Physics Performance of the ATLAS GNN4ITk Track Reconstruction Chain

Sylvain Caillou[1], *Paolo* Calafiura[2], *Xiangyang* Ju[2,*], *Daniel* Murnane[2], *Tuan* Pham[3], *Charline* Rougier[1], *Jan* Stark[1], and *Alexis* Vallier[1,**] (On behalf of the ATLAS Collaboration)

[1]L2IT, Laboratoire des 2 Infinis—Toulouse, Toulouse, France
[2]Scientific Data Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[3]Physics Department, University of Wisconsin-Madison, Madison, WI 53706, USA

Abstract.
Particle tracking is vital for the ATLAS physics programs. To cope with the increased number of particles in the High Luminosity LHC, ATLAS is building a new all-silicon Inner Tracker (ITk), consisting of a Pixel and a Strip subdetector. At the same time, ATLAS is developing new track reconstruction algorithms that can operate in the HL-LHC dense environment. A track reconstruction algorithm needs to solve two problems: track finding for building track candidates and track fitting for obtaining track parameters of those track candidates. Previously, we developed GNN4ITk, a track-finding algorithm based on a Graph Neural Network (GNN), and achieved good track-finding performance under realistic HL-LHC conditions. Our GNN pipeline relied only on the 3D spacepoint positions. This work introduces heterogeneous GNN models to fully exploit the subdetector-dependent features of ITk data, improving the performance of our GNN4ITk pipeline. In addition, we interfaced our pipeline to the standard ATLAS track-fitting algorithm and data model. With that, the GNN4ITk pipeline produces full-fledged track candidates that can be used for any downstream analyses and compared with the other track reconstruction algorithms.

1 Introduction and the GNN4ITk pipeline

The High Luminosity LHC (HL-LHC) is designed to reach a peak instantaneous luminosity of $7.5 \times 10^{34}\text{cm}^{-2}\text{s}^{-1}$, which corresponds to an average of about 200 inelastic proton-proton collisions per beam-crossing. To keep the same or better performance than the current inner detector, ATLAS will build a new all-silicon tracking system, ITk [1, 2], which consists of a Pixel detector at a small radius and a large area Strip detector surrounding it. The Pixel detector consists of about five billion finely segmented silicon sensors, most of which have a pitch of $50 \times 50\,\mu\text{m}^2$ and the rest $25 \times 100\,\mu\text{m}^2$. The Strip detector comprises 23,000 long and skinny silicon sensors ($75.5\,\mu\text{m} \times 24.1$ or 48.2 mm).

Our previous study [3] showed that a Graph Neural Network (GNN)-based pipeline, namely *GNN4ITk*, demonstrates a high tracking reconstruction efficiency for $t\bar{t}$ events at $\sqrt{s} = 14$ TeV with pileup of $\langle\mu\rangle = 200$. However, the GNN does not perform well in the

[*]e-mail: xju@lbl.gov
[**]Copyright CERN for the benefit of the ATLAS Collaboration. CC-BY-4.0 license.

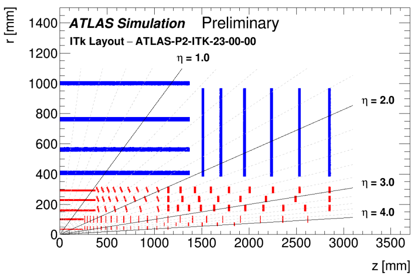

Figure 1. A schematic depiction of the ITk [6].

barrel strip detector region (see Fig. 1), partially due to the poor space point resolutions in the z-direction. Furthermore, no track fitting was performed for those track candidates. In this note, we will present an improved GNN that accommodates the heterogeneity of tracking data and show the first end-to-end GNN-based track reconstruction workflow for the ITk detector. All results are made public at Ref. [4], and the software for training and inference at Ref. [5].

2 ITk and Track reconstruction

Fig. 1 shows the ITk layout [6], which is constituted of pixel (in red) and strip (in blue) subdetectors. The pixel subdetector is near the beam pipe, contains finely segmented silicon sensors needed to separate tracks, comprises 9416 silicon modules, and drives impact parameter resolution. The strip subdetector covers a large area, is away from the beam pipe, and drives the track momentum and η resolution. It comprises 8944 silicon modules, with a strip detector tilted at a stereo angle installed on each side. The barrel strip detector region is represented by the horizontal blue segments in Fig. 1.

