Project Proposal

Google Summer of Code 2020

Implementation of Quantum Machine Learning to Perform High Energy Physics Analysis at the LHC

CERN-HSF

High Energy Physics Software Foundation

Saral Uttamani

Email: saral.uttamani@stonybrook.edu

Phone: +1 (631) 687-9227

Mentors:

Wen Guan, University of Wisconsin-Madison

Shaojun Sun, University of Wisconsin-Madison

Chen Zhou, University of Wisconsin-Madison

Sergei Gleyzer, University of Alabama

Index

Introduction

Personal Details:

Full Name	Saral Uttamani
Email	saral.uttamani@stonybrook.edu
Phone	+1 (631) 687-9227
Institute	Stony Brook University
Location	New York, USA
Timezone	EST
Github	https://github.com/MrSaral
LinkedIn	https://www.linkedin.com/in/saral-uttamani/
Resume	tiny.cc/SaralUttamani

About Me

I am a Computer Science Grad student at Stony Brook University, New York, USA. With experience in software development, research, and teaching, I have attempted to diversify my skill set. I am well versed with multiple languages such as Python, C/C++, Java, Javascript etc and their frameworks. I have experience with Machine Learning algorithms and have done various projects (present on GitHub) and published a research paper on Convolutional LSTMs. [1]

Currently I am working with Prof. Predrag Krstic, researching Quantum Machine Learning algorithms and cryptography. In my grad school, I have completed a Machine Learning course (CSE 512) under Prof. Minh and Quantum Computing course (CSE 550) under Prof. Predrag Krstic. I have experience using IBM quantum computers and Qiskit programming as well.

Being an inquisitive young professional, I love doing things that satisfy my curiosity. I am a full time physics enthusiast, so one can catch me reading Physics books in my spare time. (currently I'm reading "Physics of the Impossible" by Michio Kaku)

Link to the exercise task provided by the mentors:

https://github.com/MrSaral/GSoC_QMLHEP_Exercise

Why GSoC?

I entered the field of Quantum Computing because it combines my passions in a unique way. My love for physics and zeal for problem solving, both are captured by this project. Besides the obvious benefits of doing GSoC, what motivates me the most is the fact that I can interact with so many scientists and professionals from CERN. The prospect that my work with CERN and GSoC could potentially be of some help to researchers and programmers across the globe, is very exciting to me. This project also aligns with my current graduate research. I strongly believe I am the perfect candidate to work on this project.

Synopsis

Machine Learning (ML) is the science of getting computers to learn just like humans and improve their learning over time with a given set of data. It essentially consists of a set of algorithms that analyse the given data and identify the underlying patterns of the data. With the current state of machine learning, known as classical machine learning, we have been successful in solving problems with respect to a variety of revolutionary applications ranging from the social to the scientific. But we can only go so far with computing power, as the amount of data is growing exponentially, we run out of steam. This is where Quantum mechanics is going to play a key role.

In physical experiments, like the ones performed at CERN Large Hadron Collider(LHC), a very large amount of data (in order of petabytes) is generated. Processing this data would require excessive computing resources. Quantum Computers, where qubits are used instead of bits in classical computers, have theoretically been proven to improve the time complexities of machine learning algorithms. Instead of iteratively applying algorithms on each point in space (as seen in classical ML), we could simultaneously apply it to all points in the state space (Quantum ML), hence providing a more reliable and optimal solution. This could be the new paradigm that can potentially unlock doors to new discoveries.

While the promise of Quantum Machine Learning is substantial, we must keep in mind that we are still in the infant stages of this technology. Current Quantum Computers have noises and require error correction to perform their operations. We are currently in the Noisy Intermediate Scale Quantum(NISQ) era. In this current scenario, we are making use of hybrid algorithms which use classical as well as quantum algorithms to solve problems. One of the famous classical ML techniques is Neural Networks and it has a Quantum counterpart called Quantum Neural Networks(QNN).[2]

With this project, we intend to implement and study these various hybrid algorithms of Quantum Machine Learning for LHC HEP analysis. We also intend to develop a Quantum Neural Network algorithm based on Google OpenFermion framework.

Technical requirements

OpenFermion[4] is an open source library for compiling and analyzing quantum algorithms to simulate fermionic systems, including quantum chemistry. This framework is useful as it provides an interface for Google's Cirq(Open source Python library for writing, manipulating, and optimizing quantum circuits) and StrawberryFields (Python library for designing, simulating, and optimizing continuous variable quantum optical circuits). I also have prior experience in using PennyLane for Quantum Machine Learning.

Benefits to Community

The immediate benefits of this project is to enhance the ability of the HEP community to use Quantum Machine Learning methods. The project will facilitate the process of large scale HEP-LHC data. The identification of rare signals against immense backgrounds will become faster. This can be an enterprising moment for Quantum Computing as this project could possibly bring us one step closer to realising the objective of reliable Quantum computers. This could potentially resolve a number of difficulties we encounter today and even perhaps some that still remain unknown to us.

Project Goals

Objectives

- Implement a Quantum Variational method or a Quantum Neural Network method based on Google OpenFermion framework.
- Apply the quantum machine learning method to one or two of the LHC flagship physics channels (e.g. double-Higgs production). Compare the quantum machine learning performance to the classical machine learning performance.
- (Optional or partly) Implement a QML interface for HEP which can support different quantum frameworks such as OpenFermion.

