A practical use case for Quantum Generative Adversarial Networks in High Energy Physics

Generating top squark events decaying via the four-body mode in single-lepton final states in proton-proton collisions at $\sqrt{s} = 13$ TeV

Diogo Carlos Chasqueira de Bastos

Advanced Topics in Particle and Astroparticle Physics II

Abstract

This is a simple paragraph at the beginning of the document. A brief introduction about the main subject.

Contents

List of Figures

List of Tables

Glossary

CERN QTI CERN Quantum Technology Initiative. 1

CMS Compact Muon Solenoid. 1

HEP High Energy Physics. 1

MC Monte Carlo. 1

NISQ Noisy Intermediate-Scale Quantum. 1

QC Quantum Computing. 1

qGAN quantum Generative Adversarial Network. 1

QML Quantum Machine Learning. 1

1 Introduction

One of the main objectives of CERN Quantum Technology Initiative (CERN QTI) is to investigate if Quantum Computing (QC) can be used in the field of High Energy Physics (HEP) in the Noisy Intermediate-Scale Quantum (NISQ) era. Noisy means that, currently, we have imperfect control over qubits. Intermediate refers to the number of qubits our present quantum computers have. They range from 50 to a few hundred. In order to fully fulfill the promise of quantum computing, the challenges of noise and scalability (millions of qubits) need to be solved. Nonetheless, we can already use the available quantum computers and algorithms to tackle present challenges.

Quantum Machine Learning (QML) is a growing research area that explores the interplay of ideas from QC and machine learning. This project explores the use of quantum Generative Adversarial Networks (qGANs) [5], a QML algorithm, to learn the kinematic distributions of stop four-body decays [4] in Compact Muon Solenoid (CMS) data. The Feynman diagram for such process is represented in Figure 1. In HEP to simulate such distributions, Monte Carlo (MC) simulations are used in a very convoluted and computational resources hungry process that can take up months. As we will see, the method presented in this project could speed up this process significantly when run on quantum computers. This method could also be used for data augmentation of the MC generated samples which would be helpful in the training of the classification algorithms in many HEP searches improving the sensitivity of such searches.

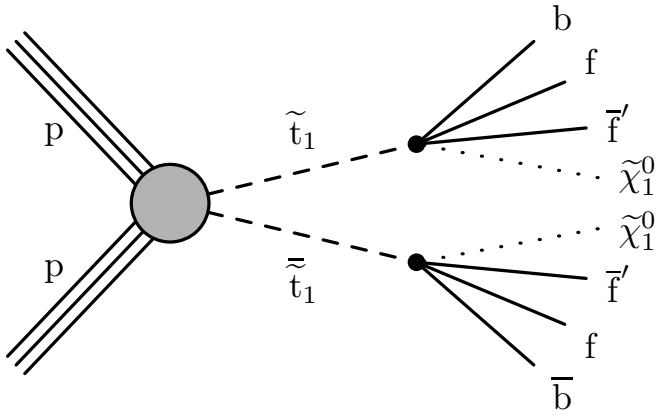

Figure 1: Diagram of top squark pair production $\widetilde{t}_1\bar{\widetilde{t}}_1$ in pp collisions, with a four-body decay of each top squark.

This project will be run on a classical computer (my personal laptop) simulating a quantum computer with 5 qubits using pennylane [3] to train and optimize the quantum algorithm, and cirq [2] for writing, manipulating, and running the quantum simulator. Since QML is a novel technique in HEP, after the present Section 1, a brief introduction to QC and the necessary operations needed for the final algorithm are presented in Section 2. The main development of this project is, as well as its implementation, shown in Section 3. The final results of the project and a brief discussion of futures steps to be built on top of this project are reserved to Section 4.

2 A Brief Introduction to Quantum Computing

2.1 Meet the Qubit

A quantum bit, know as a qubit, is the basic unit of quantum information. The principle behind QC is to take advantage of the properties of quantum mechanics for computations and information processing such as superposition and entanglement. A classic bit can only have a value of 0 or 1, a qubit can be in a superposition of 0 and 1. The Bloch sphere [1] is a geometrical representation of the pure state space of a qubit as is shown in Figure 2. This is a useful representation of a qubit where one can think of the sate of a qubit as a vector inside a 3-D sphere. The north and south poles of the Bloch sphere are typically chosen to correspond to the standard basis vectors $|0\rangle$ and $|1\rangle$ respectively.