The current ATLAS track reconstruction chain takes three steps: cluster formation, track finding, and track fitting. It starts with forming clusters from raw measurements in the pixel and strip subdetectors, namely *cluster formation* [7]. A connected component analysis is used to cluster together pixel and strip channels above a charge threshold. From these clusters, three-dimensional measurements referred to as space points are created. Each cluster equates to one space point in the pixel subdetector, while in the strip subdetector, clusters from both sides of a strip module must be combined to obtain a 3D space point. A "ghost" space point may be formed due to combinatorics of four clusters resulting from two close-by particles interacting with both sides of close-by strip modules.

The GNN4ITk pipeline [3], a *track finding* algorithm, takes a set of space points as inputs and produces a list of track candidates as outputs. Each track candidate is a list of indices of space points ordered by the distance away from the collision point. The GNN4ITk pipeline, schematically pictured in Fig. 2, contains three discrete steps: graph construction, edge labelling, and graph segmentation. Graph construction is to create connections between two space points that could potentially be from the same particle. In this way, each collision event recorded by the ITk becomes an attributed graph, in which nodes are space points and node attributes are space point coordinates, and edges are those created connections and edge attributes are the geometric quantities calculated from the node attributes. True edges are the edges that connect two space points from a target particle. Edge labelling assigns an edge classification score to each edge in the graph with a GNN, which is trained to assign a high score to true edges and a low score to other edges. Lastly, graph segmentation is to build track candidates from the graph based on the edge classification scores.

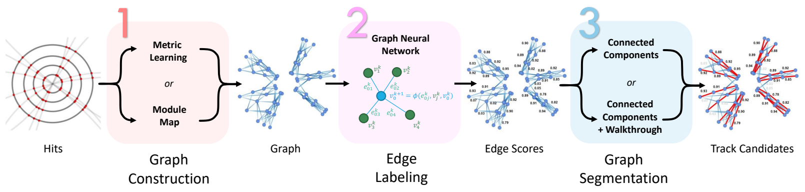

Figure 2. Schematic overview of the GNN-based track finding pipeline [3]

Finally, track candidates are fitted using a global χ^2 method to obtain a high-precision track parameter estimate. The method is based on the Newton-Raphson method and uses an interactive approach to find the best fit to a set of measurements of a track left in the detector by a charged particle traversing the detector. The track fit accounts for the effects of multiple Coulomb scattering of the particle with the detector components. A detailed description of the global χ^2 method can be found in Ref [8]. We first converted each GNN-based track candidate to *a list of raw measurements*, and then used *a pion hypothesis* to model the energy loss from the Coulomb scattering and a linear conformal mapping technique to estimate *initial track parameters*. The three elements are the inputs to the ATLAS global χ^2 track fitting.

3 Heterogeneity in the GNN4ITk pipeline

The ITk detector consists of Pixel and Strip detectors, recording different information. Specifically, each pixel space point has 3D coordinates $[r, \phi, z]$ in the global coordinate frame [1], 3D cluster position coordinates $[r_{\text{cl}}, \phi_{\text{cl}}, z_{\text{cl}}]$, and cluster shape variables [9], which are not used by the GNN. On the other hand, each strip space point has 3D coordinates and two 3D cluster positions $[r^1_{\text{cl}}, \phi^1_{\text{cl}}, z^1_{\text{cl}}, r^2_{\text{cl}}, \phi^2_{\text{cl}}, z^2_{\text{cl}}]$. Depending on how to treat the heterogeneity in the ITk data, the GNN architecture can be homogeneous or heterogeneous. Our previous publication [3] uses a homogeneous GNN, which harmonizes the ITk data by only using the space point 3D coordinates as inputs. A poor GNN performance was observed in the barrel strip region, where the GNN per-edge purity is less than 70% for a GNN per-edge efficiency of above 98%, while in other regions, the GNN per-edge purity is above 90% for the same efficiency. In this work, we append the cluster position coordinates to the space point 3D coordinates. Specifically, strip barrel space points have input features of $[r, \phi, z, r^1_{\text{cl}}, \phi^1_{\text{cl}}, z^1_{\text{cl}}, r^2_{\text{cl}}, \phi^2_{\text{cl}}, z^2_{\text{cl}}]$ and other space points have $[r, \phi, z, r, \phi, z, r, \phi, z]$ with their space point features repeated to reach the same length of features as the barrel strip space points. In addition, the pseudorapidity of the space point or the cluster position (η or η_{cl}) is added to help GNN converge faster. We call this model, which is retrained from scratch, *Extended GNN*, and re-evaluate the track reconstruction efficiency.

