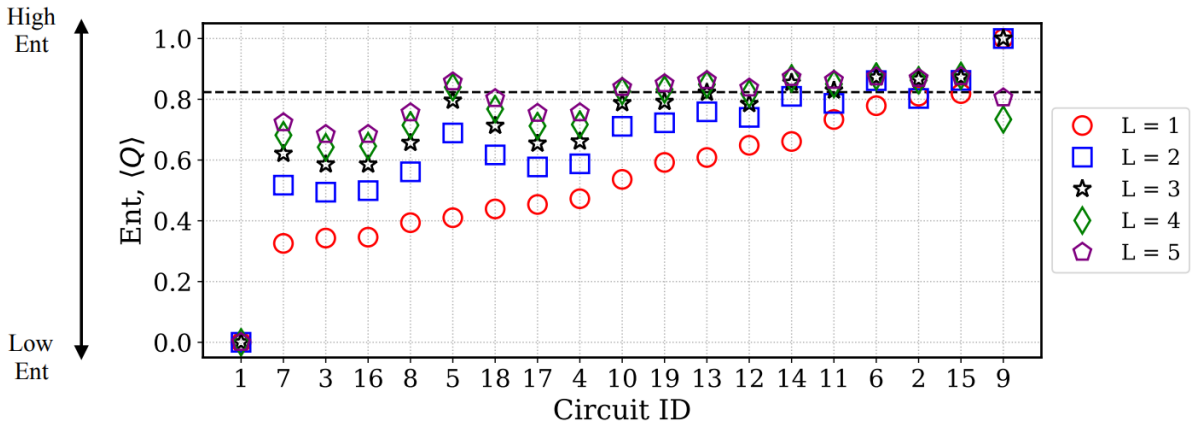

Figure 4.4: Entangling Capacity of the benchmark circuits as given in the paper [SJAG19]

by us and [SJAG19] for circuits 1 and 9 are the same. However for all the other circuits, the entangling capacity was varying on changing the number of the parameter sets (for taking expectation). As we know that as we increase the number of independent experiments (with a different parameter set each time) which are actually a sample of parameter sets, the entangling capacity should converge to the actual expectation for entire population of the parameter sets. The value of the entangling capacity remained almost the same time at each independent experiment. So our results of the Entangling Capacity should be closer to the ground truth. Figure 4.4 shows the entangling capacity given in [SJAG19] and 4.5 shows the entangling capacity that we computed.

