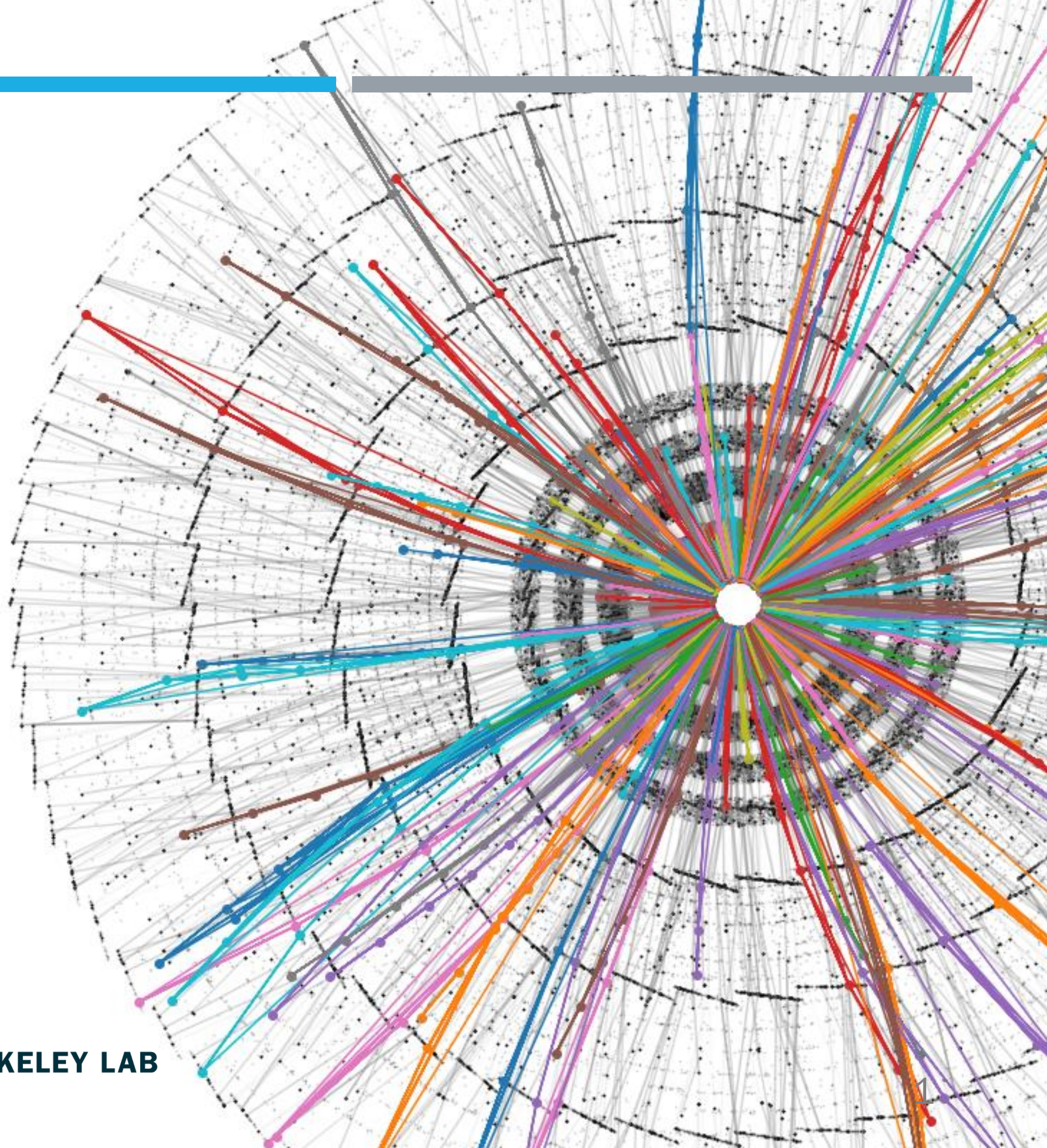
GRAPH NEURAL NETWORK TRACK RECONSTRUCTION FOR ATLAS ITK

5TH IML WORKSHOP, CERN, 13 MAY 2022



DANIEL MURNANE

ON BEHALF OF THE ATLAS COLLABORATION

ATLAS EXPERIMENT

TrkX

L2IT

BERKELEY LAB

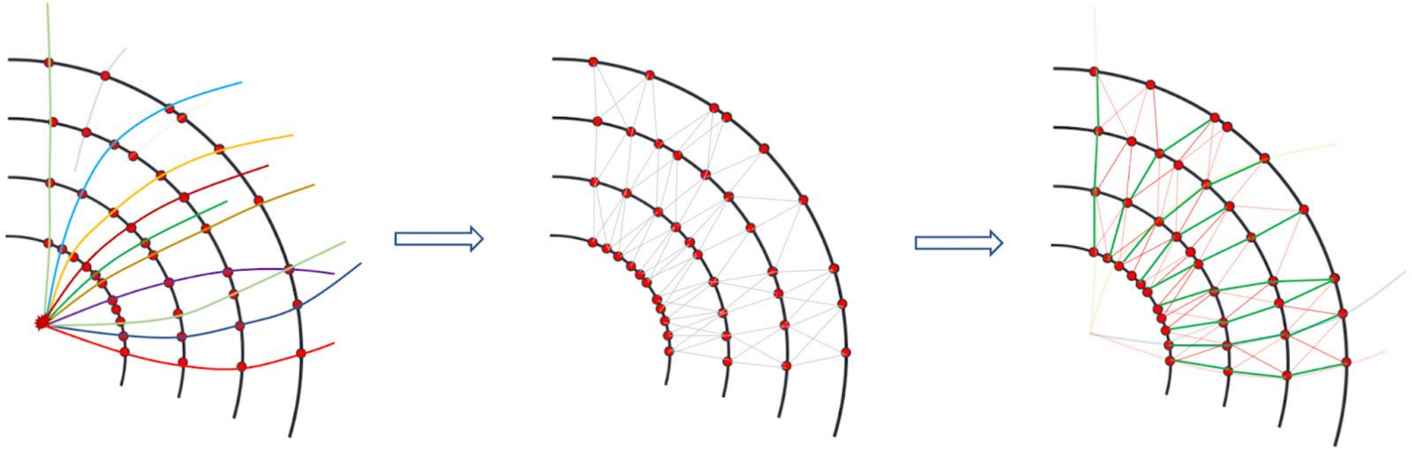
PREVIOUS WORK & BACKGROUND

GRAPH REPRESENTATION OF AN EVENT

- The goal of track reconstruction:

Given set of hits in a detector from particles, assign label(s) to each hit.

Perfect classification: All hits from a particle (*and only those hits*) share the same label



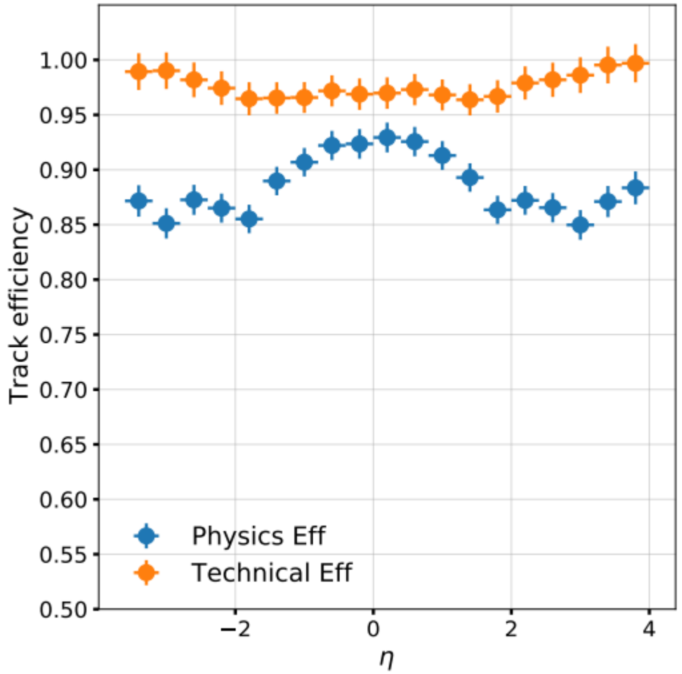
- What does it mean to represent an event with a graph?
  - Treat each hit as a **node**
  - A node can have features (e.g. position, charge deposit, etc.)
  - Nodes can be connected by **edges**, possibly in a meaningful way
- Goal: **Use ML and/or graph techniques to segment or cluster the nodes to match particle tracks**
- **Proof-of-concept:** TrackML community challenge dataset with simplified simulation

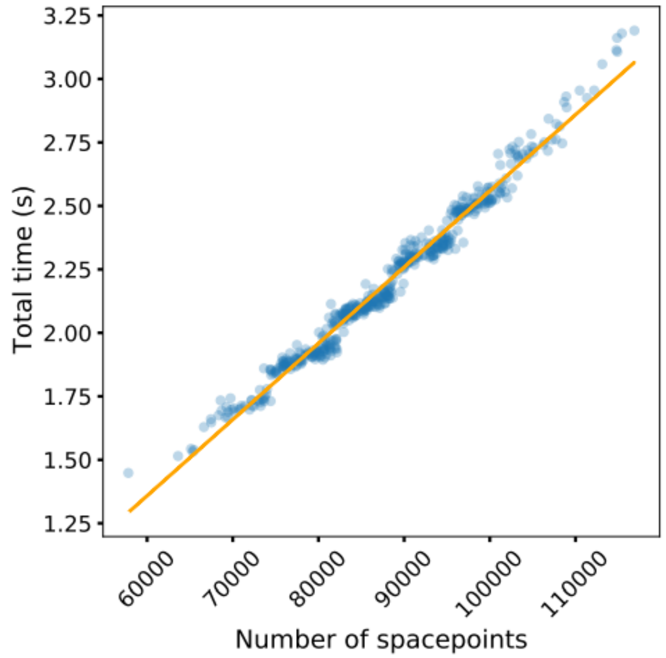
TRACKML PERFORMANCE

Two groups worked on the results in this presentation, and both first tested methods on TrackML, based on the GNN-based reconstruction introduced in arxiv:2003.11603

On this dataset, Exatrkx showed graph-based approach:

- Is competitive with highly-tuned hand-engineered solutions
- Has good scaling properties
- Is fast (~0.7s per event)
- Core algorithm is geometry-independent
- Is robust to noise and miscalibration





