Particle Track Reconstruction with Quantum Algorithms

F. Carminati[2], B. Demirköz[1], D. Dobos[3], F. Fracas[2], K. Novotny,[3] K. Potamianos[3,4], S. Vallecorsa[2], C. Tüysüz[1], J.R. Vlimant[5]

[1]Middle East Technical University, Ankara, Turkey
[2]CERN, Geneva, Switzerland
[3]gluoNNet, Geneva, Switzerland
[4]DESY, Hamburg, Germany
[5]California Institute of Technology, Pasadena, California, USA

4-8 November 2019
Adelaide, South Australia

Abstract

Accurate particle track reconstruction will be a major challenge for the High Luminosity Large Hadron Collider (HL-LHC) experiments. Increase in the expected number of simultaneous collisions and the high detector occupancy will make these algorithms extremely demanding in terms of time and computing resources. The sheer increase in the number of hits would increase the complexity exponentially, however the finite resolution of the detector and the physical "closeness" of the hits increase the ambiguity of their assignment to a trajectory, making the track reconstruction problem a major obstacle to the correct interpretation of the HL-LHC data. Most methods currently in use are based on the Kalman filter: they are shown to be robust and to provide good physics performance however, they are expected to scale worse than quadratically. Designing an algorithm capable of reducing the combinatorial background at the hit level, would provide a much "cleaner" initial seed to the Kalman filter, strongly reducing the total processing time. One of the salient features of Quantum Computers is the ability to evaluate a very large number of states simultaneously, making them an ideal instrument for searches in a large parameter space. In fact, different R&D initiatives are exploring how Quantum Tracking Algorithms[5;8;9] could leverage such capabilities. In this paper, we present our work on the implementation of a quantum-based track finding algorithm aimed at reducing combinatorial background during the initial seeding stage. We use the publicly available dataset designed for the kaggle TrackML challenge[3].

Introduction

Latest developments in **Quantum Computing showed exceptional speed-ups** for certain problems[4]. The speed-up provided by **Quantum Algorithms may play an important role in the future of track reconstruction** in particle physics experiments. In this work, we present an **exploratory look at the HepTrkX[2] project from a Quantum Computing perspective** to evaluate the capabilities of Quantum Computing along with Deep Learning algorithms[6].

The HepTrkX team proposed a **Graph Neural Network implementation for particle track reconstruction** that uses the kaggle TrackML challenge dataset. The aim here is to **provide results similar to HepTrkX** without changing the main structure of the network.

When a particle passes through a tracker plane, it creates a signal which is named a "hit". The dataset contains precise locations of these hits and their particle IDs as the ground truth. The challenge is to **associate hits that are belonging to the same initial particle/track.**

Classical Method

Hep.TrkX GNN consists of cascaded Input, Edge and Node Networks. The structure is presented in Figure 2.

- Input Network: encodes the hit features into node features.
- Edge Network (EdgeNet): outputs edge features.
- Node Network (NodeNet): outputs node features.

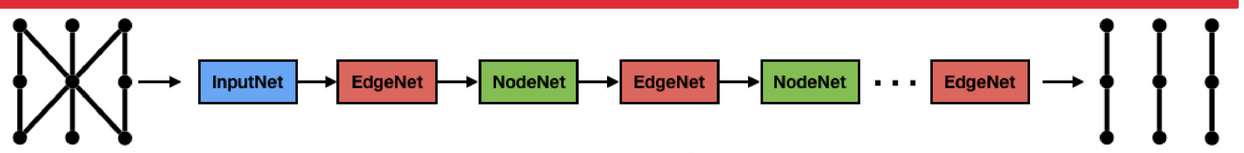

Figure 2: Network structure of HepTrkX[6] in the classical method.

