# Novel edge classification architectures for charged particle tracking with graph neural networks

**Mentors**: Kilian Lieret, Gage DeZoort

## Abstract

This project explores new edge classification architectures for a tracking approach using graph neural networks with an object condensation approach.

## Introduction

Particle tracking data can be represented as a graph where the nodes represent particle hits and edges represent their trajectories. Many tracking approaches use three stages: graph construction, edge classification and track building.

1) Track hits are embedded as graph nodes and edges are drawn between them to represent potential particle trajectories. This step is called graph construction.
2) An edge classifier is used to predict whether each edge is true (connecting two hits of the same particle) or false
3) An algorithm, e.g. connected components, graph walking, or object condensation, is used to assign hits into tracks.

We have observed that in our object condensation pipeline, edge classification performance is critical to forming particle tracks with a high efficiency.

## Architectural choices

The edge classifiers this project has used so far rely on interaction network (IN) GNN layers, which are designed to leverage both node features and edge features.This project has so far used stacked layers of INs with layer-to-layer residual connections between them. However, a comprehensive survey of IN-based edge classifier design has yet to be studied. Possible choices to study include:

- Layernorm
- Regularization (weight decay, dropout)
- Exploring different kinds of residual connections - connecting to the first layer, connecting from layer-to-layer
- The effect of message-passing depth
- Message aggregation schemes

It may additionally be interesting to try other non-IN architectures - heterogeneous architectures, for example, consider nodes (and edges) to have different “flavors.” In the context of tracking, a heterogeneous graph may have different nodes corresponding to the type of hit produced - a pixel hit vs. a hit in a strip sensor.

## Software Deliverables

The edge classification code is written in Python. We will be using Jupyter notebooks for initial model exploration. We shall then implement promising models in the main gnn_tracking package.

## Timeline

- Week 1-2:
  - Getting started with GNNs (learning pytorch geometric); getting the repository code to run
- Week 3-4:
  - Diving deeper in GNN papers, understanding existing models/approaches and developing new ideas
  - Discussion of these ideas with mentors
  - One or two simple tasks in repository (similar to framework tasks) to practice git and get a feel for the software
- Week 5-6:
  - Building prototypes of new ideas (e.g., new network architectures)
- Week 7:
  - Integrating prototype code with main repository
- Week 8-9:
  - Optimizing new approach on larger scale with full dataset & hyperparameter optimization
- Week 10:
  - Ensuring reproducibility
  - Writing final report