Tasks

1. Implement a Quantum Variational method or a Quantum Neural Network method based on Google OpenFermion framework.
    - Previous Work: As it stands, there is an implementation of Quantum Neural Networks by Xanadu.ai in it's packages of PennyLane and Quantum Machine Learning Toolkit(QMLT). That implementation is coherent with StrawberryFields. [3]
    - Description: With this task, the aim is to build an OpenFermion based library implementation of Quantum Neural Networks(QNN) and Variational methods. This will help the programmer to directly make a call to those methods using this unified library.
    - Steps:
        - Gather the requirements - Analysing the current existing frameworks for QNN and OpenFermion. Understanding the requirements of the library to be implemented.
        - Design architecture - Based on requirement specification, design the library, its various methods and their signatures.
        - Implement Method - Coding and Integrating the methods of the library.
        - Test - Using existing implementation of similar modules, verifying the working of the library. Checking the performance of the module in edge cases.
        - Error Handling - Developing error handling mechanisms for unknown test cases.
        - Document the code - Adding comments and forming the read-me documentation for the module. Uploading the code on GitHub.

2. Apply the quantum machine learning method to one or two of the LHC flagship physics channels (e.g. double-Higgs production). Compare the quantum machine learning performance to the classical machine learning performance.
    - Description: This task will essentially act as PoC for Quantum supremacy over classical machine learning algorithms. Using the data from the flagship channels of LHC, we can train models across algorithms from both sides and perform benchmark analysis based on common metrics.
    - Steps:

    - Data Gathering - Short listing the physics channels to be used and understanding the infrastructure of how the data is generated.
    - Data Preprocessing - Transforming the data dumps to formats that can be used coherently. Also visualising the data to find hidden distributions/ elimination of outliers in the data.
    - Data Modelling - Based on insights from processed data, select appropriate machine learning algorithms and quantum machine learning models.
    - Training and Optimisation - Split the original data into train and test datasets. Train the models based on the train data set. Optimise the models by tweaking its parameters.
    - Performance Analysis - Testing the optimised models against the test dataset and calculating common metrics to draw conclusions.
    - Document the study - Once the analysis is complete, document all findings.

3. Implement a QML interface for HEP which can support different quantum frameworks such as OpenFermion. (Optional or partly)
    - Description: This is an optional development which may require extensive understanding of the use case at HEP. The goal of this task would be to have a unified platform for HEP which integrates various frameworks. The output could be expanded later on as well.
    - Steps:
        - Gathering requirements - Understanding the current architecture and use cases of QML at HEP.
        - Designing interface - Designing the unified interface based on the requirements. The main challenge here is to design a scalable architecture which can be expanded later on as per need.
        - Implementing interface - Once the top features are designed, coding this module can begin. Coding can follow an iterative development approach.
        - Testing - Interestingly, testing this module must be done in batches alongside the coding phase to isolate bugs early on.
        - Error correction - Implementation of error handling mechanisms for unknown cases.
        - Documentation - Documentation of the code.

Timeline

March 31.	Deadline for submitting Project Proposal
March 31 - May 4	Research about Google OpenFermion Read documentation associated to StrawberryFields
May 4 - June 1	Official Community Bonding Period Getting involved with the community at CERN and OpenFermion Get acquainted with the tools of CERN and LHC-HEP Task 1: Requirement gathering and designing begins
June 1 - June 25	Coding Phase begins: Implementation, Testing of Task 1
June 25 - June 29	Slack time, Documentation of Task 1
June 29 - July 3	Phase 1 Evaluation Task 2 - Data preprocessing and visualisation
July 3 - July 24	Task 2 - Modelling classical and quantum ML
July 24 - July 27	Slack time for Task 2
July 27 - July 31	Phase 2 Evaluation
July 31 - August 24	Task 3 design, implement, testing and documentation.
August 24 - August 28	Slack time for Task 3
August 28.	Final Submission and Report.
September 8.	Results of GSoC 2020

My summer commitments:

I do not have any commitments in the summer post my semester end and can devote more than and on an average 40 hours per week for this project.

Deliverables

- A program that implemented a Quantum Variational method or a Quantum Neural Network method based on Google OpenFermion framework.
- Successfully apply the Quantum Machine Learning method to LHC physics analyses and obtain performance benchmarks to compare to classical machine learning methods.
- (Optional or partly) A program which can be easily used as a QML interface for HEP to support different quantum frameworks.

References

[1] Wagle S., Uttamani S., Dsouza S., Devadkar K. (2020) Predicting Surface Air Temperature Using Convolutional Long Short-Term Memory Networks. In: Kumar A., Mozar S. (eds) ICCCE 2019. Lecture Notes in Electrical Engineering, vol 570. Springer, Singapore

[2] Beer, K., Bondarenko, D., Farrelly, T. et al. Training deep quantum neural networks. Nat Commun 11, 808 (2020). https://doi.org/10.1038/s41467-020-14454-2

[3] Continuous-variable quantum neural networks. Nathan Killoran, Thomas R. Bromley, Juan Miguel Arrazola, Maria Schuld, Nicolás Quesada, and Seth Lloyd Phys. Rev. Research 1, 033063 – Published 31 October 2019

[4] Jarrod R. McClean, Kevin J. Sung, Ian D. Kivlichan, Xavier Bonet-Monroig, Yudong Cao, Chengyu Dai, E. Schuyler Fried, Craig Gidney, Brendan Gimby, Pranav Gokhale, Thomas Häner, Tarini Hardikar, Vojtěch Havlíček, Oscar Higgott, Cupjin Huang, Josh Izaac, Zhang Jiang, William Kirby, Xinle Liu, Sam McArdle, Matthew Neeley, Thomas O'Brien, Bryan O'Gorman, Isil Ozfidan, Maxwell D. Radin, Jhonathan Romero, Nicholas Rubin, Nicolas P. D. Sawaya, Kanav Setia, Sukin Sim, Damian S. Steiger, Mark Steudtner, Qiming Sun, Wei Sun, Daochen Wang, Fang Zhang and Ryan Babbush. OpenFermion: The Electronic Structure Package for Quantum Computers. arXiv:1710.07629. 2017.