Optimizing Quantum Graph Neural Networks for the Particle Tracking Problem

IBM & CERN Project Proposal

Start Date: 21/02/2022

Team Name:

The Superpositioned States of America

Abstract:

The CERN Quantum Technology Initiative declared five quantum computing and algorithms projects of interest to the HEP community. One of these projects is particle tracking reconstruction. Particle tracking reconstruction is a fundamental research aspect for particle collider experiments, which aim to validate the elementary particle theories through the observation of particle trajectories after collisions. With the upgrade of the Large Hadron Collider (LHC) to High Luminosity (HL-LHC) at CERN in the near future (circa 2027), the rate of collisions will be further ramped up, yielding many more detector hits and a larger scale of data. Therefore, being able to efficiently identify the particle trajectories from data of a large scale and increased complexity may be the key to unveil the nature of particle physics. With the development of quantum computers on its own growth curve, untangling the information from big data via quantum computing appears to be a promising solution. Our work focuses on supporting an existing approach, Quantum Graph Neural Networks (QGNNs), to solve the particle tracking reconstruction challenge. Specifically, we are looking to focus on the detailed analysis of the vanishing gradient problem, long training times, and how robust the overall approach is to noise from real quantum computers, which have been mentioned but not addressed yet in prior work. Our work aims to improve the viability of the QGNN method towards particle tracking problems.

1 Overview

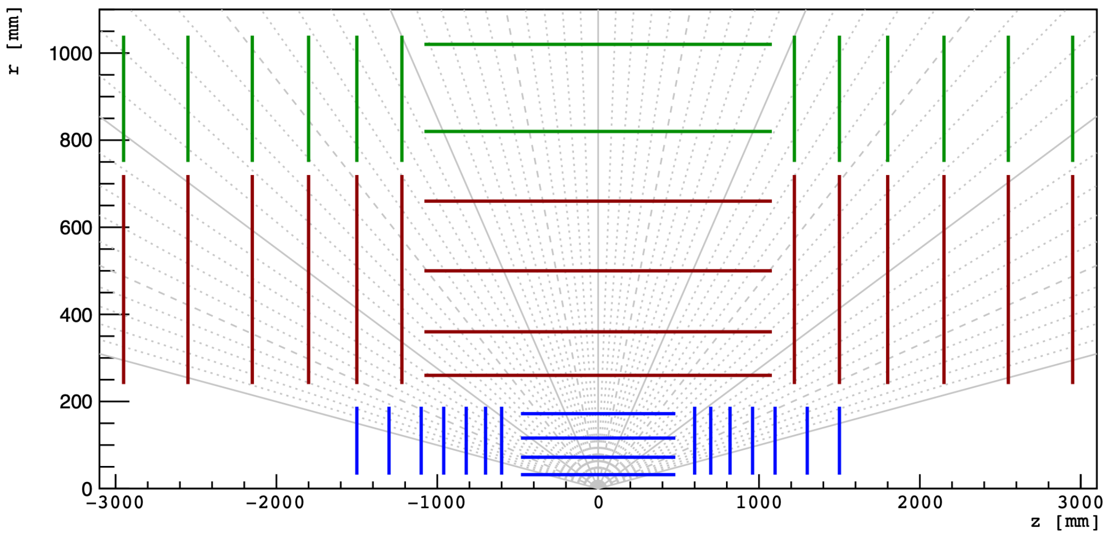

Figure 1: A scheme of the full detector layout for the virtual TrackML detector. Source: [1]

1.1 Quantum graph neural networks for particle tracking

To understand the structure of subatomic particles and the laws of nature governing them, particle physicists use colliders as a research tool to accelerate particles to very high speeds and energies in order to cause collisions to happen. In each collision, particles are scattered in every direction. The detectors, also known as "trackers," that are spread around the device can measure particles properties (such as momentum, time...). When the particles pass through the detectors, "hits" are registered without loss of energy to the particle. The challenge comes in reconstructing the particle trajectories in 3D from the detector data. Particularly, the Large Hadron Collider (LHC) [2] at CERN is planned to be completely upgraded to the High Luminosity Large Hadron Collider (HL-LHC) by the end of 2027 [3]. The significantly rising rate of collisions coming from the HL-LHC will not only bring more high-energy physics to explore, but also subsequent challenges to reconstruct the particle tracking with much larger amounts of data. Accordingly, the search for more efficient algorithms to tackle the particle track reconstruction becomes urgent and critical. CERN even hosted a Kaggle challenge named "TrackML" [5] in 2018, intending to encourage

