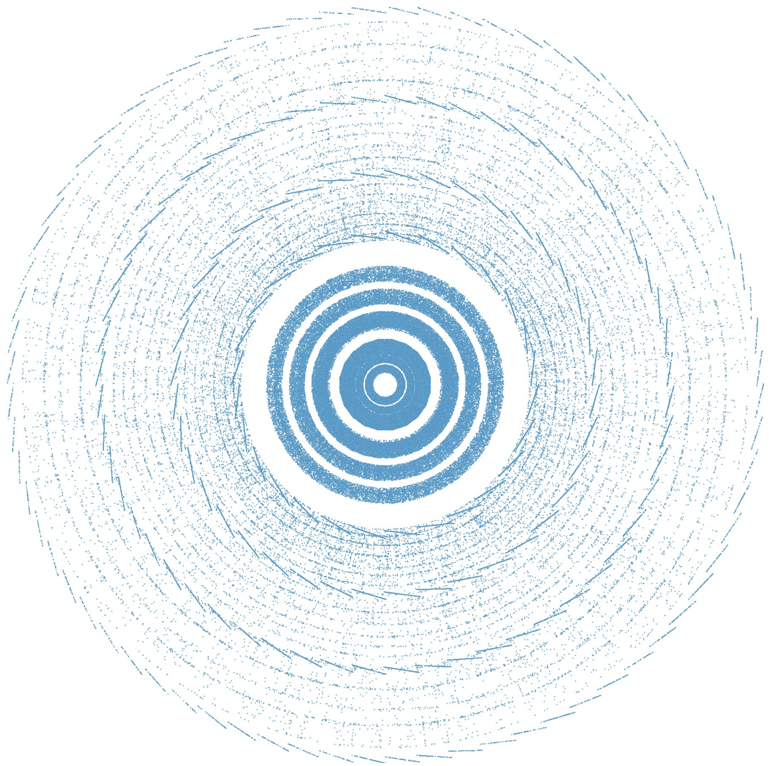
A deep learning framework for charged-particle track reconstruction in the ATLAS ITk

MINH-TUAN PHAM ON BEHALF OF THE GNN4ITK PROJECT

The problem of tracking

- In a silicon detector, charged particles leave an energy deposits (hits). Track reconstruction assigns each hit to a track.
- Very CPU-intensive and challenging to deal with increased pile-up in HL-LHC => Seek GPU-powered solutions.
- Study Graph Neural Network (GNN) approach to track reconstruction, using $t\bar{t}$ events simulated with $\langle\mu\rangle = 200.$ Compare the performance with that of the current algorithm under several physics metrics.

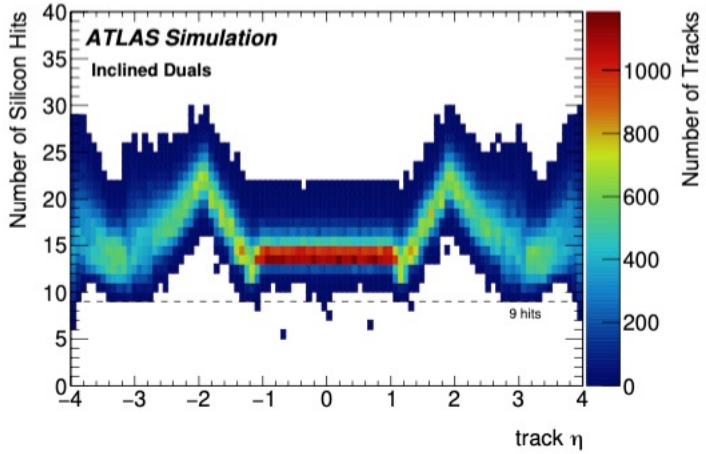


Detector	$\langle\mu\rangle$	inner tracking	muon spectrometer and calorimeter	combined reconstruction	monitoring	total
Run 2	90	1137	149	301	106	1693

https://cds.cern.ch/record/2729668/files/LHCC-G-178.pdf

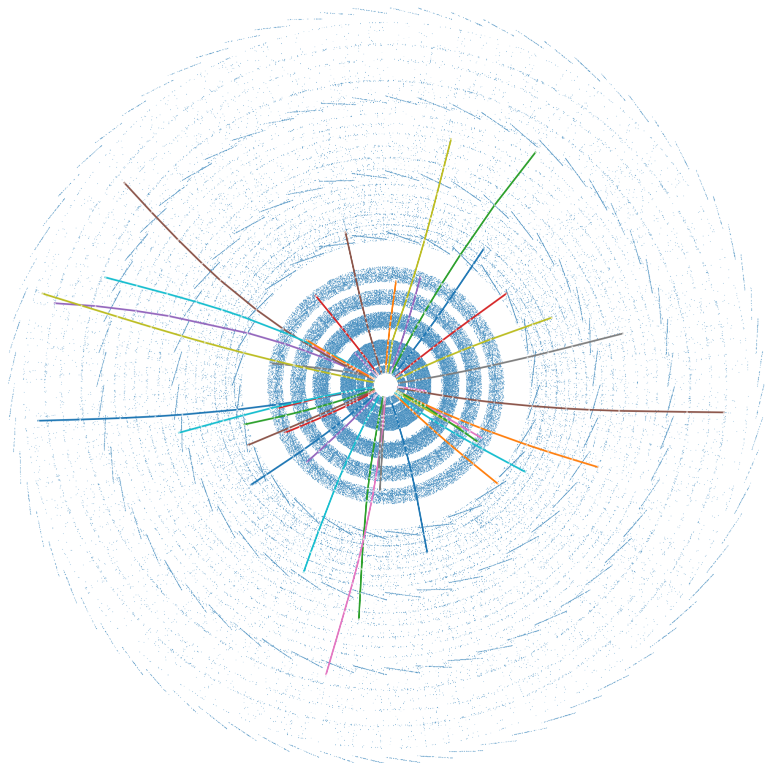
ATLAS GNN tagger takes 2 overall jet features, and *21 track features*
ATL-PHYS-SLIDE-2023-048

Jet Input	Description
p_T	Jet transverse momentum
η	Signed jet pseudorapidity
Track Input	**Description**
q/p	Track charge divided by momentum (measure of curvature)
$d\eta$	Pseudorapidity of the track, relative to the jet η
$d\phi$	Azimuthal angle of the track, relative to the jet ϕ
d_0	Closest distance from the track to the PV in the longitudinal plane
$z_0 \sin\theta$	Closest distance from the track to the PV in the transverse plane



The problem of tracking

- In a silicon detector, charged particles leave an energy deposits (hits). Track reconstruction assigns each hit to a track.
- Very CPU-intensive and challenging to deal with increased pile-up in HL-LHC => Seek GPU-powered solutions.
- Study Graph Neural Network (GNN) approach to track reconstruction, using $t\bar{t}$ events simulated with $\langle \mu \rangle = 200.$ Compare the performance with that of the current algorithm under several physics metrics.