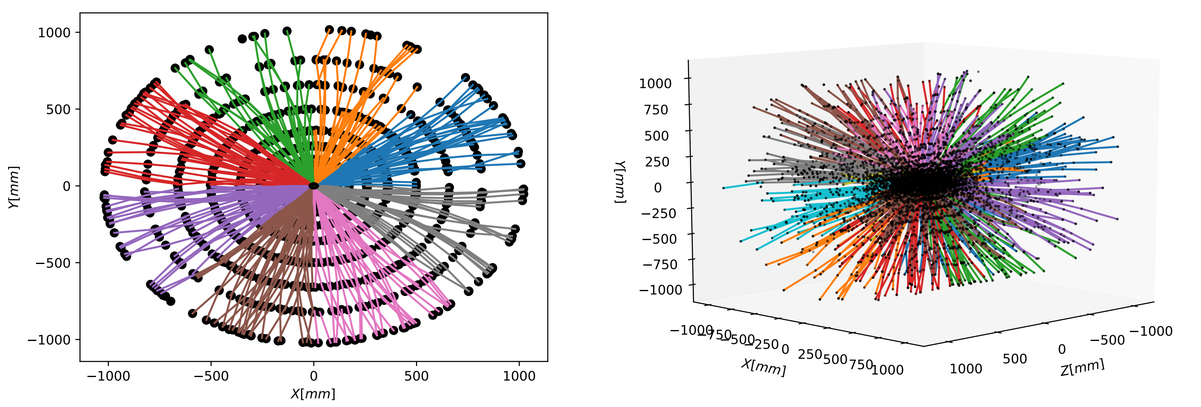

Figure 1: Cartesian projection of a single event. HepTrkX divides the graph of an event into 8 sections in the ϕ direction and into 2 sections in the η directions adding up to 16 subgraphs per event. Each subgraph is plotted with a different color.

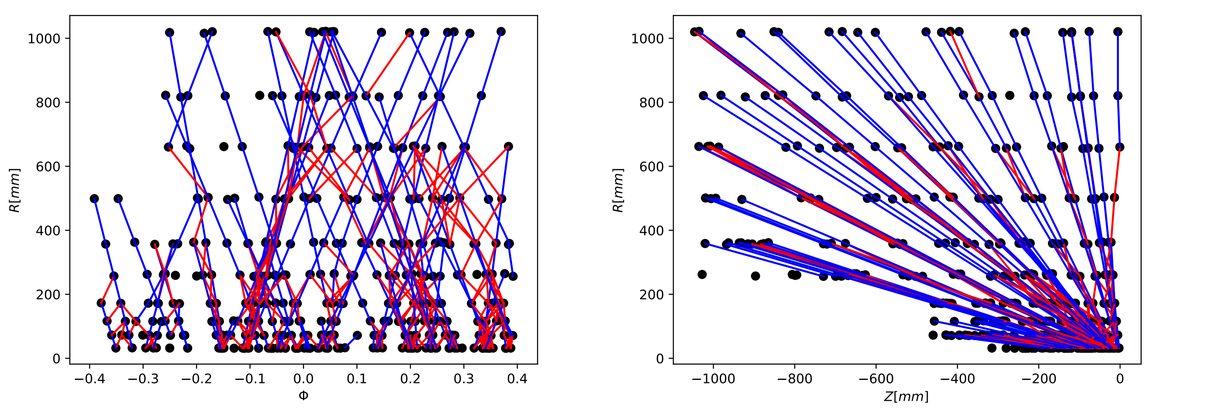

Figure 3: Projection of a subgraph in Cylindrical Coordinates. Blue lines show the ground truth edges and red lines show the fake edges created in the preprocessing phase.

Switching to Quantum Computing

We want to convert HepTrkX to a Quantum Circuit. Options are as follows;

- **Quantum** EdgeNet + Classical Node Net: constant input size (easiest implementation)
- Classical EdgeNet + **Quantum** Node Net: changing input size depending on input
- **Quantum** EdgeNet + **Quantum** Node Net: changing input size + longer circuit

In this work, the Quantum Edge Network is evaluated by itself as in Figure 4, deviating from the HepTrkX structure turning the network into a **binary classifier acting on each edge independent of each other.**

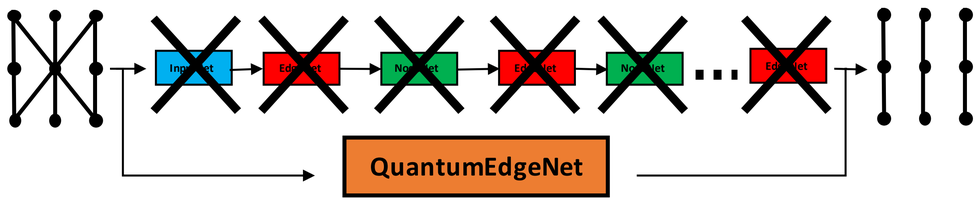

Figure 4: Evaluation of the Quantum Edge Network.