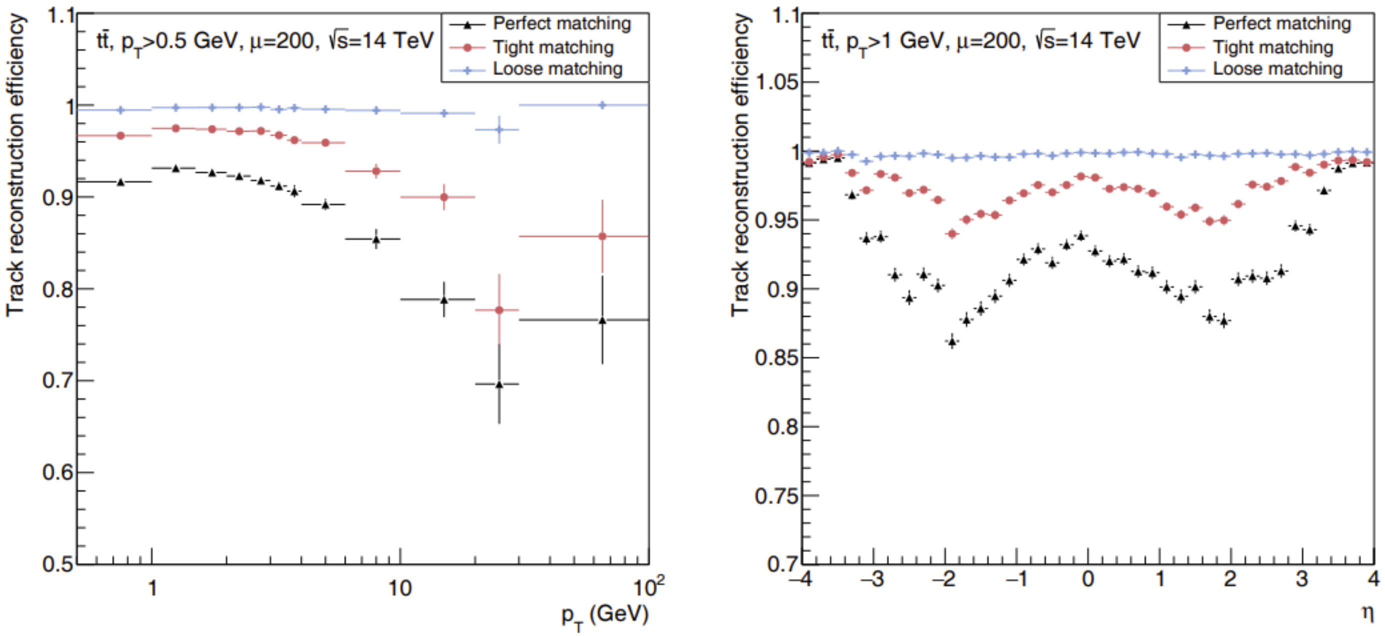
ExaTrkx collaboration, arXiv:2103.06995

TRACKML PERFORMANCE

Two groups worked on the results in this presentation, and both first tested methods on TrackML, based on the GNN-based reconstruction introduced in arxiv:2003.11603

On this dataset, L2IT showed graph-based approach:

- Able to even get high efficiency with a perfect matching scheme
- Consistent performance (with loose matching, i.e. standard ATLAS matching) across η and p_T



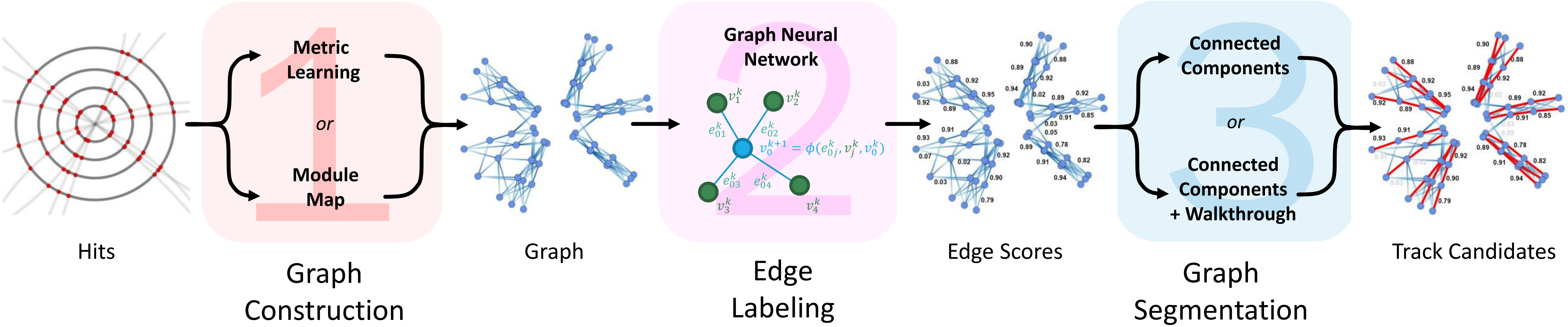
(a) Track reconstruction efficiency vs. p_T (b) Track reconstruction efficiency vs. η

L2IT, arxiv:2103.00916

GRAPH-BASED TRACK RECONSTRUCTION PIPELINE FOR ITK GEOMETRY

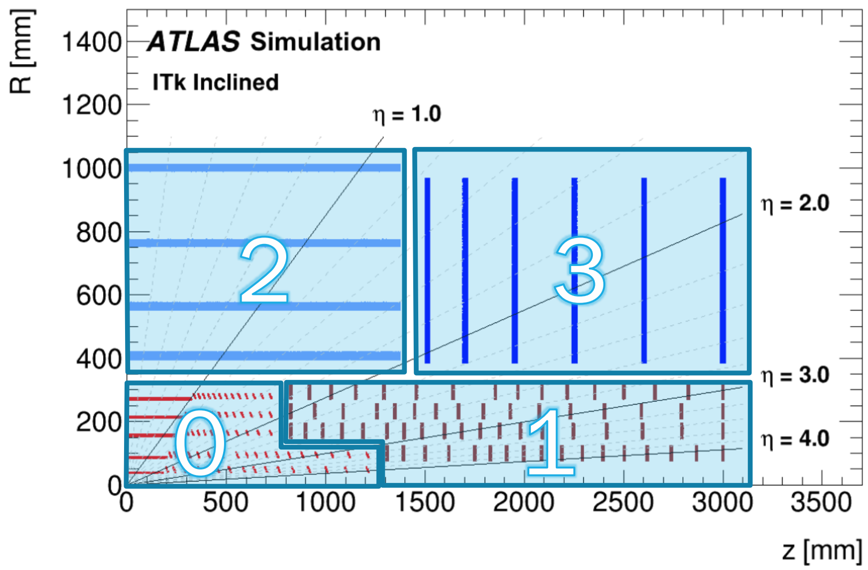
PIPELINE OVERVIEW

- Current pipeline of the L2IT-Exatrkx collaborative effort
- Each stage offers multiple independent choices, depending on hardware and time constraints



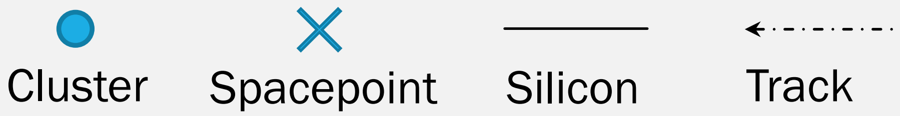
ITK GEOMETRY

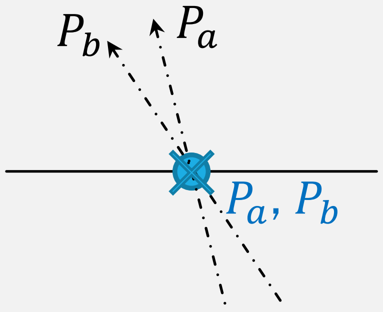
- Generation script* using Athena, $t\bar{t}$ at $\mu = \langle 200 \rangle$: with statistics dominated by soft interactions
- ITk consists of barrel and endcap, each with pixels and strips:



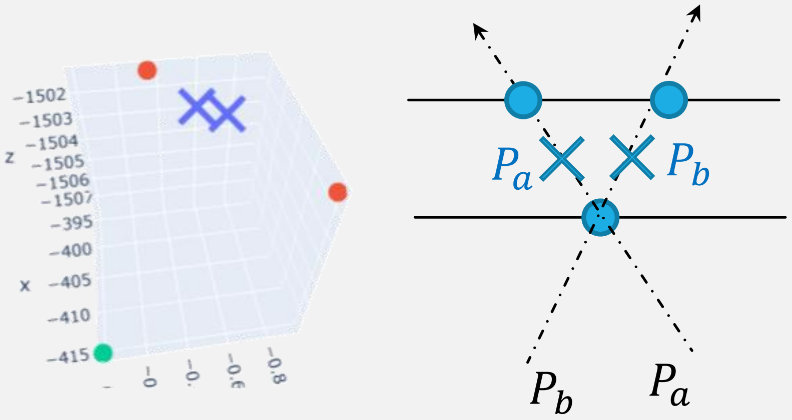
0: Pixel barrel
1: Pixel endcap
2: Strip barrel
3: Strip endcap

- Spacepoints are defined depending on strip or pixel: →





Pixel is trivial: Each spacepoint maps to one cluster, which can map to many particles

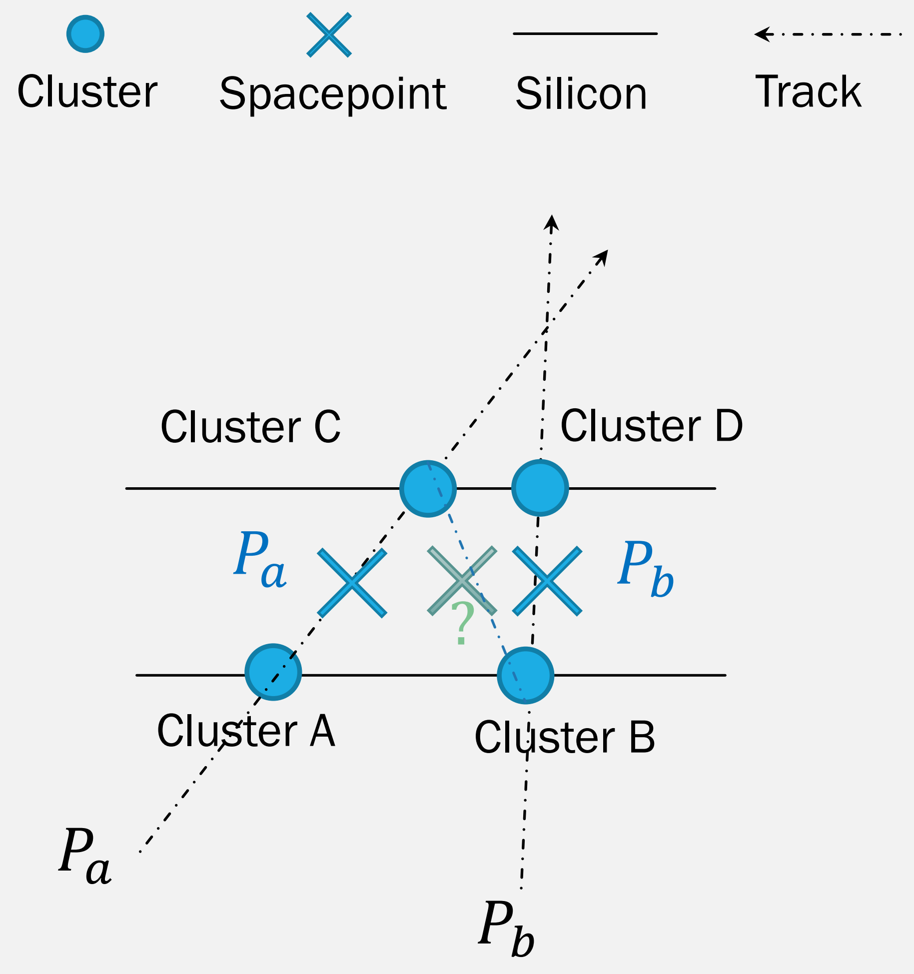


Strip: Each spacepoint maps to two clusters – one on either side of strip, which can map to many particles

