Proceedings of the CTD 2023
PROC-CTD2023-18
7th March 2024

Physics Performance of the ATLAS GNN4ITk Track Reconstruction Chain

Heberth Torres[1], Jared Burleson[2], Sylvain Caillou[1], Paolo Calafiura[3], Jay Chan[3], Christophe Collard[1], Xiangyang Ju[3], Daniel Murnane[3], Mark Neubauer[2], Minh-Tuan Pham[4], Charline Rougier[1], Jan Stark[1], Alexis Vallier[1].

[1]*Laboratoire des 2 Infinis – Toulouse, Univ. Paul Sabatier, CNRS/IN2P3, Toulouse, France;*
[2]*Department of Physics, University of Illinois, Urbana IL, USA;*
[3]*Scientific Data Division, Lawrence Berkeley National Laboratory, Berkeley CA, USA;*
[4]*Department of Physics, University of Wisconsin, Madison WI, USA.*

On behalf of the ATLAS Collaboration

ABSTRACT

Graph-based techniques and graph neural networks (GNNs) in particular are a promising solution for particle track reconstruction at the HL-LHC. Simulations of the HL-LHC environment produce noisy, heterogeneous and ambiguous data. We present an upgrade to the ATLAS GNN4ITk pipeline that allows detector regions to be handled heterogeneously. We perform direct comparisons of our results with those of existing tracking algorithms on a range of physics metrics, including reconstruction efficiency, track reconstruction performance in dense environments, and track parameter resolutions. By integrating this solution within the offline ATLAS Athena framework, we also explore different reconstruction chain configurations, for example using the GNN4ITk pipeline together with traditional techniques for track cleaning and fitting.

PRESENTED AT

Connecting the Dots Workshop (CTD 2023)
October 10-13, 2023

ATL-SOFT-PROC-2023-047
08/03/2024

1 Introduction

The reconstruction of charged particle tracks at the LHC experiments is a compute-intensive task. At the high luminosity LHC (HL-LHC) we expect a hit density in the detector four times higher than now, and upgraded detectors like the future ATLAS Inner Tracker (ITk) with extended pseudorapidity* coverage up to $|\eta| < 4$ [1–3]. ATLAS Computing Model projections [4] show that we need to do aggressive computing R&D in order to keep the resource consumption within our capacity at the HL-LHC. Track reconstruction is one of the main items to be addressed. Graph Neural Networks (GNNs) running on hardware accelerators are a promising solution for this computing challenge [5–7].

In this document, we will first describe a GNN tracking chain developed within the ATLAS Collaboration, GNN4ITk, and next we will present the physics performance achieved with it, including tracking efficiency, track hit content and the resolution of track parameters.

Three sets of plots showing performance of the GNN4ITk chain have been published by ATLAS: The first results obtained with the realistic full simulation of the ATLAS ITk, presented at CTD 2022 [8, 9]; next, results showing improved performance, presented in May at the CHEP 2023 conference [10]; finally the physics performance results [11] released for CTD 2023 and described in these proceedings.

2 Simulated event sample

MC event simulation samples are used for this study, specifically pp collisions at $\sqrt{s} = 14$ TeV, with a $t\bar{t}$ pair in the final state and at least one top quark decaying leptonically, with an average 200 pp interaction pileup per bunch crossing, as expected at the HL-LHC, and a full simulation of the ATLAS detector based on GEANT4. For the new plots showing physics performance, an updated ITk layout version 23-00-03 was used in the detector simulation [12]. Among the changes with respect to the previous layout version, the new one has a reduced radius of the innermost pixel layer, and a more detailed and accurate model of the passive material.

For the training of the GNN4ITk chain, the target particles are required at truth level to have $p_{\mathrm{T}} > 1\,\mathrm{GeV}$ and to leave at least 3 hits in ITk. Electrons are excluded from the target particles; the reconstruction of electron tracks is a complicated case that will be addressed later on. Only primary particles are considered as target particles, i.e. secondary particles resulting from interactions with the detector material are excluded. Particles resulting from B hadron decays satisfying the p_{T} and hit requirements are included within the primary and target particles; their reconstruction is of major importance for b-jet tagging.

