Project Proposal for Google
Summer of Code 2023

Graph Neural Networks for End-to-End Particle Identification with the CMS Experiment

@

Mentors:
Ruchi Chudasama
Emanuele Usai
Shravan Chaudhari
Sergei Gleyzer
Michael Andrews
Eric Reinhardt
Samuel Campbell

Devdeep Shetranjiwala
Dhirubhai Ambani Institute of Information and Communication Technology - DAIICT, India.

Contents

Personal Details

Name: Devdeep Shetranjiwala

Email: deveep0702@gmail.com, devdeepshetranjiwala@gmail.com

LinkedIn: https://www.linkedin.com/in/devdeep-shetranjiwala-4290b21ba/

Mobile: +91 9265816294

Country of Residence: India

Timezone: Indian Standard Time (UTC +5:30)

Github Handle: Devdeep-J-S

Language: English

University: Dhirubhai Ambani Institute of Information and Communication Technology - DAIICT, India.

Major: Information and Communication of Technology with a minor in CS.

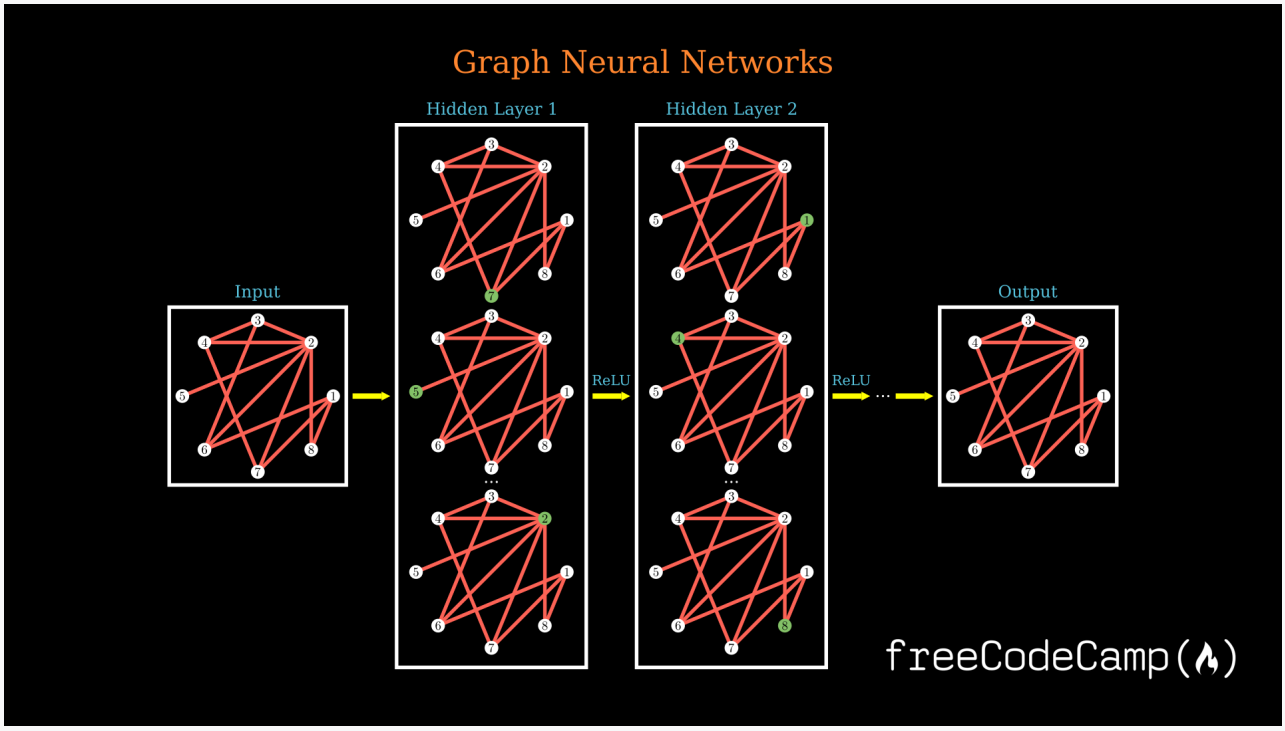
Introduction

In this project, we aim to develop and implement Graph Neural Networks (GNNs) for low-momentum tau particle identification using the CMS experiment data. Tau leptons play a crucial role in the search for new Physics, and the development of efficient tagging algorithms is necessary for signal extraction from the background. We will compare the results of GNN-based algorithms with the CNN-based algorithms for tau particle identification.

Synopsis

This project aims to develop end-to-end graph neural networks for particle (tau) identification with the CMS experiment data. The project will involve exploring different strategies for graph construction, developing different graph architectures, and testing and benchmarking GNN performance.

We will analyse the model performance from multiple perspectives and interpret what the networks have learned through training. We will benchmark model inference on GPU and provide user guidance through documentation and code.