*Thanks Noemi Calace

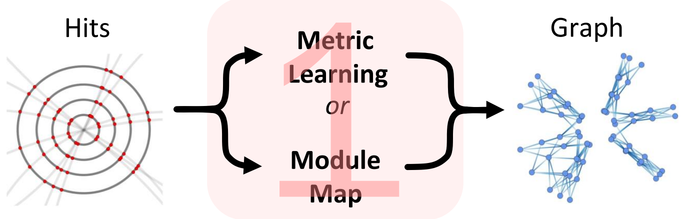
ITK GEOMETRY

- Fiducial particles are **charged, with $\eta \in [-4, 4]$, and production radius < 260mm**
- Each event has O(15k) fiducial particles, O(300k) spacepoints
- We define **background** spacepoints as including:
  - Those left by non-fiducial or intermediate particles (i.e. any particle barcodes not retained during simulation), or
  - Those mis-constructed in the strip regions as **ghost** spacepoints
- An event has O(170k) background spacepoints



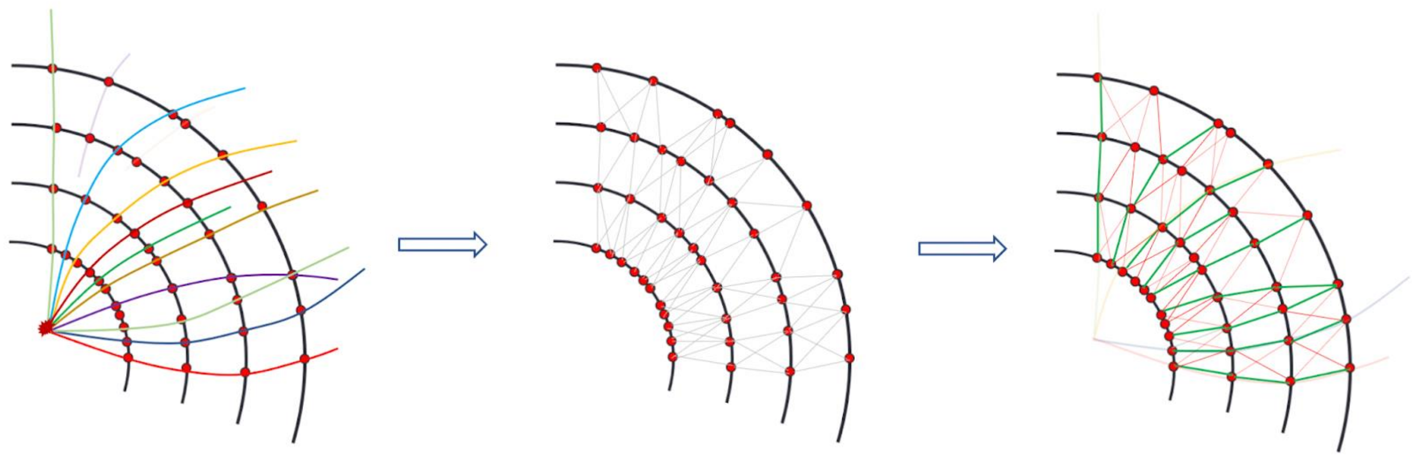
Ghost spacepoint: Incorrectly constructed from clusters left by different particles

GRAPH CONSTRUCTION

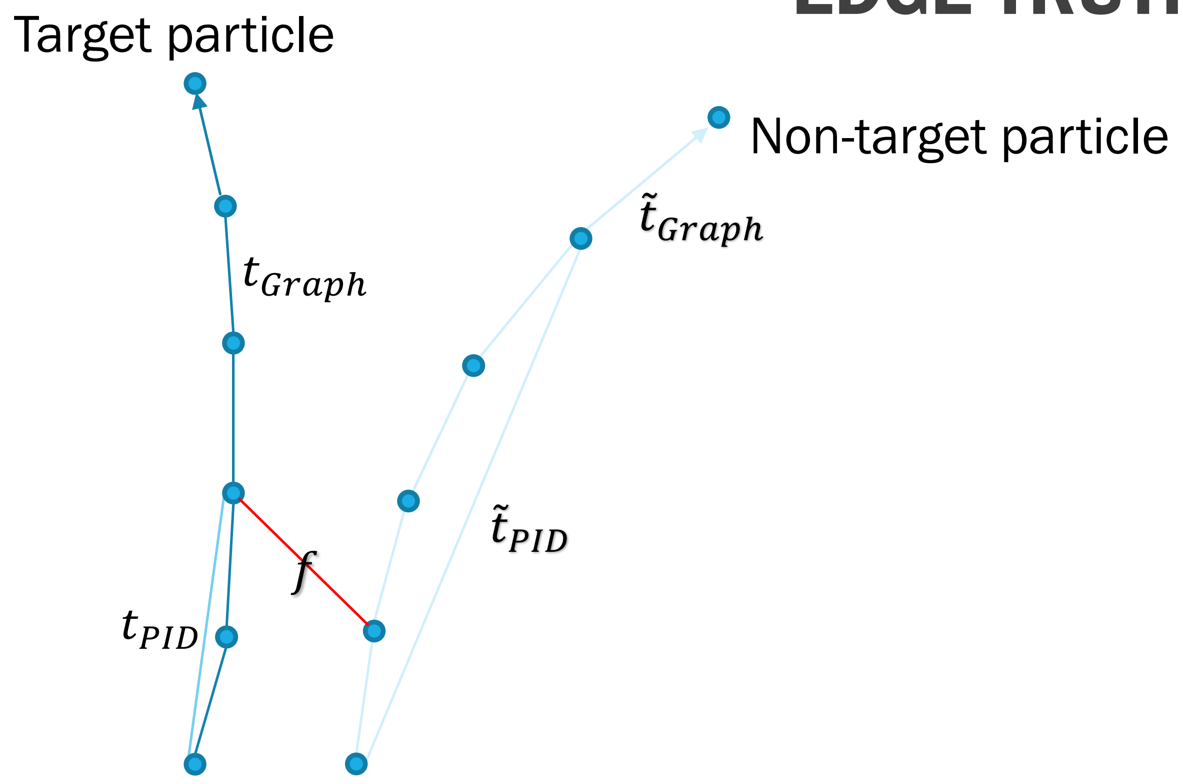


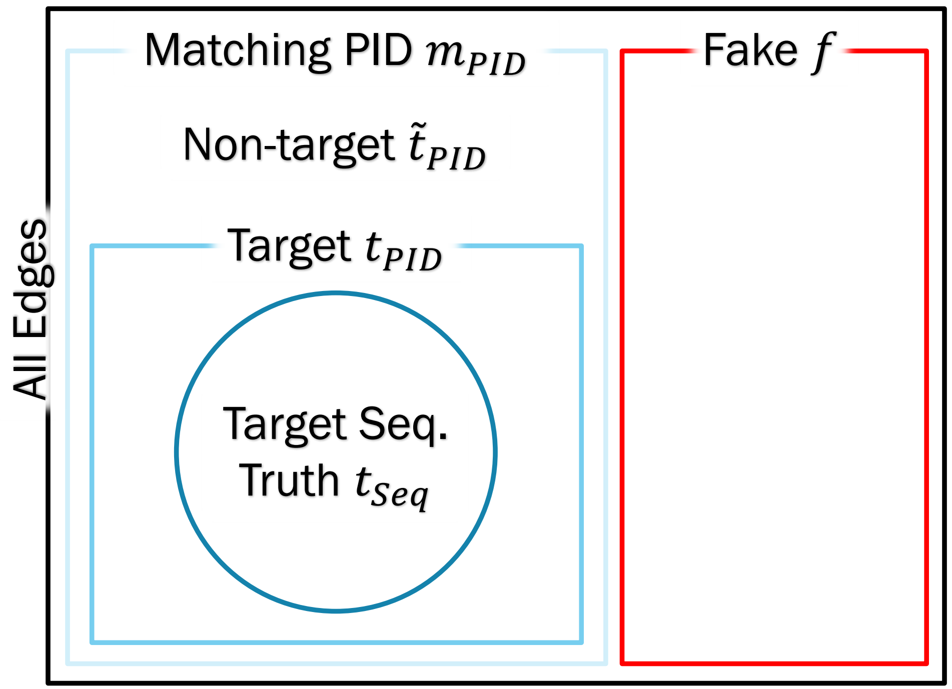
Graph Construction

- What is the goal of graph construction?
  - To apply a GNN, need a graph structure from spacepoint data
  - Depending on our target particles, an edge can have several types of truth
- First need to define target graph to construct



EDGE TRUTH DEFINITIONS





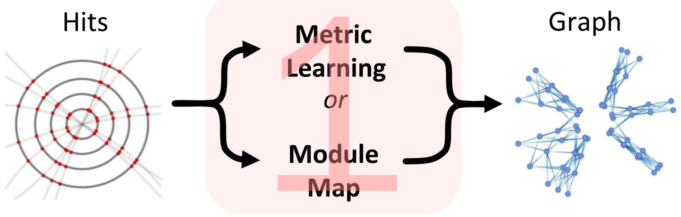
Target particle:

- $p_T > 1GeV$, and
- At least 3 SP on different modules, and
- Primary

Therefore, define efficiency and purity (note that we mask out sequential non-target) for a graph with edges e

$$\text{Efficiency} = \frac{|e \cap t_{Seq}|}{|t_{Seq}|}, \quad \text{Purity} = \frac{|e \cap t_{Seq} - \tilde{t}_{Seq}|}{|e - \tilde{t}_{Seq}|}$$

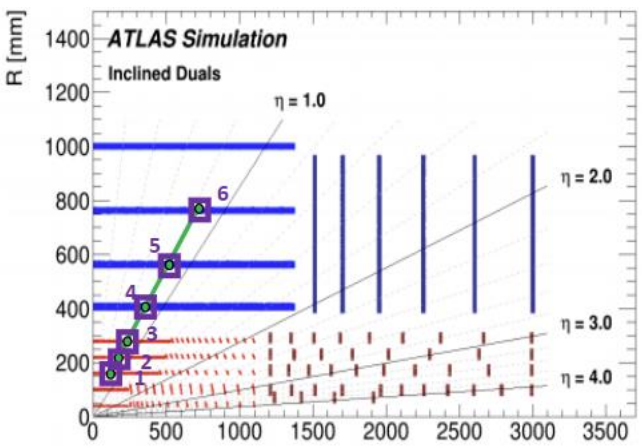
MODULE MAP



Graph Construction

- **The idea:** Build a map of detector modules, where a mapping between two modules means a particle could sequentially pass from one to the other
- Can make this more powerful by mapping triplets, where a connection *from* module A *to* module B *to* module C means that at least one true track has passed sequentially through A to B to C
- Step 1: Build all combinations of sequential triplets for an event, register an A-to-B-to-C entry if a triplet passes through
- Step 2: For *each* A-to-B-to-C entry, also register/update the max and min values of a set of geometric observables. Apply these cuts when building the graph in inference. O(90k) events used to train module map.