3 Description of the GNN4ITk tracking chain

A graph is a set of nodes and edges, with each edge connecting a node pair. In our case, the **nodes** represent ITk detector measurements (hits) belonging to an event. An **edge** represents the hypothesis that the two nodes associated to it correspond to two successive hits from the same particle.

The GNN4ITk consists of three steps:

1. In the first step the hits or nodes are connected together to build up a graph per event.
2. Next comes the core of the GNN4ITk, in which a graph neural network is used to classify the edges as true or fake hit-pair connections. In this step, a score is assigned to each edge.
3. Finally the graph is segmented based on the edge scores, to build up the track candidates, i.e. collections of hits likely coming from the same particle.

The following sections describe each of these three steps. This tracking chain is implemented in the ACORN framework [13].

*ATLAS uses a right-handed coordinate system with its origin at the nominal interaction point (IP) in the centre of the detector and the z-axis along the beam pipe. The x-axis points from the IP to the centre 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 $\eta = -\ln\tan(\theta/2)$.

3.1 Graph construction

Currently we have two alternative methods for the graph construction: Metric Learning and Module Map. Both methods produce graphs with around 300 000 nodes and a few million edges per event.

In the **Metric Learning** method, a multilayer perceptron (MLP) is trained to embed the nodes into a latent space, in which hits from the same particle are close to each other. Using a clustering algorithm, nodes that are close to each other in that latent space are connected together introducing edges in the graph. In a second step, another MLP is used to filter the edges, to reduce the size of the graphs.

For the **Module Map** method, a lookup table is built containing all the possible triple-module directional connections. It has about one million possible combinations. This module map has been prepared using 90 000 simulation events. For the graph construction, in a first step, edges are introduced in the graph based on the module map. Next, the edges are filtered using geometric cuts.

The physics performance results shown in Section 4 were obtained using the Module Map method for the graph construction.

The per-edge efficiency for both methods is close to 100%, with inefficiencies reaching just a few percent in specific regions as a function of the truth particle pseudorapidity η or transverse momentum p_{T} [8]. We believe that these inefficiencies can be reduced using better simulation samples, either of larger size or with a modified p_{T} spectrum to better populate the low statistics tail.

3.2 Edge classification

A graph neural network (GNN) is trained to identify the true edges, i.e. those really connecting two successive hits from the same particle. It is done based on geometric node and edge features, including among others the hit coordinates r and ϕ as node features, and variables like $\Delta\eta$ and $\Delta\phi$ as edge features. Cutting on the GNN edge score, we can reduce the number of edges by two orders of magnitude, to the order of 10^4 edges per event.

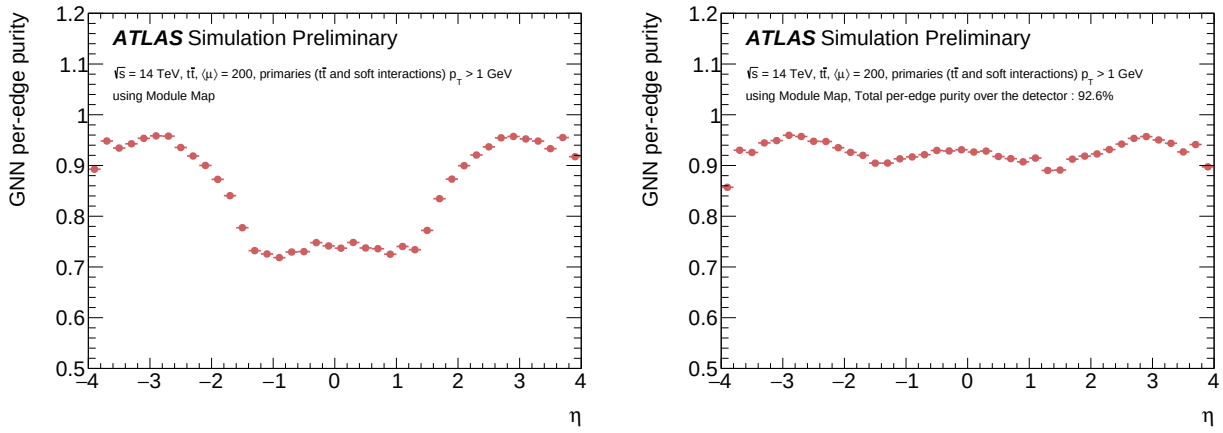
Within our GNN model, in a first step the node and edge features are encoded into a latent space of 128 dimensions. It is done using two independent MLPs, one to encode node features and another for the edge features. Next, within the core of the GNN architecture, composed of Interaction Network Layers [14], the node and edge latent features are combined and projected into new latent spaces using two MLP blocks, and the information is propagated through the graph to neighboring nodes and edges. The propagation of the information is done through eight message-passing iterations. Finally, the edge features in the latent space resulting from the last iteration are decoded with an MLP into a classification edge score. Each MLP within this model consists of three layers.