Credit: freeCodeCamp

MileStones

I. Project Goals

1. Data preparation: The first step would be to obtain and preprocess the raw data for the Tau identification task.
2. Graph construction: The next step would be to construct the graph representation of the data. Different node and edge construction strategies and input representations will be explored and analysed.
3. GNN architecture: Various graph neural network architectures will be experimented with and compared to identify the best-performing architecture for the task.
4. Hyperparameter tuning: Hyperparameters of the best-performing architecture will be tuned for optimal performance.
5. Testing and evaluation: The trained model will be tested on a separate test dataset, and its performance will be evaluated using metrics such as accuracy, precision, recall, and F1 score.
6. Inference and deployment: The trained model will be integrated with the CMSSW inference engine for offline and high-level trigger systems.

II. Expected Results

1. Trained end-to-end graph neural network model for tau identification.
2. Analysis of the design choices and performance of the GNN model.
3. Comparison of the results of GNN-based algorithms with the CNN-based algorithms for tau particle identification.
4. Optimized and scalable GNN algorithm for efficient training and inference.
5. Benchmark of end-to-end GNN inference on GPU.
6. Final inference performance benchmark on the CMSSW inference engine.

III. Outline of Approach and Implementation Plan

Starting Research and Understanding:

- Read relevant literature on Graph Neural Networks (GNNs) and their applications in particle physics.
- Understand the CMS experiment and its data format.
- Study the existing algorithms for tau identification using CMS data.
- Analyse the limitations of existing methods and explore how GNNs can be used to overcome them.

Data Generation:

- Download the CMS Open Data and preprocess it for GNN input.
- Generate simulated data for training and testing purposes.
- Create a data pipeline for efficient data loading and processing.

Graph Construction:

- Explore different graph construction strategies, including node and edge construction.
- Experiment with pruning, preprocessing, and input representations to optimise graph construction.
- Analyze the performance of different graph construction strategies.

GNN Architecture:

- Conduct an exploratory analysis of different GNN architectures.
- Experiment with novel architectures inspired by exploratory analysis.
- Select the best-performing GNN architecture for tau identification.

Training and Optimization:

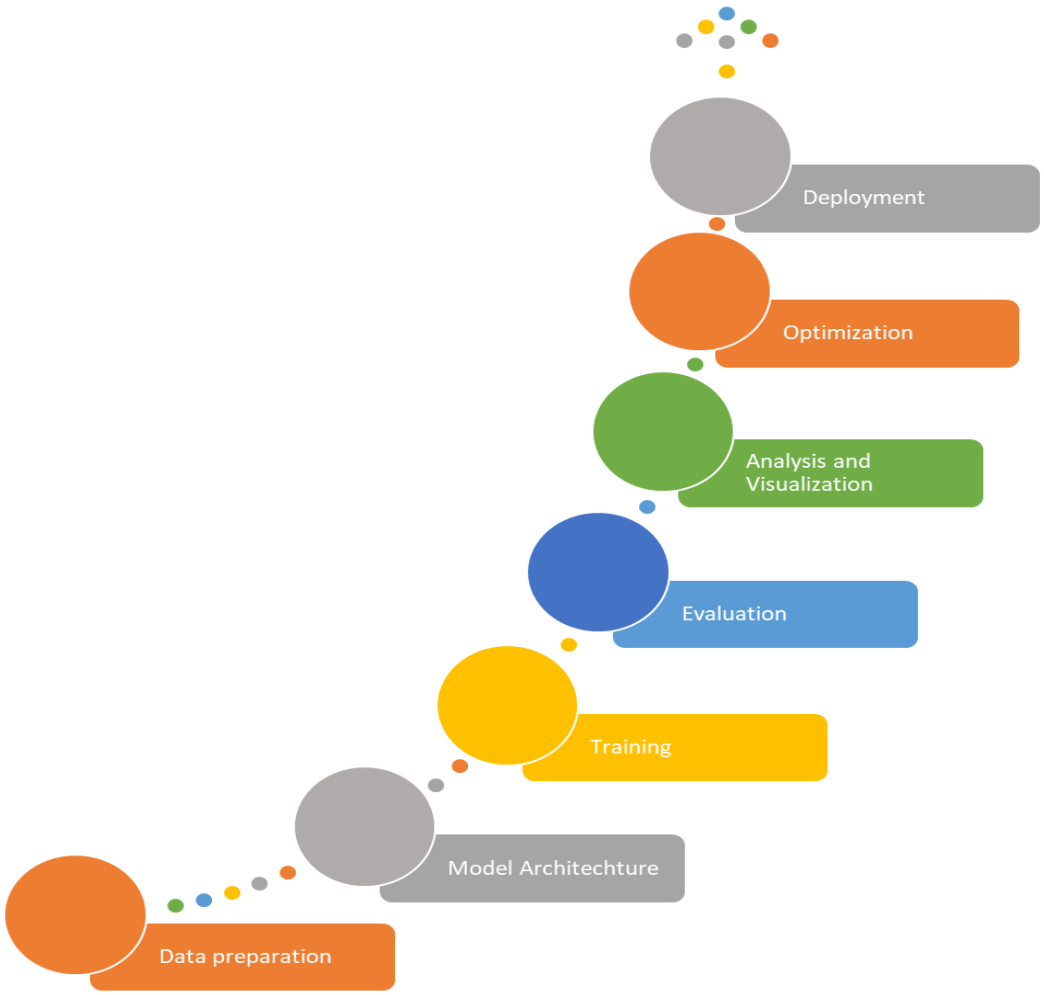
- Train the selected GNN architecture using the generated data.
- Optimise hyperparameters to improve performance.
- Analyze the performance of the model on validation and test sets.