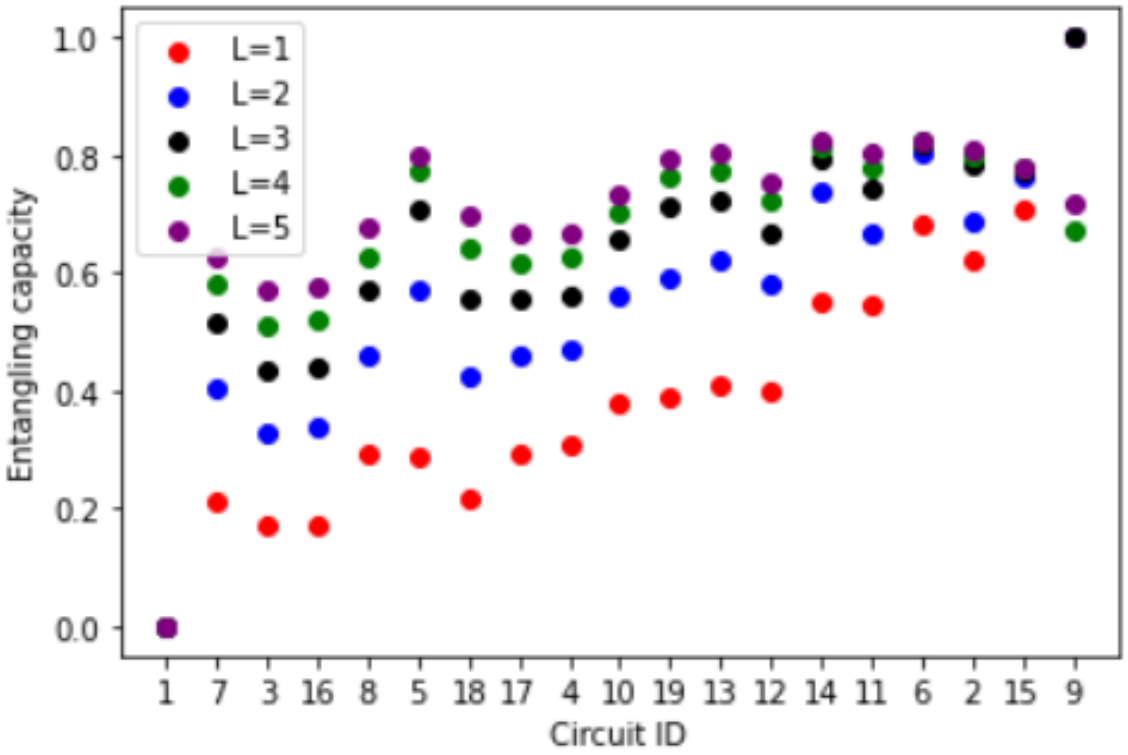

Figure 4.5: Entangling Capacity of benchmark circuits that we computed

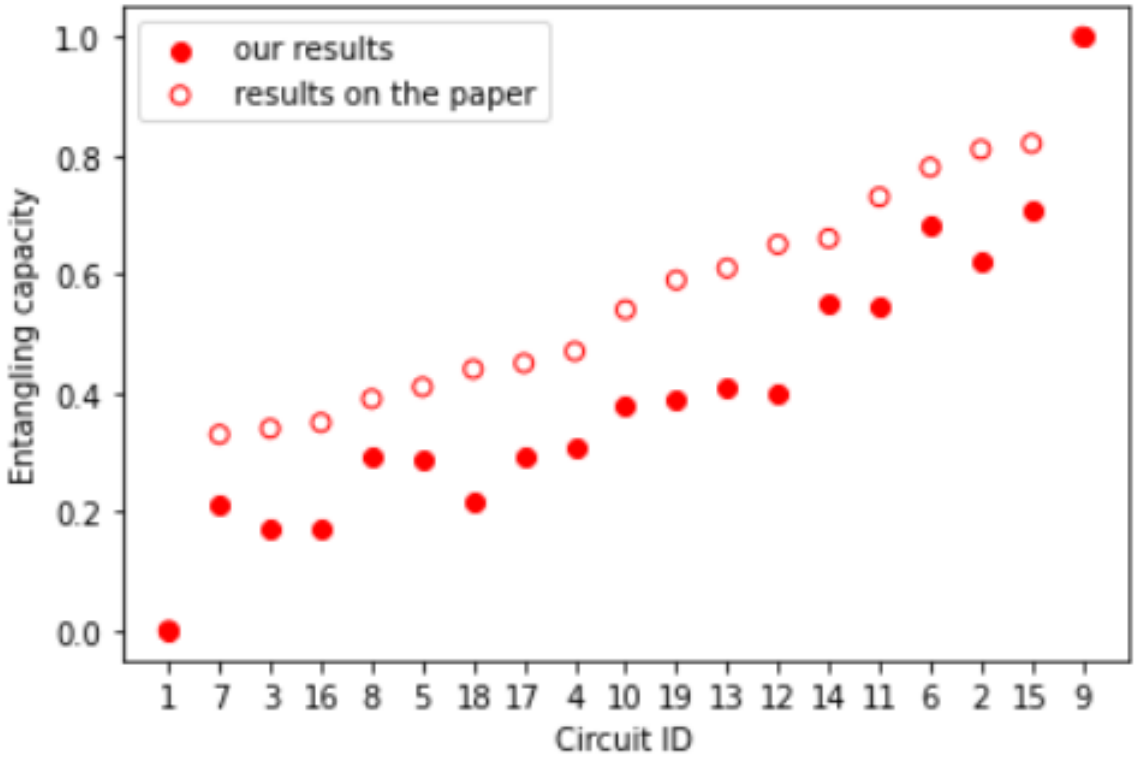

Figure 4.6: Entangling Capacity for single layer circuit as computed by us vs [SJAG19]

Chapter 5

Importance of Expressibility and Entangling Capacity in Quantum Neural Networks

In previous chapters, we discussed in detail about expressibility and entangling capacity. In this chapter we discuss the results given by the PQCs (in Figure 3.2) when trained as a Quantum Neural Network and try to find the importance of the above two metrics when compared with accuracy, precision and recall on a "Fraud Detection Synthetic Dataset" that we trained these circuits on.

5.1 Classical Neural Networks

Classical Neural Networks (or simply Neural Networks) which are often referred as Artificial Neural Networks (ANN) or Multi Layer Perceptrons are a class of machine learning model inspired from the human brain mimicking the way the biological neurons send message / signal to each other. These are comprised of multiple layers consisting of nodes (often also referred as neurons and are inspired from biological neurons). The first layer is the input layers. The number of nodes in the input layer has to be equal to the number of features we are taking in our training dataset. The features in a dataset are nothing but columns of data that are used to predict its output on the relevant task (regression or classification). After the first layer, there are generally one or more hidden layers. The number of hidden layers determine the complexity and expressible power of the the model. The number of nodes in the hidden layer is hyper-parameter changing which changes the model. The last hidden layer is connected to the output layer. The number of nodes in the output layer depends on our data and problem. Each node has an "Activation Function" which is generally non linear for introducing non linearity in the model in order for it to be able to learn more complex functions. For regression, we have only one node in the output layer where as for classification, the number of nodes in the output layer can be the number of classes in our data. The output of each node in the output layer gives the predicted probability for that class. In case of regression, the last layer gives the predicted value.

When the data features from the input layer pass through multiple layers by multiplying weight of each layer and gives the predicted value in the output layer, this is called as forward pass in a training session. The output that we get from the output layer is compared with the ground truth value which gives us the loss (the extent by which the predicted value is different from the ground truth). In Figure 5.1 which shows an Fully Connected ANN,

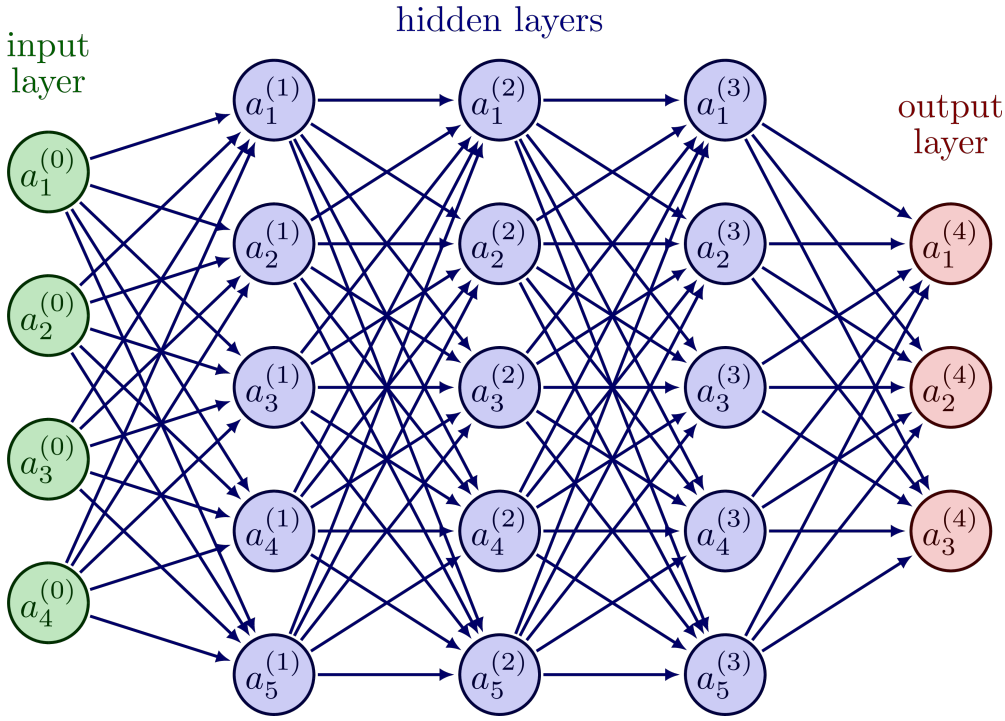

Figure 5.1: An Artificial Neural Network (courtesy: https://tikz.net/neural_networks/)

$a_i^{(j)}$ refers to i^{th} node of j^{th} layer. Fully connects refers to the fact that each node in a layer is connected to each node of the next layer. The links connecting the nodes contain weight which actually are the trainable parameters. Theoretically, an ANN can represent any complex function hence can learn anything. The update in the trainable parameter is done iteratively using some Optimization technique. We consider a loss function which depends on the ground truth value and the predicted value. This loss function is optimized using some optimization technique to reduce is to minimum and the weights of the neural network is changed accordingly. This step is called feedback or backward pass in a training session.