The network starts with input encoding by applying a set of unitary rotations to **map the input to** $[0, 2\pi]$. Then, it applies the Tree Tensor Network (TTN)[7] which consists of unitary Ry rotations and CNOT gates using the higher dimension Hilbert Space. The output is a measurement from a single qubit. **The TTN has 11 parameters** which are the angles of rotations in Y direction on the Bloch Sphere. These **parameters are optimized using stochastic gradient** descent and weighted binary cross entropy error.

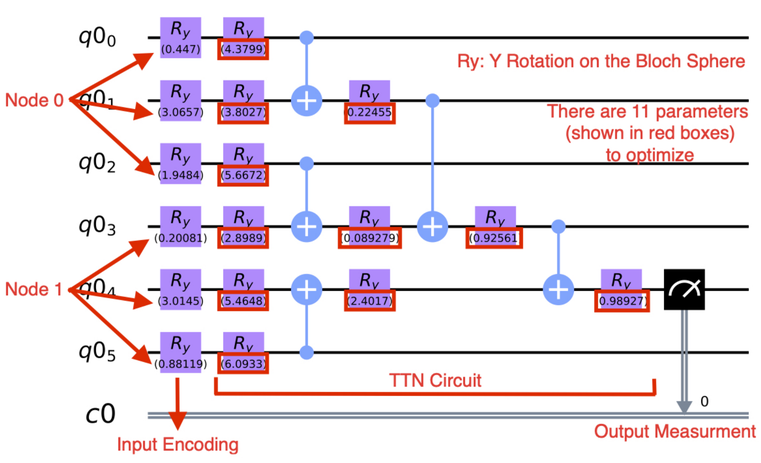

Figure 5: 6 Qubit TTN representation of the Quantum Edge Network.

A detailed analysis on gradient calculations are done. The **gradients efficiently calculated** by evaluating the same circuit at $\pm\pi/2$ angles of the parameter of interest. The amount of sampling is crucial in this method and its effect can be seen in Figure 6 where we only change the sampling amount which is called "shots". The network is trained with **1440 subgraphs produced from TrackML dataset**, which is approximately 1% of the dataset. The results are presented in Figure 7. The increase of accuracy shows that the numerical approach to gradient taking is successful. Although it seems that the accuracy is low, this is **due to oversimplification of the problem** and is not a result of TTN being adapted.

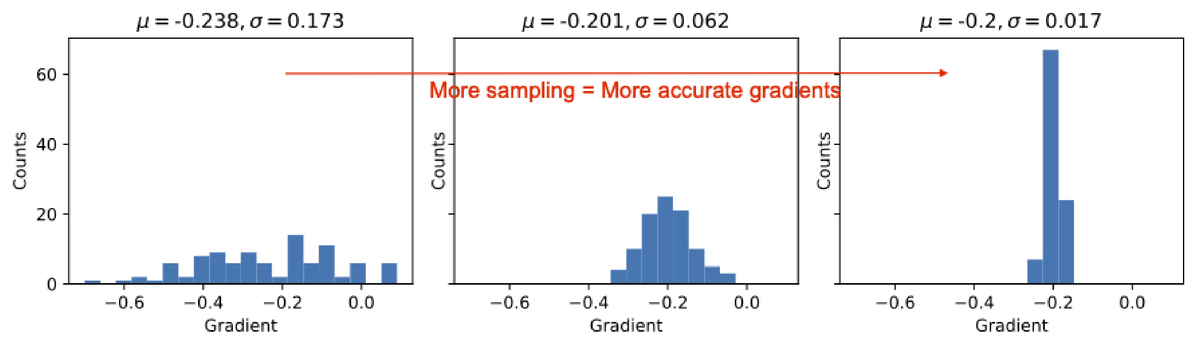

Figure 6: Gradients of the same circuit for a single parameter evaluated at different shots 100 times each. From left to right shots: 10,100,1000.