Compared to the GNN presented at CTD 2022, the new version uses a non-recurrent interaction network, which means that the sets of parameters of the MLP blocks used in each of the eight iterations are different from each other; before, a common set of weight and bias parameters were optimized for the eight iterations. In addition, Batch Normalization [15] layers have been added after each linear layer of encode and Interaction Network MLPs. The most important change is that now the input data has an heterogeneous format matching the pixel and strip technology heterogeneity of the detector; more details about this heterogeneous data format are given below.

The new GNN has a significantly better performance than the CTD 2022 version. For score cuts optimized to have an inclusive 98% edge efficiency, the purity in the central η region improved from 70% to 90% (Figure 1). The source of the low purity for the previous GNN was identified to be poor spacial resolution on the barrel-strip hits.

Instead of a 3D measurement like in a pixel layer, when a particle traverses a strip module we get two 2D measurements, which we need to combine to get the 3D one. As shown in Figure 2(a), in a strip module, we have two strip sensor planes with a small stereo angle between the direction of the strips. In this example, when traversing a barrel-strip module, a particle fires the two strips (or two strip clusters) highlighted in brown, one at the radially inner sensor plane and one at the outer plane. The question is, where along the brown-strip length did the particle hit the inner sensor plane?[†]

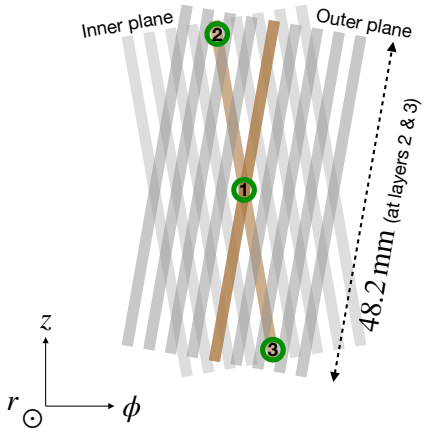
[†]In ATLAS, we reconstruct the strip hit as the point where the particle traverses the inner plane of the module.



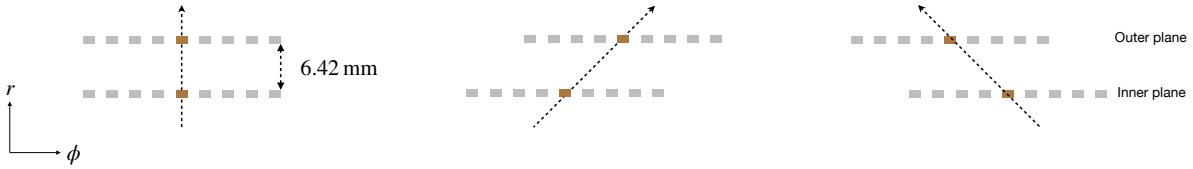
(a) Old CTD 2022 edge purity [8].

(b) CHEP 2023 edge purity [10].

Figure 1: Graph Neural Network per-edge purity (fraction of edges corresponding to true edges) versus η, for edges passing a threshold on the edge classification score that provides 98% per-edge efficiency.



(a) ϕz view.



(b) Case 1, ϕr view.

(c) Case 2, ϕr view.

(d) Case 3, ϕr view.

Figure 2: In this example, when traversing a barrel-strip module, a particle flying out of the page in Figure (a) has fired the two strips (or two strip clusters) highlighted in brown, one at the inner sensor plane and one at the outer plane. The question is, where along the brown-strip length did the particle hit the inner sensor plane? As illustrated in this figure, we cannot estimate the hit position along the strip length precisely without knowing the direction of the particle traversing the module.