For instance, lets consider the example of House Price Prediction (taken from Andrew Ng's Machine Learning Course notes [Ng12]). We have four features in our dataset (House size, number of bedrooms, Zip Code (for location) and Wealth (of the locality). Accordingly in 5.2, we have four nodes in the input layer (representing Size, Bedrooms, Zip Code and Wealth respectively). Let *ReLU* (Rectified Linear Unit) be the non linear activation function at each node. The *ReLU* activation function is defined as follows:

$$ReLU(x) = \begin{cases} x, & \text{if x>0} \\ 0, & \text{otherwise} \end{cases}$$

Now lets suppose that the price of the house depends on the the size of the family (capacity of the house in terms of the size of the family it can accommodate may be a function of the house size and rooms), and how walk-able is the neighborhood, for example

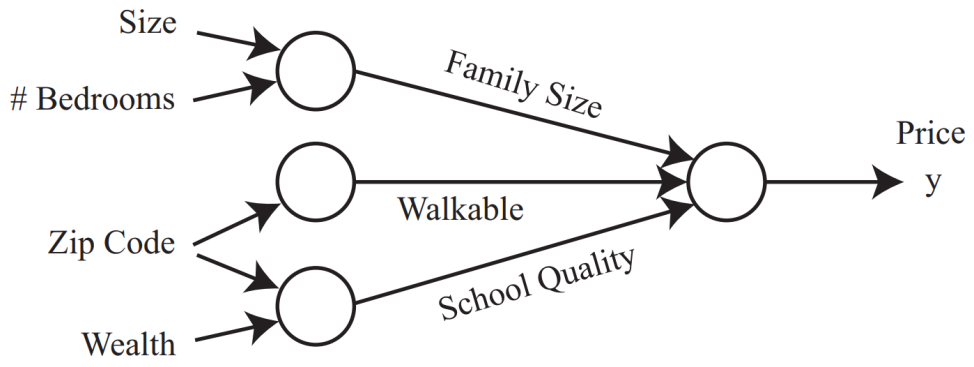

Figure 5.2: A small neural network for predicting housing prices. (courtesy: [Ng12])

is every basic need (like grocery shops) are fulfilled in walking distance (this can be a function of the location of the are or the Zip Code) and the School Quality in the locality (which could be a function of the Zip Code and wealth of the people living in the locality). Hence we get the derived features (Family Size, Walk-able and School Quality) which can be represented by the nodes in the hidden layer. The hidden layer is connected to the output layer which would give us price. Now lets say that the output of each node of the hidden layer is given by a_1,a_2 and a_3 and the inputs are x_1, x_2, x_3 and x_4. Then a_1,a_2 and a_3 can be written as:

$$a_1 = ReLU(w_1x_1 + w_2x_2 + w_3)$$

$$a_2 = ReLU(w_4x_3 + w_5)$$

$$a_3 = ReLU(w_6x_3 + w_7x_4 + w_8)$$

where $w_1, w_2, w_3, w_4, w_5, w_6, w_7$ and w_8 are learn-able weights or parameters. The final output that we will get from the output layer that will be the predicted house price (if the activation function is linear / identity, can be written as:

$$h_w(x) = w_9a_1 + w_{10}a_2 + w_{11}a_3 + w_{12}$$

where a_1,a_2 and a_3 are the outputs from the previous layer and w_9, w_{10}, w_{11} and w_{12} are learn-able weights or parameters. These learn-able weights are tuned using iterative optimization to minimize the loss function. What exactly the model learns through hidden layer is not interpret-able. We took the example of Family Size, Walk-able and School Quality for the sake of understanding what a neural network can learn using optimization and how the length of hidden layer is an important hyper-parameter.

In general, we represent these w_is, x_isanda_is as vectors. We have m data points in the dataset. The updated equation would be:

$$a_j = ReLU(\theta_j^{[1]^T}x + b_j[1]), \theta_j^{[1]} \in R^4, b_j^{[1]} \in R$$

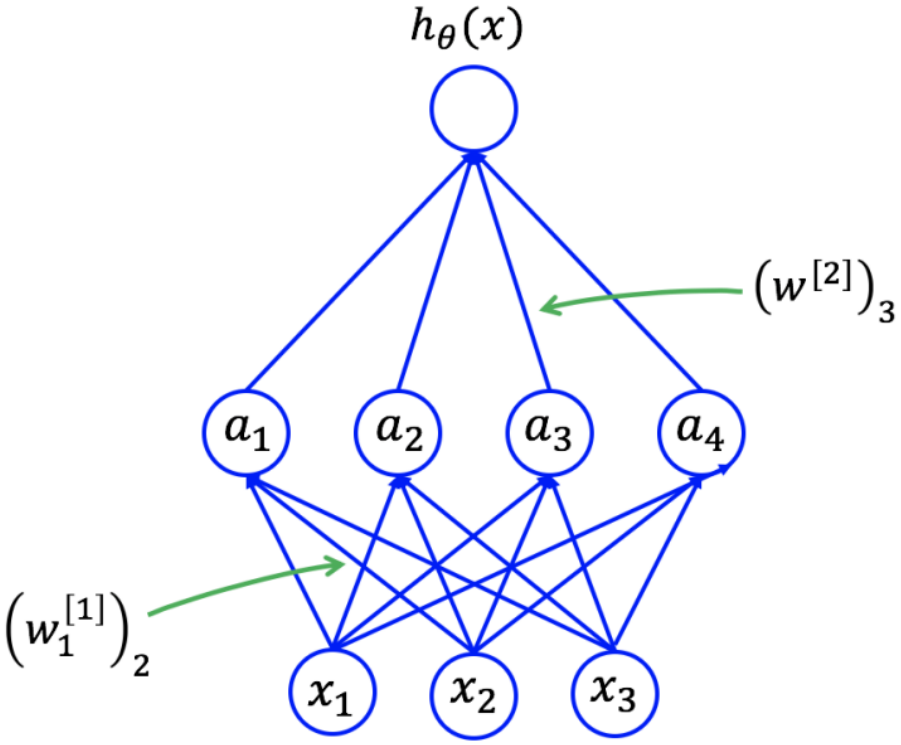

Figure 5.3: A representation of neural network for house price prediction example (courtesy: [Ng12])

$$a = [a_1, \ldots a_m]^T \in R^m$$

$$h_\theta(x) = \theta^{[2]^T} + b^{[2]}, \theta^{[2]} \in R^m, b^{[2]} \in R$$

Figure 5.3 shows the above in diagrammatic format in a neural network.