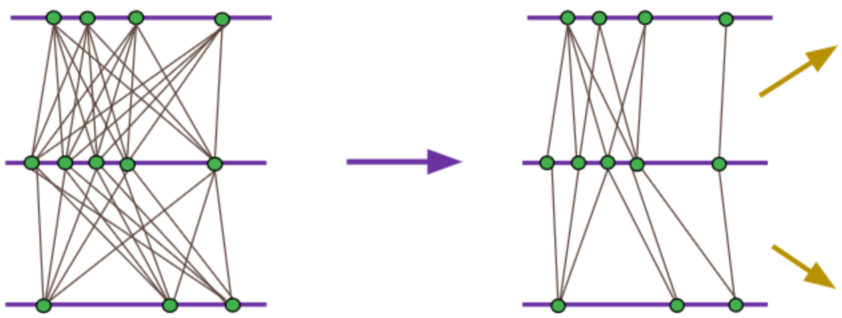
Step 1



Connections added:

1 -> 2 -> 3
2 -> 3 -> 4
3 -> 4 -> 5
4 -> 5 -> 6

Step 2

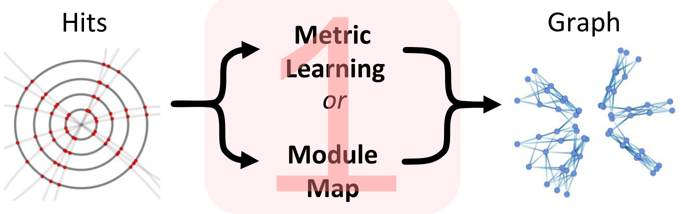


- $z_0 = z_{h1} - r_{h1} \times \left(\frac{\Delta z}{\Delta r}\right)$
- $\phi_{\text{slope}} = \frac{\Delta\phi}{\Delta r}$
- $\Delta\phi = \phi_{h2} - \phi_{h1}$
- $\Delta\eta = \eta_{h2} - \eta_{h1}$

- $z_0 = z_{h1} - r_{h1} \times \left(\frac{\Delta z}{\Delta r}\right)$
- $\phi_{\text{slope}} = \frac{\Delta\phi}{\Delta r}$
- $\Delta\phi = \phi_{h2} - \phi_{h1}$
- $\Delta\eta = \eta_{h2} - \eta_{h1}$

- $\Delta\frac{\Delta y}{\Delta x} = \frac{\Delta y_{12}}{\Delta x_{12}} - \frac{\Delta y_{23}}{\Delta x_{23}}$
- $\Delta\frac{\Delta z}{\Delta r} = \frac{\Delta z_{12}}{\Delta r_{12}} - \frac{\Delta z_{23}}{\Delta r_{23}}$

METRIC LEARNING



Graph Construction

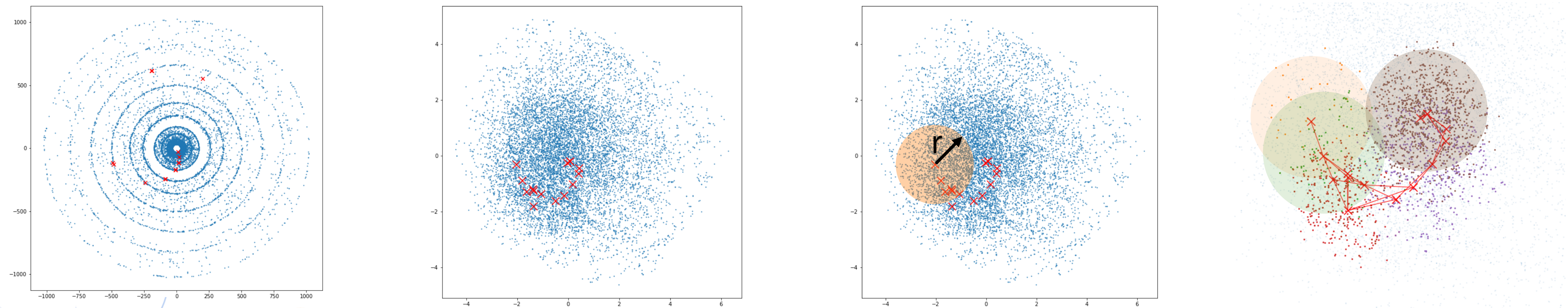
- **The idea:** Teach a multilayer perceptron (MLP) to embed spacepoint features (spatial and cell information)
- In this embedded space, all doublets in a given particle track are **trained to be near each other (Euclidean distance x)**, using a contrastive loss function L:
- A hit in a track is trained to be **closest** to its preceeding and succeeding track hits

$$L = \begin{cases} x, & \text{if true pair} \\ \max(0, r - x), & \text{if false pair} \end{cases}$$

