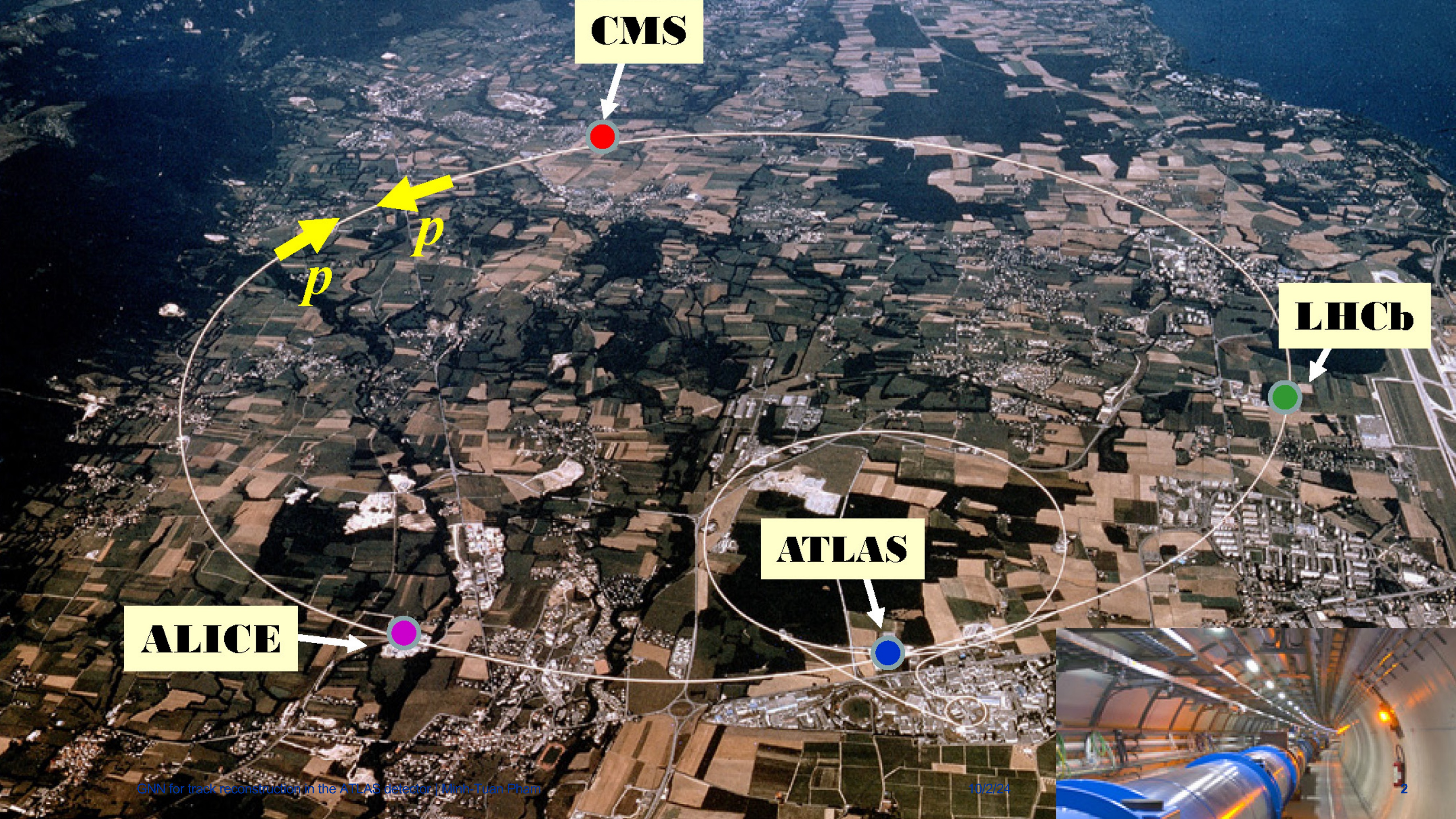
Graph neural network for charge—particle track reconstruction in the ATLAS ITk

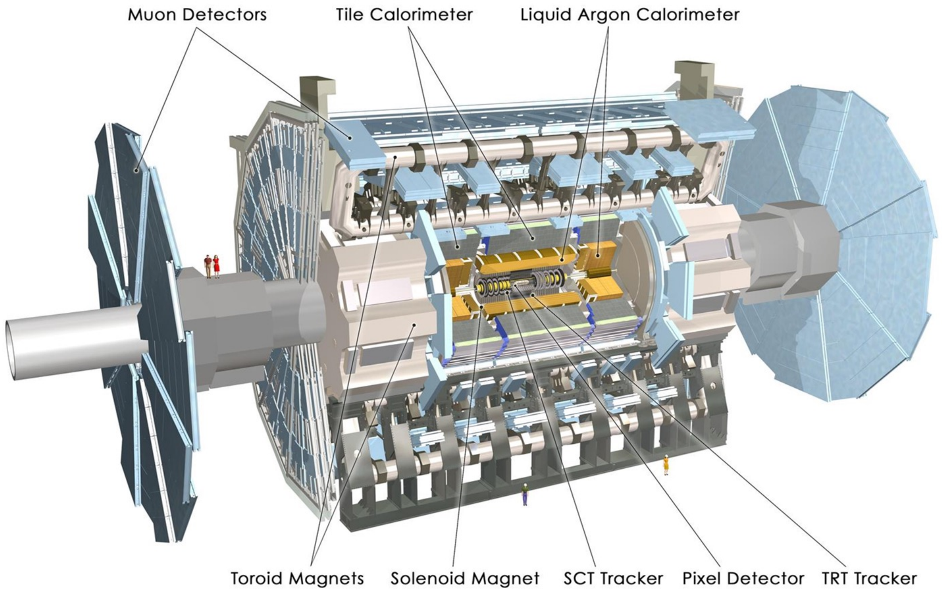
Sylvain Caillou, Daniel Murnane, Jan Stark, **Minh-Tuan Pham**

Heterogeneous Data and Large Representational Models in Science Workshop, Toulouse, France

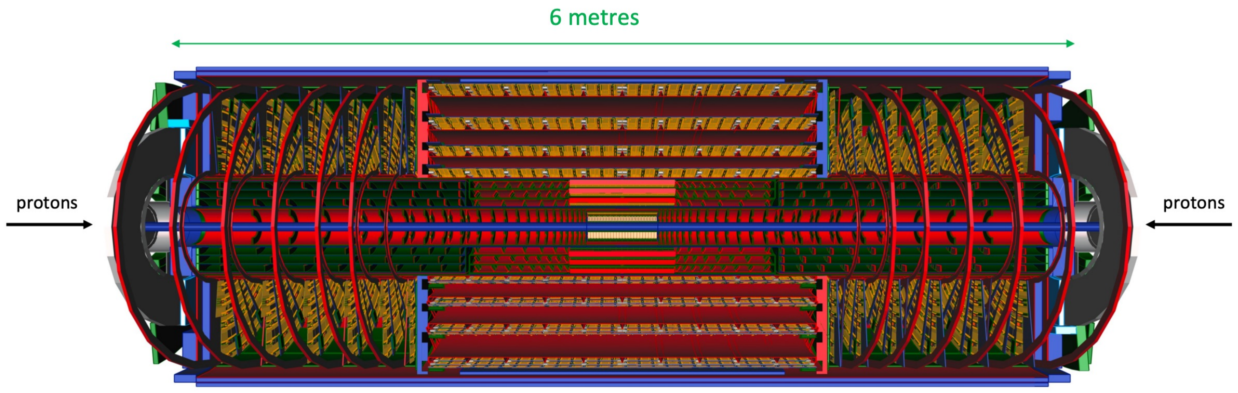

CMS
p
p
LHCb
ATLAS
ALICE

The ATLAS detector

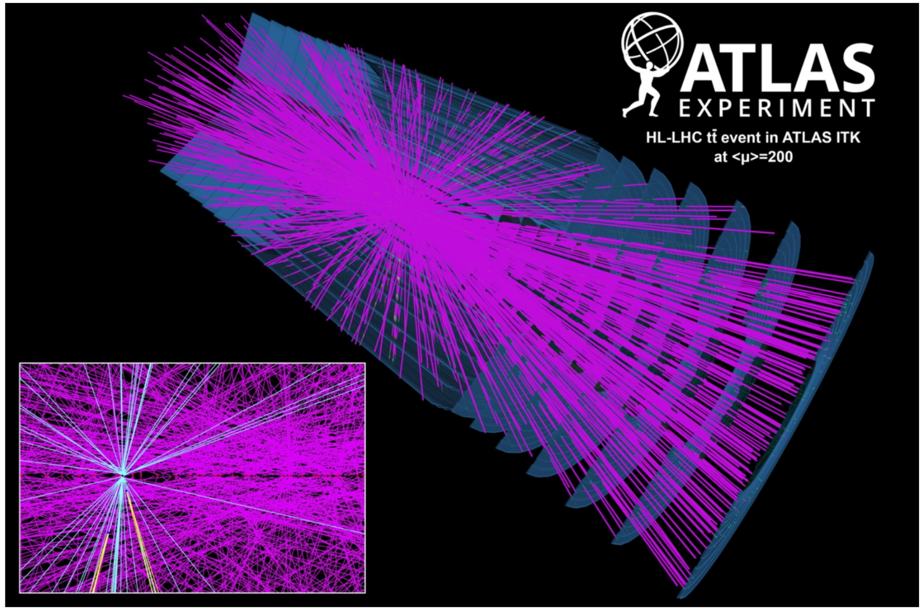
- The Large Hadron Collider (LHC) collides energetic protons in discrete packets (bunches).
- 1 bunch crossing = 1 collision event
- ATLAS detector records electronic signals from collision debris to reconstruct the event.
- The Inner Tracker (ITk) reconstructs the trajectory (track) of charged particles.