5.2 Quantum Neural Networks (QNN)

Quantum Neural Network (QNN) often also referred as Variational Quantum Classifier / Regressor is nothing but a PQC with its parameters trained just like the parameters of a Neural Network. This is the best example of HQC Algorithms as the the forward pass of the training happens in the quantum circuit. A QNN comprises of two parts as shown in 5.4:

- Feature Map
- Variational Model

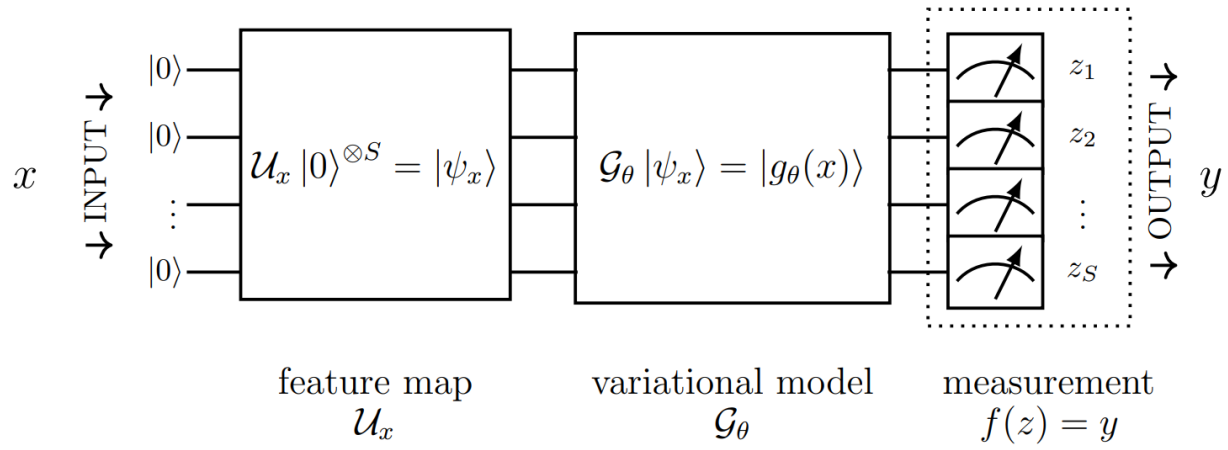

Figure 5.4: General architecture of QNN (courtesy: [ASZ+21])

5.2.1 Feature Map

A QNN being based on quantum circuits can't directly learn from the classical data as an ANN does. It needs to convert the classical data into a quantum state in order to work in a QNN. This is what the feature map does. A feature map is basically a quantum circuit that encodes the classical data (basically its features) into a quantum state. Generally the feature map is kept fixed and its parameters are not changed while optimization and learning. Features may be represented as angle of the quantum gates in the Feature Map or the Amplitude of the quantum state depending on the feature map we used. Some specific feature maps are discussed below:

- **Amplitude Encoding:** As the name suggests, in this encoding technique each feature/column of our dataset is taken as amplitude of the Quantum State (which are 2^n for n qubits). So to encode a n dimensional data, we need a $\log_2 n$ qubit circuit. The data is first normalized to 1 such that the sum of their square is 1 so as to get a normalized quantum state with the features as amplitudes. Hence the number of amplitudes to be encoded would be the product of number of dimensions an number of data points.

  $$|\psi(x)\rangle = \sum_{i=1}^{N} x_i |i\rangle$$

  For an n dimensional dataset $\alpha = \{a_1, a_2, \ldots, a_n\}$, $|\alpha|^2 = 1$

- **Basis Encoding:** In this encoding technique, the feature is first converted to a binary string which is then encoded to a computational basis state. If the the binary form of a data is 1010 then it would be encoded as $|1010\rangle$. For some data D with number of samples M and number of dimensions m, it can be represented as follows:

  $$|D\rangle = \frac{1}{\sqrt{M}} \sum_{i=1}^{M} |x^{(m)}\rangle$$

- **Angle Encoding:** In this feature map, the features are encoded as angles of R_X or R_Y gates. For an n dimensional data, we need an n qubit system with R_X (or R_Y gates) in each wire in the Feature Map.

5.2.2 Variational Form

In the variational form, a PQC is used which is has changeable parameters which can be learned using optimization in the same way as we learn the trainable weights in ANN. So in the encoded data (in form of Quantum State), this variational form is applied and optimized for the required task [ZLW19] [ROAG17b] [DB18] [CCL19]. The result generated by the QNN is then given to a classical computer for optimization by calculating its loss by comparing with the ground truth values. The feedback (backward pass) is given to the Quantum Computer and the parameters are changed accordingly.

5.2.3 Why QNN?

The question may arise: "What exactly is the point of a neural network of quantum analog if we still need a classical computer with it?" A properly designed Neural Network is advantageous then the classical neural networks in two very important aspects viz faster training ability (can learn same things in few iterations) and a higher Effective Dimension [ASZ+21] which means that a QNN can have increased capability and train-ability. *Effective Dimension* is basically defined as the proportion of the dimensions that are actually actively being used with the overall dimension being used by a neural network.

One big motivation for QNN is the difficulty we face in training ANN in big data applications as a Quantum Neural Network has faster training ability owing to quantum parallelism. However, since the actual technological implementation of Quantum Computer is in very premature stage, most of these ideas of Quantum Neural Networks are only theoretical proposals and full fledged physical experiments are still awaited.

5.3 Fraud Transaction Detection using Quantum Neural Network

We considered the Fraud Transaction Detection problem and used the 19 circuits in Figure 3.2 to train them as the variational form of QNN. The goal of this study was to find the importance of expressibility and entangling capacity of the circuit used for QNN in its ability to learn the distribution of a dataset. The Details of the dataset, feature map, optimizer used along with the feature engineering and the train-test data set split is discussed below.

5.3.1 Dataset used: Fraud Detection Synthetic Dataset

This dataset was generated by PaySim, a synthetic dataset generator on Mobile Money Transactions based on a sample of real transactions extracted from one month of from financial logs from a mobile money service implemented in an African country [Elm16]. The dataset is available on Kaggle (https://www.kaggle.com/datasets/ealaxi/paysim1). This dataset has 10 columns: step (which maps to time unit), type (cash out or online payment), amount (the transaction amount), nameOrig (name / primary key of the person who did the transaction, say money sender), oldbalanceOrg (old bank account balance of the money sender, before the transaction), newbalanceOrig (the new account balance of the money sender after the transaction), nameDest (name / primary key of the person who received the amount, say the money reciever), oldbalanceDest (old bank account balance of the money receiver, before the transaction), newbalanceDest (the new account balance of the money receiver after the transaction), isFraud (if its a fraud transaction).

5.3.2 Feature Map Used: Amplitude Encoding

QNN can't directly learn from classical data. We first need to convert the data into quantum state to be able to optimize the PQC so as to learn the distribution of the data. These encoding again are actually done by specific Quantum Circuits when data is given to them in input. We used two amplitude encoding as the feature map for our experiments.

As the name suggests, in this encoding technique each feature / column of our dataset is taken as amplitude of the Quantum State (which are 2^n for n qubits). So for 4 qubits PQCs we could train datasets of upto 16 features. Hence we used this technique for our problem as we had total 9 features. We used RawFeatureVector circuit available in Qiskit for this purpose.