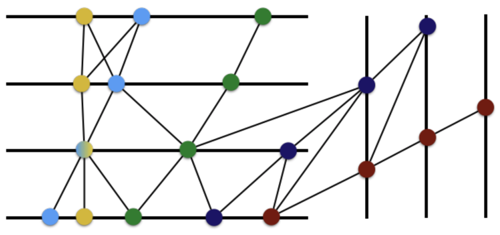

Figure 2: A graph representation of track hit data. Source: [4]

people of different academic/professional backgrounds to try and solve this particle tracking reconstruction problem. The TrackML Challenge dataset contains 10000 events to simulate the HL-LHC conditions [1]. Each event has $O(10^5)$ space-point hits, corresponding to $O(10^4)$ particles. With such an exponentially large amount of data, it is conceivable to seek out solutions that may exist from Maching Learning (ML) or

Deep Learning (DL) approaches. The idea to implement Quantum Graph Neural Networks (QGNNs) for the particle tracking problem is as follows: Firstly, in the TrackML detector layout (Fig. 1), the geometry of the detected "hits" distributed in real space is layer-by-layer (Fig. 2). The hits can be the "nodes" while the traveling path between hits (from one layer to the next layer) can be the "edges" if we borrow the language from Graph Neural Networks (GNNs). Therefore, applying GNNs to the measured particle data appears to be a promising way to reconstruct the particle trajectories [4]. Secondly, with the rapid development and growth of quantum computing (QC), a fertile area of research is Quantum Machine Learning (QML) where qubits can be used directly in a neural network. By replacing each node of the GNN structure as a qubit and constructing the corresponding quantum circuits, we can combine both DL and QC into the QGNN approach to attack the particle tracking reconstruction problem.

In prior work [6], the authors mentioned problems with training times and vanishing gradients. We hope during this short period of time to look into these issues as well as attempting to benchmark these circuits on actual quantum hardware instead of the classical simulators that the authors used.

1.2 Vanishing gradient problem

A known issue of recurrent neural networks is the vanishing gradient problem and the quantum graph neural network used in this work is no exception. Training an RNN requires using back-propagation through time (BPTT), which means that you choose a number of time steps N, and you unroll your network so that it becomes a feed-forward neural network (FFNN) with N duplicates of the original network.

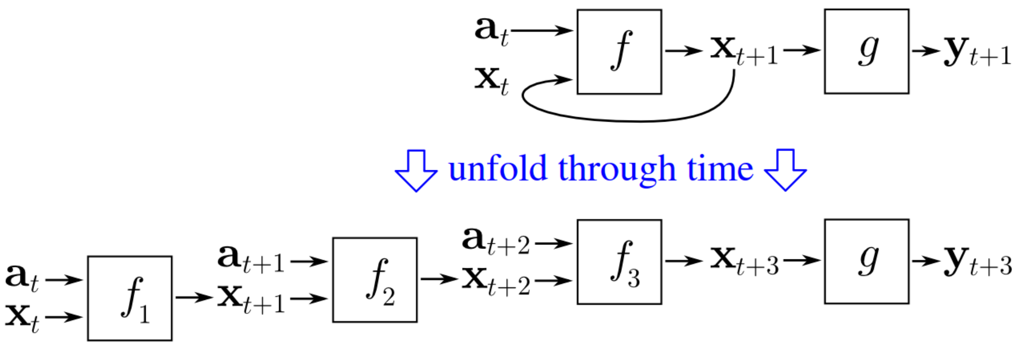

Figure 3: An unrolled recurrent neural network. Source: [7]

So training an RNN with BPTT is equivalent to unrolling the RNN into a deep FFNN and proceeding to calculate the gradients as you normally would with a FFNN (backpropagation). However, we soon face difficulties with this approach. It becomes unwieldy to compute the gradients of the early layers of the unrolled RNN as N scales. This is due to the nature of BPTT, where the early layer's gradients are the products of all succeeding terms and thus are prone to vanishing or exploding gradients.

The quantum graph neural network proposed in existing literature uses a similar recurrent architecture where each layer/step of message passing can be thought of as a time step, and thus faces the vanishing gradient problem as the number of cycles of message passing increase. Our project aims to provide a solution to the vanishing gradient problem faced by quantum graph neural networks. In the following section, we consider how we draw from the progress of RNNs in classical machine learning to introduce a novel solution to help with the training of QGNNs.