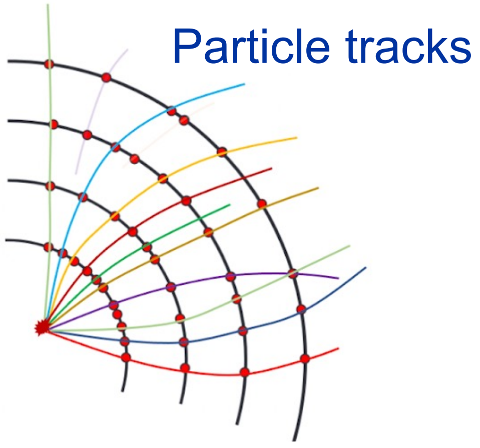
For proton 7 TeV = 0.999999991c

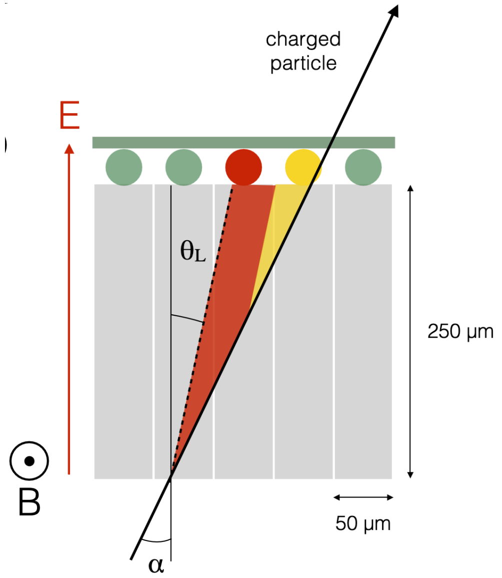


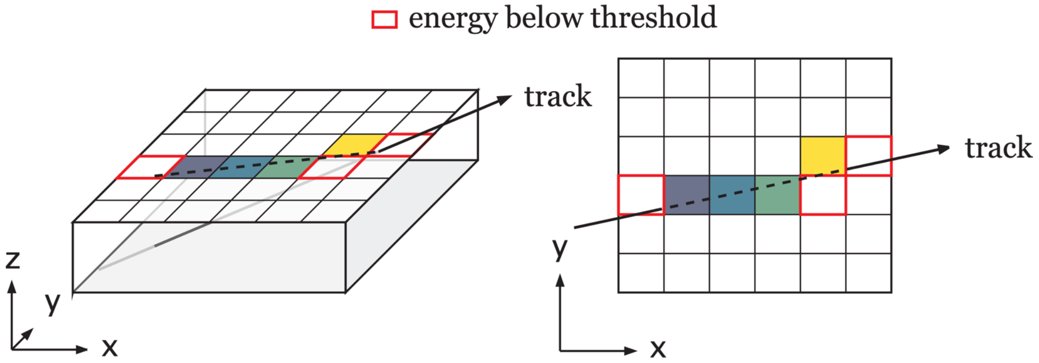
27k silicon modules



Building blocks of tracks



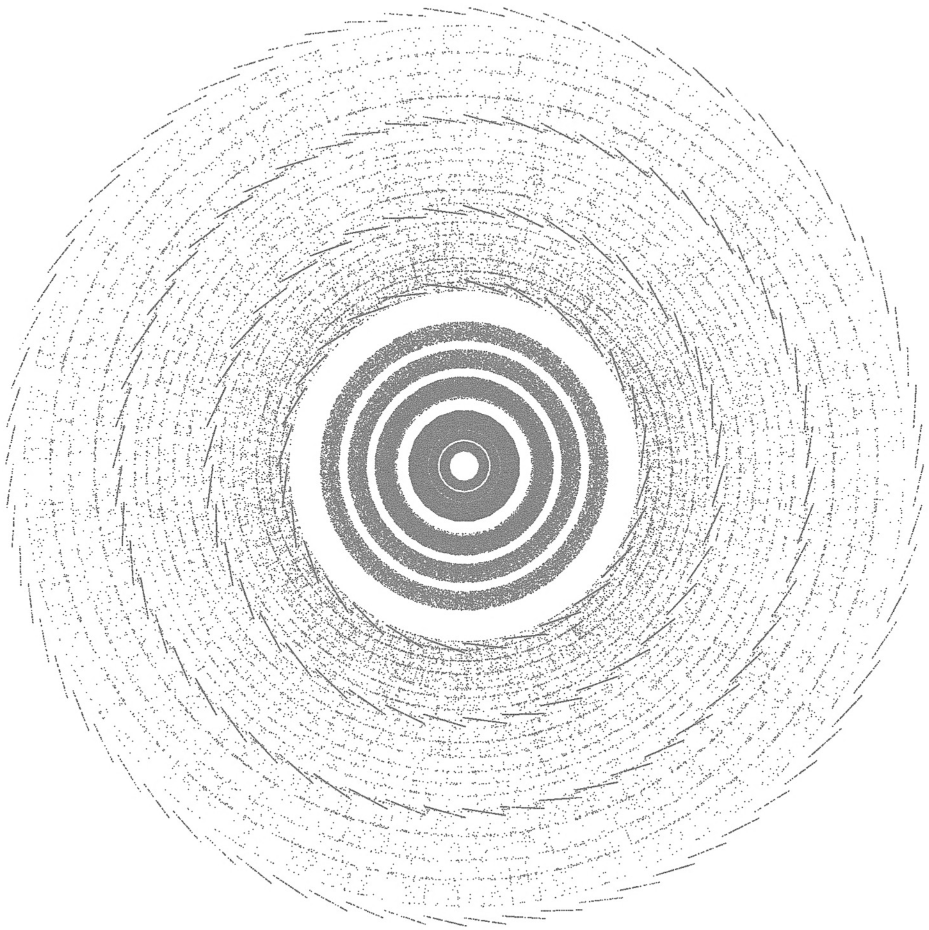


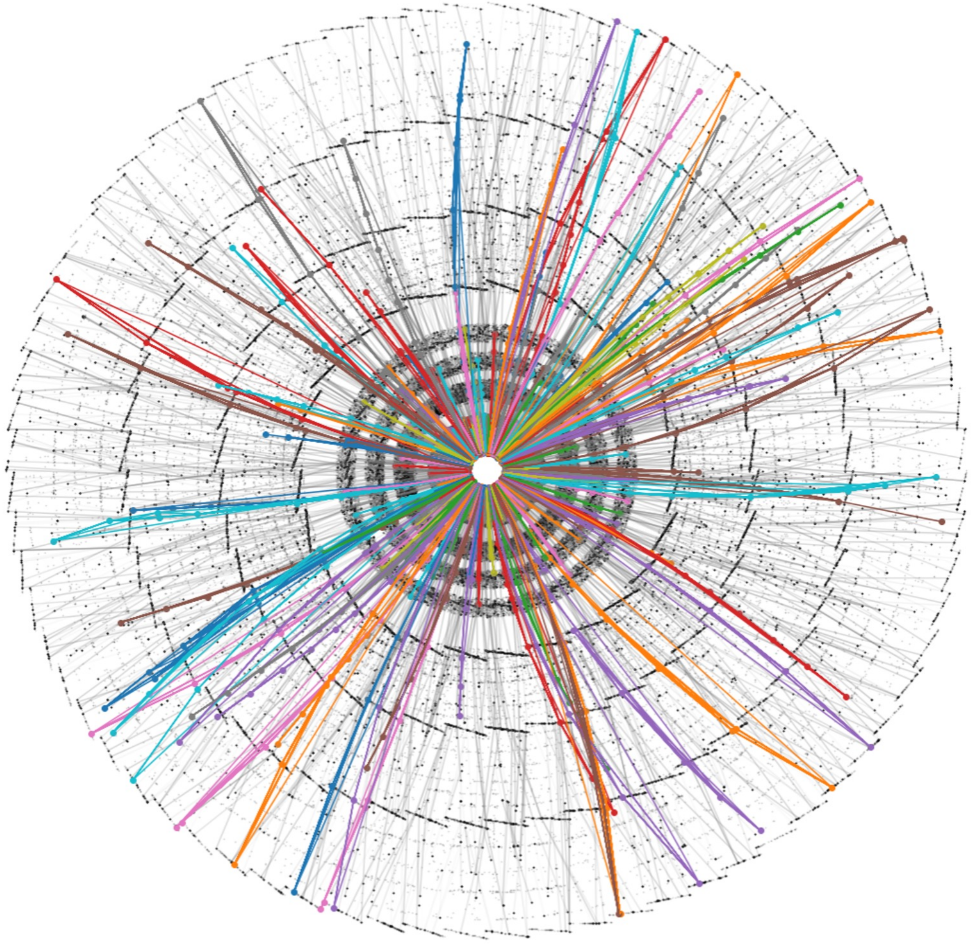


$$\vec{r} = \frac{1}{N}\sum_{i=1}^{N} \vec{l}_i, \qquad \vec{r} = \frac{1}{\sum_{i=1}^{N} q_i}\sum_{i=1}^{N} q_i \vec{l}_i.$$

- Charged particles ionize silicon sensors. Measure the charge collected in each cell as raw readout.
- Cells simultaneously hit by the particle form a measurement.
- Given all clusters from a BX, a tracking algorithm assigns each cluster to a track candidate.

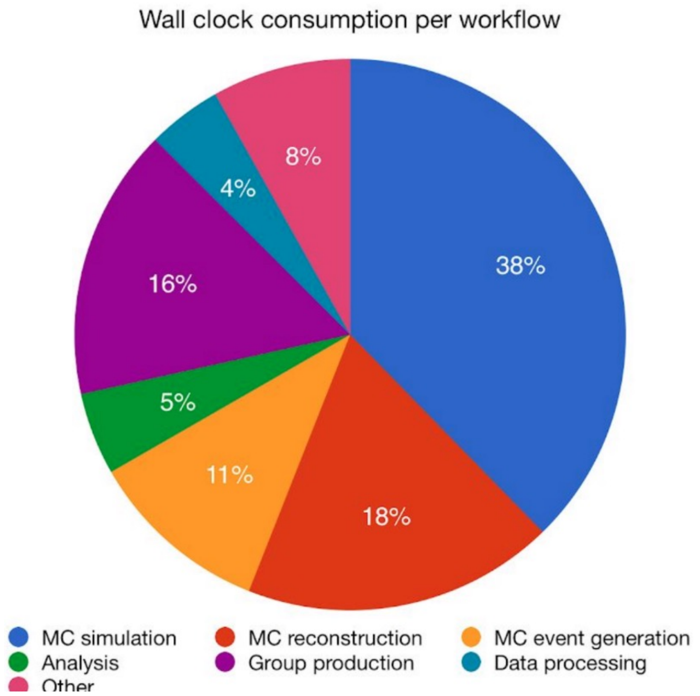
How do we get from





here to here

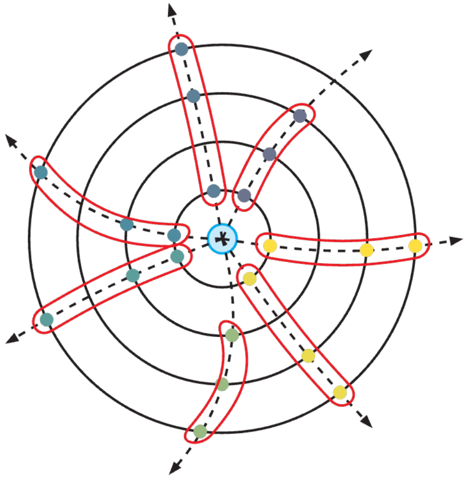
The cost of tracking

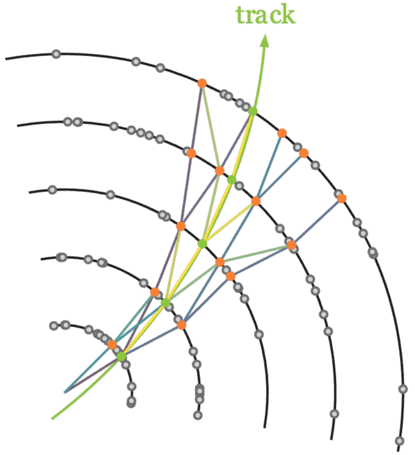


Current algorithm: Combinatorial Kalman Filter (CKF)

- Start from seeds, estimate track parameters and a search road.
- Iteratively incorporate the hit on the road most consistent with current track until no more compatible hit.

TL;DR: Inner tracking is the most expensive component. Seek alternatives for High Luminosity LHC.





ATLAS-TDR-029-ADD-1

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

Inner tracking takes 67% of reconstruction time.