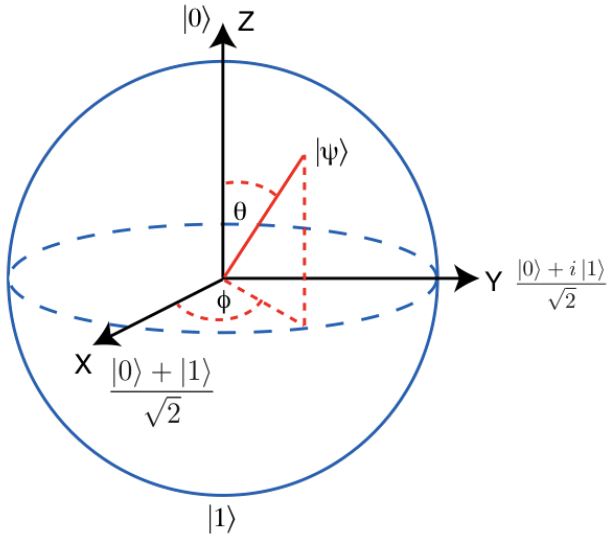

Figure 2: Representation of the qubit in the state $|\psi\rangle$ in the Bloch sphere.

The qubit's 0 and 1 states are mathematically represented as follows:

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

As a consequence of this mathematical description, unlike a bit, a qubit is not limited to being in one of these two states. Qubits can exist in what's known as a superposition of states. A superposition is a linear combination of two basis vectors. Mathematically:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{2}$$

where α and β are complex numbers that satisfy:

$$\alpha\alpha^* + \beta\beta^* = 1 \tag{3}$$

2.2 Quantum Circuits

Quantum circuits are a common means of visually representing the sequence of operations that are performed on qubits during a quantum computation. They consist of a set of operations (gates) applied to a set of qubits (wires). Each wire in the diagram represents a qubit. Circuits are read from left to right, and this is the order in which operations are applied. An example of a quantum circuit is shown in Figure 3.

A quantum computation is all about manipulating qubit states by using gates and seeing the outcome of such manipulations by measuring the qubits.

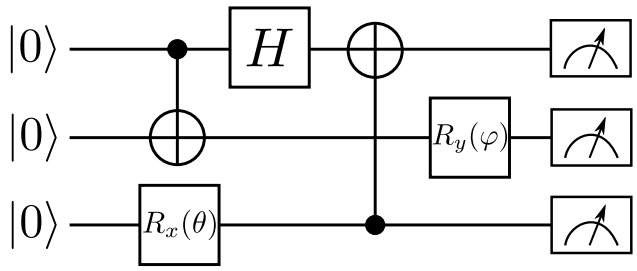

Figure 3: Example of a quantum circuit. This circuit has three qubits that start in the state $|0\rangle$, performs five operations, and measures every qubit at the end of the circuit.

2.3 Flipping qubits

Performing quantum operations involves multiplication by matrices that send valid, normalized quantum states to other normalized quantum states. The simplest quantum operation as the following effect:

$$X|0\rangle = |1\rangle,\ X|1\rangle = |0\rangle \tag{4}$$

Known as bit flip, $Pauli-\mathrm{X}$ or NOT gate, it can be mathematically represented as:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{5}$$

Figure 4 represents the qubit flip gate in a quantum circuit diagram.

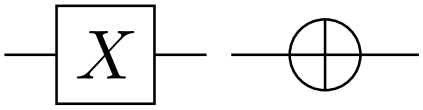

Figure 4: The $Pauli-\mathrm{X}$ or NOT gate in a quantum circuit.

Equation 6 shows the mathematical operation of applying the NOT gate to a qubit in state $|0\rangle$ flipping it to state $|1\rangle$.

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle \tag{6}$$

2.4 Rotations, rotations, rotations: the $R_X(\theta)$, $R_Y(\theta)$, and $R_Z(\theta)$ Gates

One of the most relevant operations to the qubit in the context QML are the $R_X(\theta)$, $R_Y(\theta)$, and $R_Z(\theta)$ gates because they can be used to rotate the qubit state in the Bloch sphere. This means qubits can be manipulated as a function of a parameter, an angle, in order to find an optimum in the quantum algorithm being designed. This concept is analogous to using weights in Neural Networks. In the same manner, the value of the angles used in the rotations are the parameter upon which the QML algorithm is going to be trained.

As it was shown, a qubit can be represented in 3D-space by the Bloch sphere. Using the same representation, $R_X(\theta)$, $R_Y(\theta)$, and $R_Z(\theta)$ are shown in Figure 5.