To answer this question, the first thing that comes to our mind when looking at this figure is: The particle should have passed through the intersection of the two strips in this ϕz view, as pointed out by the green circle 1. However the two sensor planes are not directly one on top of each other, but there is a gap of 6.42 mm in between, as shown in Figure 2(b). Therefore that hypothesis is only valid if the particle traversed the module in a direction perpendicular to the module plane as in Figure 2(b). A particle traversing the module in a different direction leaning right or left as in Figures 2(c) or 2(d) could hit the inner plane at a different place, for instance the positions indicated by the green circles 2 or 3, still firing the same brown strip pair. It means that actually we cannot precisely estimate the hit position along the strip length without knowing the direction of the particle when traversing the module.

By default, the strip hits are reconstructed assuming that the particle follows a straight trajectory from the interaction point, which is a good assumption for high p_{T} particles. However it is not the right assumption for most of the particles we are tracking starting at 1 GeV, as they are bent in the transverse plane by the magnetic field, and therefore we obtain poor resolution hits. In the barrel, the resolution for the default strip hits is of the order of centimeters in the z direction, at the level of the length of the strips.

This issue has been addressed by passing to the GNN as input node features the coordinates of the two individual strip clusters, in addition to the coordinates of the hit reconstructed by default. That is done for strip-barrel hits, while for the rest of the hits we use only the reconstructed hit, which results in a heterogeneous format for the list of node features. This extra strip-cluster information fed to the GNN is the main source of the 20% purity improvement shown in Figure 1. Other alternatives currently under study include hand-engineered strip-edge features, based on an estimation of the particle direction when traversing the strip module, and the usage of a heterogeneous GNN model.

3.3 Graph segmentation

The graph segmentation is done in two steps:

- First a loose cut on the edge score is applied and the resulting graphs are processed with a connected component algorithm [9]. After this first stage, the set of hits with an unambiguous connection sequence is already considered a track candidate, and no further filtering is applied to it.
- On the other hand, sets of hits with ambiguous connection sequences, i.e. including multiple alternative paths, are processed with a walk-through algorithm [9] using a tighter edge score cut, to get the final track candidates.

The better the GNN edge classification performance, the more track candidates are produced by the connected component stage and the faster the reconstruction process.

3.4 Technical performance of the entire GNN4ITk chain

Based on the target particles defined in Section 2, a technical tracking efficiency is computed for two track-to-particle matching criteria, "standard" and "strict" matching. In the standard matching, a track candidate is required to have at least 50% of its hits originating from the target particle matched to it, in other words, 50% hit purity. In strict matching, we require 100% hit purity and, in addition, 100% of the target particle's hits to be present in the track candidate, in other words, 100% hit efficiency. The resulting tracking efficiency is around 93% for the strict matching, with variations between 90% and 95% as a function of η. In the case of the standard matching, the efficiency is around 99% and more uniform over η [10].

Fake tracks are defined as track candidates not matched to any particle; the rate of these is of the order of one per mil for the standard matching.

We had a first look at the technical performance of the GNN4ITk chain on single particle events using a dedicated GNN trained with a simplified dataset. Only particles leaving at least 7 (9) hits for $|\eta| < 2$ ($|\eta| > 2$) were considered. The resulting efficiency is shown in Figure 3 as a function of η, for muons and electrons with $p_{\mathrm{T}} = 10\,\mathrm{GeV}$.

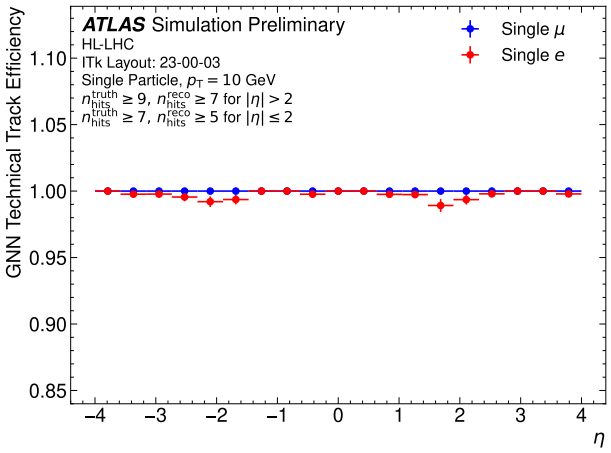


Figure 3: Track reconstruction technical efficiency as a function of the true pseudorapidity η for single particle events, for $p_T = 10$ GeV muons and electrons, without pile-up [11]. Particles are required to leave at least 7 (9) space points for $|\eta| < 2$ ($|\eta| > 2$). Track candidates are required to contain at least 5 (7) space points for $|\eta| < 2$ ($|\eta| > 2$).