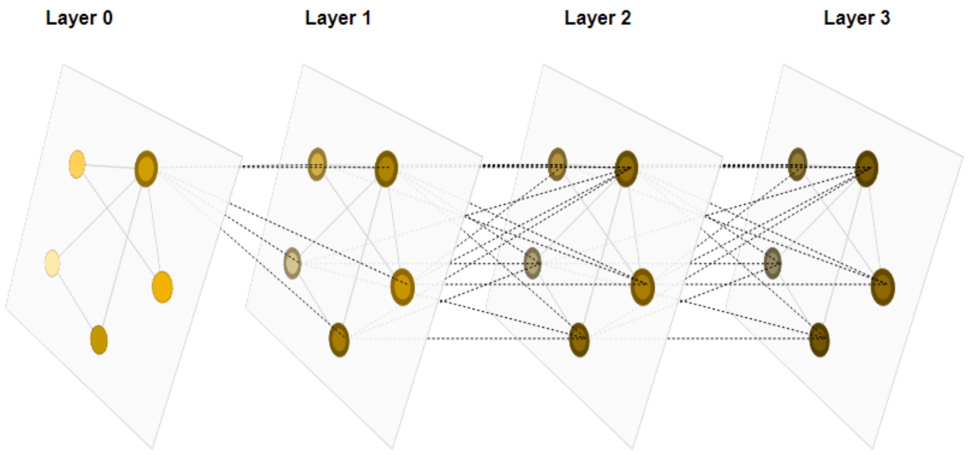

Figure 4: The information of the highlighted initial node propagates through the graph network with each layer of message passing (propagation). Source: [8]

2 Proposal to solve the vanishing gradient problem within QGNNs

The QGNN model we will use to sort out the reconstructed particle paths is summarized in Fig. 5. Since we aim to solve the vanishing gradient problem as an extension of the earlier work [6] without changing the main structure of the QGNN model, we borrow their figure of the QGNN model. The input would be the detected data as a graph in spatial space, which has three dimensions. The model starts with a single-layer Neural Network (NN) as an input layer to increase the dimension of data first. Then a Quantum Edge Network (QEN) is applied to all the edges of the input layer before outputting an edge feature that is used to update the initial edge values. Then the Quantum Node Network (QNoN) is applied to all the nodes of the updated graph (i.e., the one with updated edge values) before the output updates the nodes values in the hidden layers. The quantum circuits of both QEN and QNoN are shown in Fig. 6. The alternate implementations between QEN and QNoN in the hidden layers will propagate the data information from closer detector layers to farther layers. In the end, a QEN is applied to produce the segment classification. Our draft source code for this project is here.

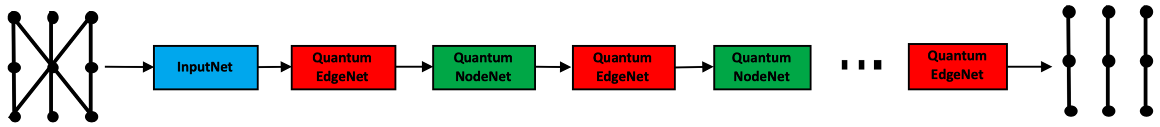

Figure 5: The Quantum Graph Neural Network model used in this work. Source: [6]

There is a body of research for in classical ML for the vanishing gradient problem. Additionally, there is research re: “barren plateaus” in QML. We will be surveying the effectiveness of these techniques in the QGNN model.

Other than utilizing algorithmic techniques, the effect from data pre-processing can also affect the vanishing gradient issue. Since the vanishing gradient problem arises from the accumulated effects of multiple layers as illustrated in Sec. 1.2, keeping less layers might just scale down the vanishing gradient problem. The vanishing gradient issue might just artificially vanish, which is not our original intention. Therefore, we have to stay cautious towards the vanishing gradient problem when we pre-process data.