5.3.3 Optimizer Used: COBYLA

COBYLA (Constrained optimization by linear approximation) is an Optimization Technique for Constrained Optimization Problems where the derivative of the objective function is not used [Pow07]. As of now, the features in Amplitude Encoding is not differentiable in Both of the major Quantum Machine Learning frameworks (qiskit and pennylane). Because of this we used COBYLA as our optimizer.

In this Optimization Technique, the Constrained Optimization Problem is iteratively approximated by Linear Programming problem. At each iteration, an approximating linear programming problem is solved to obtain a candidate for the optimal solution. This candidate solution is then evaluated with the original and constrained function so as to find

a new data point in the Optimization Space. The step size of finding a new data point is reduced when the algorithm is unable to improve. The algorithm stops when the step size becomes sufficiently small.

5.3.4 Feature Engineering

As per the PaySim Thesis [Elm16], most often method used by fraudsters is to send an amount of money from one account to a cash out counter of the Company using online transaction. And then take cash from this cash out counter as cash transactions are easier to hide. This process may include a chain of online transactions from one account to another finally reaching the cash counter and taking cash from there. This makes all the transaction of this chain as fraud transactions. However in our dataset, a single data point does not contain any information about such a chain coming from other transaction. It only has the general information of the amount of cash taken out, previous account balance of both, the sender and the receiver with their IDs. Graph Neural Network can be a better alternative to solve such a problem than a simple neural network but we don't have Quantum Graph Neural Network on any framework as of now. So we had to add features in this dataset itself so that a normal Neural Network can also learn properly in this.

For this, we added two new features "t2t" and "t2c" in our data. We sorted our dataset in increasing order of the transaction amount and time stamp. We ran a loop over all the data points of the sorted dataset and considered two data points at a time. If the two data points have same transaction amount and both are online transactions, we set t2t as 1 and if one transaction is cash out, we flag t2c as 1.

We removed the step (time stamp), the nameOrig and nameDest (names of sender and receiver) features from the dataset and used the remaining dataset for training.

5.3.5 Dataset Division for Training and Testing

The dataset had 2770393 data points for transactions including "Cash out" and "Mobile Transfer" Category (No other category of transaction had any fraud cases). Out of these, only 8213 data points represent fraud transaction. Since we were running our experiments on Simulator of the actual machine (which will actually run on the classical computer), we need a long training time for each training session. Hence we were forced to take limited data instances. We kept the size of our training dataset as 8000. Now that we are taking only this small fraction of the entire data, we can actually selectively take data points to get a well balanced training data set. So we took 4000 data points of fraud transactions and 4000 data points of non fraud transactions in our training set. For the test set, we took the rest of 4213 fraud transaction data points and 8000 non fraud transaction data points. So

the size of our test data set was 12213. We divided the data points on the training and test set randomly so as ensure that a a variety of data points are covered.

5.4 Experiments and Results

5.4.1 Experimental Setup

We trained all the 19 circuits in Figure 3.2 as variational Form (for single and double layer of circuits) of QNN with amplitude encoding as the feature map with the "Fraud Detection Synthetic Dataset" to see the improvements in the F1 Score and Accuracy with increase in expressibility and entangling capacity so as to see the importance of these measures while deciding a PQC for using in QNN. Statevector simulator is used for this task. We use two scores to grade the quality of training viz Accuracy and f1 score.

We are considering the Fraud Class as the positive class.

If total number of data points $= n$,
total number of correctly predicted data points $= c$,
total number of true positives $= TP$,
total number of false positives $= FP$,
total number of true negatives $= TN$ and
total number of false negatives $= FN$.

Then Accuracy is defined as:

$$Accuracy = \frac{c}{n} = \frac{TP + TN}{n}$$

Precision is defined as:

$$Precision = \frac{TP}{TP + FP}$$

Recall is defined as:

$$Recall = \frac{TP}{TP + FN}$$

F1 score is defined as the Harmonic mean of precision and recall:

$$f1score = \frac{Precision \times Recall}{Precision + Recall}$$

Tools Used: Python programming language, qiskit_machine_learning framework.

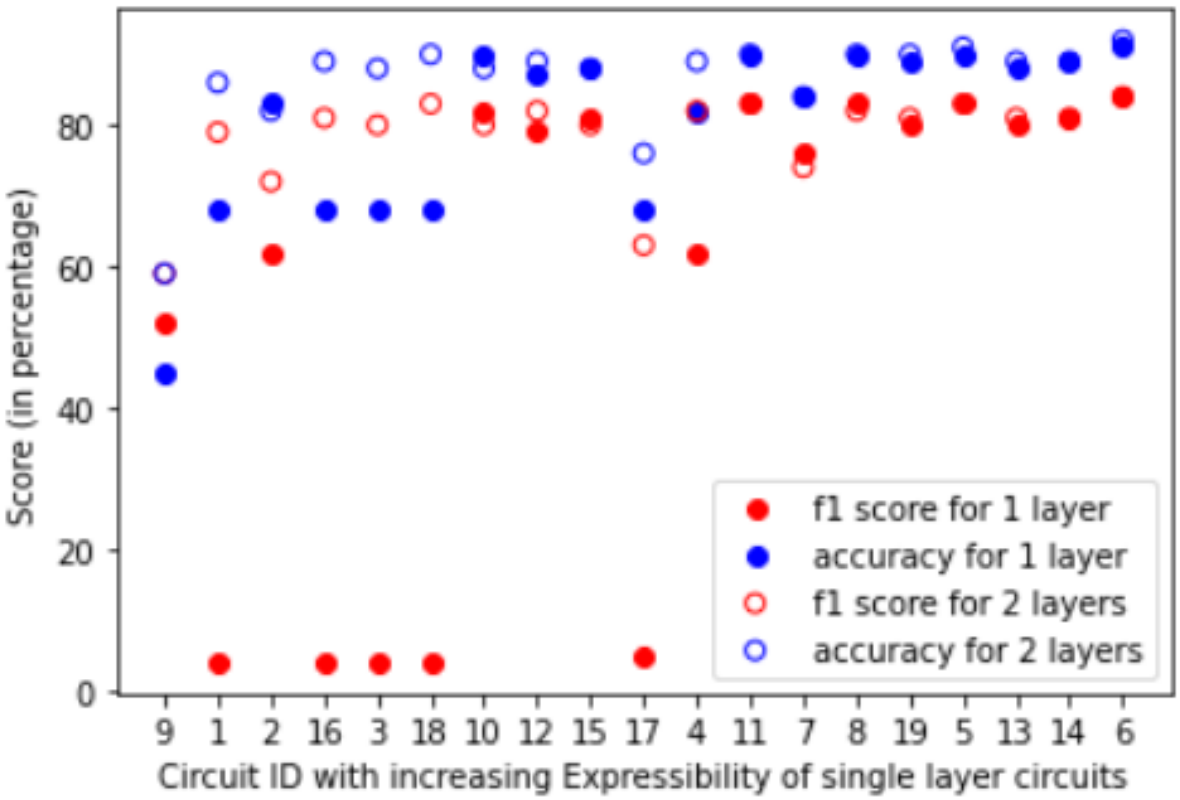

Figure 5.5: Accuracy and F1 Score on different circuits with increasing expressibility of single layer circuits

5.4.2 Results and Observations

Figure 5.5 shows the accuracy and f1 Scores we got with circuits ordered in increasing expressibility of single layer circuits and Figure 5.6 shows the accuracy and f1 Scores we got with circuits ordered in increasing entangling capacity of single layer circuits .