The second approach is to use a heterogeneous graph neural network (*HeteroGNN*), where we use different neural networks to encode the pixel and strip space points. Simi-

[1]ATLAS uses a right-handed coordinate system with its origin at the nominal interaction point (IP) in the center of the detector and the z-axis along the beam pipe. The x-axis points from the IP to the center of the LHC ring, and the y-axis points upwards. Polar coordinates (r, ϕ) are used in the transverse plane, ϕ being the azimuthal angle around the z-axis. The pseudorapidity is defined in terms of the polar angle θ as $\eta = -\ln\tan(\theta/2)$. Angular distance is measured in units of $\Delta R \equiv \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$.

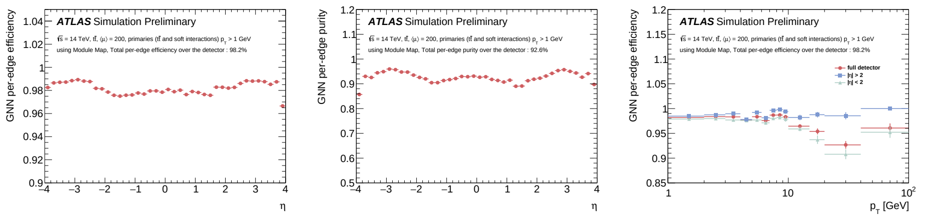

Figure 3. Graph Neural Network per-edge efficiency (left) and purity (middle) as a function of particle η and per-edge efficiency as a function of p_{T} (right).

larly, we use different encoders for edges connecting different sub-detectors. In this way, the heterogeneity of the input data is kept and made homogeneous by the encoder networks in the GNN. We name this GNN as *Heterogeneous GNN*.

4 Results

The GNN4ITk aims to reconstruct primary particles while explicitly excluding electrons as well as secondary particles from interactions with the detector material. These primary particles should have $p_{\mathrm{T}} > 1$ GeV, be produced at $R < 26$ cm and $|\eta| < 4$, and leave at least three space points. They are referred to as "target particles" in the following.

To evaluate the GNN performance, we define the per-edge efficiency as the ratio of the number of true edges passing a given threshold on the *Extended GNN* score and the number of true edges. True edges are defined as edges connecting successive space points of target particles. Figure 3 shows the results for GNN score $s > 0.5$ using graphs constructed with a Module Map [3] that is a dictionary-based graph construction algorithm and 100% efficient (i.e. that includes all true edges). The small inefficiency in the last bin (highest η) is expected to be recoverable by using more simulated data for training. The per-edge efficiency is the same as the previous publication [3], but the per-edge purity is improved from about 70% to over 90% in the barrel region, thanks to the added cluster information for the strip barrel space points. The per-edge efficiency is above 98% for particles with $p_{\mathrm{T}} < 10$ GeV and decreases to about 90% for higher p_{T} particles, which may be caused by a much lower number of high p_{T} tracks in the training data. The degraded GNN performance for high p_{T} particles can be mitigated by using larger weights for high p_{T} particles in the training loss function. Figure 4 shows the per-edge efficiency and purity in the (z, r) plane of true edges. The z and r position of an edge is the position of the inner-side space point. The per-edge purity in the strip barrel region is greatly improved compared with the previous results [3].