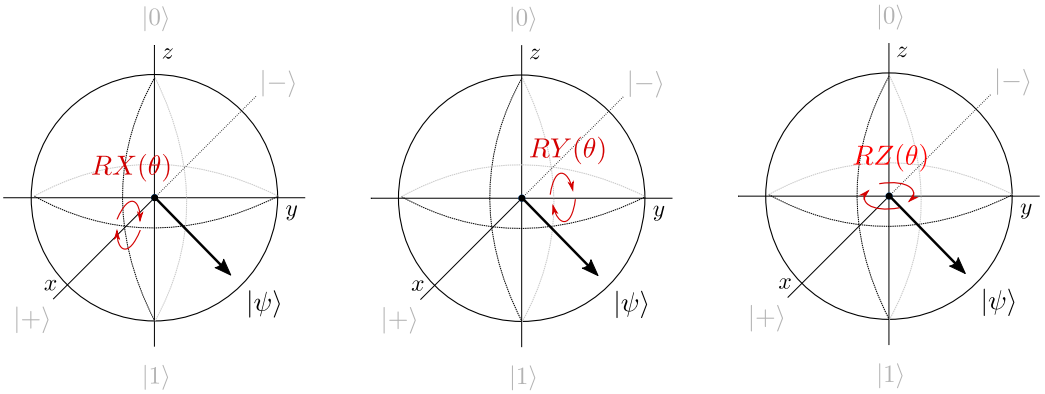

Figure 5: The $R_X(\theta)$, $R_Y(\theta)$, and $R_Z(\theta)$ gates in a the Bloch sphere.

The matrix representation of these rotations:

$$R_X(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -i\sin(\frac{\theta}{2}) \\ -i\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \tag{7}$$

$$R_Y(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \tag{8}$$

$$R_Z(\theta) = \begin{pmatrix} e^{\frac{-i\theta}{2}} & 0 \\ 0 & e^{\frac{i\theta}{2}} \end{pmatrix} \tag{9}$$

Figure 6 represents the $R_X(\theta)$, $R_Y(\theta)$, and $R_Z(\theta)$ gates in a quantum circuit diagram.

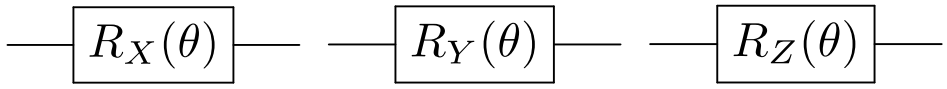

Figure 6: The $R_X(\theta)$, $R_Y(\theta)$, and $R_Z(\theta)$ gates in a quantum circuit diagram.

2.5 Superposition and the *Hadamard* Gate

The Hadamard gate, H, is a fundamental quantum gate that allows to move away from the poles of the Bloch sphere, and create a superposition of $|0\rangle$ and $|1\rangle$ as follows:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{10}$$

It is mathematically represented as:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{11}$$

Figure 7 represents the *Hadamard* gate in a quantum circuit diagram.

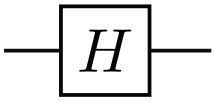

Figure 7: The H gate in a quantum circuit diagram.

Equation 12 shows the mathematical operation of applying the H gate to a qubit in state $|0\rangle$ mapping it to a state in a superposition, commonly written as $|+\rangle$.

$$H|0\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\ 1\end{bmatrix} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \quad (12)$$

2.6 Multiple Qubit States

To describe a state of 2 qubits, 4 complex amplitudes are required:

$$|\psi\rangle = \psi_{00}|00\rangle + \psi_{01}|01\rangle + \psi_{10}|10\rangle + \psi_{11}|11\rangle = \begin{bmatrix}\psi_{00}\\ \psi_{01}\\ \psi_{10}\\ \psi_{11}\end{bmatrix} \quad (13)$$

Such that the normalization condition is respected:

$$|\psi_{00}|^2 + |\psi_{01}|^2 + |\psi_{10}|^2 + |\psi_{11}|^2 = 1 \quad (14)$$

Qubits $|a\rangle$ and $|b\rangle$ are two separate qubits such that:

$$|a\rangle = \begin{bmatrix}a_0\\ a_1\end{bmatrix}, |b\rangle = \begin{bmatrix}b_0\\ b_1\end{bmatrix} \quad (15)$$