APPLICATION OF KALMAN FILTERING TO TRACK AND VERTEX FITTING

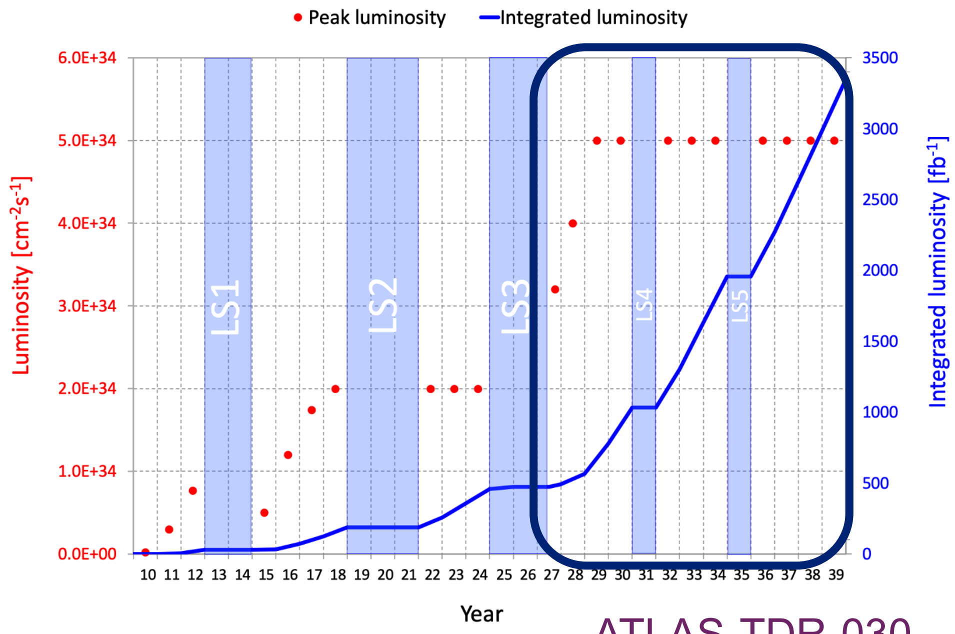
R. FRÜHWIRTH

Institut für Hochenergiephysik der Österreichischen Akademie der Wissenschaften, Vienna, Austria

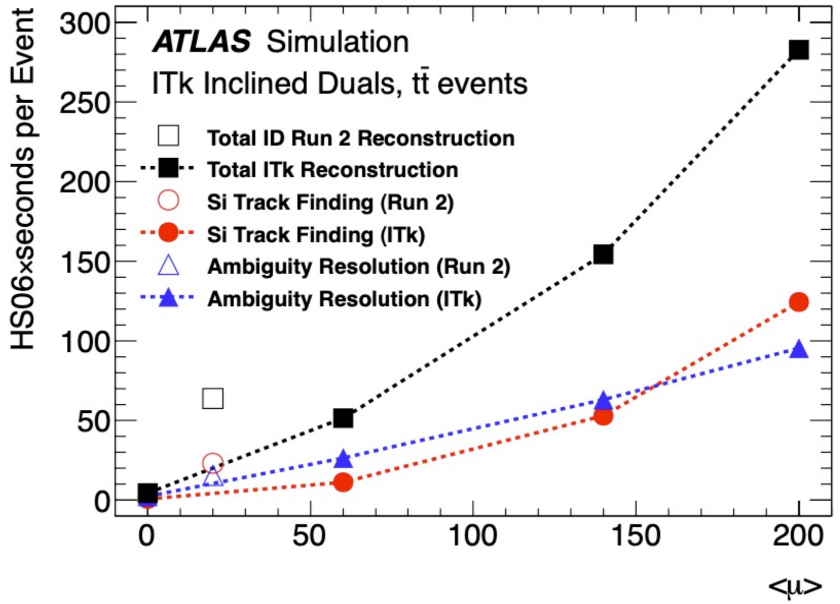
Received 30 June 1987

The High Luminosity LHC

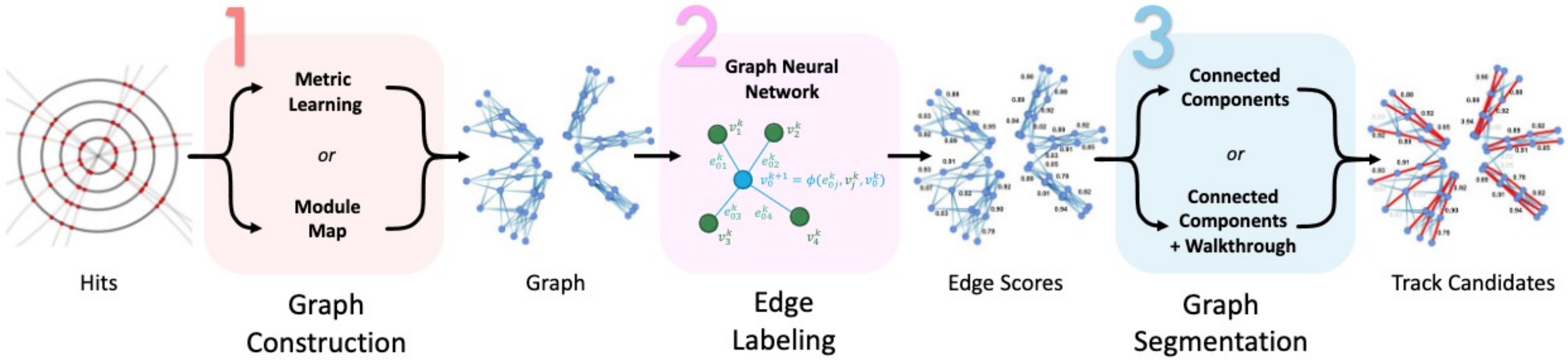
- In 2029, the avg. number of interactions per event will increase to 200 (2.5x current rate).
- Challenging to meet throughput requirement.
- Several directions to cope:
  - Optimize the CKF on CPU (ATL-PHYS-PUB-2019-041)
  - CKF on accelerators (arXiv:2105.01796v2)
  - **Novel machine learning algorithms on accelerators.**
- Goal: Find a faster ML-based algorithm that runs in <1s/event, achieving the same tracking performance.



ATLAS-TDR-030



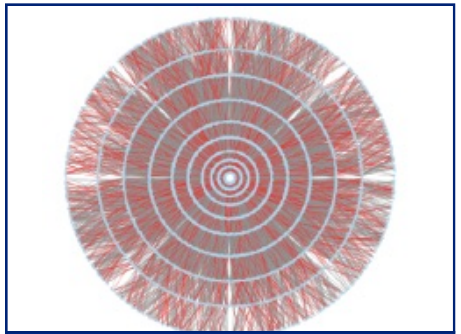
A graph-based approach to tracking



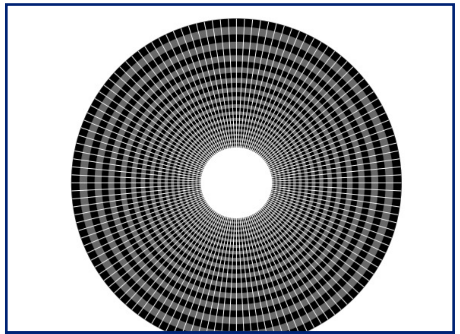
1. **Construct a graph from detector hits.**
   - 1 node = 1 hit.
   - 1 edge = A hypothesis of nodes being consecutive hits on a track.
2. **Classify edges using graph neural network (GNN).**
3. **Segment the graph to build track candidates.**

Git repo, documentation.

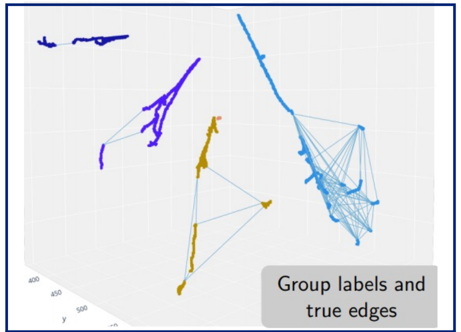
Graph neural network in particle physics



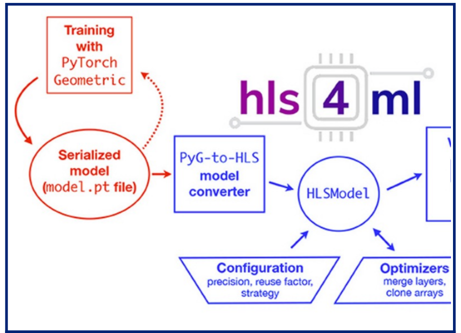
High Lumi Generic Tracking



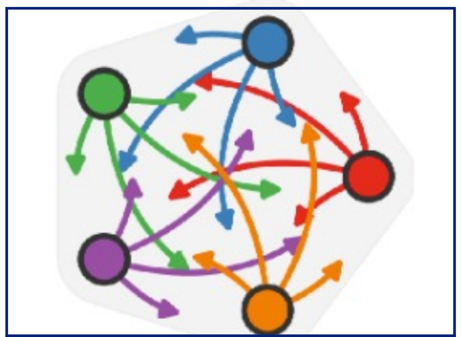
High Lumi CMS Calorimetry



LArTPC Particle Reconstruction



FPGA-based Track Reconstruction



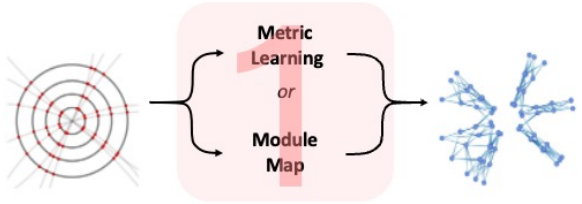
GNN Flavour Tagging

Very large and active field of study!

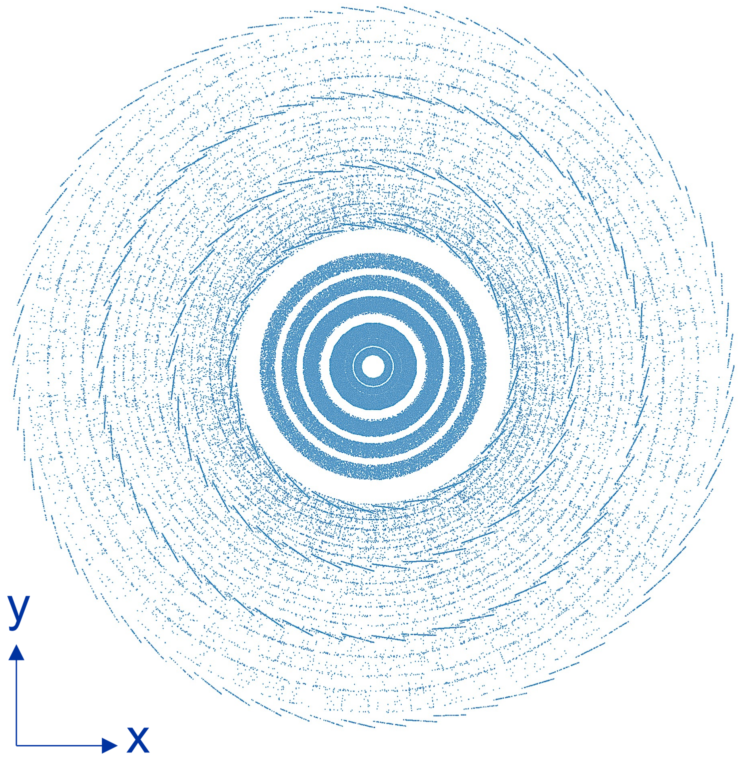
Comprehensive review of GNNs for Track Reconstruction - arXiv:2012.01249

White paper on progress and future of the field – arXiv:2203.12852

Graph construction

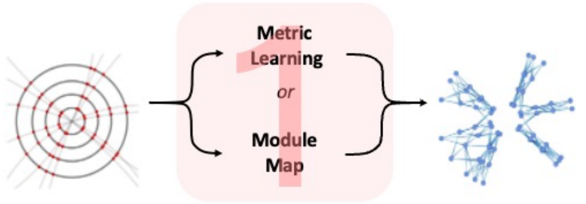


- **A typical collision contains O(300k) nodes. Fully connected graph is unfeasible.**
- **Must include $> 99\%$ true connections.**
- **Must constrain graph to O(10^6) edges, to guarantee memory and throughput.**



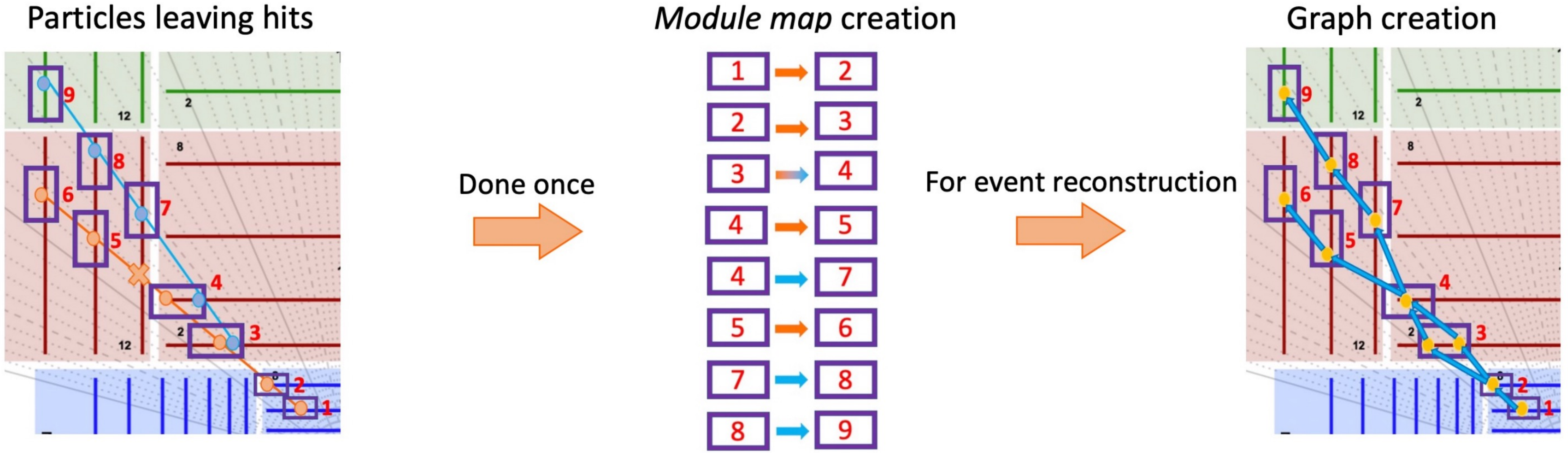
Collision event projected to xy-plane. Very high hit density near collision point