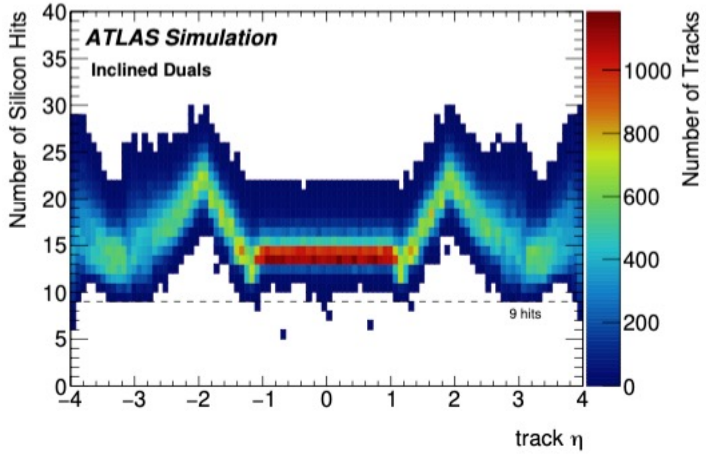


Detector	$\langle \mu \rangle$	inner tracking	muon spectrometer and calorimeter	combined reconstruction	monitoring	total
Run 2	90	1137	149	301	106	1693

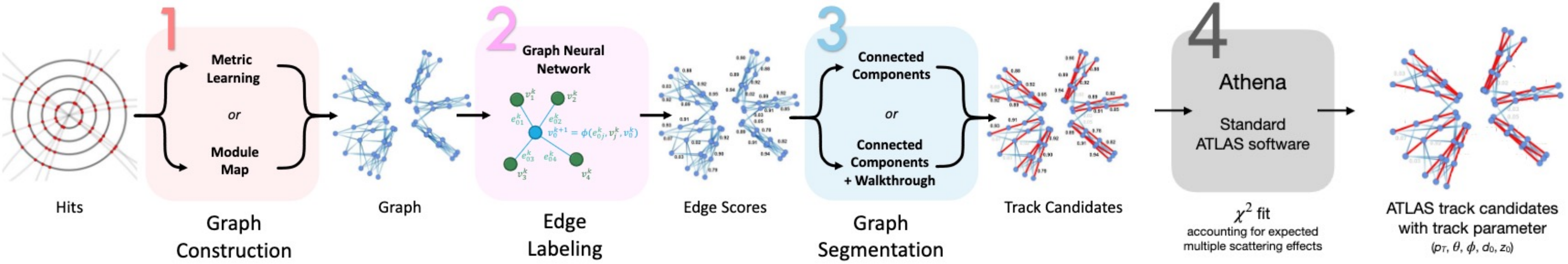
https://cds.cern.ch/record/2729668/files/LHCC-G-178.pdf

ATLAS GNN tagger takes 2 overall jet features, and *21 track features*
ATL-PHYS-SLIDE-2023-048

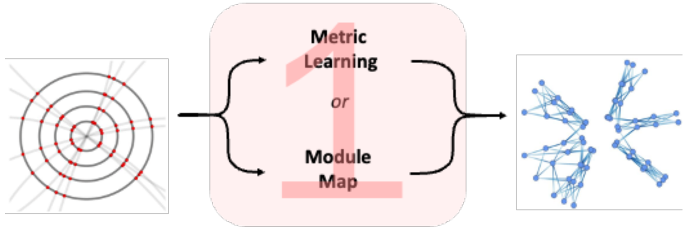
Jet Input	Description
p_T	Jet transverse momentum
η	Signed jet pseudorapidity
Track Input	**Description**
q/p	Track charge divided by momentum (measure of curvature)
$d\eta$	Pseudorapidity of the track, relative to the jet η
$d\phi$	Azimuthal angle of the track, relative to the jet ϕ
d_0	Closest distance from the track to the PV in the longitudinal plane
$z_0 \sin\theta$	Closest distance from the track to the PV in the transverse plane



Overview of the GNN4ITk pipeline



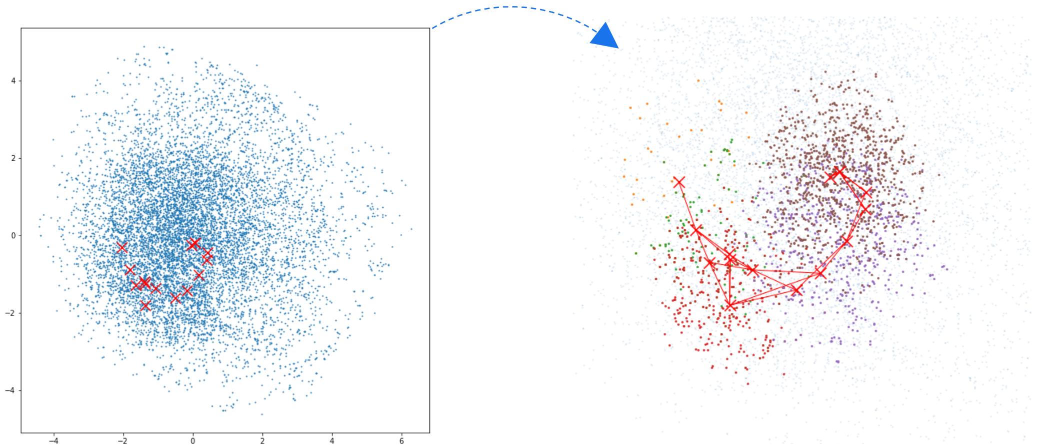
- Represent a pp-collision event as a graph. Treat each **hit** as a **node**, and each **edge** connecting 2 nodes as a hypothesis that they are 2 consecutive hits on a particle track.
- Classify edges using a Graph Neural Network (GNN), then segment the graph to build track candidates.
- Git repo, documentation (WIP).



Graph construction

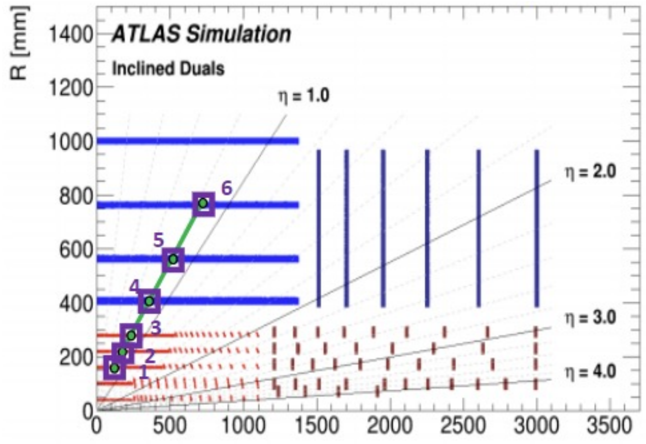
Machine learning approach: Metric Learning

- Train a DNN to project hits to an embedding space, such that Hits from the same particles are near each other by L_2-distance. Constructs graphs using kNN.
- Clean up easy fake edges by a DNN or a shallow GNN to reduce graph size and fit on GPU.



Data driven approach: Module Map

- Build a map of detector modules, where a triplet of hits ABC means at least 1 true track has passed sequentially through A, B, and C.
- Register a triplet ABC if all 3 modules get hit in the event.