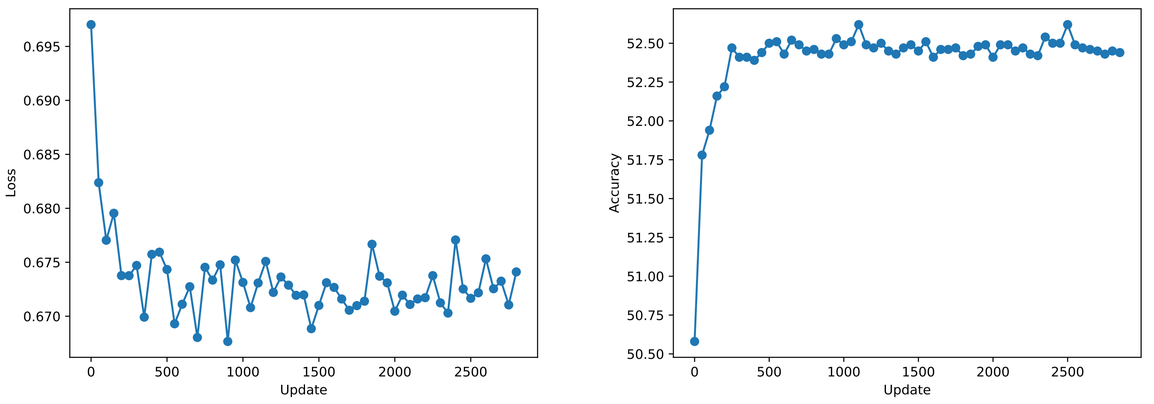

Figure 7: Training Loss (on the left) and Validation Accuracy (on the right) of the Quantum TTN in 2 full epochs. (1 epoch = 1440 updates)

On-going Work

This work presents a **first attempt at appyling Quantum Computing to Graph Neural Networks** for particle track reconstruction. The convergence to a low accuracy is expected as the model is oversimplified. HepTrkX uses hidden layers with high dimensions which is essential in Deep Learning. Therefore, to improve the performance, next steps **include merging the TTN with the complete HepTrkX GNN** and using **higher dimensions in the Hilbert Space.**

Future work will benefit from pennylane[1], which is a python library for quantum machine learning, automatic differentiation, and optimization of hybrid quantum-classical computations.

References

[1] https://github.com/XanaduAI/pennylane.
[2] https://heptrkx.github.io/.
[3] S. Amrouche et al. The Tracking Machine Learning challenge : Accuracy phase. 2019.
[4] F. Arute et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(July), 2019.
[5] F. Bapst et al. A pattern recognition algorithm for quantum annealers. http://arxiv.org/abs/1902.08324. 2019.
[6] S. Farrell et al. Novel deep learning methods for track reconstruction. http://arxiv.org/abs/1810.06111. 2018.
[7] E. Grant et al. Hierarchical quantum classifiers. *npj Quantum Information*, 4(1):17–19, 2018.
[8] I. Shapoval et al. Quantum Associative Memory in HEP Track Pattern Recognition. http://arxiv.org/abs/1902.00498. 1, 2019.
[9] A. Zlokapa et al. Charged particle tracking with quantum annealing-inspired optimization. http://arxiv.org/abs/1908.04475. 2019.

Part of this work was conducted at "*iBanks*", the AI GPU cluster at Caltech. We acknowledge NVIDIA, SuperMicro and the Kavli Foundation for their support of "*iBanks*".

This work is partially supported by Turkish Atomic Energy Agenecy (TAEK).

Contact Information: ctuysuz@cern.ch

Code Available @ github.com/cnktysz/HEPTrkX-quantum