Observations on Scores of Single Layer Circuit based QNN

As we can clearly observe, the f1 Score in Circuits (for single layer) 1,3,16,17 and 18 is very low. This is complemented with the fact that Circuits 1,3,16 and 18 are low in both expressibility (KL Divergences $= 0.35, 0.24, 0.3, 0.27$ respectively) and Entangling Capacity of $0, 0.19, 0.19, 0.22$ respectively. For Circuit 17, the Entangling Capacity is around 0.3 and expressibility (KL Divergence $= 0.14$) also not much higher than other lowest expressible circuits considered for the Study.

Circuit 9 is lowest expressible but has entangling capacity of 1, so we get an f1 score of 52 with accuracy even lower than the f1 score which means that this model could not even learn the "Non Fraud" class. From this, it can be concluded that a circuit with extremely low expressibility (KL Divergence $= 0.67$) should not be used for QNN even if it is maximally

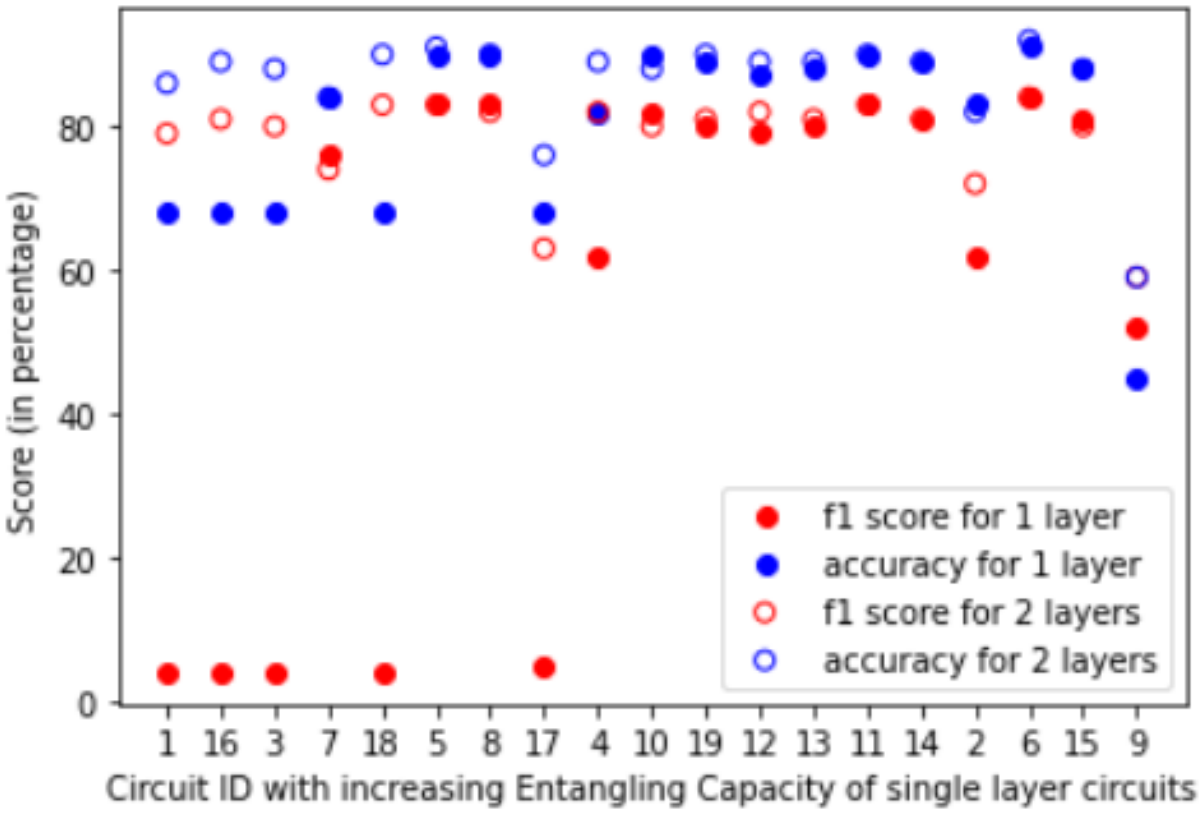

Figure 5.6: Accuracy and F1 Score on different circuits with increasing entangling capacity of single layer circuits

entangled. QNN with Circuit 9 is acting like a dumb model which is randomly predicting anything without learning anything significant.

Circuit 2 has very low expressibility (KL Divergence = 0.3) and but high entangling capacity of 0.6. So we get f1 Score of 62% and accuracy of 82%. This means that entangling capacity is also a useful measure to consider while choosing a PQC for QNN. A circuit of high entangling capacity and as compared lower expressibility is able to learn to a good extent. However circuits with extremely low expressibilty (with KL Divergence > 0.5) should never be taken. An unexpressible circuit can't learn anything even if its maximally entangled. This makes expressibility a more important measure for this task but that doesn't mean that Entangling Capacity is not useful which can be proved by the example of Circuit 2.

For Circuit 4, the f1 score we got was same as what we got from Circit 2 (62%) and the accuracy was also almost the same (83% and 82% respectively). However in this case, both the expressibility and entangling capacity was very close to Circuit 17 (which has f1 score of 4% and apparently can't learn the distribution of fraud class). In fact Circuits 17 and 4 are very much similar in structure with all the gates same just with order of one R_X gate different. This proves that expressibility and entangling capacity are useful to decide a circuit but these are not full proof.

In Circuit 7, the expressibility is on a higher side as compared to the other circuits (KL Divergence = 0.1) but the Entangling Capacity is very low (0.2) which is impacting the overall score (f1 score of 76%) as compared to other Circuits.

In Circuits 10 and 12, we get a very good score (f1 score close to 80%). But these have entangling capacity a little higher than Circuits 17 and 4 (0.4). and expressibility a little lower than Circuits 17 and 4 (KL Divergence = $0.25, 0.19$ respectively). Which on one hand again points that expressibility and entangling capacity are useful to decide a circuit but these are not full proof but on the other hand also points that entangling capacity of 0.4 is good enough.