4 Physics performance

Now with the results presented in this document, we reach another stage in the development and validation of this novel tracking algorithm: instead of monitoring only technical efficiencies, we are starting to study a more realistic efficiency, applying the requirements used in ATLAS to select tracks for the physics analysis, as well as efficiency for tracking within jets and impact parameter resolutions.

We are using the standard ATLAS software (Athena) to execute the χ^2 fit of the collections of hits resulting from the GNN4ITk. This fit takes into account the expected multiple scattering effects, as well as a detailed map of the magnetic field in ITk. From this fit, we obtain measurements of the five physics parameters characteristic of ATLAS track candidates, p_T, θ, ϕ, and the transverse and longitudinal impact parameters d_0 and z_0.

In addition, we have performed a direct comparison against the ATLAS implementation of the combinatorial Kalman filter (CKF). We have processed the same simulation sample with both tracking chains, the GNN4ITk and the CKF.

The tracks have been selected using the requirements listed in Table 3 in [12]. For the GNN4ITk tracks, three cuts are slightly looser than those in the table:

- Number of pixel plus strip hits ≥ 8,
- $|d_0| < 20$ mm,
- $|z_0| < 25$ cm.

Out of these three changes, the main difference is in the requirement for the total number of hits. The reason for applying this looser cut is the difference in the treatment of strip hits between the GNN4ITk and the CKF; more details are discussed below. In addition, all tracks are required to have $p_T > 1$ GeV, and we continue excluding electrons, which were not considered for the training of the GNN4ITk. The simulated particles matched to reconstructed tracks are required to satisfy $p_T > 2$ GeV to avoid turn-on effects.

The resulting efficiency is shown as a function of η and p_T in Figures 4(a) and 4(b). The blue points and yellow triangles correspond to the CKF and GNN4ITk respectively. As one can see, the efficiency reached by the GNN4ITk is roughly at the same level as the CKF one, with the GNN4ITk efficiency just a few percent below the CKF one. Given that it is the first time we have looked at this efficiency, and that the GNN4ITk has not been trained taking it into account, we are very satisfied with this result.