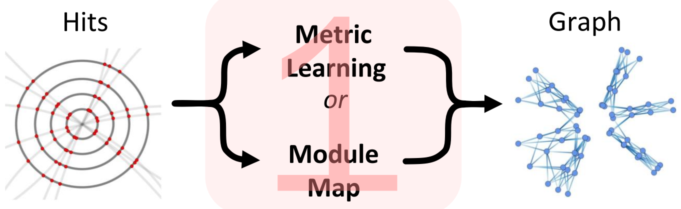
Embed into learned latent space

Connect all spacepoints within radius r

All spacepoint pairs joined into graph

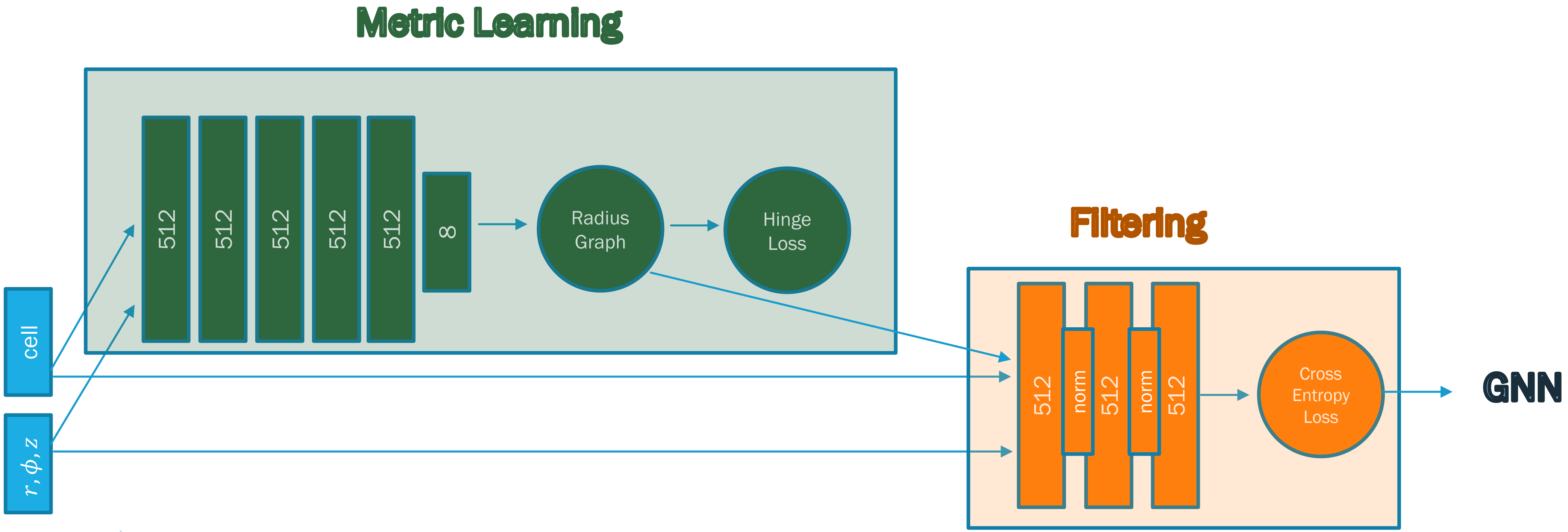


METRIC LEARNING - FILTERING

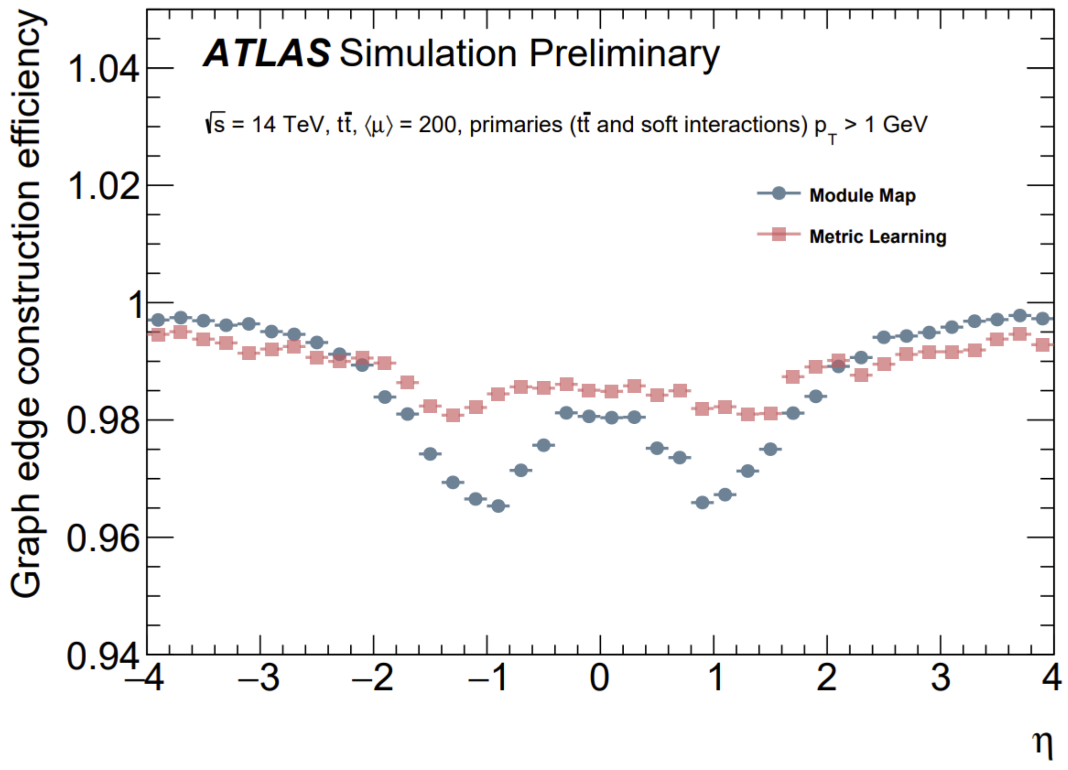


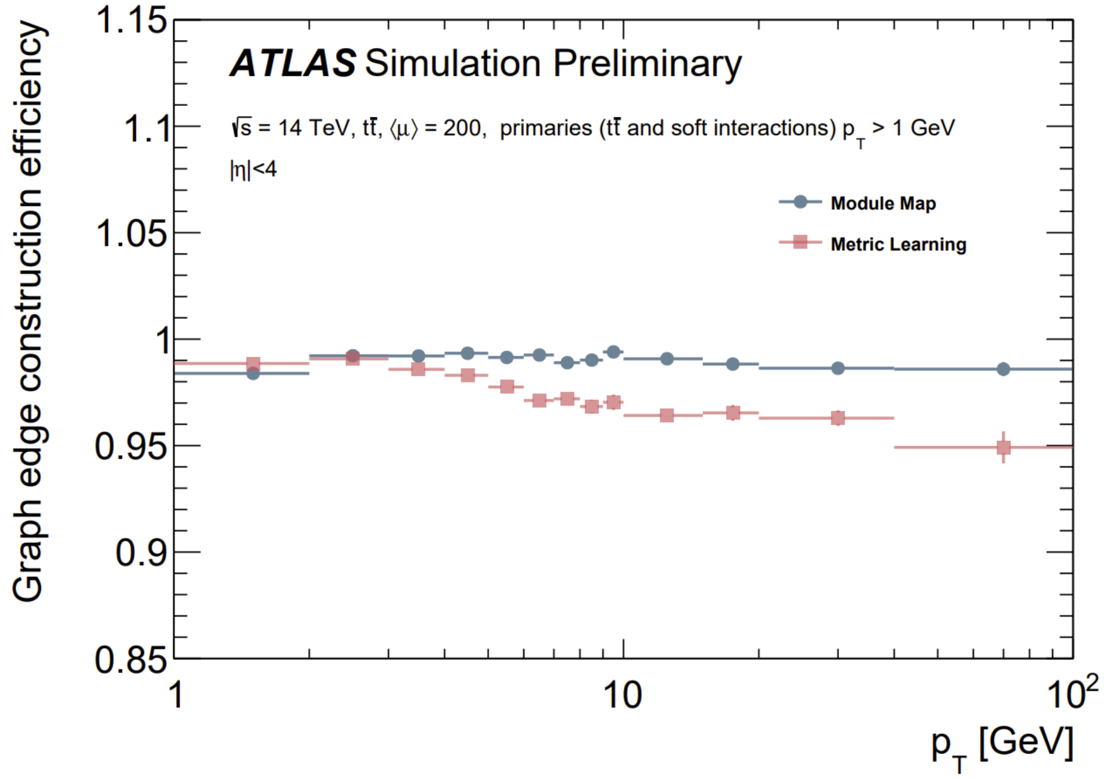
Graph Construction

- Output graph of metric learning is impure: 0.2%
- Can pass edges through a simple MLP filter to filter out the easy fakes
- Improves purity to 2%, so graph can be trained entirely on a single GPU



GRAPH CONSTRUCTION RESULTS





- Drop in efficiency at low η due to poor barrel strip resolution (can discuss further!)
- Drop in efficiency at high p_T due to low training statistics

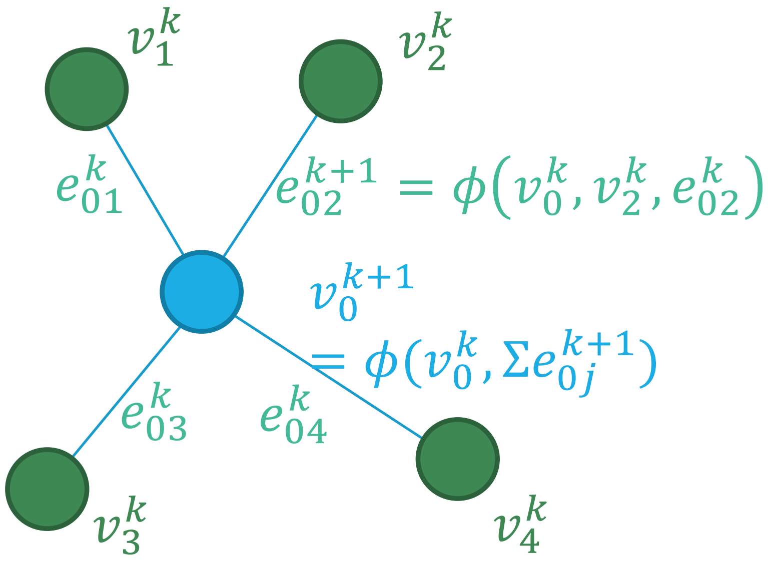
EDGE CLASSIFICATION WITH GRAPH NEURAL NETWORK

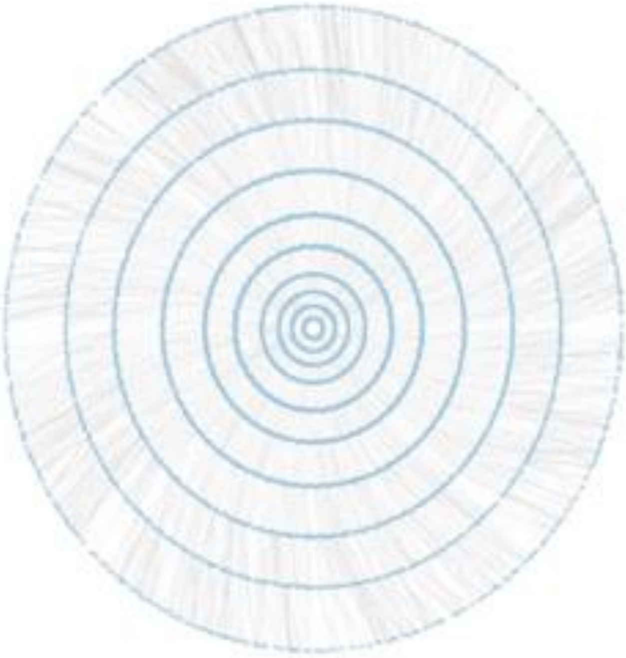
1. Node features (spatial position) are encoded
2. Encoded features are concatenated and encoded to create edge features
3. Edge features are aggregated around nodes to create next round of encoded node features (i.e. message passing)
4. Each iteration of message passing improves discrimination power

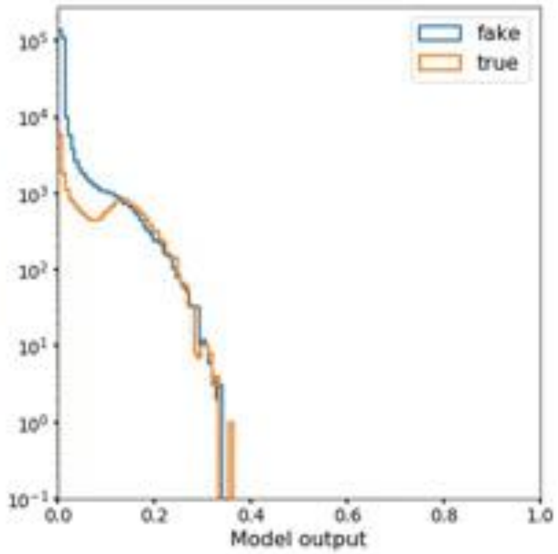
v_i^k node features
e_{ij}^k edge features
at iteration k

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." 2016.

INTERACTION NETWORK

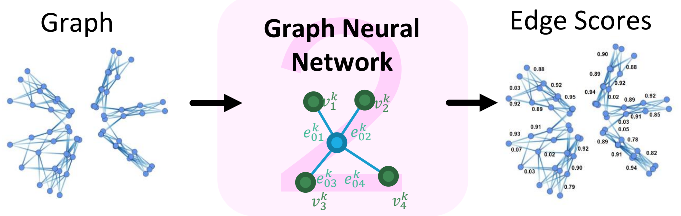






TrkX
BERKELEY LAB

TRAINING CHALLENGES & SOLUTIONS



Edge Labeling

Memory requirements

- Challenge: Training graphs are very large – O(1m) edges
- Solution A: Gradient checkpointing
- Solution B: Model offloading

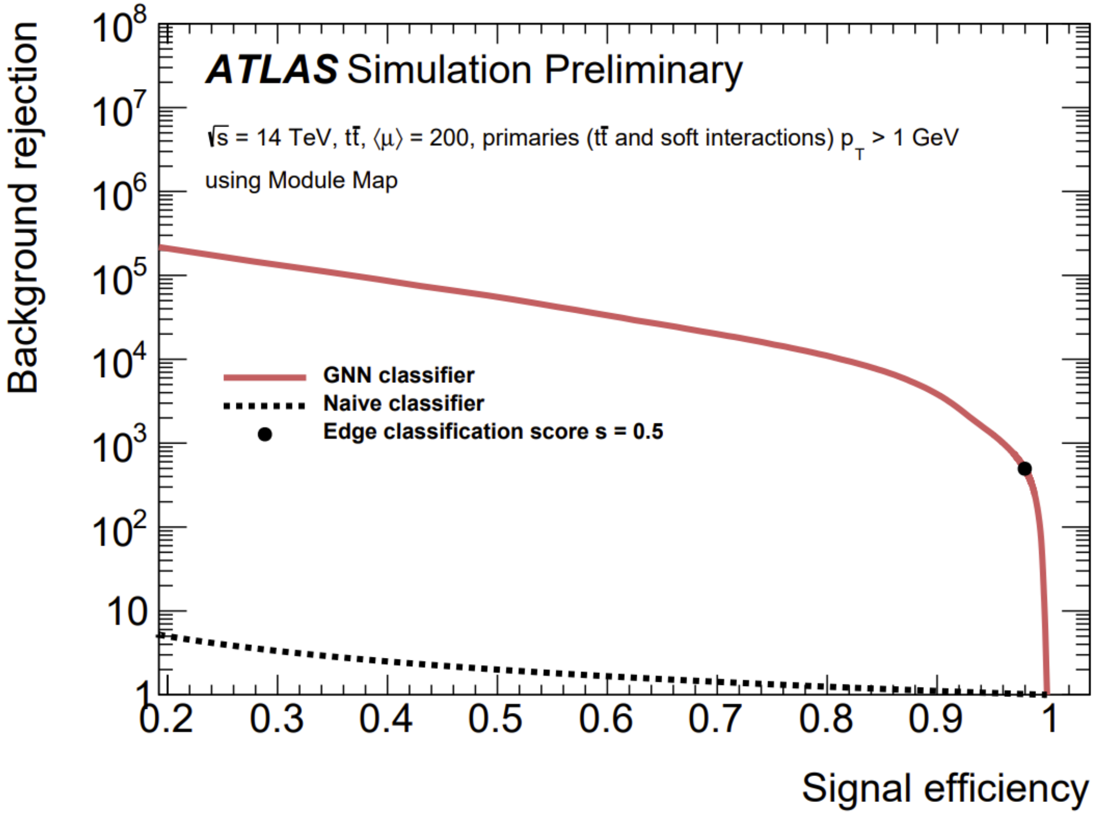
(Can discuss these further if interested)