All the other circuits are high enough on both expressibility and entangling capacity, assuming 0.4 is good enough for entangling capacity and KL Divergence less than 0.12 is good enough for expressibility.

Observations on Scores of Double Layer Circuit based QNN

As we append one more layer of the same circuit in the QNN, both accuracy and f1 score of almost all the circuits either increased (if it was low) or remained the same (if it was already high enough).

For circuit 1, even though there is very minute improvement in expressibility (KL Divergence decrease from 0.35 to 0.2) and no improvement in entangling capacity but the

accuracy and precision score shooted up to a great extent (f1 score went up from 4% to 79%). This is consistent with the observations from single layer circuit that we need only a threshold expressibility to make a good enough QNN model. Similar improvements in score can be observed with circuits 3,16,17 and 18 (in which the score were otherwise too poor for single layer).

The improvement of score is not very significant for circuit 9 (f1 score increased from 52% to 59%). This is because the the expressibility of even the double layer of circuit 9 is less than even single layer of other circuits considered (KL Divergence is 0.41). And the entangling capacity remains the same with this increase in layer. So this minute improvement is expressibility (KL Divergence decreased from 0.66 to 0.41), the scores have also improved accordingly.

However, we don't see any improvements in the scores we get by increasing the layer of circuit 7 even though both expressibility and entangling capacity improve to a significant extent on doing this. This again shows that expressibility and entangling capacity are very helpful in finding a good PQC for QNN but is not full proof.

5.4.3 Concluding Remarks

Our experiments show that expressibility and entangling capacity can be very useful measures while deciding a PQC for training as a QNN. Even though these measures are very useful but these are not fool proof and should not be completely relied on. An Entangling Capacity of more than 0.4 and expressibility with KL Divergence less than 0.12 was found to be good enough for a circuit to be used as the Variational form of QNN.

.

Chapter 6

Discussion and Conclusion

In this study, we did in depth study especially on two particular metrics used to quantify the solution space covered by a PQC. As discussed previously, the expressibility is a statistical property and the way we compute it is affected greatly by the noise of the machine that we are using to compute it since we need to find the fidelity and actually run the circuits for doing that. On the other hand, computing the entangling capacity doesn't need us to actually run the circuits. We can compute it directly using the density matrix of the circuit.

With our experiments we found that even though the topology of the machine doesn't have much effect on the expressibilty value that we get, however for some specific circuits some specific topology were a little different than the QASM Simulator. Experimenting more on it and studying it theoretically more in depth is a good future direction. Some new algorithms may also be devised to compute expressibilty which are more immune to the noise.

We found that the entangling capacity based on the MW Measure can not be used to calculate the effect of quantum channels on the Circuit as it evolves the density matrix of the circuit to mixed state and the MW Measure is not applicable for mixed states. If we are able to decompose the density matrix to two components one pure entangled state and other being mixed pure state than we can use the density matrix of the pure entangled state to compute the entangling capacity but this decomposition is computationally too expensive.

We used the 19 circuits (in Figure 3.2) that we considered for this study as the variational form of QNN to see the importance of expressibility and Entangling Capacity in deciding a circuit for QNN. We found that we don't really need a maximally entangled circuit nor a very high expressible circuit as these comes with the cost of circuit complexity and high expenses. We can smartly use a circuit with good enough expressibility (for our case we found it with KL Divergence less than 0.11) and Entangling Capacity (for our case we found entangling capacity more than 0.4 good enough). These measures can surely be used to decide the PQC for QNN but this should also be kept in mind that these are not full proof and we should use them carefully.

Appendix A

Code Snippets Supporting this Study

This is the code snippet to Estimating Probability Distribution of Haar Random States:

```
#Possible Bin with bin list length of 75
bins_list=[];
for i in range(76):
    bins_list.append((i)/75)
#Center of the Bean
bins_x=[]
for i in range(75):
    bins_x.append(bins_list[1]+bins_list[i])
def P_harr(l,u,N):
    return (1-l)**(N-1)-(1-u)**(N-1)
#Harr historgram
P_harr_hist=[]
for i in range(75):
    P_harr_hist.append(P_harr(bins_list[i],bins_list[i+1],2))
```

The following code snippet illustates how we computed fidelity for circuit A in Figure 3.1 as discussed in Chapter 3.

```
nshot=10000
nparam=4000
fidelity=[]
for x in range(nparam):
    th1=2*pi*random()
    th2=2*pi*random()
    qr = QuantumRegister(1)
    cr = ClassicalRegister(1)
    qc = QuantumCircuit(qr, cr)
    qc.h(qr[0])
    qc.rz(th1, qr[0])
    qc.rz(th2, qr[0])
```

```
    qc.h(qr[0])
    qc.measure(qr[0],cr[0])
    job = execute(qc, backend, shots=nshot)
    result = job.result()
    count =result.get_counts()

    if '0' in count and '1' in count:
        ratio=count['0']/nshot
    elif '0' in count and '1' not in count:
        ratio=count['0']/nshot
    else:
        ratio=0

    fidelity.append(ratio)
```

The following code snippet illustates how we computed fidelity for circuit A in Figure 3.2 as discussed in Chapter 3.

```
backend = Aer.get_backend('qasm_simulator')

arr = []
for kk in range(19):
    arr.append([])
    for lo in [1,2,3,4,5]:
        nshot=1000
        nparam=2000
        fidelity=[]
        for x in range(nparam):
            qr = QuantumRegister(4)
            cr = ClassicalRegister(4)
            qc = QuantumCircuit(qr, cr)

            theta=[];
            for y in range(500):
                theta.append(2*pi*random())

            qc=list_of_circuit[kk](qc,qr,theta,lo,1)


            qc.measure(qr[:],cr[:])
```

```
            job = execute(qc, backend, shots=nshot)
            result = job.result()
            count =result.get_counts()

            if '0000' in count and '1' in count:
                ratio=count['0000']/nshot
            elif '0000' in count and '1' not in count:
                ratio=count['0000']/nshot
            else:
                ratio=0

            fidelity.append(ratio)

        weights = np.ones_like(fidelity)/float(len(fidelity))
```

This is the code snippet to compute the Meyer Wallach Measure:

```
def entangle_measure_nsv(qc,n):
    dm=DensityMatrix(qc)
    qb=list(range(n))
    traces = []
    entropy=0
    for i in range(n):
        pt= partial_trace(dm,qb[:i]+qb[i+1:]).data
        trace = np.dot(pt,pt).trace()
        entropy+=trace
    entropy/=n
    return abs(2*(1-entropy))
```

Bibliography

[ASZ+21] Amira Abbas, David Sutter, Christa Zoufal, Aurelien Lucchi, Alessio Figalli, and Stefan Woerner. The power of quantum neural networks. *Nature Computational Science*, 1(6):403–409, jun 2021.