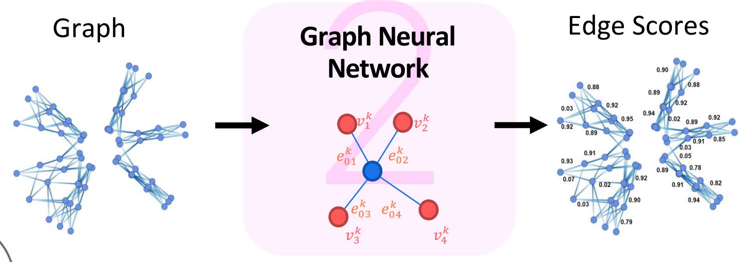


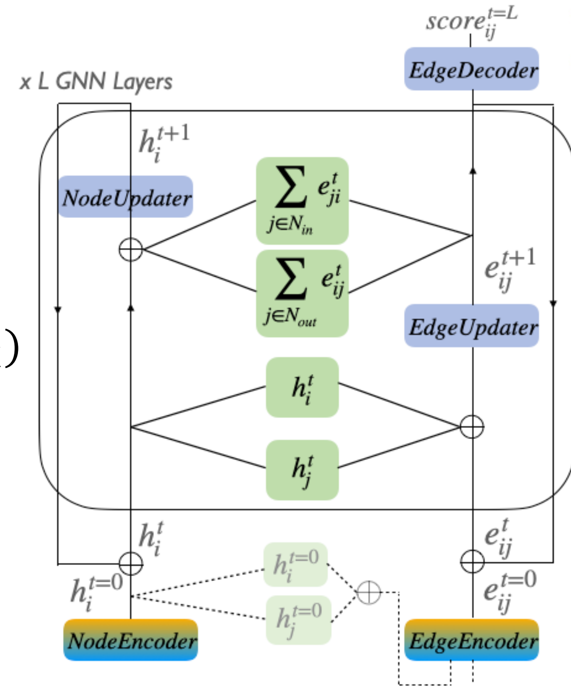
Connections added:

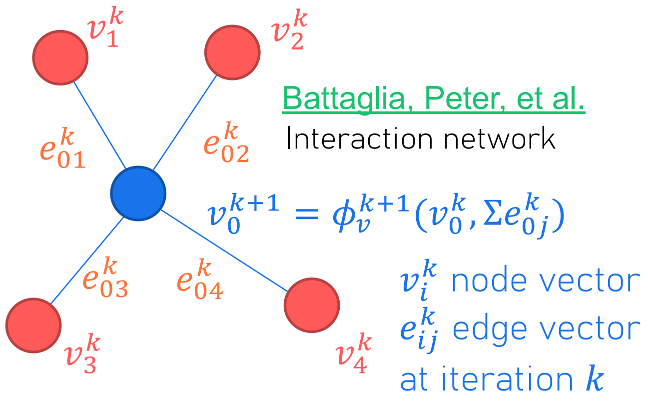
1 -> 2 -> 3
2 -> 3 -> 4
3 -> 4 -> 5
4 -> 5 -> 6

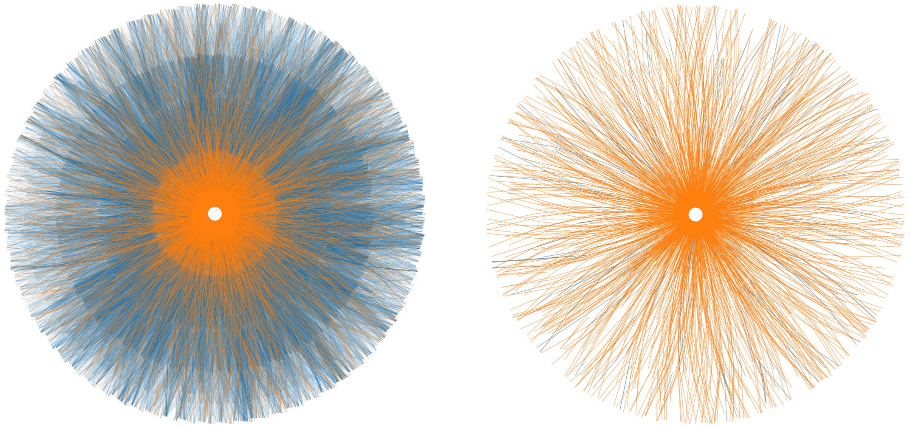
GNN edge classification

1. Encode nodes features (position, charge count, local measurements, etc.) to a latent node vector $v_i^0 = \phi_v(x_i)$
2. Concatenate node vectors of two hits connected by an edge and encode to edge vector, $e_{ij}^0 = \phi_e(v_i^0, v_j^0)$
3. **Aggregate edge vectors, acting as messages between nodes, $m_i^0 = \sum_j e_{ij}^0$**
4. **Update node features using aggregated message, $v_i^1 = \psi_v^1(v_i^0, m_i^0)$. Update edge features using updated node features, $e_{ij}^1 = \psi_e^1(v_i^1, v_j^1, e_{ij}^0)$.**
5. Repeat steps 3 and 4 $\boldsymbol{n = 8}$ times.
6. Compute an edge score representing the probability of being a true edge, $s_{ij} = \psi_d(e_{ij}^n)$

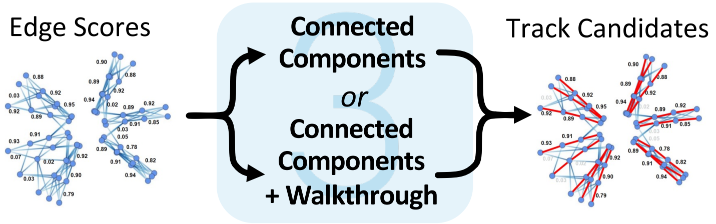






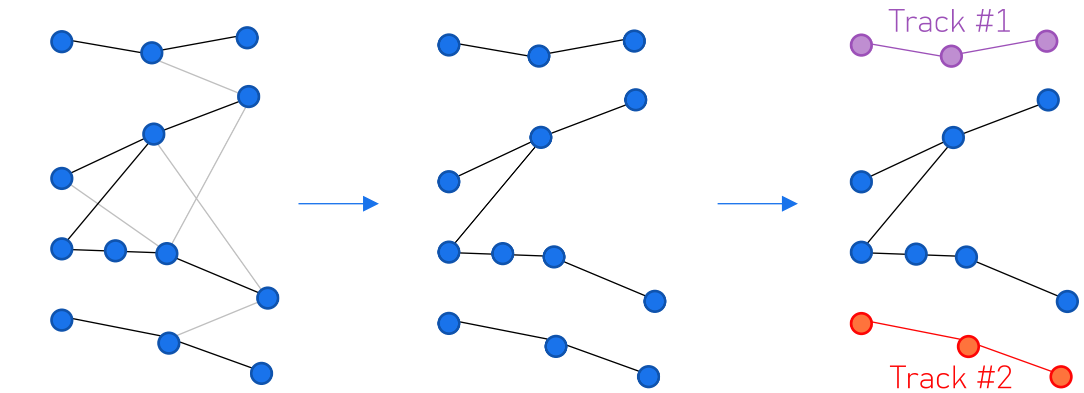


Input graph (left) and classified graph (right). Fake = blue. True = orange



Track construction

1. Connected Components

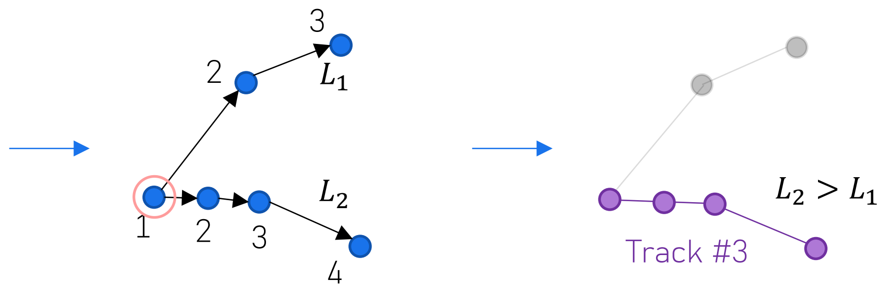


Classified edges

Loose score cut

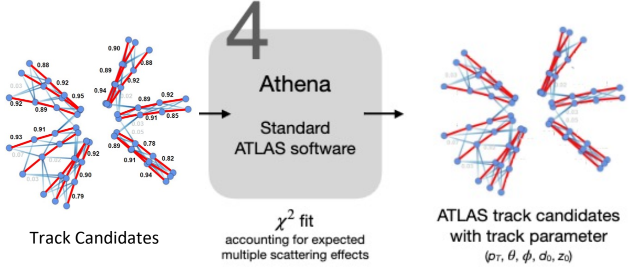
Label simple candidates

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



Walk through paths from starting node, count length L

Assign longest path as candidate



Physics performance of the GNN4ITk pipeline

- Perform a global χ^2 fit on GNN track candidates. Evaluate the performance and compare to that of tracks found by the CKF.
- GNN tracks are selected using ATLAS requirements, with some selection cuts loosen.