Graph construction – Module Map

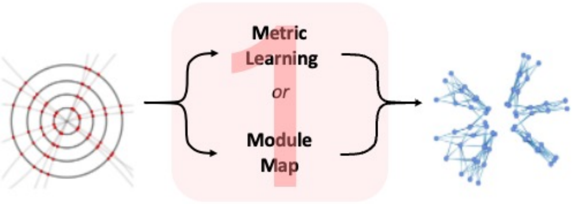


Data driven method

1. **Build a list of possible connections between 2 detector modules, from observing 90k simulated events.**
2. **On a new event, connect 2 hits of their modules are connected by a connection in the MM.**

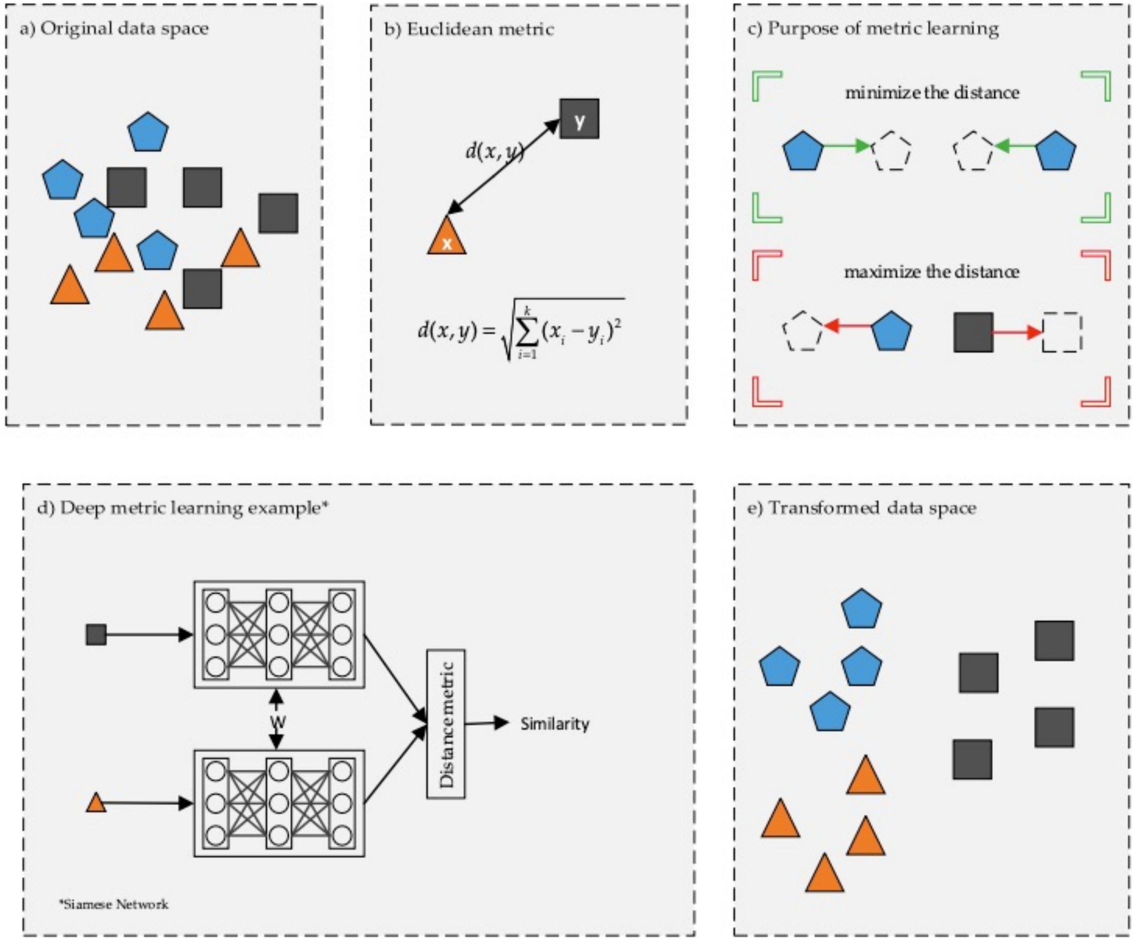


Graph construction – Metric Learning



1. Define a distance metric.
2. Learn a MLP transformation that minimizes the distance between true hit pairs and maximizes otherwise.
3. In the transformed space, use kNN algorithm to connect the nodes.
4. Train another MLP to eliminate easy fakes.

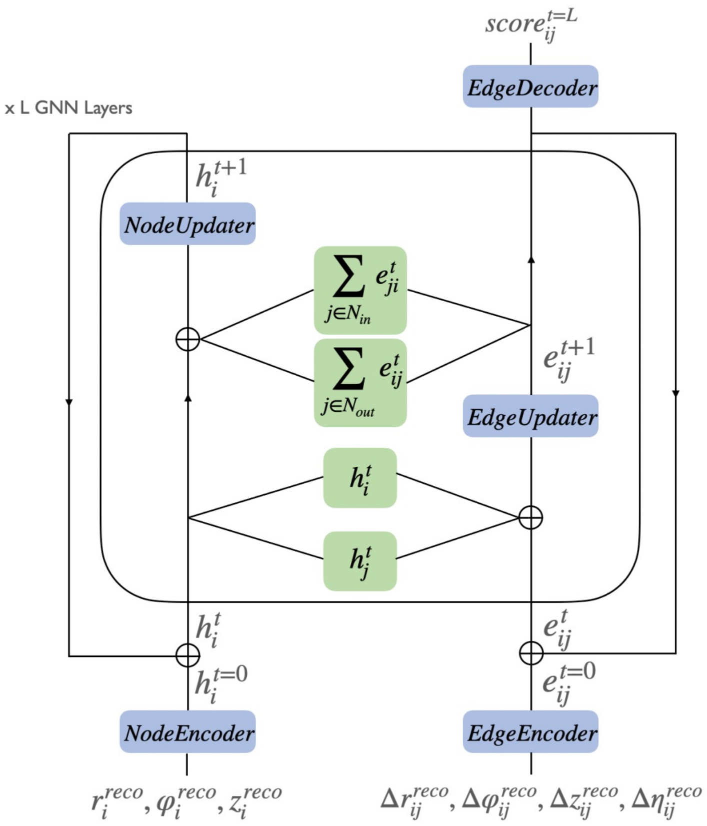
Both methods create graphs < 2e6 edges and containing > 99% true connections.



Symmetry 2019,11(9),1066

Edge classification

Encoders: map input node and edge features to latent space.
Message passing: Update node and edge features using aggregated edge messages.
Edge Decoder: map latent edge feature to an edge score.



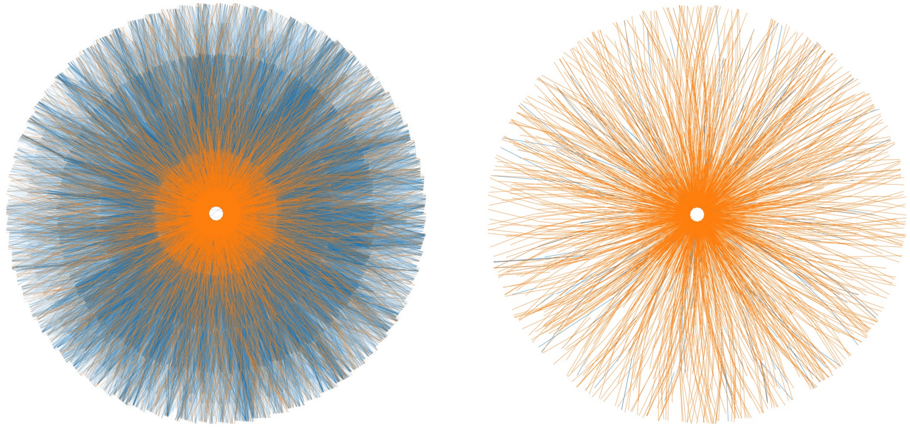
$y_{ij} = EdgeDecoder(e_{ij}^{t=N})$

$e_{ij}^{t=0} \leftarrow EdgeEncoder(\vec{e}_{ij}^{\,t-1}, h_i^t, h_j^t)$

$h_j^t \leftarrow NodeUpdater(h_i^{t-1}, m_j^t)$

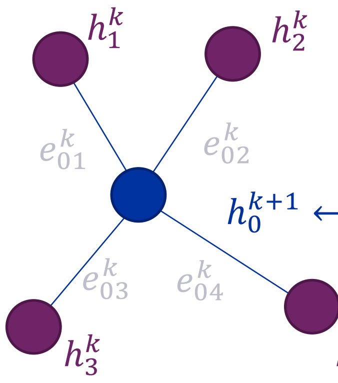
$m_j^t = Agg_{i=1}^{N_j} e_{ij}^t$

$h_i^{t=0} \leftarrow NodeEncoder(\vec{x}_i^{\,reco})$
$e_{ij}^{t=0} \leftarrow EdgeEncoder(\vec{e}_i^{\,reco})$