Track candidates are built from the graph and the classification scores using a connected components algorithm with a loose requirement on the classification scores followed by a walk-through algorithm. For the efficiency calculation, tracks are required to match generated target particles. The Module Map is used to create the graphs. The efficiency is reported for two alternative truth-matching requirements: the standard truth-matching uses the same requirement as in Ref [2] ($P_{match} > 0.5$), and the strict truth-matching requires tracks to include all space points from one generated particle and only space points from that particle, thus resulting in lower efficiency. Figure 5 compares the tracking reconstruction efficiency between the previous publication ("CTD 2022") and the *Extended GNN* ("HeteroData"). In both cases, the fake track rate and duplication rate are each found to be $O(10^{-3})$ using standard truth-matching. The fraction of tracks produced by the connected components algorithm

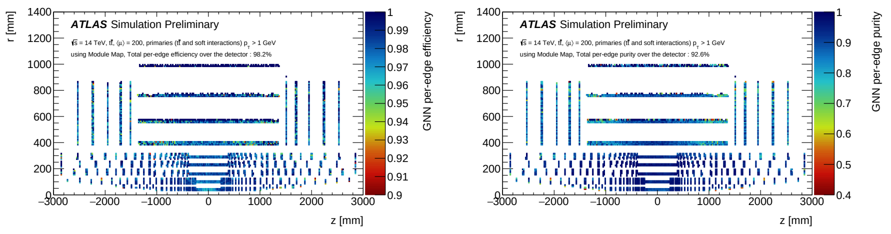

Figure 4. Performance of the *Extended GNN* edge classifier in the (z, r) plane. The position in (z, r) of an edge is defined by the position of its source node.

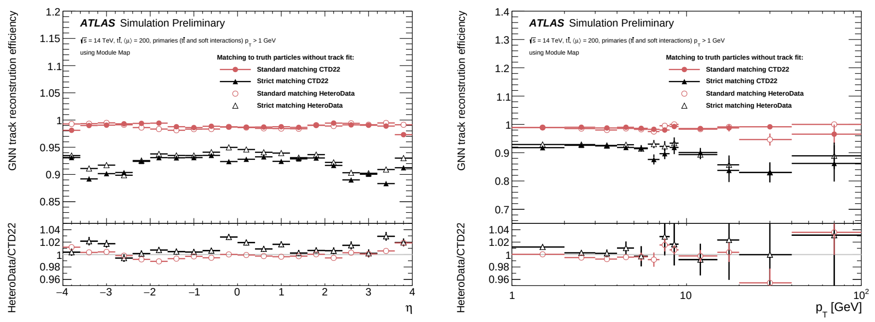

Figure 5. Graph Neural Network track reconstruction efficiency versus η for target particles.

without any further processing by the walk-through algorithm is 35% for the initial GNN, and 42% for the new GNN.

We compare the track contents and track parameter resolutions of the GNN4ITk-based track finding with those of the standard tracking reconstruction for ITk [10], which uses the Combinatoric Kalman Filter (CKF) for track finding. Tracks found by the CKF are selected using the standard quality requirements listed in Ref. [10] and are required to satisfy $p_\mathrm{T} > 1$ GeV. Tracks found by the GNN are required to satisfy criteria that are slightly less strict: at least 6 silicon hits, transverse impact parameter $|d_0| < 20$ mm, longitudinal impact parameter $|z_0| < 25$ cm and $p_\mathrm{T} > 1$ GeV. Figure 6 compares the average number of pixel and strip clusters per track as a function of the target particle η. Tracks from the GNN have the same number of pixel clusters per track as the default ITk reconstruction, but have fewer strip clusters per track, possibly due to missing clusters and wrongly assigned clusters. Clusters that do not form a space point will never enter the GNN4ITk pipeline. If a ghost space point is used in the GNN tracks, one of the true clusters from the particle may be missing. Figure 5 also shows the relative track p_T resolution as a function of target particle η. The target particles are required to satisfy $p_\mathrm{T} > 2$ GeV to avoid turn-on effects. The same set of 100 simulated events is used for evaluation.