Requirements	Pseudorapidity interval		
	$\|\eta\| < 2.0$	$2.0 < \|\eta\| < 2.6$	$2.6 < \|\eta\| < 4.0$
pixel + strip hits	≥ 9	≥ 8	≥ 7
pixel hits	≥ 1	≥ 1	≥ 1
holes	≤ 2	≤ 2	≤ 2
p_T [MeV]	> 1000		
$\|d_0\|$ [mm]	≤ 2.0	≤ 2.0	≤ 10.0
$\|z_0\|$ [cm]	≤ 20.0	≤ 20.0	≤ 20.0

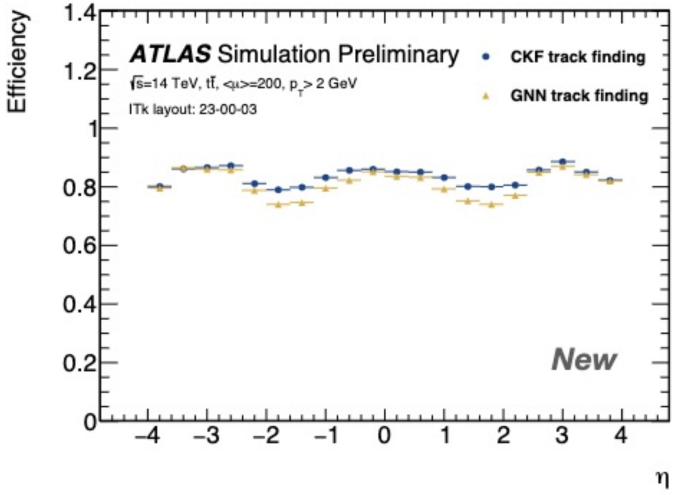
For GNN4ITk 3 cuts are looser:
pixel + strip hits ≥ 8,
$|d_0| < 20$ mm
$|z_0| < 25$ cm

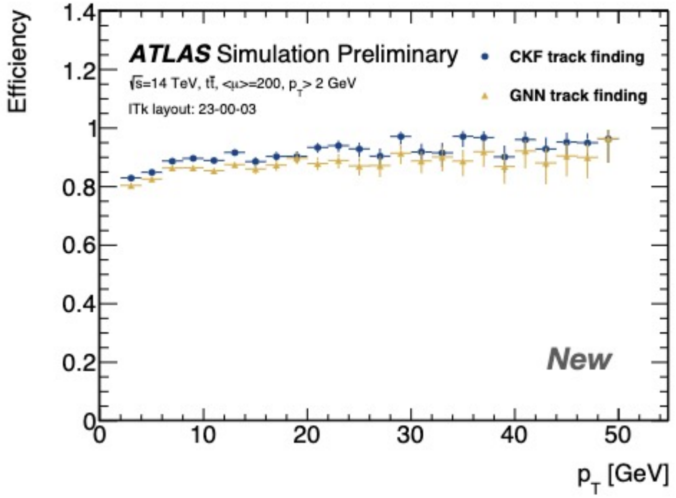
ATL-PHYS-PUB-2021-024

Tracking efficiency

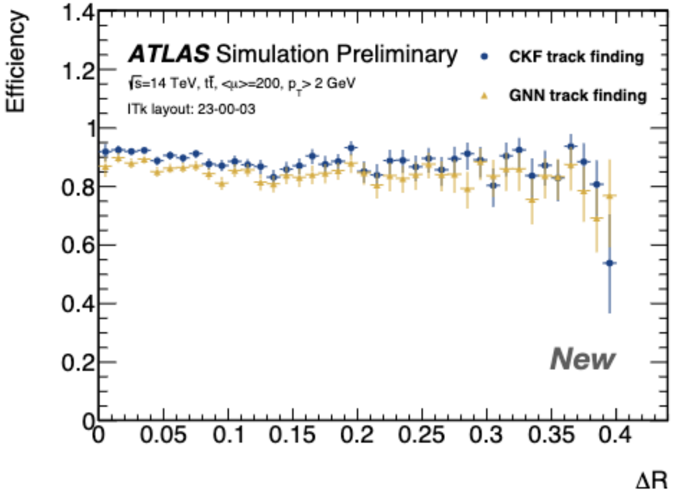
- Similarly efficient to the CKF in the central and forward region.
- Most inefficiency is located near $|\eta| \approx 1.8$. Overall competitive performance.
- Tracking efficiency inside jets approaching the CKF, with relatively uniform level over ΔR (distance to jet center) and jet p_T.