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

Imbalanced data:
True: $O(10^4)$
Fake: $O(10^6)$

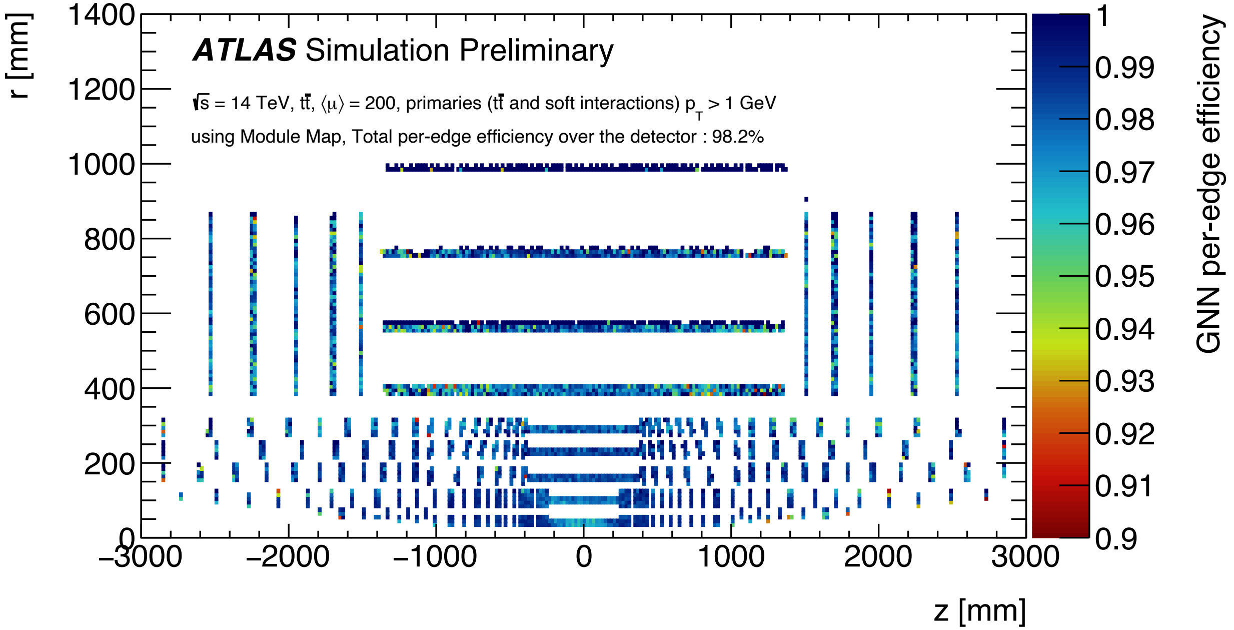


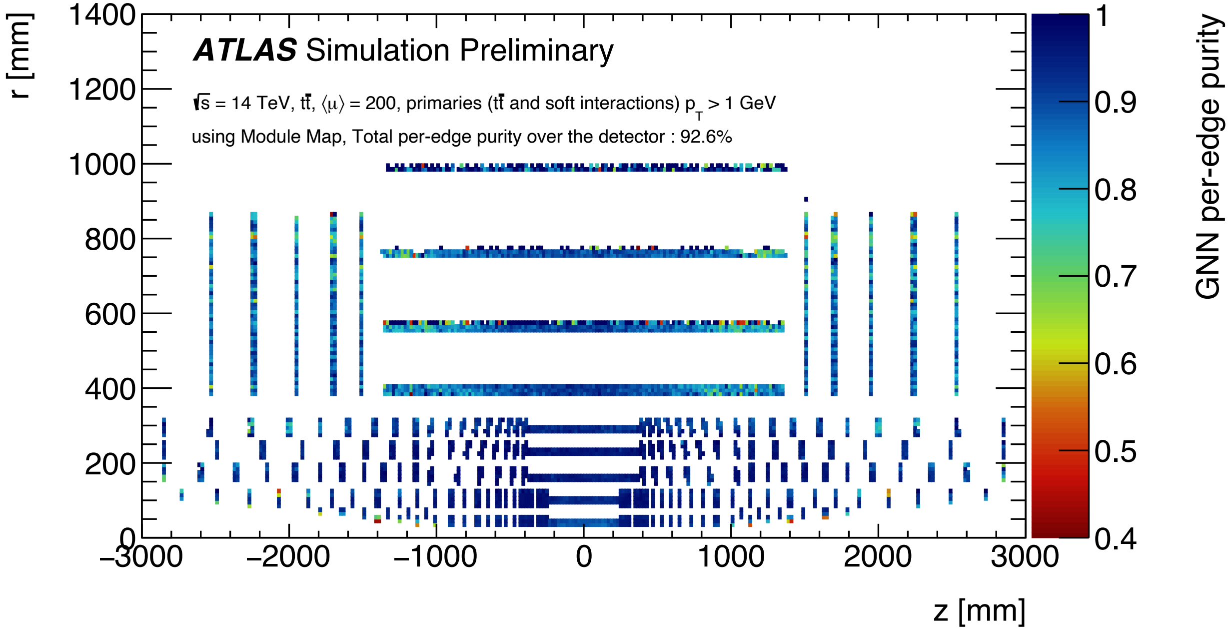
Battaglia, Peter, et al.
Interaction network

$h_0^{k+1} \leftarrow NodeUpdater(h_0^k,\ m_j^t)$

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

Edge classification performance





Apply edge cut 0.5, define efficiency (recall)

$$\epsilon_{model} = \frac{\left|\{e_{ij} | y_{ij} = 1; \phi(e_{ij}) > 0.5\}\right|}{\left|\{e_{ij} | y_{ij} = 1\}\right|}$$

High efficiency throughout detector, average **98.2%**

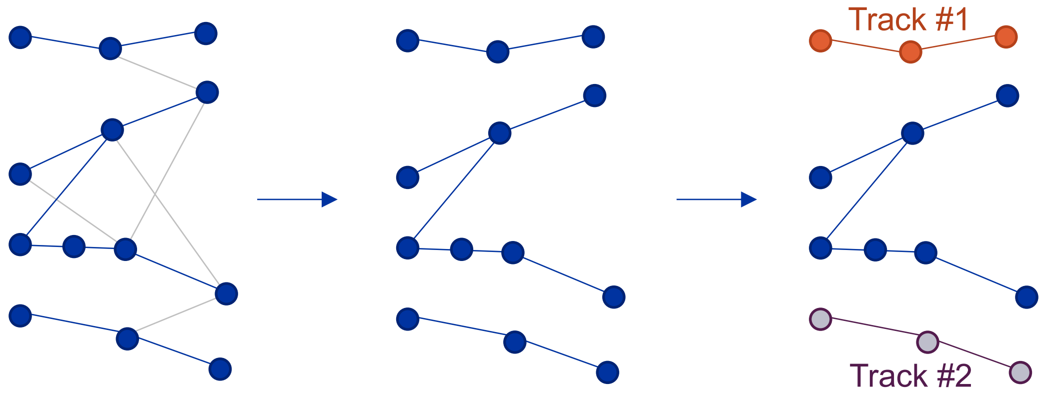
Define purity (precision)

$$p_{model} = \frac{\left|\{e_{ij} | y_{ij} = 1; \phi(e_{ij}) > 0.5\}\right|}{\left|\{e_{ij} | \phi(e_{ij}) > 0.5\}\right|}$$

Average purity **92.6%**

Graph segmentation

Connected Components

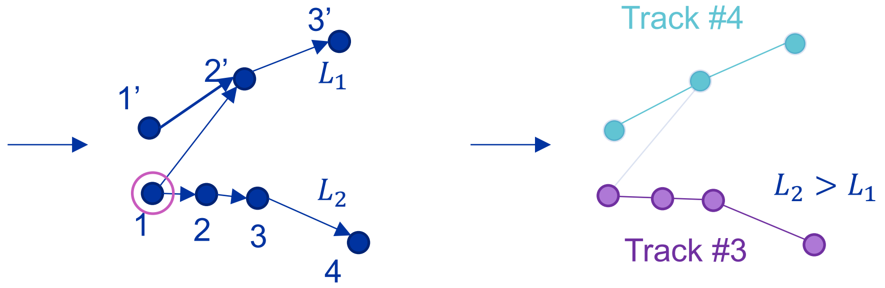


Classified edges

Loose score cut

Label simple candidates

Walkthrough



Walk through paths from starting node, count length L

Assign longest path as candidate

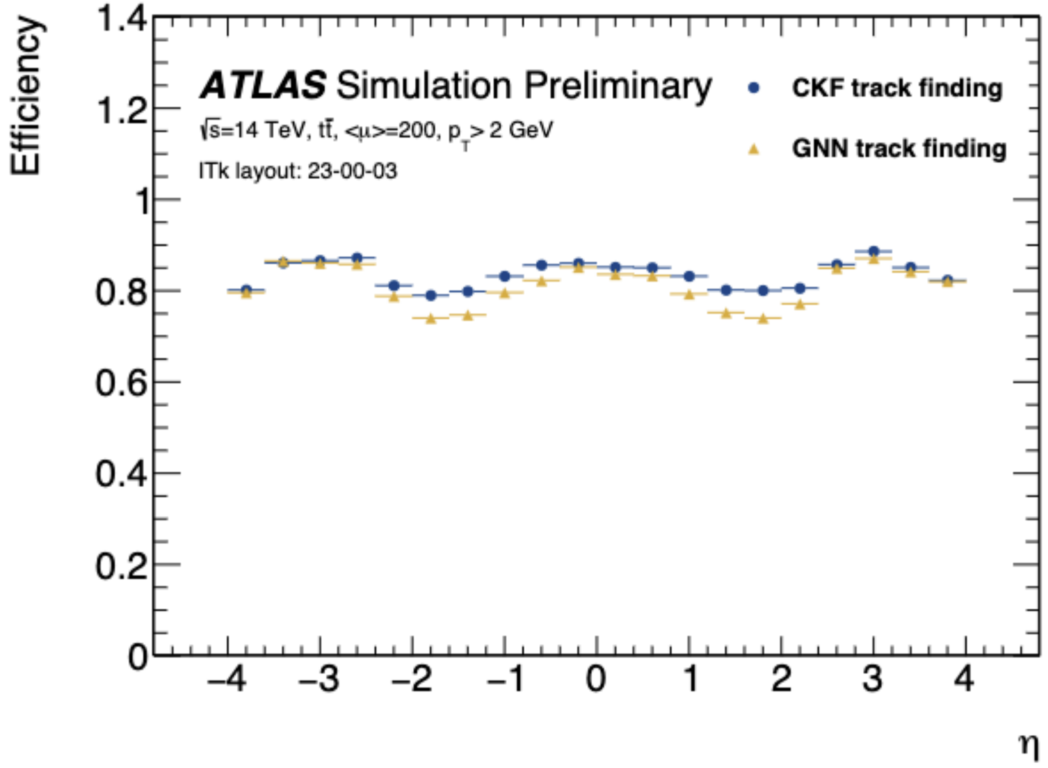
2-step sequence: Connected components (CC) and walkthrough:

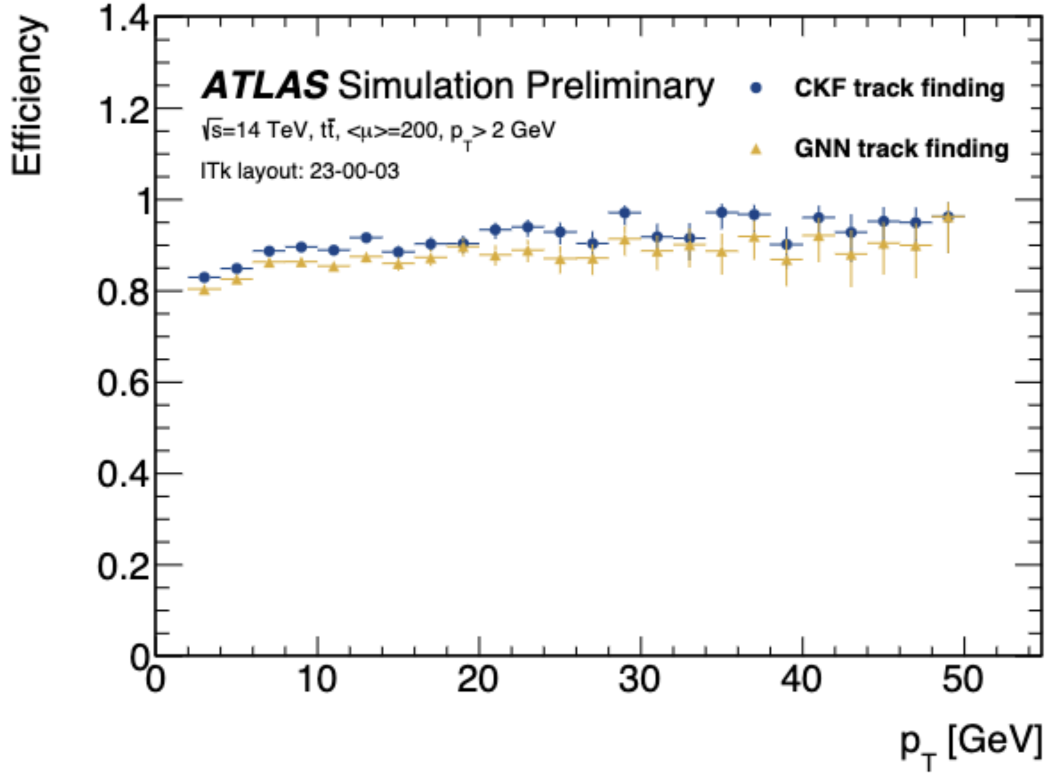
1. Use CC to isolate subgraphs with no branching.
2. On subgraphs with branching, use walkthrough to separate track candidates.

Each track candidate is a list of hits => extract track parameters by a χ^2 fit

Track reconstruction efficiency

ATL-SOFT-PROC-2023-047





Competitive performance compared to the CKF, matching in the central and forward region, while lagging in “transition” region. The difference strongly depends on particle η, suggesting the transition region is particularly difficult for the GNN.