Heterogeneous GNN (*HeteroGNN*) can use different input data types by construction. We used graphs constructed using Metric Learning [3] and trained *Extended GNN* and *HeteroGNN* for comparisons. We select a GNN score of 0.09 for *Extended GNN* and 0.08 for *HeteroGNN* so that the average per-edge efficiency is 98%. The average per-edge purity is 94% for both models. No significant improvement is seen for *HeteroGNN*. However, the

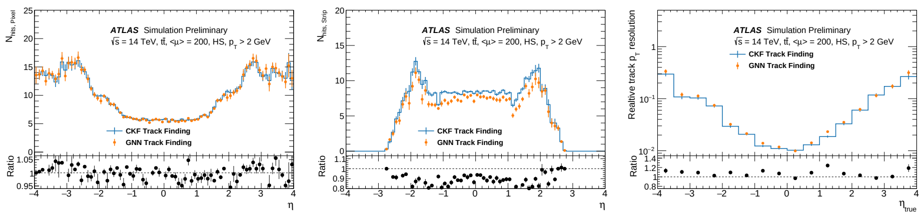

Figure 6. Average number of pixel (left) and strip clusters (middle) per track, and the relative track p_{T} resolution (right) as a function of η of the associated hard scattering primary particle.

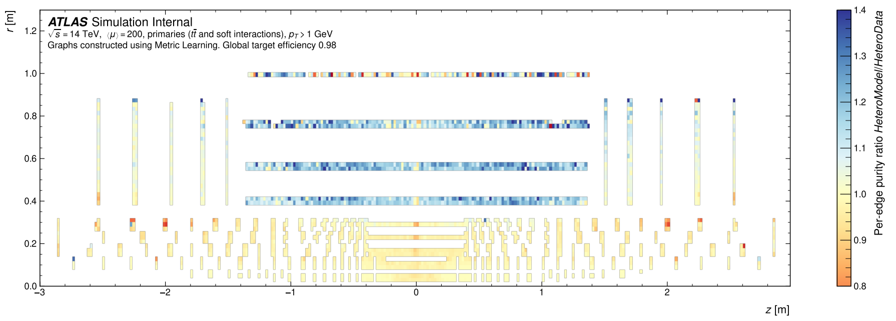

Figure 7. Per-edge purity ratio between *HeteroGNN* ("*HeteroModel*") and *Extended GNN* ("*HeteroData*") in the (z, r) plane.

HeteroGNN results in an averaged improvement of 11% in per-edge purity in the strip barrel and an averaged loss of 1%in the pixel sub-detector as shown in Fig. 7.

5 Conclusion

The GNN4ITk pipeline provides not only competitive track efficiency but also high quality track parameter resolutions. The GNN-based track finding is integrated into the ATLAS tracking framework, enabling us to use the existing tools to perform track fitting and evaluate tracking performance. With the above improvements, the GNN-based particle tracking steps steadily towards production-level quality. We are investigating further different heterogeneous GNN architectures and their impacts on particle tracking.

This work was supported through the Scientific Discovery through Advanced Computing (SciDAC) program funded by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research, under Contract DE-AC02-06CH11357.

References

[1] ATLAS Collaboration, *ATLAS Inner Tracker Strip Detector: Technical Design Report*, ATLAS-TDR-025; CERN-LHCC-2017-005 (2017)

[2] ATLAS Collaboration, *ATLAS Inner Tracker Pixel Detector: Technical Design Report*, ATLAS-TDR-030; CERN-LHCC-2017-021 (2017)

[3] ATLAS Collaboration, *ATLAS ITk Track Reconstruction with a GNN-based pipeline*, ATL-ITK-PROC-2022-006 (2022)

[4] ATLAS Collaboration, https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PLOTS/IDTR-2023-01/ (2023)

[5] GNN4ITk Team, https://github.com/GNN4ITkTeam/CommonFramework (2023)

[6] ATLAS Collaboration, *Expected tracking and related performance with the updated ATLAS Inner Tracker layout at the High-Luminosity LHC*, ATL-PHYS-PUB-2021-024 (2021)

[7] ATLAS Collaboration, Eur. Phys. J. C **77**, 673 (2017), `1704.07983`

[8] ATLAS Collaboration, Eur. Phys. J. C **80**, 1194 (2020), `2007.07624`

[9] P.J. Fox, S. Huang, J. Isaacson, X. Ju, B. Nachman, JINST **16**, P05001 (2021), `2012.04533`

[10] ATLAS Collaboration, *Expected Tracking Performance of the ATLAS Inner Tracker at the HL-LHC*, ATL-PHYS-PUB-2019-014 (2019)