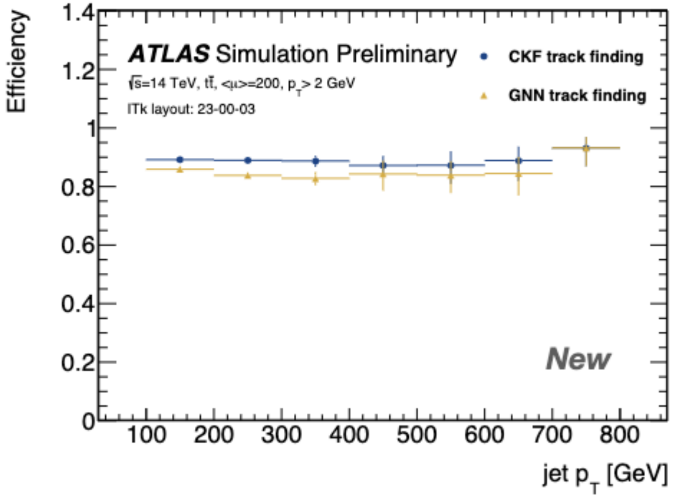
Tracking efficiency in ttbar events





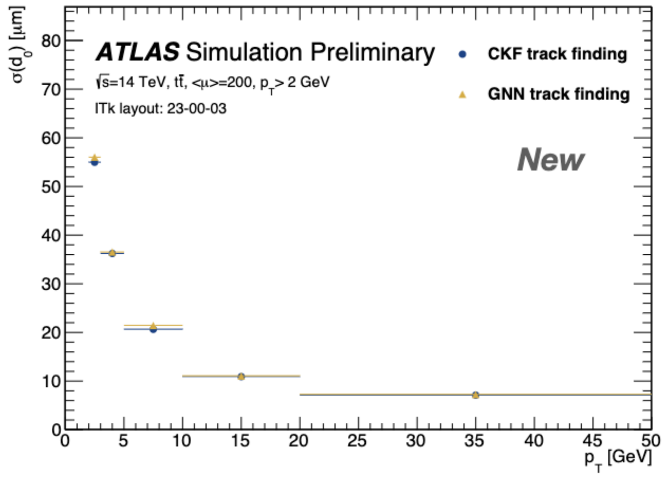
Tracking efficiency in jets

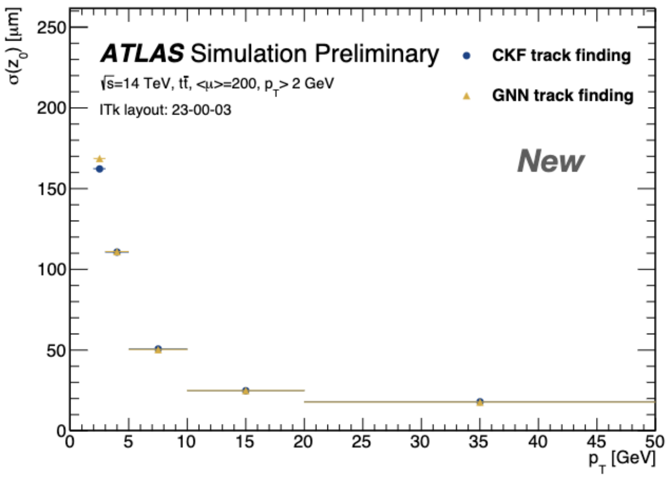


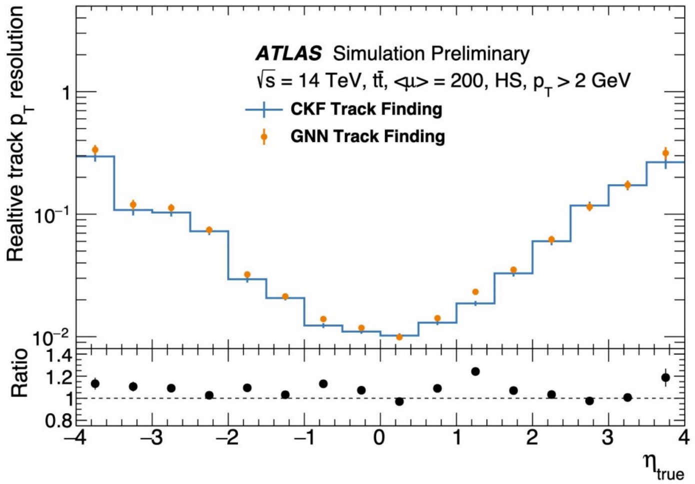


Track parameter resolution

- Good impact parameter resolution ($\sigma(d_0)$, $\sigma(z_0)$), comparable with CKF.
- Achieve p_T resolution compatible with the CKF.