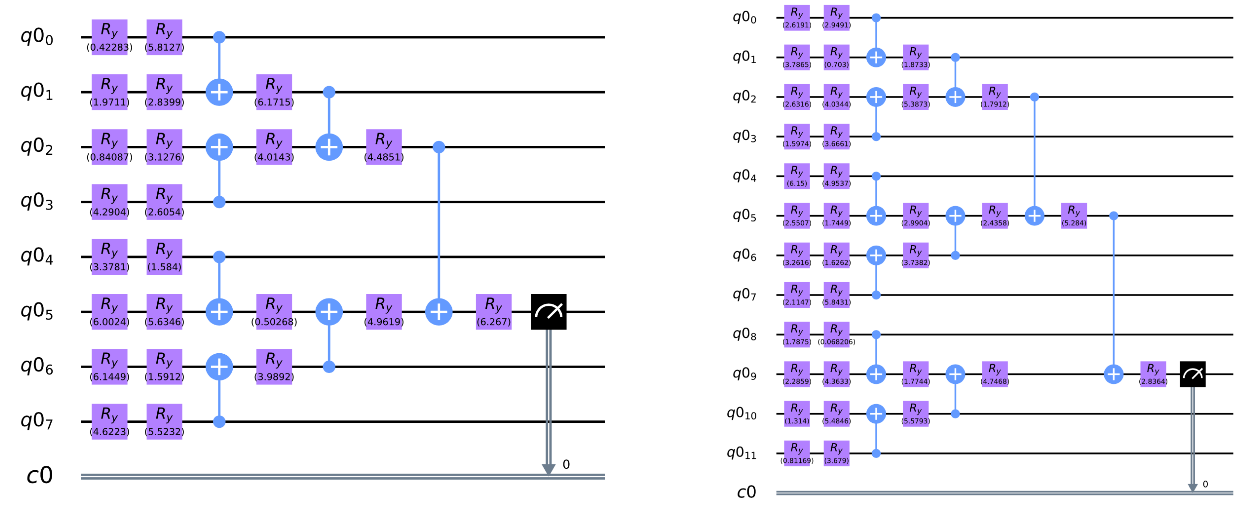

Figure 6: Quantum Circuit of QEN is on the left. Quantum Circuit of QNoN is on the right. The numerical values are from an example data. Source: [6]

3 Proposal to improve training times

In the original QGNN paper [6], the authors mention that training time was on the order of a week even with a reduced dataset. For this project, we unfortunately don't have a full week and are hoping to test multiple iterations, so it will be necessary to find a set of criteria to decrease the amount of data we process, all-the-while keeping important connections and original distributions. If successful, then the results we obtain should offer insight into how an *even smaller* dataset might extrapolate to the overall, larger dataset.

3.1 Data Preprocessing

Some of the ideas to make the graph more sparse and decrease the computational load:

1. Limit the hits to the initial "horizontal barrels", as referred to in the literature, shown in blue in Fig. 1.
2. Use the ground-truth data to eliminate whole trajectories of hits in order to make the graph more sparse.
3. Use less events.

Also, the HepTrkX team set a selection criteria for pre-processing TrackML dataset [4], which are parameters we could also tweak. We will likely figure out and settle on one set of criteria after we experiment with the data.

3.2 Quantum execution

Separately, while Qulacs is mentioned as the simulator of choice it would be interesting to test training on other high-performance quantum computing simulators and/or tweak the parameters of these simulations. In prior experience, reducing double floating-point precision to single floating-point precision has impacted run-times considerably. Proper profiling of the training pipeline might reveal bottlenecks that can be alleviated.

Finally, it is worth considering using actual quantum hardware in addition to or in place of classical simulations. At the very least it will be interesting to see how robust the QGNN algorithms are to noise.

4 Required resources

Currently, our team has secured initial power-ups in the form of Amazon Braket credits and access to a 7-qubit IBM quantum machine. Luckily, the original QEN circuit used in the QGNN paper [6] utilizes 7 qubits (Fig. 6). However the QNoN circuit uses 11 qubits, which is why **having access to IBM's 16-qubit machine would be helpful for benchmarking purposes**.

5 Summary

In summary, the CERN Quantum Technology Initiative declared five quantum computing and algorithms projects of interest to the HEP community, which can be of benefit to humanity. The ambitions of CERN and the HEP community are large and it is reasonable that quantum computing has been folded into that large ambition. We have chosen to focus our efforts on assisting one of these projects, namely particle tracking reconstruction via QGNNs, in the hopes that we can make some small contributions back to the researchers of the project.

References

[1] S. Amrouche et al., *The Tracking Machine Learning challenge : Accuracy phase*, arXiv:1904.06778 [hep-ex, physics:physics] (2020), pp. 231–264. http://arxiv.org/abs/1904.06778. arXiv:1904.06778, doi:10.1007/978-3-030-29135-8_9.

[2] *The Large Hadron Collider*. https://home.cern/science/accelerators/large-hadron-collider.

[3] *High-Luminosity LHC*. https://home.cern/science/accelerators/high-luminosity-lhc.

[4] S. Farrell et al., *Novel deep learning methods for track reconstruction*, arXiv:1810.06111 [hep-ex, physics:physics] (2018). http://arxiv.org/abs/1810.06111. arXiv:1810.06111.

[5] *TrackML particle tracking challenge*. https://sites.google.com/site/trackmlparticle/.

[6] C. Tüysüz et al., *A Quantum Graph Neural Network Approach to Particle Track Reconstruction*, arXiv:2007.06868 [quant-ph] (2020). http://arxiv.org/abs/2007.06868. arXiv:2007.06868, doi:10.5281/zenodo.4088474.

[7] *Backpropagation through time*, Wikipedia (2021). https://en.wikipedia.org/w/index.php?title=Backpropagation_through_time&oldid=1051436108.

[8] B. Sanchez-Lengeling, E. Reif, A. Pearce and A. B. Wiltschko, *A Gentle Introduction to Graph Neural Networks*, Distill **6** (2021), p. e33. https://distill.pub/2021/gnn-intro. doi:10.23915/distill.00033.