Evaluation of Scalable Quantum and Classical Machine Learning for Particle Tracking Classification In Nuclear Physics

Polykarpos Thomadakis, Emmanuel Billias, and Nikos Chrisochoides

ebill007@odu.edu

Introduction

Future particle accelerators will exceed by far the current data size (10^{15}) per experiment, and high-luminosity program(s) will produce more than 300 times as much data. Classical Machine Learning (ML) likely will benefit from new tools based on quantum computing. Particle track reconstruction is the most computationally intensive process in nuclear physics experiments. A combinatorial approach exhaustively tests track measurements ("hits"), represented as images, to identify those that form an actual particle trajectory, which is then used to reconstruct track parameters necessary for the physics experiment. Quantum Machine Learning (QML) could improve this process in multiple ways, including denoising the data, classifying candidate tracks, and reconstructing the track parameters without conventional processing.

We will present our contributions to the candidate track classification problem using a quantum convolutional network, for Noisy Intermediate-Scale Quantum (NISQ) computers: (1) an artifact-free image decomposition of the particle images into sub-images to cope with the 5% fidelity of a 12-qubit circuit required for the encoding of the full image (currently used in classical ML); (2) improve fidelity by 70% utilizing Distributed NISQ (D-NSIQ) model that utilizes 128 cores to run our simulations on 5-qubit Quantum Processor Units with different and realistic noise models; (3) evaluation of D-NISQ QML in terms of accuracy against the 99% precision of an optimized classical convolutional network we published recently.

Overview

The CEBAF Large Acceptance Spectrometer at 12 GeV (CLAS12) is located at Hall B, one of the experimental halls at the Jefferson Lab in Newport News, VA, serving a variety of physics experiments with different running conditions. The forward part of CLAS12 is built around a superconducting toroidal magnet (Figure 1). The six coils of the toroid divide the detector azimuthally into six sectors. Each sector contains three multi-layer drift chamber (DC) for reconstructing the trajectories of charged particles originating from a fixed target. One sector is composed of three drift chambers (called "regions"), each consisting of two sections (called "super-layers").

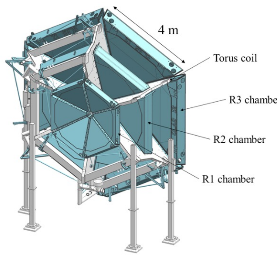

Fig. 1: View of CLAS12 detector showing Drift Chambers and Toroidal Magnet

- The Drift Chambers are used for tracking charged particles. Each super-layer has six layers of wires (12 wire planes in each "region") and each layer of wires has 112 hexagonal cells.
- Figure 2 presents some example events shown for one sector at a time.
- The efficiency of track reconstruction relies on cleanly identifying segments (clusters) in each super-layer. With increased noise, which arises from running with high beam intensity, removal of noise hits with conventional algorithms becomes less efficient, and with loss of segment the tracking efficiency suffers.
- With this work, we study how Convolutional Auto-Encoders can improve the segment finding algorithm by removing uncorrelated hits from noisy raw data.

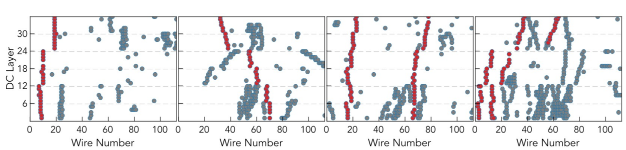

Fig. 2: Example of Drift Chamber data with wire hits (gray circles) and hits identified as belonging to a track by tracking algorithm (red circles). Each plot presents data from one sector from different events. Cases with one and two tricks in one sector are shown.

Methodology

- CNNs are a type of neural network known to perform better on data where spatial locality is important.
- The convolutional layer reads an N-dimensional array as an initial input to the network or produced by a previous layer. A kernel is a 2-dimensional array of weights, usually 3x3, that is used to apply convolution on the input.
- The pooling layer reduces the dimensionality of the convolutional output
- The weights of the kernel are the parameters learned by the neural network and are adjusted between iterations through the process of backpropagation and gradient decent.

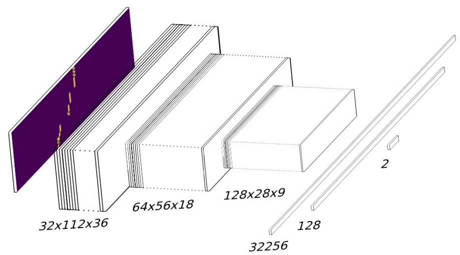

Fig. 3: Architecture of the CNN network including the feature maps of each convolutional layer and fully connected layer. The number of neurons per layer is also included.

We also explore the use of an analogous quantum circuit, the Quantum Convolutional Neural Network (QCNN) shown in Figure 4.