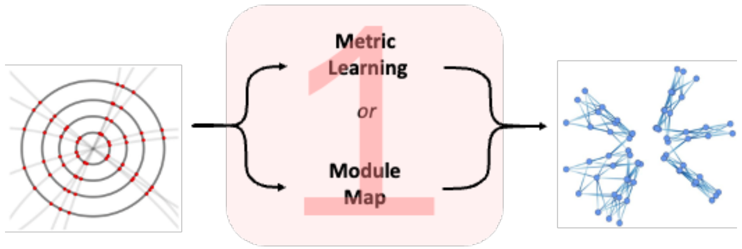


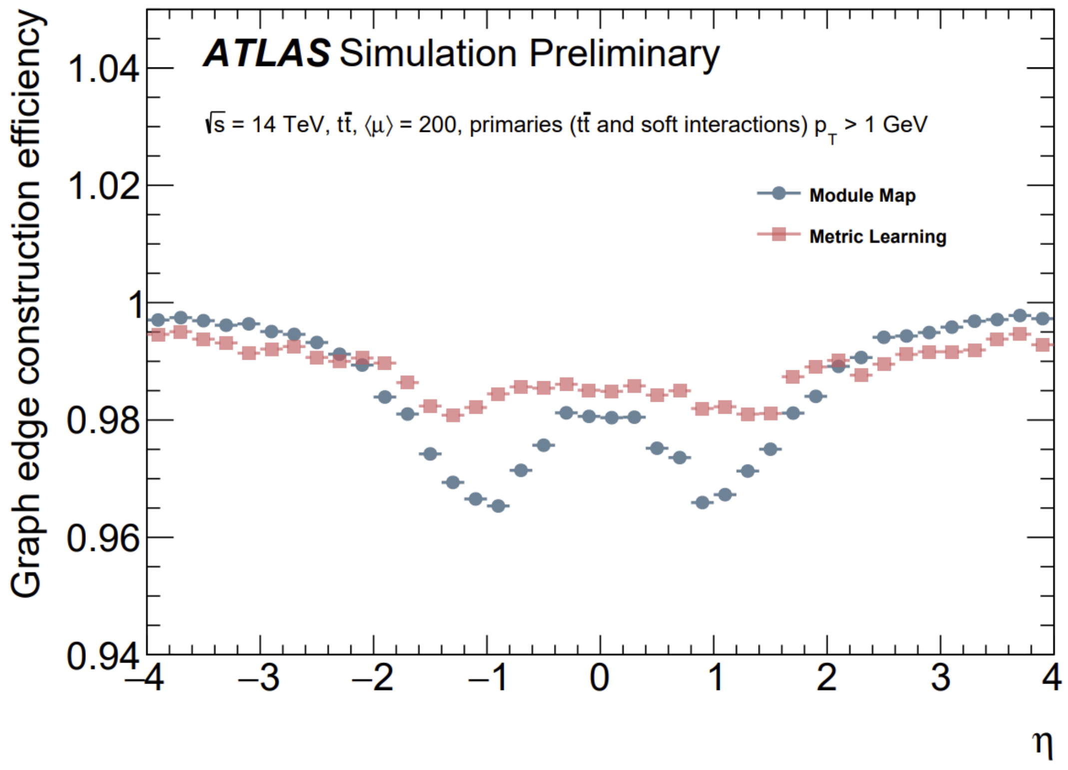
Summary

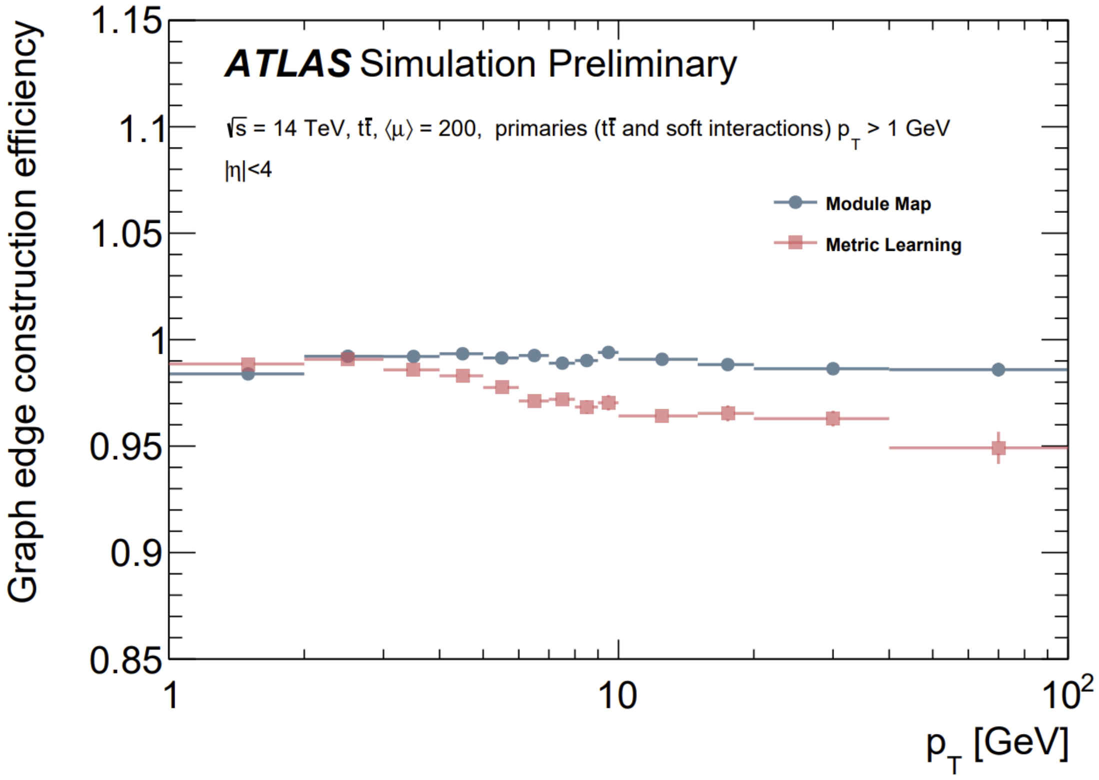
- We present the first ML pipeline for charged-particle track reconstruct from detector hits.
- Demonstrate a competitive physics performance of this pipeline on data simulated under $\langle\mu\rangle = 200$ condition and the latest detector geometry, in comparison with the current algorithm.
- Future work:
  - Machine learning optimization, further improving GNN reconstruction efficiency and purity.
  - Validate performance on other physics processes: Z(ll), dark matter, long-lived particles, etc.
  - Timing study and optimization, regional tracking, quantization, model optimization.
  - Integration into ATLAS analysis software package (Athena) and full-chain testing.

Back-up slides

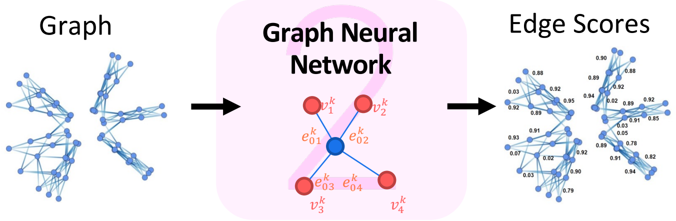
Graph construction efficiency



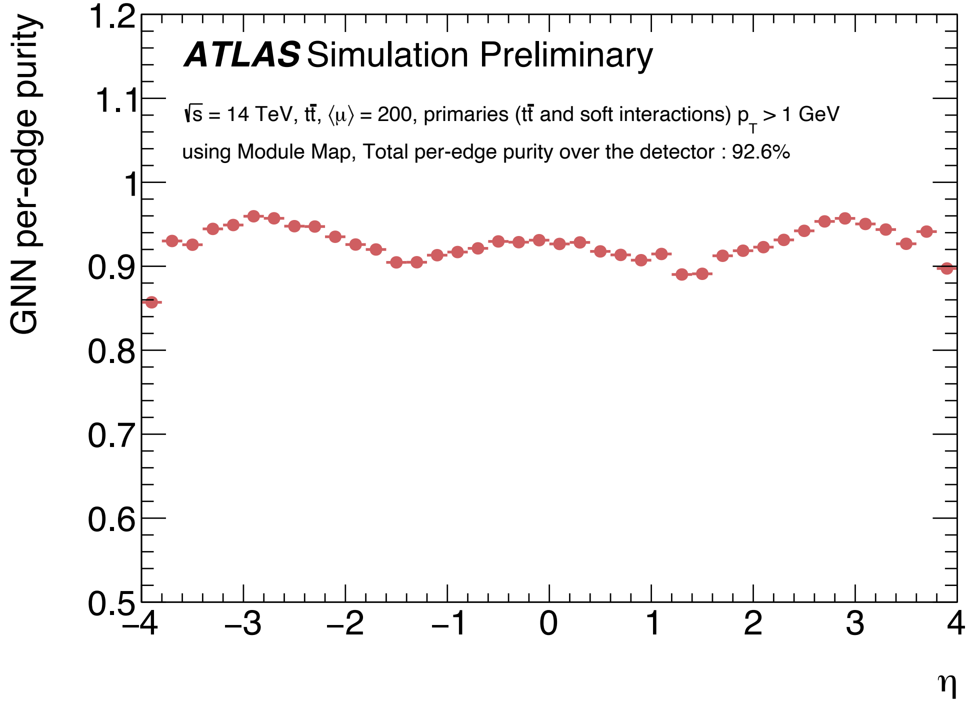


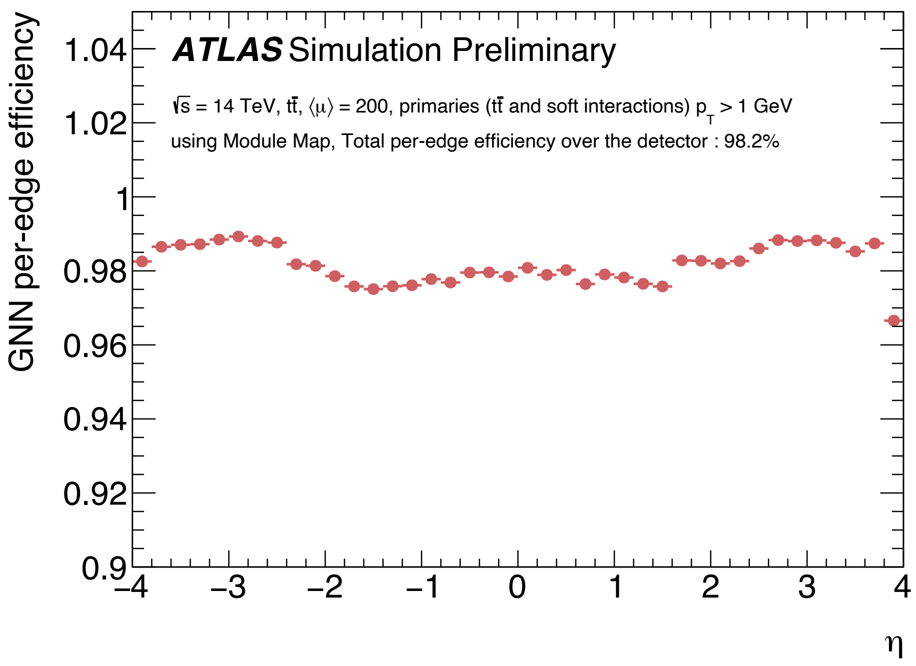


- Drop in efficiency at low η due to poor barrel strip resolution (will discuss further!)
- Drop in efficiency at high p_T due to low training statistics



GNN edge classification performance





(Left) Edge classification **Efficiency**, defined by $\frac{N_{TP}}{N_T}$ and (right) **Purity**, defined as $\frac{N_{TP}}{N_P}$ as functions of η.