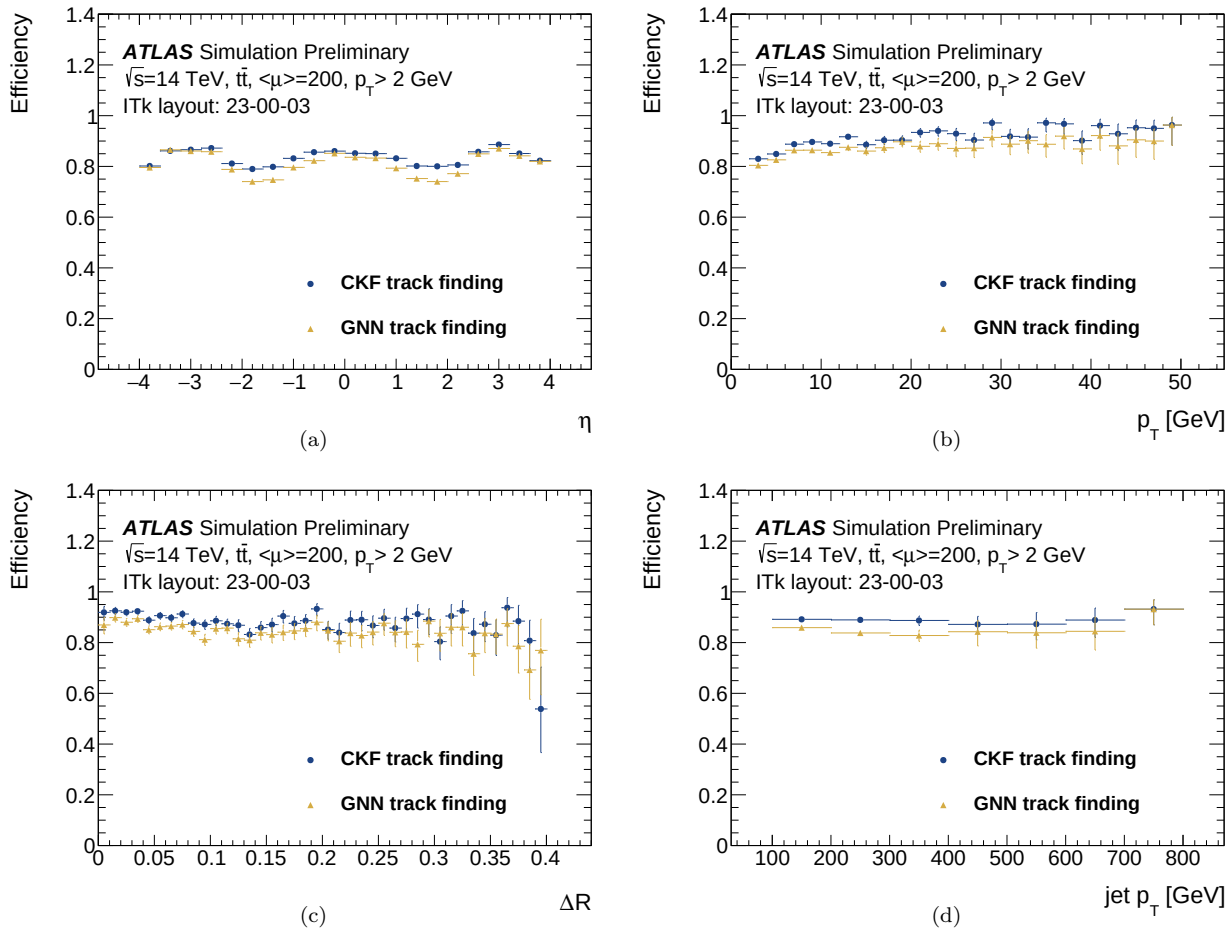
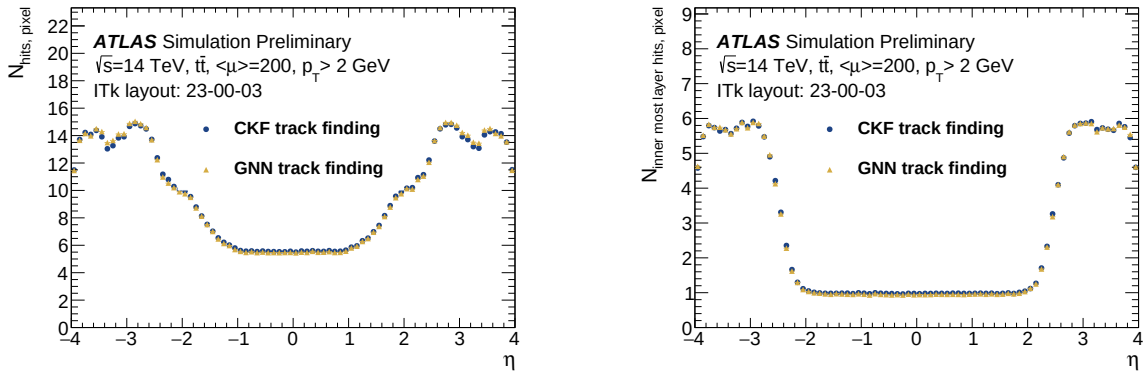


Figure 4: Track reconstruction efficiency as a function of the true pseudorapidity η (a) and p_{T} (b), and for tracks inside jets as a function of the angular separation ΔR between each track and the jet axis (c) and the jet p_{T} (d) [11].

Figures 4(c) and 4(d) show efficiency for tracking inside jets. Inside a jet, we have an environment dense in particles and hits in which track reconstruction is more challenging. Figure 4(c) shows the efficiency as a function of the angular separation ΔR between each track and the jet axis. The density of particles and hits in the core of a jet at small ΔR values is higher than in the external part of the jets. However the efficiency obtained is rather flat versus ΔR, without showing any sign of degradation at small ΔR. Figure 4(d) shows the efficiency as a function of the jet p_{T}. The higher is the jet p_{T}, the more dense is the core of the jet. Still the GNN4ITk provides a uniform efficiency as a function of jet p_{T} without degradation at high p_{T} values.