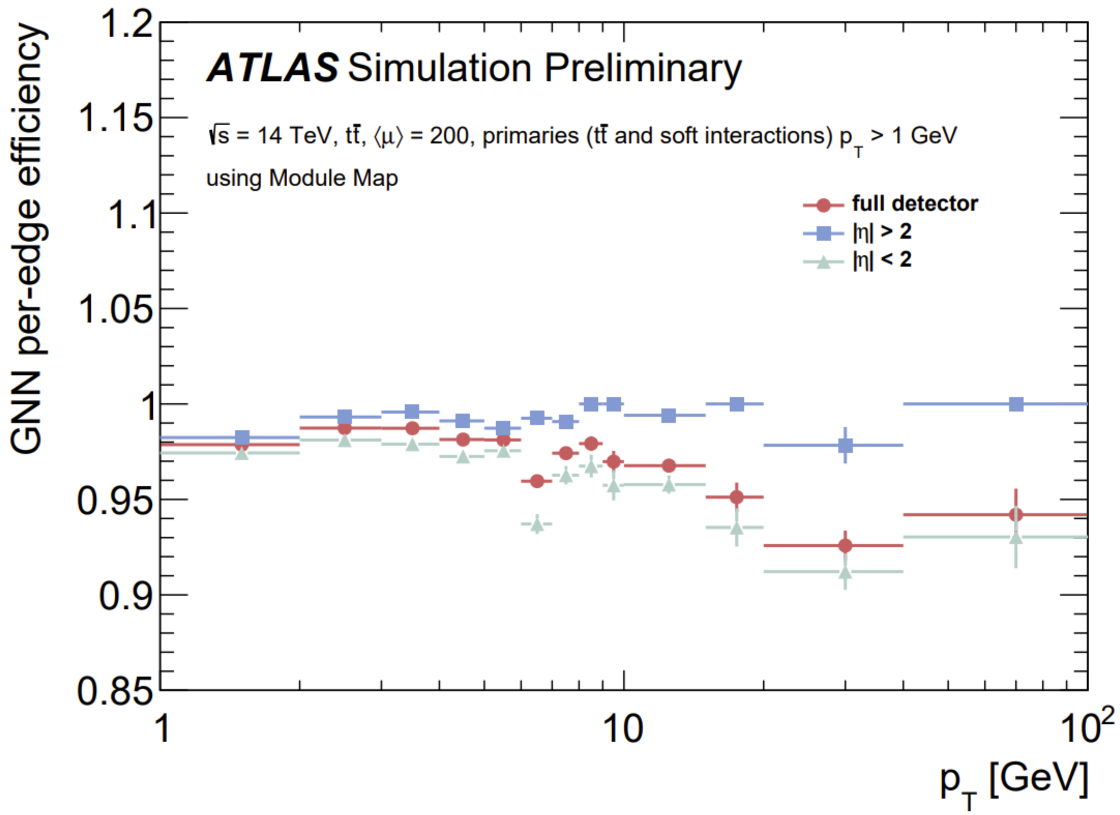
Loss masking and balancing

- Challenge: Target vs. background edges are highly imbalanced (1:100)
- Challenge: Non-target edges are not all equally “wrong” – don’t want to confuse GNN
- Solution: Weight target edges up by x10 and mask out sequential non-target edges