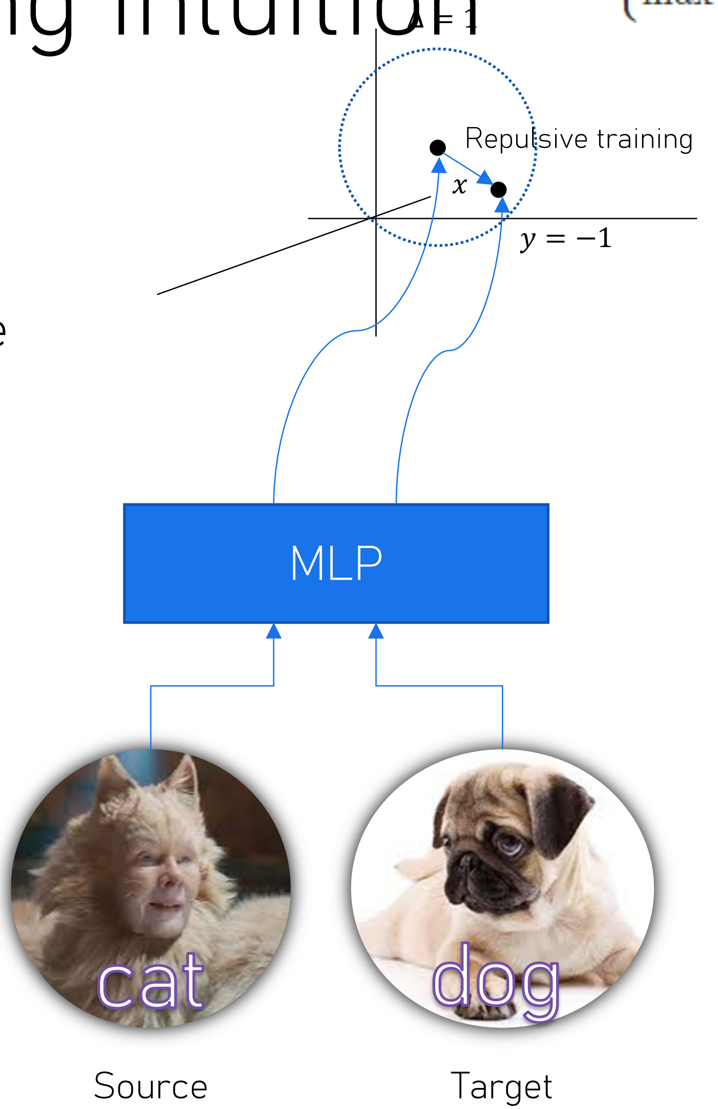
The GNN correctly identifies most True edges while rejecting the majority of Fake edges, evidenced by high efficiency and purity.

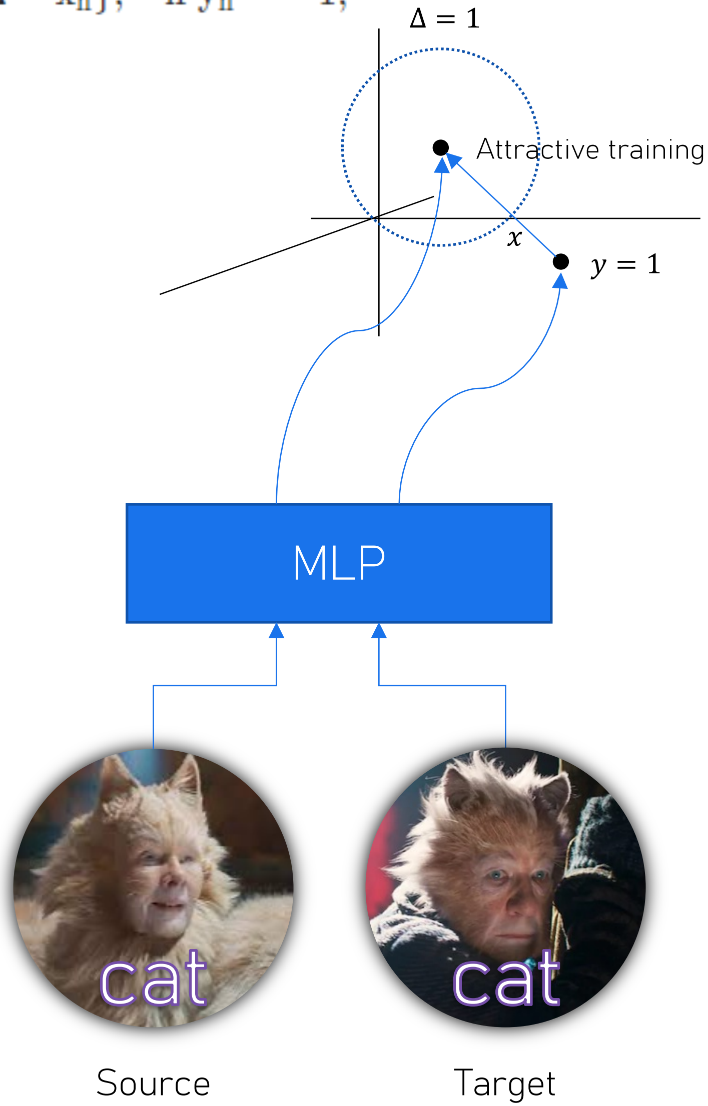
Metric learning intuition

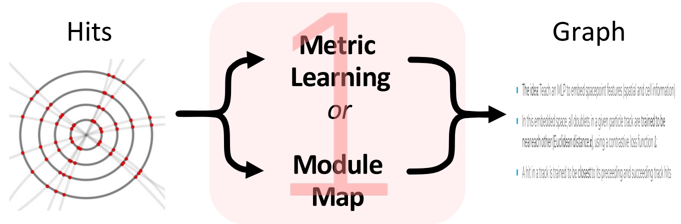
"Contrastive" hinge loss

$$l_n = \begin{cases} x_n, & \text{if } y_n = 1, \\ \max\{0, \Delta - x_n\}, & \text{if } y_n = -1, \end{cases}$$

- Encode / embed input into N-dimensional space
- Reward (low loss) matching pairs within unit distance
- Punish (high loss) mismatching pairs within unit distance
- Repeat for many pairs







Metric learning

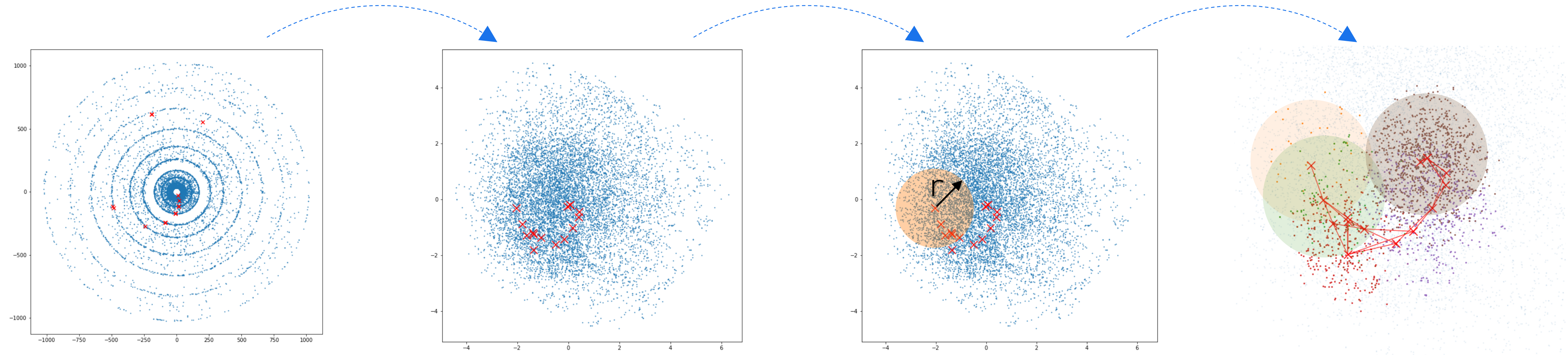
- **The idea**: Teach an MLP to embed spacepoint features (spatial and cell information)
- In this embedded space, all doublets in a given particle track are **trained to be near each other (Euclidean distance $\boldsymbol{x}$)**, using a contrastive loss function L:
- A hit in a track is trained to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