Computational performance

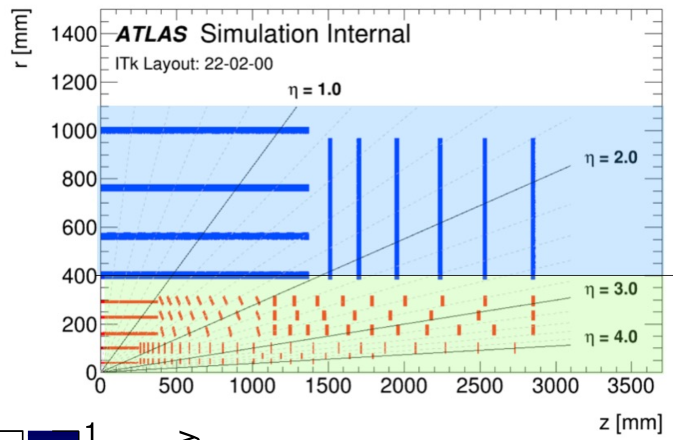
- **Achieve goal of sub-second throughput via several optimizations.**
  - A CUDA-native implementation of the module map,
  - ML model inference with Automatic Mixed Precision (AMP) and JIT compilation in Pytorch, (Alvaro's talk)
  - Walkthrough algorithm optimized with JIT compilation.
- **Compress models with Quantization and Pruning to apply in Event Filter.**
- **Currently pursue ideas to maximize throughput, e.g. knowledge distillation**
- **Contributions in upcoming CHEP2024 (links in back-up)**

Inference throughput in produced with AMP and model compilation in Python.

Learning heterogeneous detector data

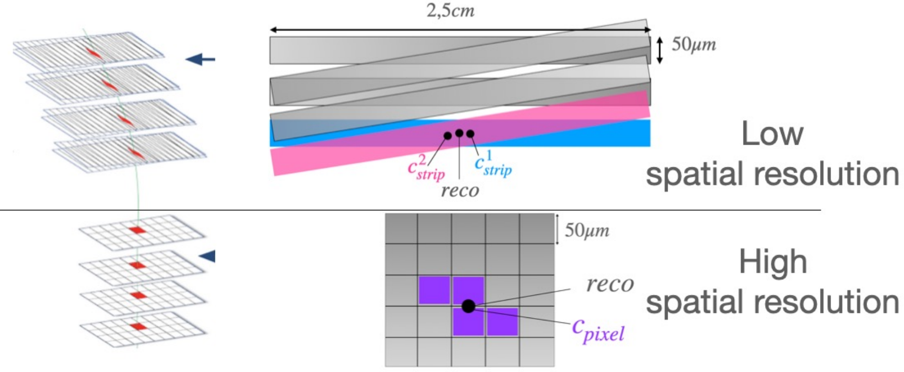
Challenges from a heterogeneous detector

2 sensor technologies having different spatial resolutions.
→ Positional hit inputs from STRIPS has lower positional precision than PIXEL.



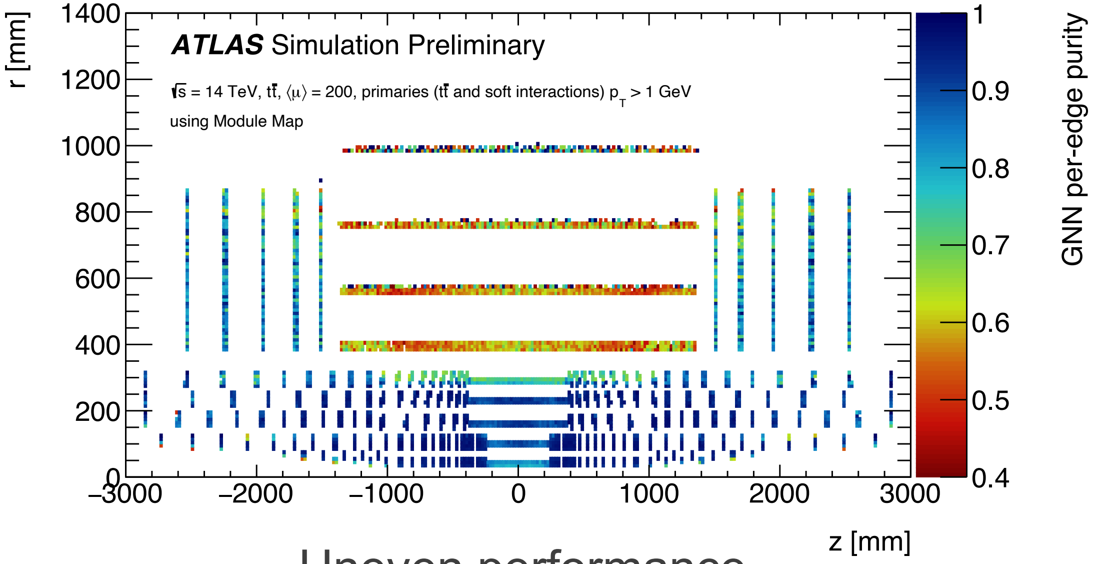
STRIP technology
Space Point reconstructed from 2 STRIP clusters

$$r^{reco}, \varphi^{reco}, z^{reco} = f_{strip}(c^1_{strip}, c^2_{strip})$$

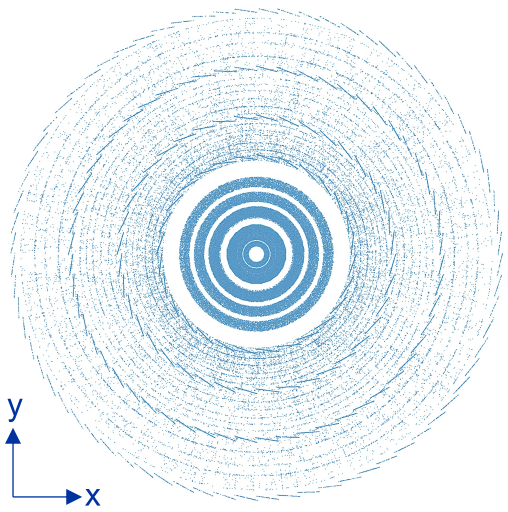


PIXEL technology
Space Point reconstructed from 1 PIXEL cluster

$$r^{reco}, \varphi^{reco}, z^{reco} = f_{pixel}(c_{pixel})$$



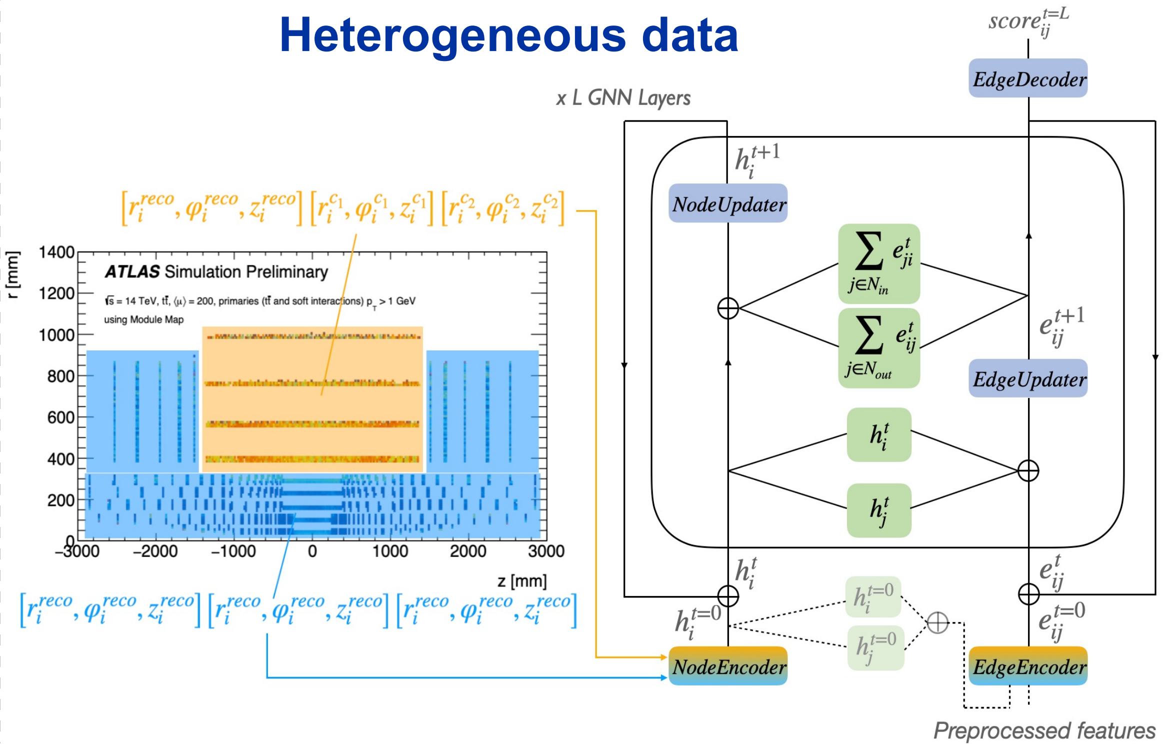
Uneven performance



Different hit and edge density from PIXEL to STRIP sub-detector.

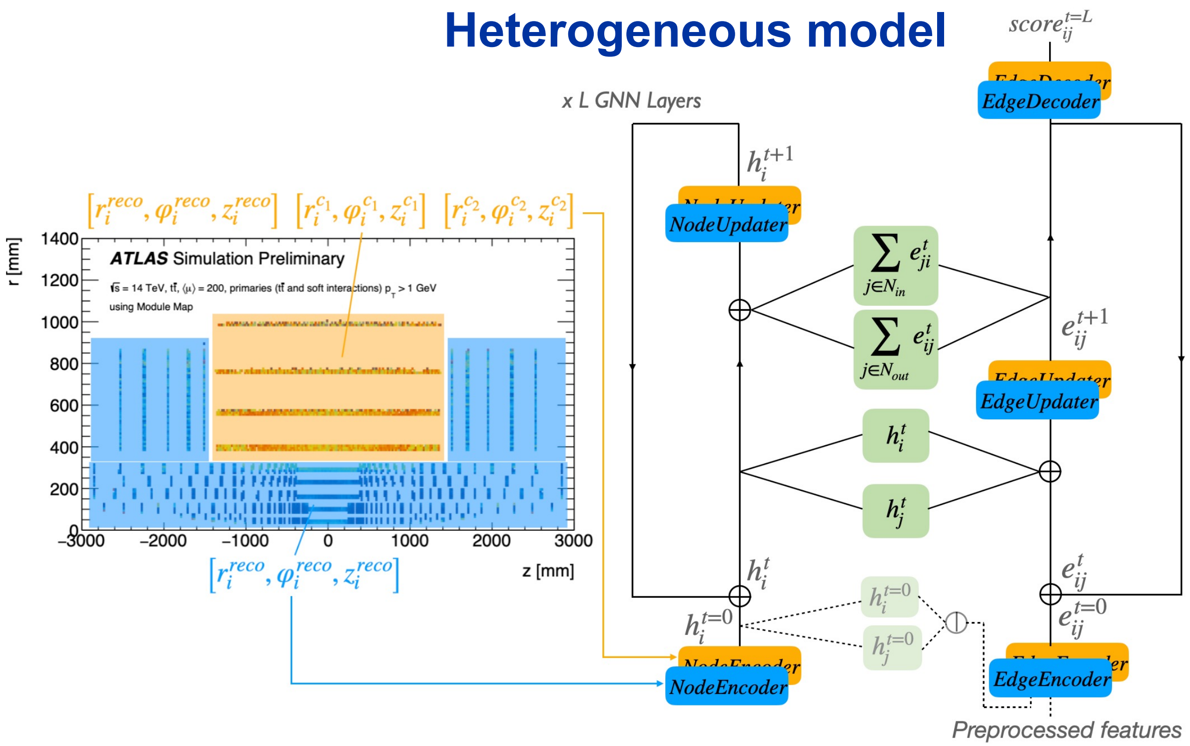
Dealing with heterogeneous input data

Heterogeneous data



Solution 1: Input heterogeneous low-level data.
PIXEL input = Spatial coordinates + padding.
STRIP input = Spatial coordinates + cluster information