[CCL19] Iris Cong, Soonwon Choi, and Mikhail D. Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, aug 2019.

[CRO+19] Yudong Cao, Jonathan Romero, Jonathan P. Olson, Matthias Degroote, Peter D. Johnson, Mária Kieferová, Ian D. Kivlichan, Tim Menke, Borja Peropadre, Nicolas P. D. Sawaya, Sukin Sim, Libor Veis, and Alán Aspuru-Guzik. Quantum chemistry in the age of quantum computing. *Chemical Reviews*, 119(19):10856–10915, Aug 2019.

[DB18] Vedran Dunjko and Hans J Briegel. Machine learning &amp artificial intelligence in the quantum domain: a review of recent progress. *Reports on Progress in Physics*, 81(7):074001, jun 2018.

[Elm16] Ahmad Elmir. Paysim financial simulator : Paysim financial simulator, 2016.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[FN20] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. Dec 2020.

[HCT+19] Vojtěch Havlíček, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, mar 2019.

[idZS05] Karol Życzkowski and Hans-Jürgen Sommers. Average fidelity between random quantum states. *Phys. Rev. A*, 71:032313, Mar 2005.

[JOCAG17] Peter Johnson, Jonathan Olson, Yudong Cao, and Alán Aspuru-Guzik. Qvector: an algorithm for device-tailored quantum error correction. 11 2017.

[KL51] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79 – 86, 1951.

[KMT+17] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549:242–246, 2017.

[LS98] Maciej Lewenstein and Anna Sanpera. Separability and entanglement of composite quantum systems. *Physical Review Letters*, 80(11):2261–2264, mar 1998.

[MRBAG16] Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, feb 2016.

[MTK+00] C Monroe, Q Turchette, D Kielpinski, B King, V Meyer, Wayne Itano, David Wineland, C Myatt, M Rowe, Christopher Langer, and C Sackett. *Scalable Entanglement of Trapped Ions*. AIP Proceedings 551, Melville, NY, 2000-01-01 00:01:00 2000.

[MW02] David A. Meyer and Nolan R. Wallach. Global entanglement in multiparticle systems. *Journal of Mathematical Physics*, 43(9):4273–4278, 2002.

[NC11] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, USA, 10th edition, 2011.

[NDD+03] Michael A. Nielsen, Christopher M. Dawson, Jennifer L. Dodd, Alexei Gilchrist, Duncan Mortimer, Tobias J. Osborne, Michael J. Bremner, Aram W. Harrow, and Andrew Hines. Quantum dynamics as a physical resource. *Phys. Rev. A*, 67:052301, May 2003.

[Ng12] Andrew Ng. Cs229 lecture notes - supervised learning. 2012.

[Pow07] M. J. D. Powell. A view of algorithms for optimization without derivatives 1. 2007.

[Pre18] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.

[ROAG17a] Jonathan Romero, Jonathan Olson, and Alán Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *arXiv: Quantum Physics*, 2:045001, 2017.

[ROAG17b] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, aug 2017.

[SBSW20] Maria Schuld, Alex Bocharov, Krysta M. Svore, and Nathan Wiebe. Circuit-centric quantum classifiers. *Physical Review A*, 101(3), mar 2020.

[SJAG19] Sukin Sim, Peter D. Johnson, and Alá n Aspuru-Guzik. Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. *Advanced Quantum Technologies*, 2(12):1900070, oct 2019.

[WBLV21] Manuela Weigold, Johanna Barzen, Frank Leymann, and Daniel Vietz. Patterns for hybrid quantum algorithms. In Johanna Barzen, editor, *Service-Oriented Computing*, pages 34–51, Cham, 2021. Springer International Publishing.

[ZLW19] Christa Zoufal, Aurélien Lucchi, and Stefan Woerner. Quantum generative adversarial networks for learning and loading random distributions. *npj Quantum Information*, 5(1), nov 2019.

A Study on Expressibility and Entangling Capacity in Parametrized Quantum Circuits

ORIGINALITY REPORT

7%
SIMILARITY INDEX

PRIMARY SOURCES

1	library.isical.ac.in:8080 Internet	419 words — 3%
2	mafiadoc.com Internet	108 words — 1%
3	www.arxiv-vanity.com Internet	99 words — 1%
4	nozdr.ru Internet	61 words — < 1%
5	ipfs.io Internet	43 words — < 1%
6	www.eecs.berkeley.edu Internet	42 words — < 1%
7	Somshubhro Bandyopadhyay. "Origin of noisy states whose teleportation fidelity can be enhanced through dissipation", Physical Review A, 2002 Crossref	39 words — < 1%
8	Gennaro De Luca. "Survey of NISQ Era Hybrid Quantum-Classical Machine Learning Research", Journal of Artificial Intelligence and Technology, 2021 Crossref	30 words — < 1%

Srishti Sur. Kalay 21.07.2022

9	csse.xjtlu.edu.cn Internet	28 words —	< 1%
10	ediss.uni-goettingen.de Internet	27 words —	< 1%
11	Sukin Sim, Peter D. Johnson, Alán Aspuru - Guzik. "Expressibility and Entangling Capability of Parameterized Quantum Circuits for Hybrid Quantum - Classical Algorithms", Advanced Quantum Technologies, 2019 Crossref	24 words —	< 1%
12	hdl.handle.net Internet	24 words —	< 1%
13	Darren M. Chan, Angelique Taylor, Laurel D. Riek. "Faster robot perception using Salient Depth Partitioning", 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017 Crossref	20 words —	< 1%
14	Nicolas Veyrat-Charvillon. "Mutual Information Analysis: How, When and Why?", Lecture Notes in Computer Science, 2009 Crossref	17 words —	< 1%
15	Yaninee Jungjarassub, Krerk Piromsopa. "A Performance Optimization of Quantum Computing Simulation using FPGA", 2022 19th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2022 Crossref	17 words —	< 1%
16	Stig Elkjær Rasmussen, Niels Jakob Søe Loft, Thomas Bækkegaard, Michael Kues, Nikolaj Thomas Zinner. "Reducing the Amount of Single - Qubit	16 words —	< 1%

Srismita Sur. Kolay 21.07.2022

Rotations in VQE and Related Algorithms", Advanced Quantum Technologies, 2020
Crossref

17	csis.pace.edu Internet	16 words — < 1%
18	tesi.cab.unipd.it Internet	15 words — < 1%
19	catalog.uttyler.edu Internet	14 words — < 1%
20	dbkgroup.org Internet	14 words — < 1%
21	repository.iiitd.edu.in Internet	14 words — < 1%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	< 14 WORDS
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	< 14 WORDS

Susmita Sur-Kolay 21.07.2022