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

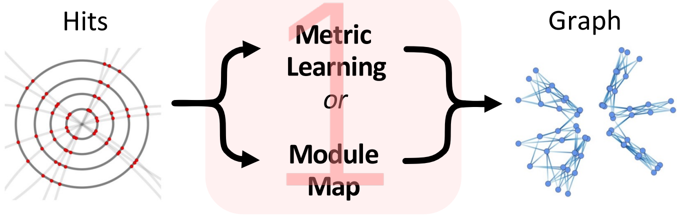
Embed into learned latent space

Connect all spacepoints within radius r

All spacepoint pairs joined into graph

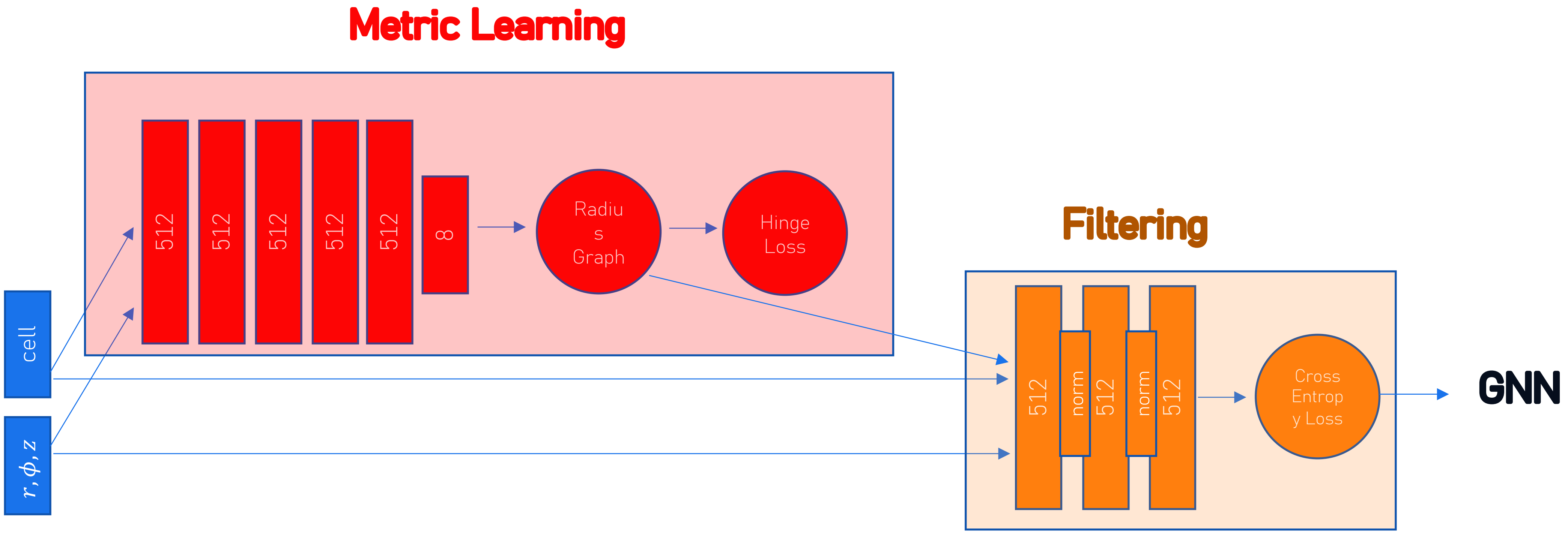


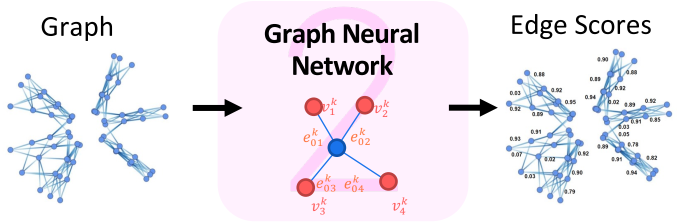
Metric learning – filtering



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

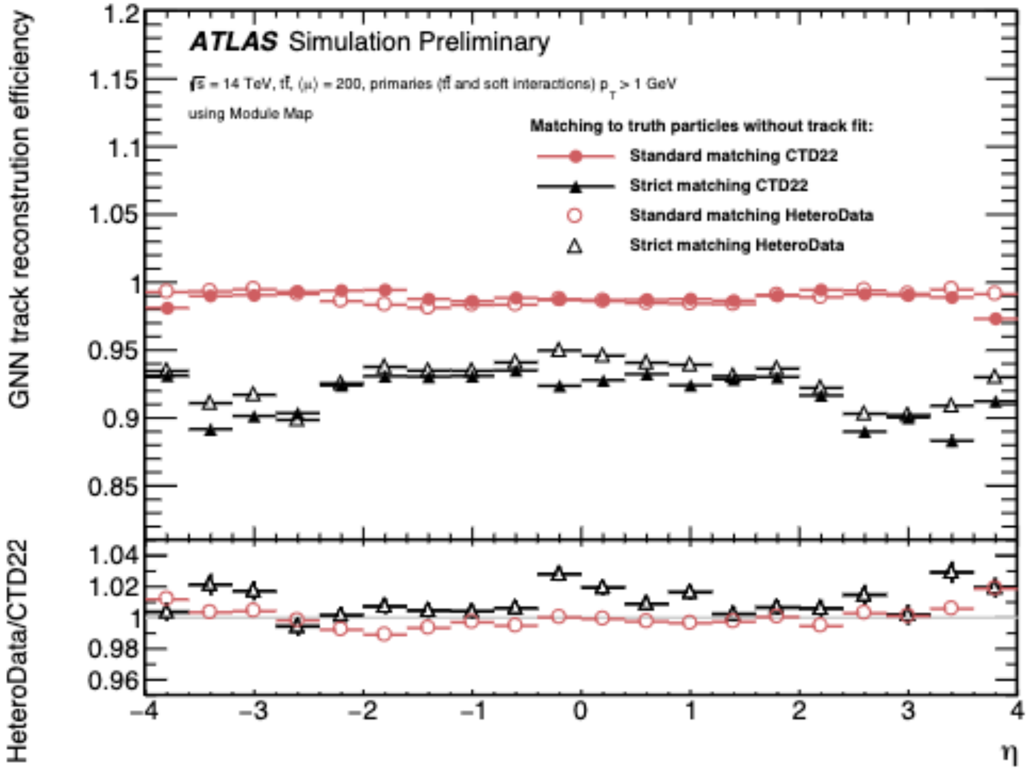




Loss function design

- The **target** of the GNN and track reconstruction is edges from primary particles with pT>1 GeV that have left at least 3 hits on different modules in the detector (see slide 12)
- Have very small set of target edges (1-2% of edges are true target t_{Seq})
- **Solution**: $t_{Seq}\ y = 1$ weighted up by $\times 10$, sequential background $\tilde{t}_{Seq}$ masked, all others $y = 0$
  - Weighting gives much better performance at high-efficiency
  - Masking gives much better performance around the 1 GeV cutoff

Technical efficiency



Standard matching: At least 50% of the space points (hits) in the track candidate belongs to the target particle (50% hit purity).

Strict matching: 100% hit purity and 100% of the hits in the particle found in the track candidate (100% hit efficiency).

Fake rate is $O(10^{-3})$ using standard truth matching