Heterogeneous model

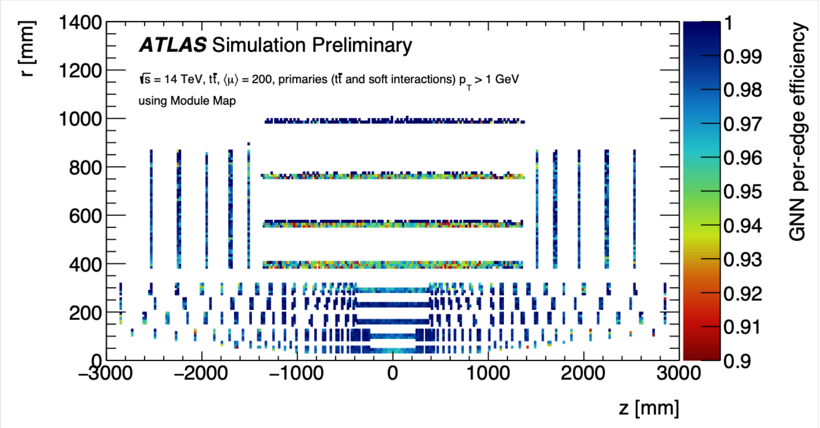


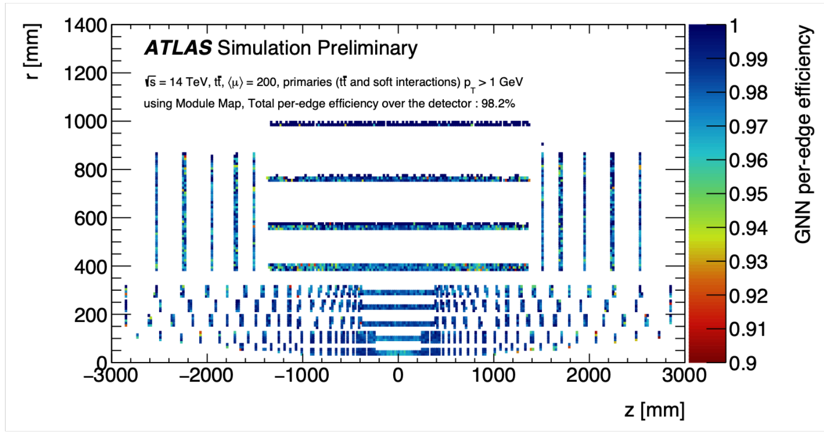
Solution 2: Train separate MLPs to handle data from each technology. E.g.
PIXEL input → Pixel Node Encoder
STRIP input → Strip Node Encoder

Solution 1: Heterogeneous data performance

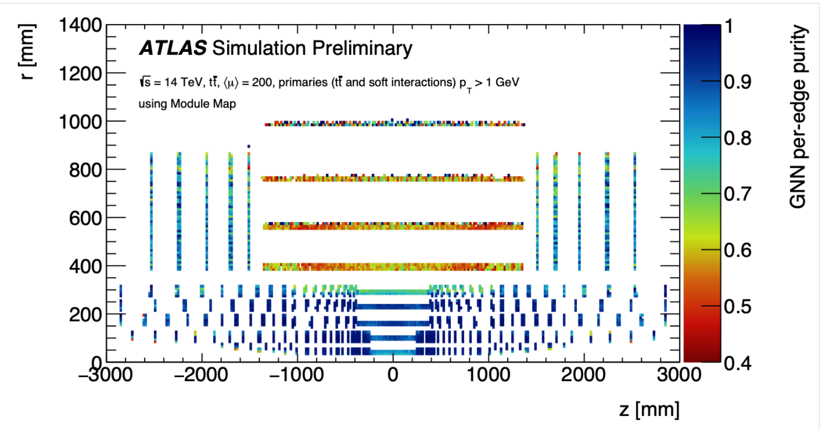
Homogeneous data

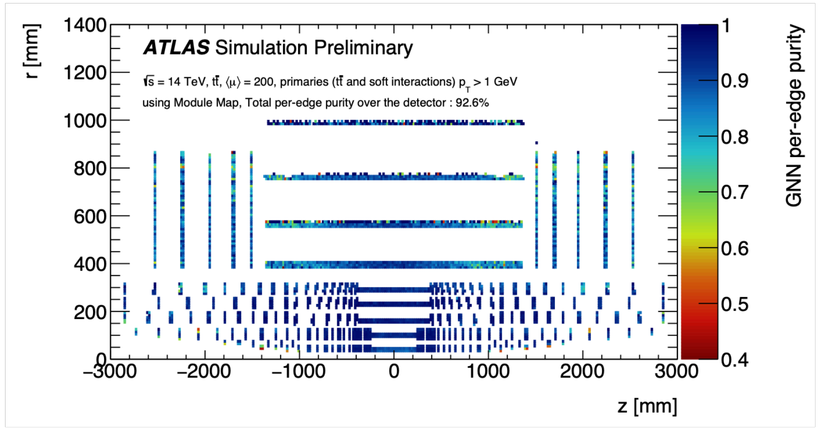
Heterogeneous data





Small improvement in efficiency (recall) in STRIP





Large improvement in purity (precision) in STRIP

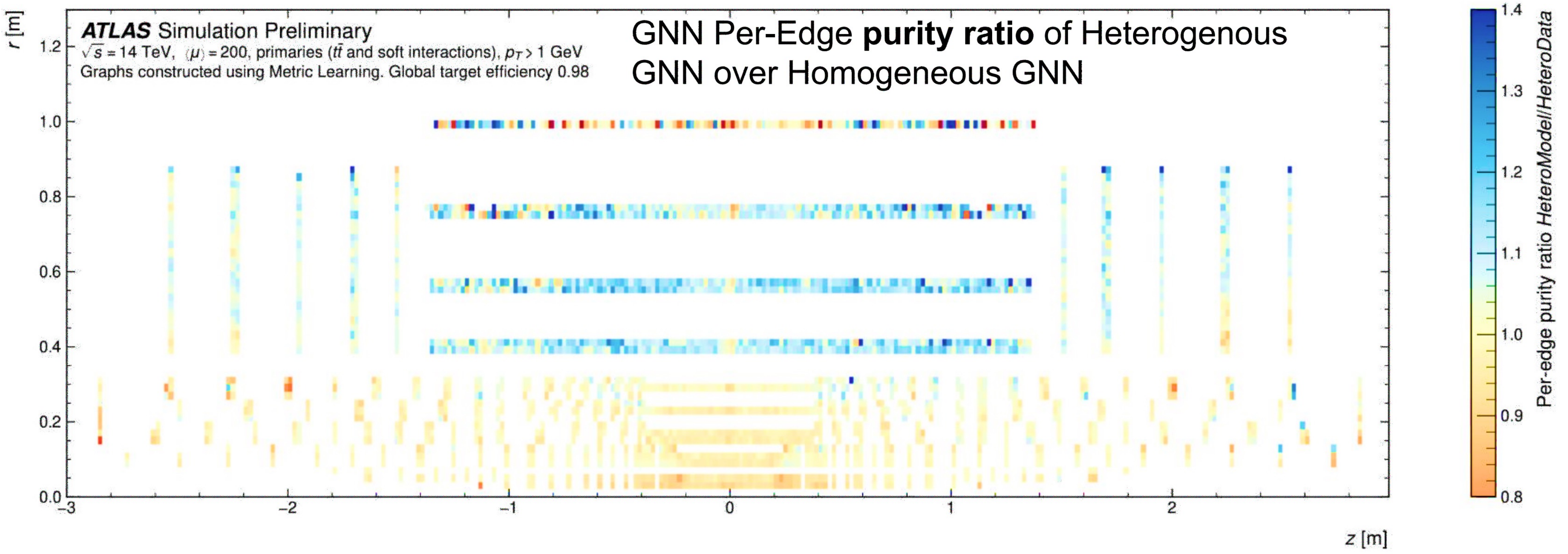
Model learns better representation of strip hits with low-level input.

Solution 2: Training heterogeneous model

Compare homogeneous model+heterogeneous data to heterogeneous model, keep $\epsilon_{model} = 98\%$, plot p_{hetero}/p_{homo} across the detector.

Purity in STRIP improved by 11%, but with 1% loss in PIXEL.

Similar performance averaged over detector. Investigating ways to improve purity in PIXEL.



Summary and prospect

- Demonstrate a physics performance close to legacy track reconstruction algorithm
- Achieve speed compatible with ITk throughput requirement
- Room for improvement in both speed and precision toward full deployment in production
- Pursue novel ideas to learn better from heterogeneous data from the tracker and other sub-detectors.

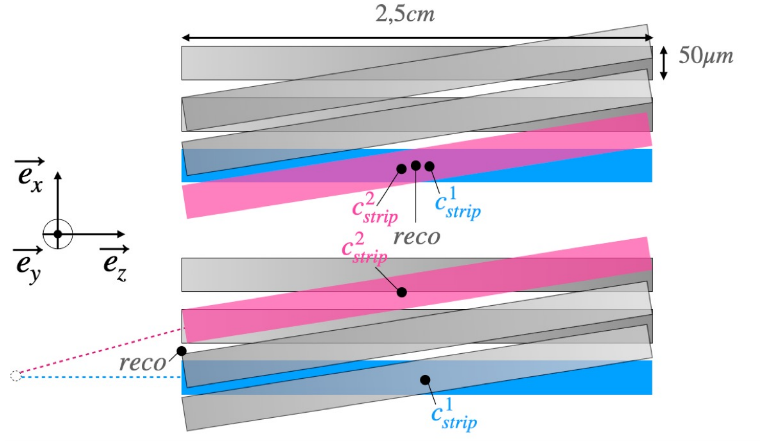
Acknowledgement

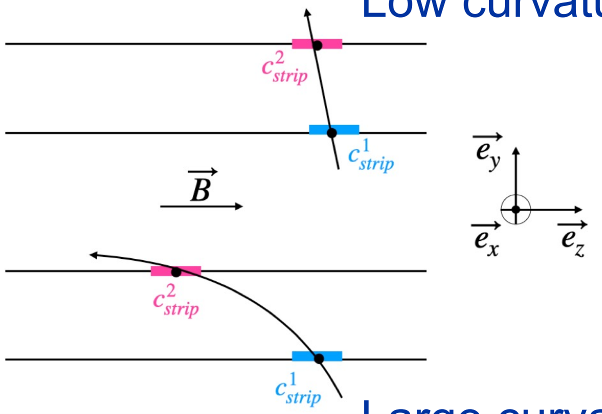
I have used materials from the following presentations to prepare these slides

1. Graph Neural Networks for charged-particle track reconstruction (speaker: Jan Stark)
2. Novel fully-heterogeneous GNN designs for track reconstruction at the HL-LHC (speaker: Sylvain Caillou)
3. Performance of GNN-based tracking for ATLAS Itk (speaker: Xiangyang Ju)

Back-ups

Why different resolutions?