- First, a feature mapping of our image is encoded as a quantum circuit.
- The QCNN pooling layer reduces the dimensionality by discarding dependent qubits between layers.
- Each layer of the QCNN is parameterized meaning the circuit output can be attenuated towards a minimized loss function by classical computational means.

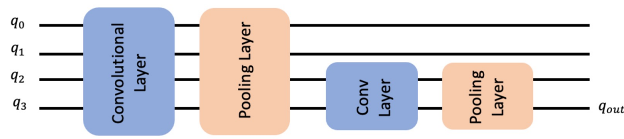

Fig. 4: Four qubit example of a QCNN architecture. Note that after the first pooling layer, qubits q_0 and q_1 are "discarded" and are no longer part of the QCNN reducing the input dimensionality of the next layer. The final output is reliant on a single qubit existing in one of two states. Image taken from IBM's Qiskit.

- Each layer can be constructed as a series of two qubit unitary gates $U = (A_0 \otimes A_1)N(\alpha,\beta,\gamma)(A_2 \otimes A_3)$
- $N(\alpha,\beta,\gamma) \in SU(4)$ constructed with three parameterized rotations
- $A_j \in SU(2)$ which gives $U(4)$ a total of 15 parameters to span the entire Hilbert space to find the optimal QCNN solution
- The required number of parameters required will most likely rely on the encoding method chosen

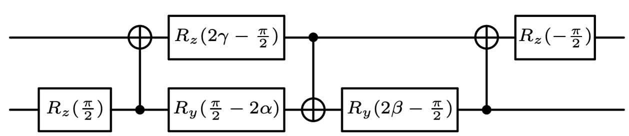

Fig. 5: Circuit implementation of $N(\alpha,\beta,\gamma)$ comprised of 5 single qubit rotations, 3 of which are parameterized, with 3 CNOT gates. This two qubit unitary covers a subspace of the full $SU(4)$. Vatan, Farrokh, and Colin Williams. "Optimal quantum circuits for general two-qubit gates."

- The number of parameters to attune the network will rely heavily on our initial number of encoding qubits n
- Prior methods have shown that a feature mapping of encoding n qubits for an n pixel image yields quality results for the QCNN
- We will experiment with using an amplitude encoded feature mapping requiring only $n = \log_2 ML$ qubits for an $M \times L$ pixel image

We will be exploring a variety of parameterized unitary matrices for both the pooling and convolutional layers. These are to include the full generalized 15 parameter unitary gates as well as the sub space encoded 3 parameter $N(\alpha,\beta,\gamma)$ gate. The computational bottleneck lies in the attenuation of an increasing number of parameters depending on the composite unitary chosen as well as the exponential time it takes to execute quantum simulations. To mitigate this computational bottleneck on modern hardware, we propose to execute our computations on a simulated Distributed Noisy Intermediate Scale Quantum (D-NISQ) system.

- The first of four D-NISQ layers is the decomposition which is a partition of the image data as is common with classical neural networks.
- The classical to quantum layer is second with the distributed calculation of the quantum encoding (feature mapping) portion of the newly partitioned data.
- The quantum computation layer is the parallel execution of our encoded circuits for an epoch.
- The final layer is the quantum fusion layer where we may coalesce our data to find the cumulative error, update the QCNN parameters, then begin the process again.
- Past experiments have shown that the Old Dominion University Wahab cluster is an ideal candidate for the execution of these simulations in parallel on both GPU's and CPU's.

Results

We first show promising image encoding results on NISQ era hardware. The image encoding portion of the QCNN methodology is applied before the first convolutional layer.

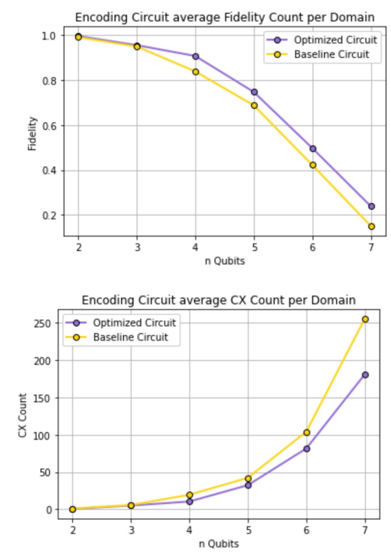

Fig. 6: Amplitude encoding circuit for a binary particle trace image broken into domain sizes of n qubits. Metrics are average across all circuits for each domain.

Acknowledgements

This work was funded in part by the Center for Nuclear Femtography (CNF), which is administrated by the Southeastern Universities Research Association (SURA) under an appropriation from the Commonwealth of Virginia, and the Richard T. Cheng Endowment at Old Dominion University. The performance evaluation was performed using the Wahab computing cluster at Old Dominion University.