The kronecker product is used to describe the collective state. The collective state of qubits $|a\rangle$ and $|b\rangle$ is described as follows:

$$|ba\rangle = |b\rangle \otimes |a\rangle = \begin{bmatrix} b_0 \times \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \\ b_1 \times \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} b_0 a_0 \\ b_0 a_1 \\ b_1 a_0 \\ b_1 a_1 \end{bmatrix} \tag{16}$$

$$|ba\rangle = b_0 a_0 |00\rangle + b_0 a_1 |01\rangle + b_1 a_0 |10\rangle + b_1 a_1 |11\rangle \tag{17}$$

For a computation with n qubits, a computer needs to keep track of 2^n complex amplitudes. This is the reason why quantum computers are difficult to simulate. A modern laptop can simulate a general quantum state of around 10 qubits, but simulating 100 qubits is too difficult even for the largest supercomputers.

2.7 Entanglement and the $CNOT$ Gate

Until now, only gates that act on a single qubit have been described. In order to complete all the "ingredients" necessary, gates that act on multiple qubits are needed in order to entangle multiple qubits.

When 2 qubits are entangled, by knowing the state of one qubit, the state of other becomes known as well. The $CNOT$ operation changes the state of a target qubit depending on the state of a control qubit. When these two concepts are merged, qubits can be entangled in quantum computers.

By definition, a state is entangled if it cannot be described as a tensor product of individual qubit states. An entangled state can only be described by specifying the full state. One example of an entangled state:

$$|ba\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{18}$$

In order to describe this state using the kronecker tensor one needs to find a_0, a_1, b_0 and b_0 such that:

$$b_0a_0 = 1, b_0a_1 = 0, b_1a_0 = 0, b_1a_1 = 1 \tag{19}$$

Each variable appears in two equations, one of which is equal to 1, and the other to 0. But for any of the 0 ones to be true, at least one variable needs to be 0, and that would immediately contradict the other equation. Therefore, there is no solution, it's not possible to describe this state as two separate qubits. They are entangled.

The $controlled-$NOT or $CNOT$ gate can make an entangled state by performing the $Pauli-$X operation on one target qubit depending on the state of another control qubit. Mathematically represented as:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{20}$$

Figure 8 represents the $CNOT$ gate in a quantum circuit diagram.

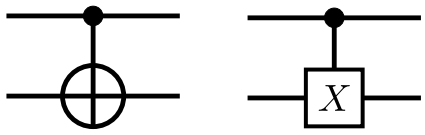

Figure 8: The $controlled-$NOT or $CNOT$ gate in a quantum circuit.

Qubits $|q_0\rangle$ and $|q_1\rangle$ are two separate qubits such that:

$$|q_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{21}$$

$$|q_1\rangle = |0\rangle \tag{22}$$

The collective state is defined as $|q_0 q_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle)$ which is not entangled. Now, if the gate $CNOT$ is applied to the collective state as in Equation 23, an entangled state is achieved.

$$CNOT|q_0 q_1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{23}$$

$$CNOT|q_0 q_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{24}$$

2.8 qGANS

QGANs are fundamentally composed of two Quantum Neural Networks (QNNs) - a generator and a discriminator one - that compete between each other. The generative network generates candidates while the discriminative network evaluates them. The contest operates in terms of data distributions. Typically, the generative network learns to map from a latent space to a data distribution of interest, while the discriminative network distinguishes candidates produced by the generator from the true data distribution. The generative network's training objective is to increase the error rate of the discriminative network (i.e., "fool" the discriminator network by producing novel candidates that the discriminator thinks are not synthesized (are part of the true data distribution)).

3 DEMO

asd

```
def f(x):
    return x
```

4 Results

asd

References

[1] F. Bloch. "Nuclear Induction". In: *Phys. Rev.* 70 (7-8 Oct. 1946), pp. 460–474. DOI: 10.1103/PhysRev.70.460. URL: https://link.aps.org/doi/10.1103/PhysRev.70.460.

[2] *Cirq*. 2022. URL: https://quantumai.google/cirq.

[3] *Pennylane*. 2022. URL: https://pennylane.ai/.

[4] *Searching for top squarks with CMS data*. 2022. URL: https://cms.cern/news/searching-top-squarks-cms-data.

[5] Christa Zoufal, Aurélien Lucchi, and Stefan Woerner. "Quantum Generative Adversarial Networks for learning and loading random distributions". In: *npj Quantum Information* 5.1 (Nov. 2019). DOI: 10.1038/s41534-019-0223-2. URL: https://doi.org/10.1038%2Fs41534-019-0223-2.