GNN EDGE CLASSIFICATION RESULTS

ROC CURVE & EDGEWISE PERFORMANCE VS. p_T

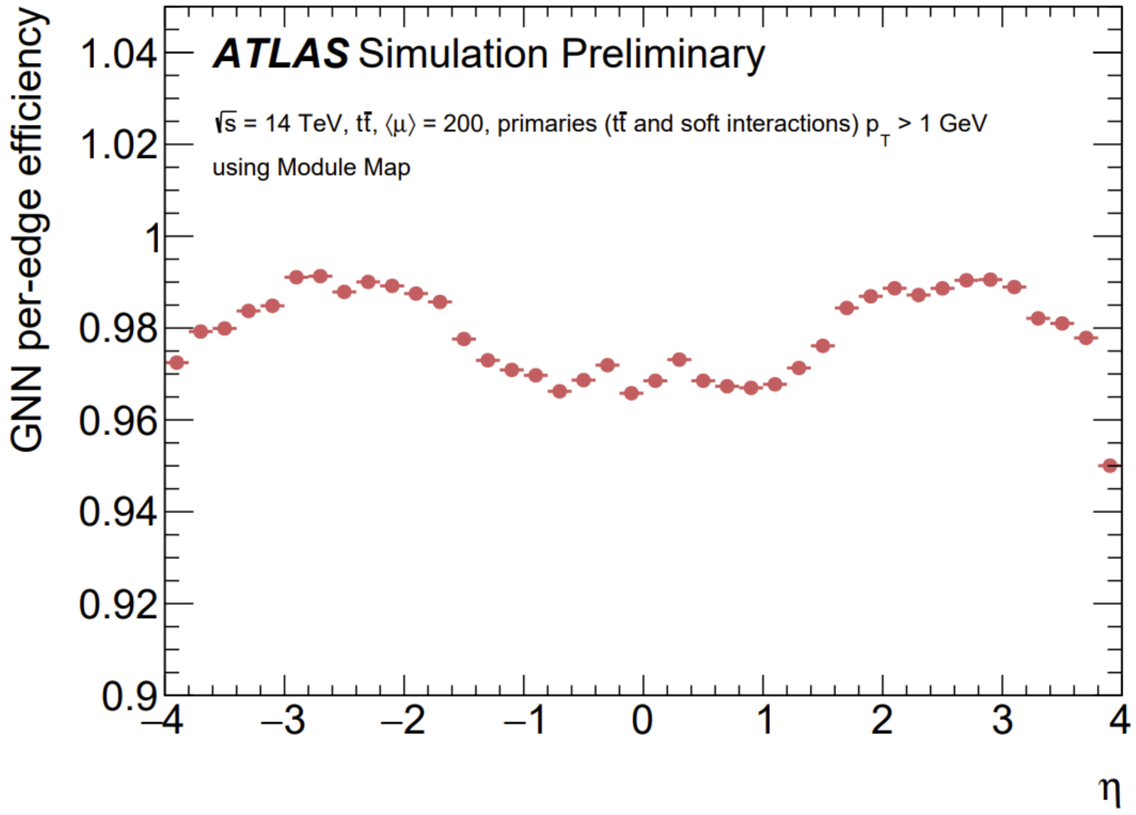


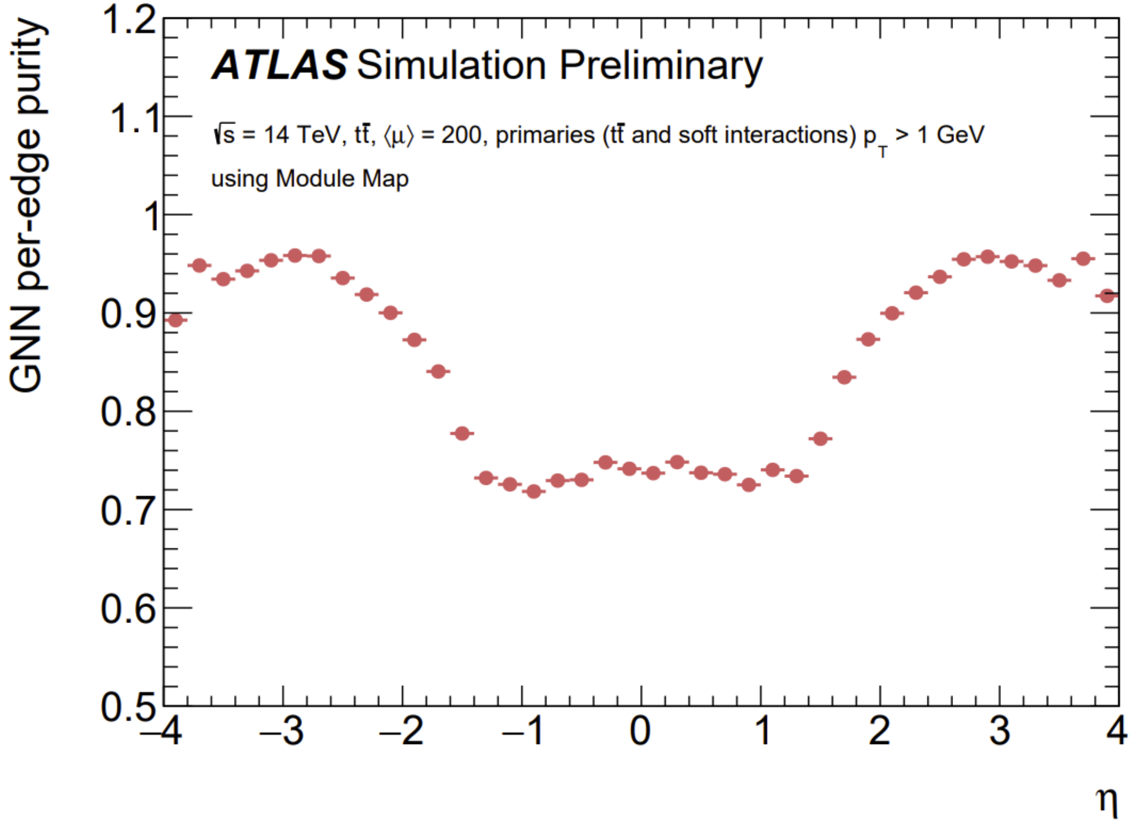


- Edge cut of 0.5 on output of GNN edge classifier

GNN EDGE CLASSIFICATION RESULTS

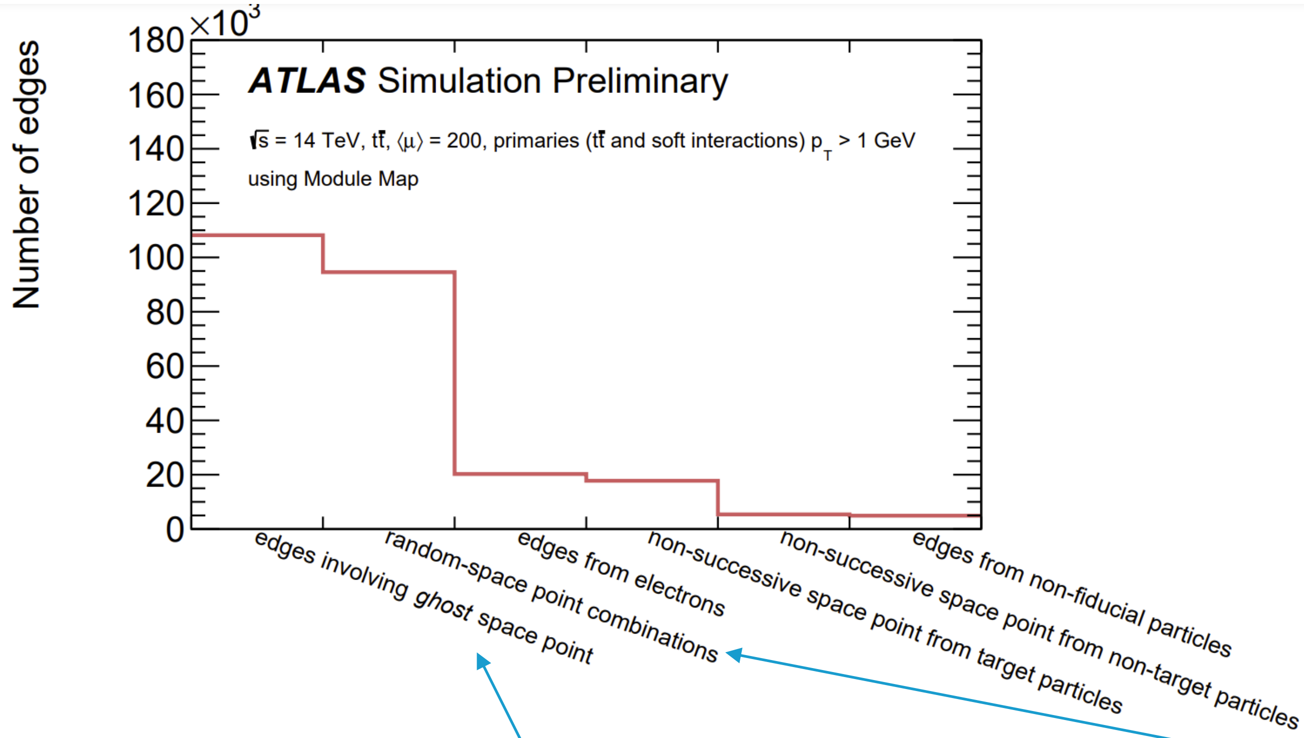
EDGEWISE PERFORMANCE VS. η

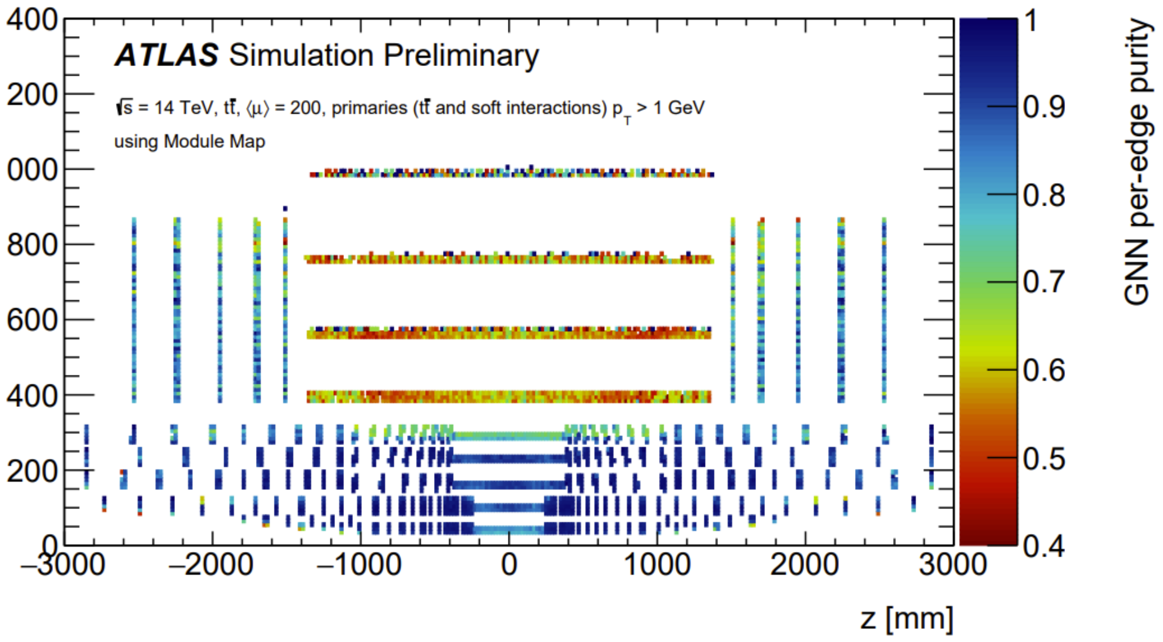


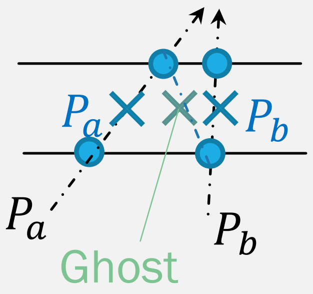


- Again, see a drop in performance at low η

BARREL STRIP MISCLASSIFICATION

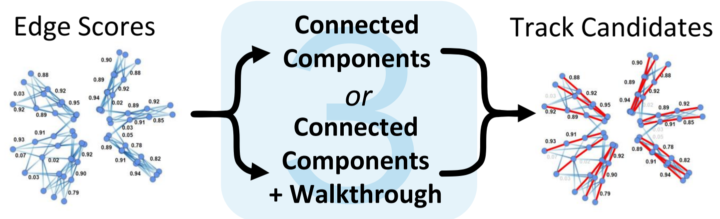






Edges between SP from particle A, and a ghost SP of clusters from particle A and particle B. I.e. The GNN is “right”, the construction is “wrong”

Edges between SP from particle A and particle B. i.e. The GNN is “wrong”



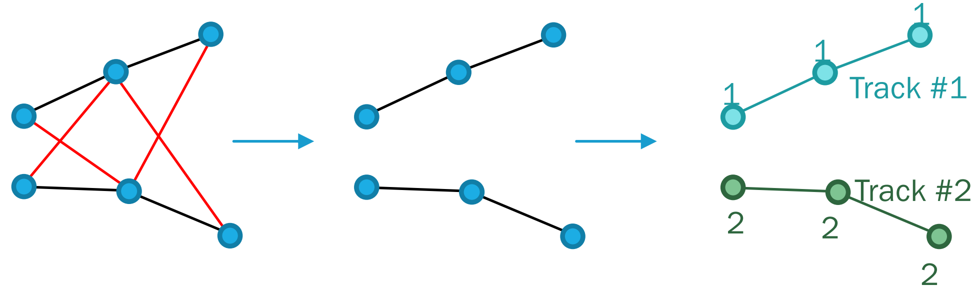
Graph Segmentation

TRACK CANDIDATES CONSTRUCTION

- We now have labelled edges. Want to now label each *node* depending on connectivity. A node is allowed to have more than one label.
- Two distinct approaches: **component-based** segmentation, or **path-based** segmentation.

Component-based

E.g. connected components algorithm:

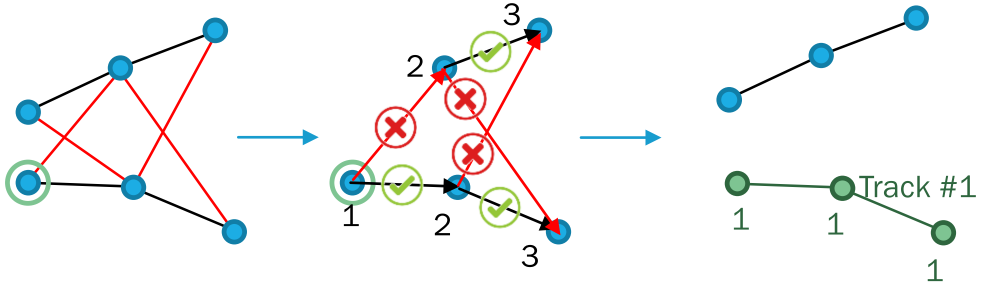


Classified edges | Ignore cut edges | Label connected components

- Pros: Fast, embarrassingly parallelizable
- Cons: Easily merges tracks into one candidate

Path-based

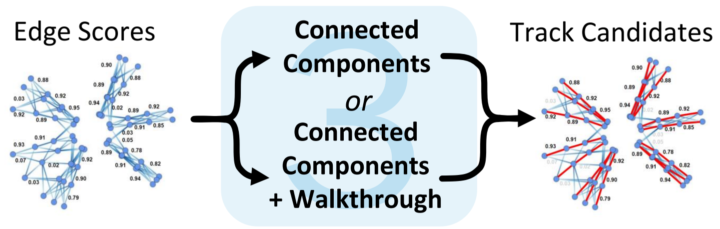
E.g. walkthrough algorithm:



Classified edges, Starting node | Choose high score junctions | Remove a high-scoring path

- Pros: Handles hits as a sequence, as a track should be
- Cons: May not parallelize well, depending on algorithm, needs a *directed* graph

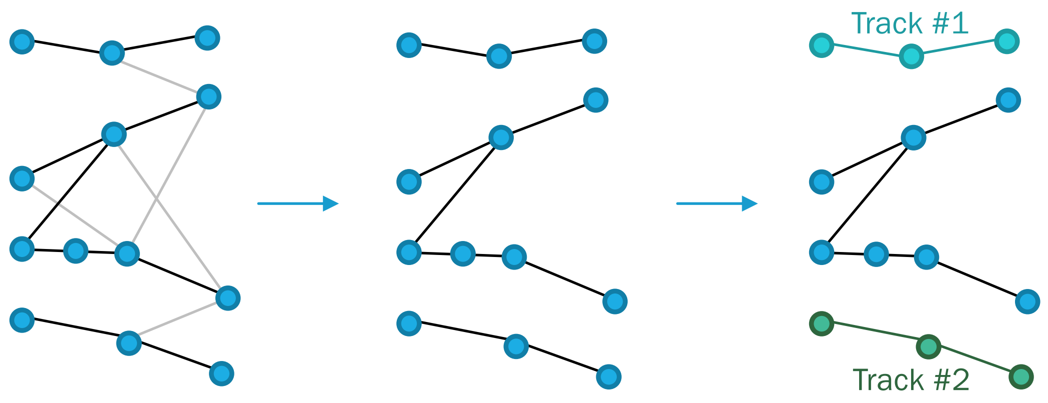
TRACK CANDIDATES CONSTRUCTION



Graph Segmentation

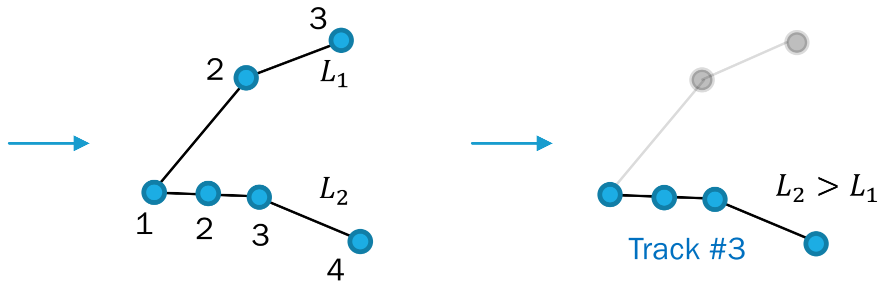
- Our specific algorithm combines the good features of each approach:

1. Connected Components



Classified edges | Cut score < 0.2 | Label simple candidates

2. Walkthrough, a.k.a "Wrangler"

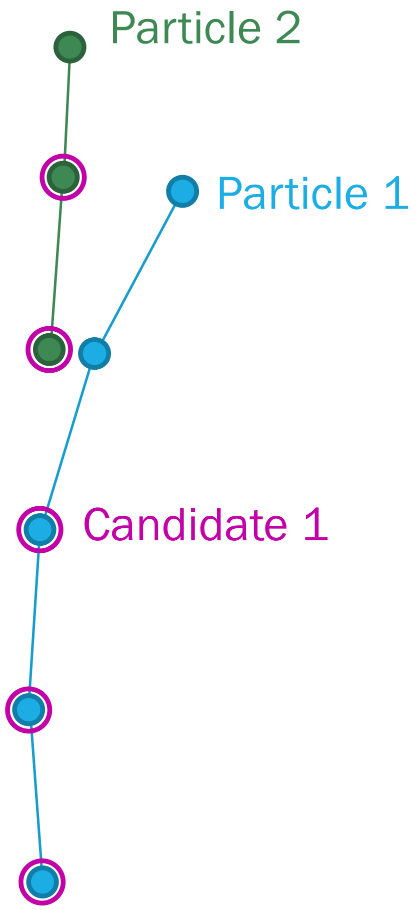


Walk through paths, Counting length L | Assign longest path as candidate

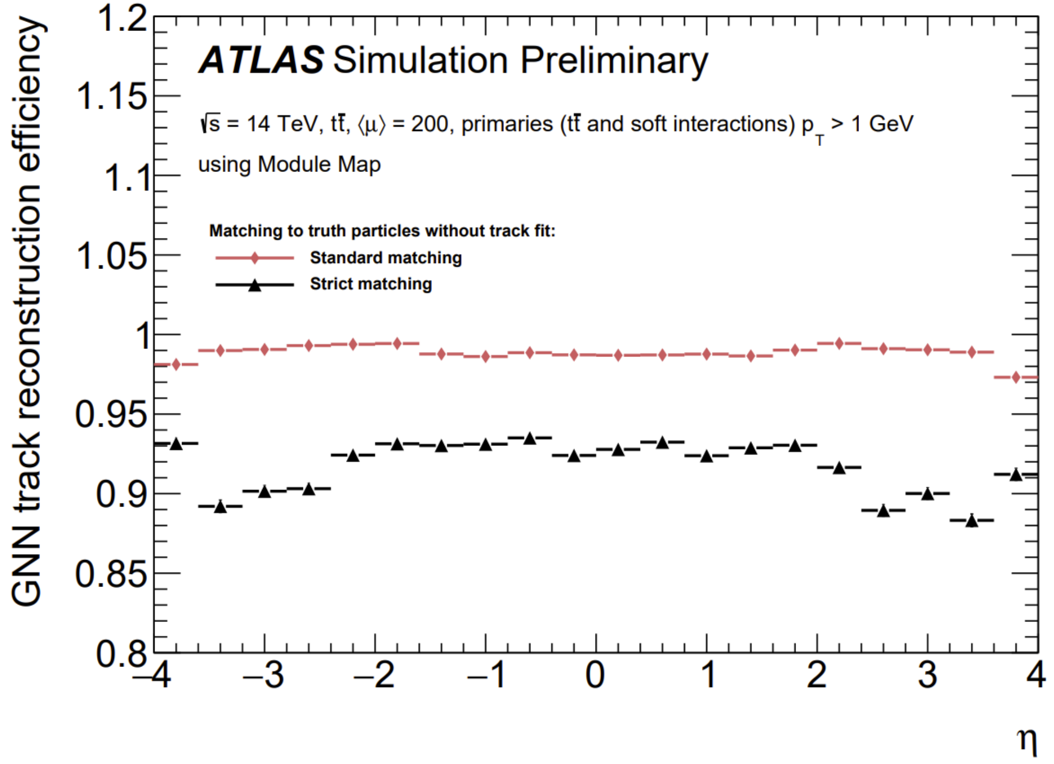
TRACK MATCHING DEFINITIONS

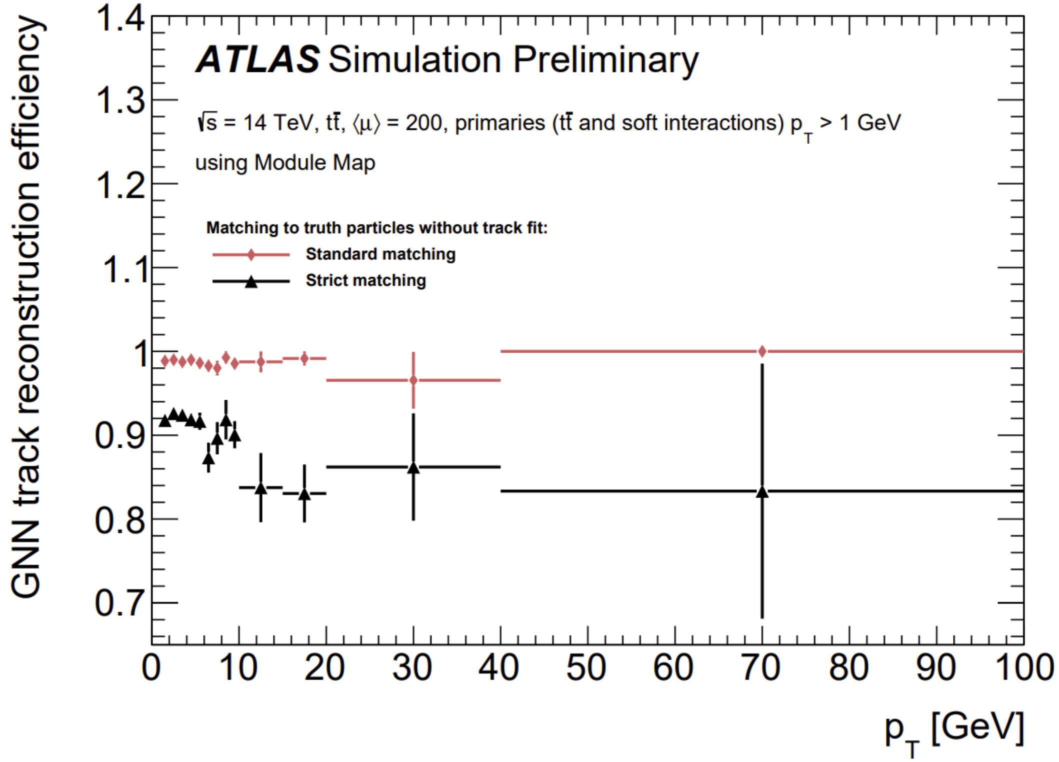
- $N(P_i, C_j)$ is the number of spacepoints shared by particle i and candidate j
- Particle i is called "matched" if, for some j, $\frac{N(P_i,C_j)}{N(P_i)} > f_{truth}$
- Candidate j is called "matched" if, for some i, $\frac{N(P_i,C_j)}{N(C_j)} > f_{reco}$
- Particle i and candidate j are called "double matched" if, for some i and j, $\frac{N(P_i,C_j)}{N(P_i)} > f_{truth}$ and $\frac{N(P_i,C_j)}{N(C_j)} > f_{reco}$
- $eff = \frac{\sum_i P_i(matching\ condition)}{\sum_i P_i}$, $pur = \frac{\sum_j C_j(matching\ condition)}{\sum_j C_j}$

Standard matching: single-matched particles with $f_{truth} = 0.5$
Strict matching: double-matched particles with $f_{reco} = 1.0$



TRACK RECONSTRUCTION RESULTS



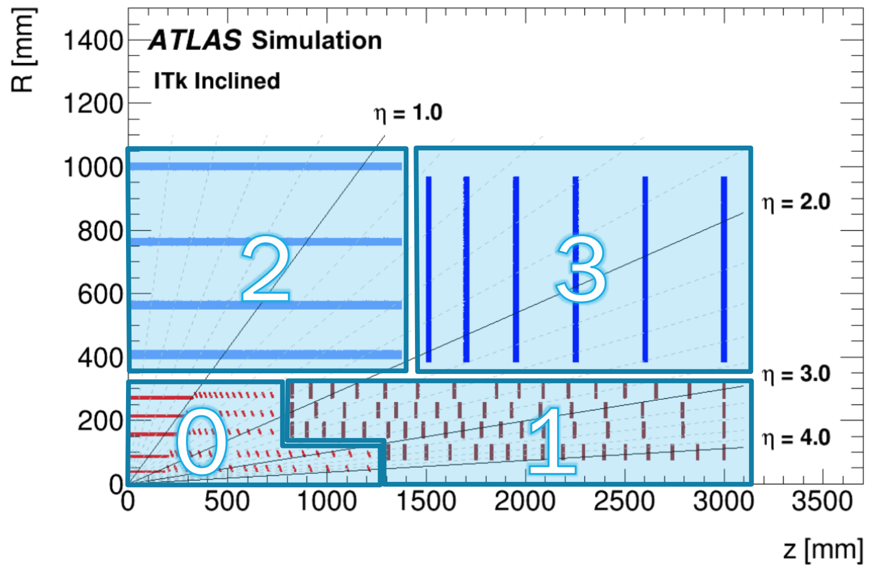


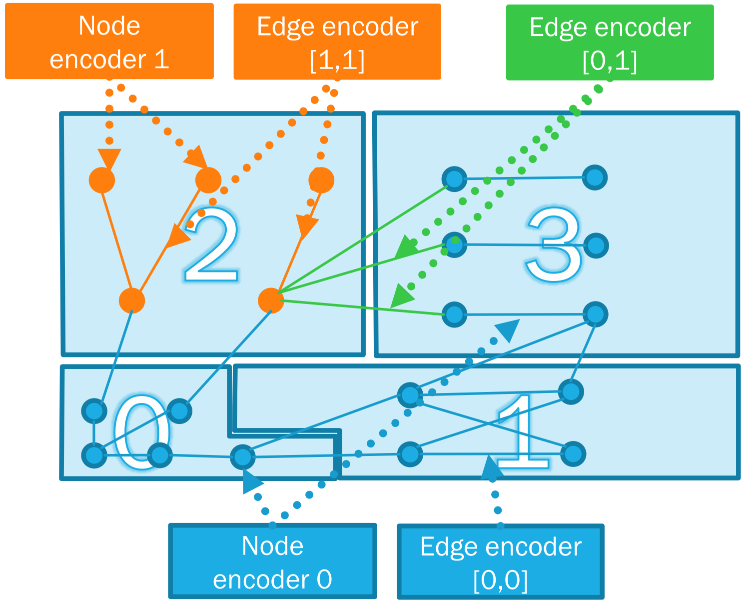
Standard matching: single-matched particles with $f_{truth} = 0.5$
Strict matching: double-matched particles with $f_{reco} = 1.0$

- Fake rate is $O(10^{-3})$, and duplicate rate is $O(10^{-3})$, for the standard matching

ONGOING WORK

- Extending TrackML inference timing and scaling studies to ITk dataset
- Incorporating strip cluster features into heterogeneous graph construction and GNN classification – already showing significant boost in performance
- Investigating training and inference performance on lower p_T tracks (i.e. $< 1GeV$) and high p_T tracks (i.e. $> 1GeV$)
- Investigating performance on large radius tracks and dense track environments
- Incorporating pipeline into ACTS (done!) and Athena reconstruction chain – for direct comparison with CKF, and to study track parameter resolution





CONCLUSION

- Produced first public results on official ATLAS ITk geometry using GNN-based track reconstruction pipeline
- Promising reconstruction performance, well-positioned for comparison with traditional algorithms

THANKS! AND…

Please tune into the “mini-workshop” for GNNs in HEP, co-located with Connecting the Dots.
Consider submitting an abstract to show some ongoing work, to
ctd.gnn.workshop@gmail.com (by Monday!)