Inference and Deployment:

- Test and benchmark the trained model's inference speed on GPUs.
- Integrate the GNN algorithm into the CMS Software (CMSSW) inference engine.
- Optimise the algorithm for efficient training and inference on heterogeneous computing

IV. Optional Milestones

1. Incorporating additional features: Explore incorporating additional features such as timing and energy information in the graph representation.
2. Comparison with CNNs: Compare the performance of GNN-based algorithms with CNN-based algorithms for Tau identification.
3. Optimizing training on multiple GPUs: Scale the GNN algorithm to multiple GPUs and optimise for efficient training and inference using heterogeneous computing.



V. References

Paper 1: [1807.11916] End-to-End Physics Event Classification with CMS Open Data: Applying Image-Based Deep Learning to Detector Data for the Direct Classification of Collision Events at the LHC

Paper 2: [1902.08276] End-to-End Jet Classification of Quarks and Gluons with the CMS Open Data

The Large Hadron Collider | CERN

GitHub - cms-sw/cmssw: CMS Offline Software

CMS | CERN

VI. Other Deliverables

Weekly report and blog

Timeline

Period	Deliverables	Specific Goals
(Communicating Bonding period) *May 4 - May 28*	➔ Community bonding report ➔ Establish communication with mentors ➔ Set up development environment ➔ Familiarize with CMS experiment data and tools	➔ Communicate and bond with students and mentors ➔ Know the project in more detail ➔ Set up the necessary software and tools required for the project ➔ Read documentation on CMS experiment data and tools

(Week 1) *May 28 - June 3*	➔ Data preparation and exploration report	➔ Explore and analyze different strategies for graph construction ➔ Perform data preprocessing and input representation ➔ Create a report on the data preparation and exploration process
(Week 2) *June 4 - June 10*		
(Week 3) *June 11 - June 17*	➔ Graph neural network architecture report	➔ Experiment with different graph architectures ➔ Identify the most promising architecture for low-momentum tau identification ➔ Create a report on the graph neural network architecture experimentation
(Week 4) *June 18 - June 24*		
(Week 5) *June 25 - July 1*	➔ Trained graph neural network model for tau identification	➔ Train the selected graph neural network model for low-momentum tau identification ➔ Optimize hyperparameters for the best model performance ➔ Create a trained model for low-momentum tau identification
(Week 6) *July 2 - July 8*		
Mid-term Evaluation *(July 10 - July 14)*		
(Week 7) *July 9 - July 15*	➔ Model testing and benchmarking report	➔ Test and benchmark the trained graph neural network model on different cases

(Week 8) *July 16 - July 22*		➔ Analyze the performance of the model in terms of inference speed, compute requirements, and robustness ➔ Create a report on the model testing and benchmarking process
(Week 9) *July 23 - July 29*	➔ Integration with CMSSW inference engine report	➔ Integrate the trained graph neural network model with the CMSSW inference engine ➔ Test the integration and ensure compatibility with offline and high-level trigger systems of the CMS experiment ➔ Create a report on the integration with the CMSSW inference engine
(Week 10) *July 30 - August 5*		
(Week 11) *August 6 - August 12*	➔ Final report and documentation	➔ Write a final report summarizing the project's objectives, methodology, and results ➔ Create documentation for the code and the trained model
(Week 12) *August 13 - August 19*		
(Final Week: Submit final work product to final mentor evaluation) *August 20 - August 27*	➔ Final evaluation ➔ Project submission	➔ Evaluate the final product and ensure that all project goals have been met. ➔ Submit the code developed during the project and instructions for running and testing it. ➔ Prepare a final presentation to showcase the project's achievements

Final Evaluation *August 28 - September 4*
Results are announced *September 5*

Prerequest challenges

Github repo link:
https://github.com/Devdeep-J-S/Graph-Neural-Networks-CMS

Learning from challenge

One of the major challenges I faced in particle identification with the CMS experiment was dealing with high-dimensional and complex datasets. These datasets are often sparse, noisy, and difficult to interpret, requiring a deep understanding of physics and advanced machine-learning techniques. As a result, I faced challenges in developing the necessary skills and knowledge to work with this type of data.