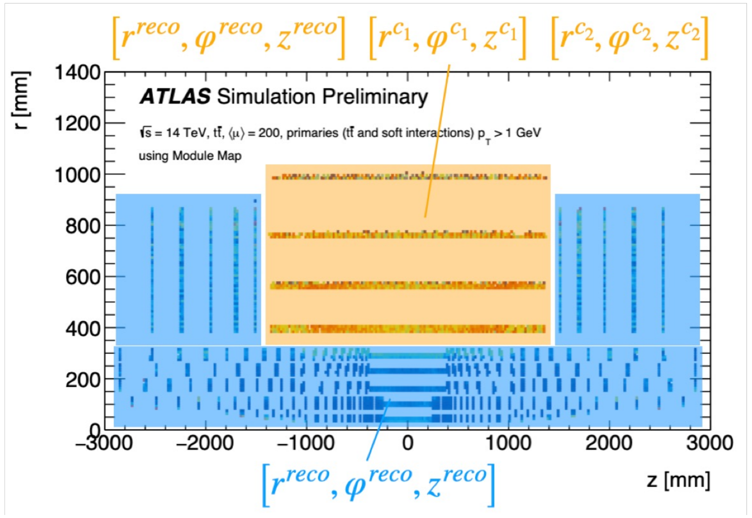




Low curvature: looks like a straight line

ATLAS assumes a straight line connecting true strip hit positions and beam spot, bad for tracks with large curvature
Hit position estimator has no access to curvature
→ poor resolution in z, O(cm)

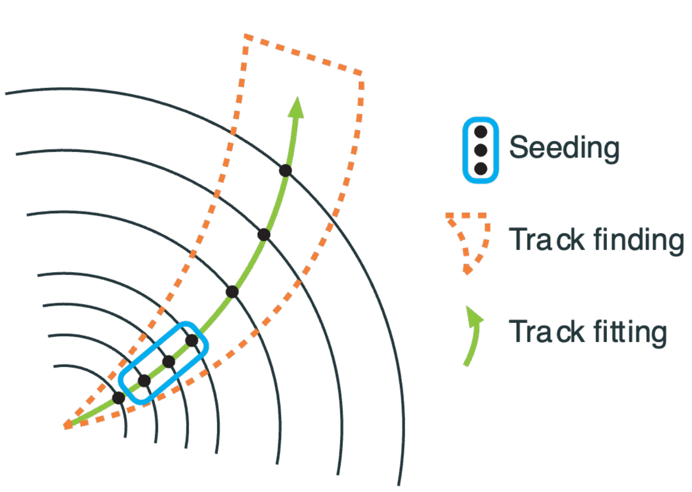
Large curvature: looks like an **arc**



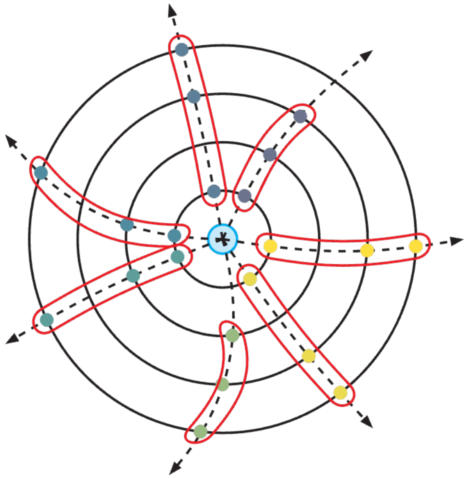
GNN trained to identify true connection → has indirect access to curvature, can learn better representation if given low-level cluster information.

The legacy Kalman Filter

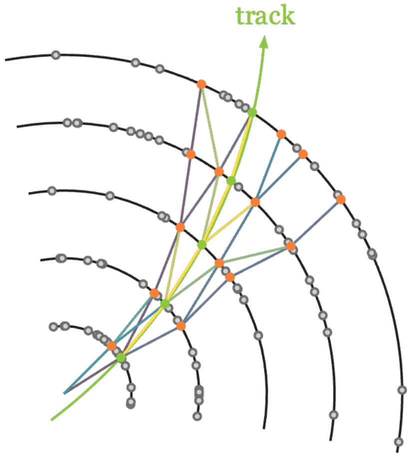
1. A large number of 3-hit tracklets are generated by a seed maker.
2. Estimate track parameters and a compatible search road of detector modules.
3. At each iteration, predict the next intersection of track with detector modules on the road, incorporate the hit on predicted module and update track params, modify the search road.
4. Estimate the track params from all hits on track.
5. Reverse the search direction to incorporate hits prior to the first seed hit.



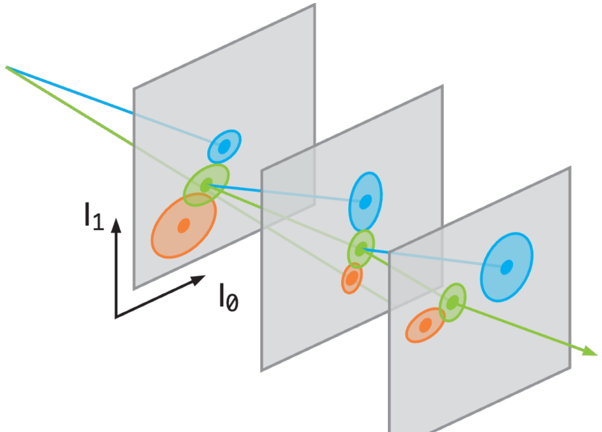
Overview



Seed Maker



Track Maker



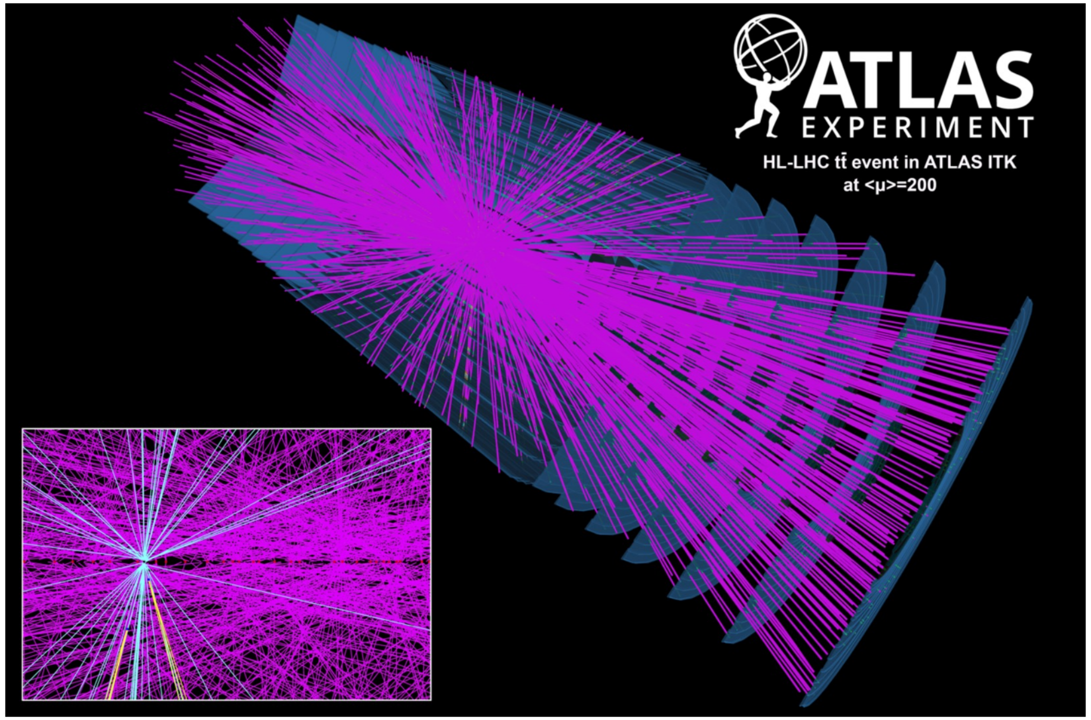
Iterative hit search

Challenge to the Inner Tracker under HL-LHC

Increased luminosity = increased number of p-p interactions per BX (pile-up). In HL-LHC, pile-up ranges 140-200. Run 3 pile-up $\langle\mu\rangle = 60$.

High pile-up $\longrightarrow$ higher occupancy in the tracker $\longrightarrow$ Replace the Inner Detector (ID) by a high-granularity Inner Tracker (ITk), PIXEL 50x50 μm^2 and 25x100 μm^2,vs. 50x400 μm^2 and 50x250 μm^2 in ID.

More granularity = more computing resources and time required $\longrightarrow$ **seek new software technologies for tracking**.



Event display of a top quark pair production simulated collision at $\langle\mu\rangle = 200$, with only pT > 1 GeV tracks selected. (ATLAS upgrade tracking event display)

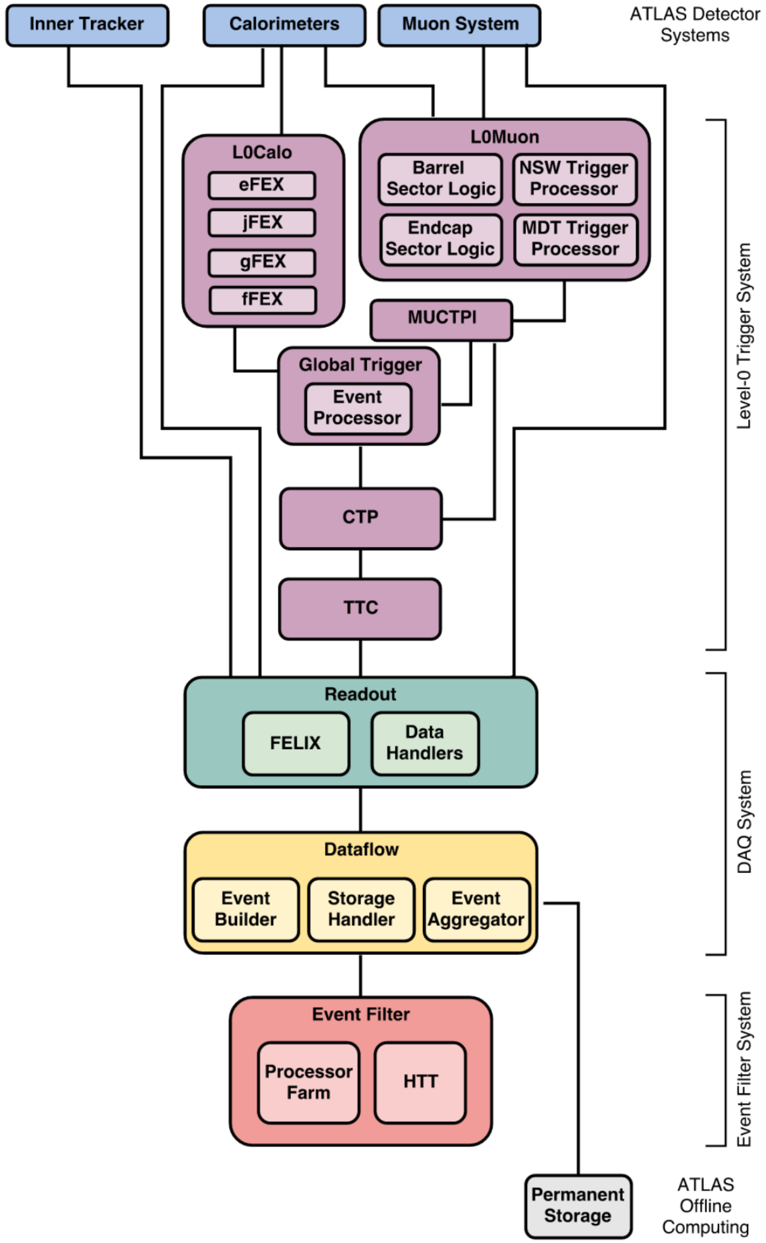
$$\langle\mu\rangle = \frac{\mathcal{L}\sigma_{tot}(pp)}{f_{BX}} = \frac{5\times10^{-5} fb^{-1} s^{-1} \times 1.17 \times 10^{14} fb}{40\times10^{6} BX.s^{-1}} \simeq 146\ BX^{-1}$$

GNN4ITk CHEP2024 contributions

1. Improving Computational Performance of ATLAS GNN Track Reconstruction Pipeline (speaker: Alina Lazar)
2. High Performance Graph Segmentation for ATLAS GNN Track Reconstruction (speaker: Daniel Murnane)
3. EggNet: An Evolving Graph-based Graph Attention Network for End-to-end Particle Track Recontruction (speaker: Jay Chan)
4. Energy-efficient graph-based algorithm for tracking at the HL-LHC (speaker: Heberth Torres)

ATLAS trigger system

10 kHz event written to tape.
Must do track reconstruction on Hz scale.



Event rate: 40 MHz

after hardware-based L0 trigger: 1 MHz

detailed detector readout after L0 accept

after event filter (to tape): 10 kHz