However, overcoming these challenges, I developed a strong interest in graph neural networks (GNNs) for end-to-end particle identification. GNNs have shown great promise in this area, allowing for the representation of particle data as graphs, where each particle is represented as a node and their relationships are represented as edges. This allows GNNs to learn complex relationships and patterns within the data, improving the accuracy of particle identification.

By learning about GNNs and their applications in particle identification, I developed a deep understanding of machine-learning techniques and physics concepts. This led to exciting opportunities to contribute to the field and make new discoveries and develop new and innovative approaches to analysing particle data.

Ultimately, working on these tasks has inspired me to pursue innovative solutions for particle physics problems using machine learning and graph neural networks and has also contributed to my interest in developing solutions for the CMS Trigger System.

Biographical Information

I. Educational background

I am a 3rd-year student pursuing a Bachelor's in Information and Communication Technology with a minor in Computational Science from Dhirubhai Ambani Institute of Information and Communication Technology-DAIICT. I am hardworking when it comes to academics and hold an 8 CGPA.

- EXTRACURRICULAR

IEEE IAS STUDENT BRANCH COMMITTEE DAIICT: Webmaster

February 2022 – Present | Gandhinagar, India

- Worked on websites and apps for numerous IEEE events, like i.Fest, Summer school, etc.
- Led team of web developers to develop events websites.

II. Technical Interest

I am passionate about software development and research, niching down to web development and machine learning and its applications.

I have completed several courses in machine learning, including ones that cover classification, regression, and clustering algorithms.

I have hands-on experience implementing these algorithms using Python and libraries like Scikit-learn, PyTorch and TensorFlow. I have also worked on several projects that involve processing and analysing large datasets, which would be useful for this project.

Regarding programming skills, I am proficient in Python, C, C++, and JavaScript and have experience with data manipulation and visualisation libraries like Pandas and Matplotlib.

I am comfortable working with Jupyter Notebooks and have experience with version control tools like Git. I am also good at web development using the Django-python framework. I am looking forward to growing and developing these skills more and more by contributing to this organisation.

III. Software Development Experience/ Projects

1. EVENT MANAGER WEB APP

Tech used: Python-Django, Twilio API, Google SMTP, SQLite

- Event Manager is a tool to record and retrieve the data of events effectively.
- An application with CRUD operations and user authentication.
- The web app has features like host sign-in, email and SMS notifications, Database Management and easy-to-use UI.
- Impact : Won WOC4.0 Web development Compition 1st Prize.

2. RESTAURANT AUTOMATION

Tech used: C, C++, Shell Script

- The problem is an advanced version of the Dining-Philosopher problem.
- Developed a code to handle the production and distribution of a product and applied it to manage a Restaurant.
- A program that uses the concepts like Concurrent Programming and Process management.

3. Woc 5.0 - WEATHER PREDICTION APP

Tech used: Python, Sklearn, Tensorflow, Flask

- Implemented an app to predict the weather by analysing data and categorising them into specific labels. Such a problem can be solved using various Machine learning techniques.

4. LABOR LIST AND WAGES MANAGEMENT TOOL

Tech used: Python-Django, PostgreSQL, HTML, CSS, Bootstrap

- A tool to effectively record and retrieve the data of Labor and Supervisors.
- An application with CRUD operations.
- The web app has features and database management.

5. OPEN SOURCE - VIDEO STREAMING - RTSP CLIENT-SERVER

Tech used: Python, Tkinter

- Implementation of a video streaming server and client that communicate using the Real-Time Streaming Protocol (RTSP) and Realtime Transfer Protocol (RTP).

IV. Communication

Working Hours - I am flexible with my timings and can work dedicatedly for hours together. Also, I will ensure that GSoC will be my only significant commitment this summer and give it my full attention.

Communication preferences - I am comfortable with any platform and can adjust whatever suits the mentors.

In case of work delay due to personal or other reasons, I assure you that I will work longer and more efficiently to complete the backlog in the available time. I plan to write weekly progress blogs on my medium publication to establish a systematic solution to ensure I get all week's work. I would share the same with my mentors while actively communicating with them.

V. Resume

GSOC_Resume_Devdeep.pdf

VI. Engagements during Summer

I have no commitments in the summer. I'll be staying back home for most of it. I have mentioned my typical working hours above; on average, I can spend 35-40 hours per week on the project.

Note: If the project goes to 350 hr length due to the complexity of the task or additional future milestones implementation, I am also comfortable with it.