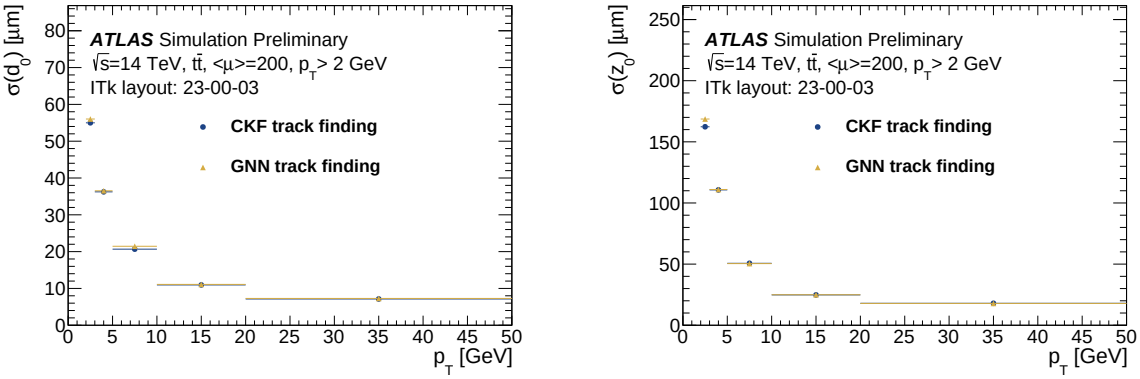
Figure 5 shows the pixel hit content, considering all pixel layers in Figure 5(a) and only the innermost pixel layer in Figure 5(b). The innermost pixel layer is of particular importance for the measurement of the track impact parameters, and therefore relevant for jet b-tagging. We observe that the pixel hit content for the GNN4ITK is compatible with that of the CKF, and both of them are compatible with what is expected for the ITk detector. Given this result for the pixel hit content, we expect good resolution for the impact parameters d_0 and z_0, and this is the case as shown in Figure 6, which presents the d_0 and z_0 resolution as a function of p_{T}.

Figure 7(a) shows the strip hit content, specifically counts of strip clusters. In the case of the GNN4ITk, we only consider full strip hits, each hit consisting of two strip clusters, with one cluster in each strip sensor plane of the hit module. It means that for the GNN4ITk we are only counting strip cluster pairs. If there



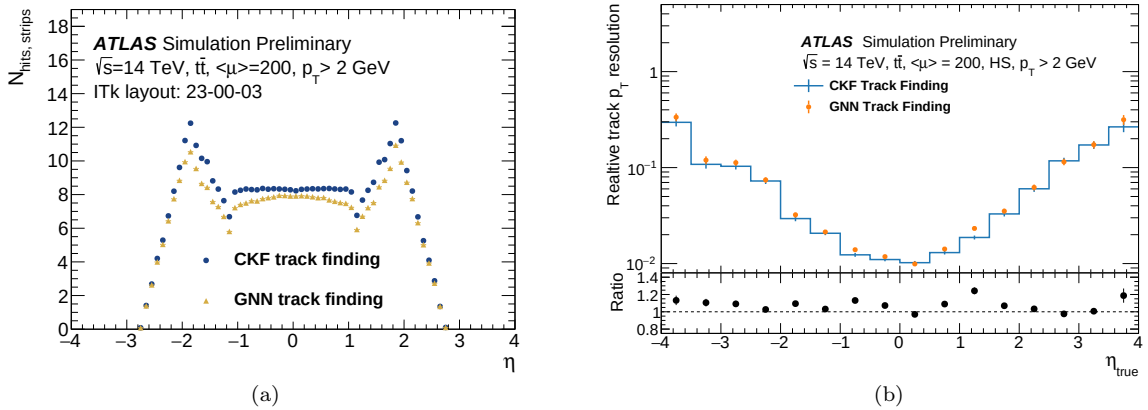
(a) (b)

Figure 5: Average number of pixel hits (a) and of pixel hits on the radially innermost layer (b) per track as a function of the true pseudorapidity η [11]. In the end-caps, the radially innermost modules at various disks are counted within this innermost layer.



(a) (b)

Figure 6: Track parameter resolution in d_0 (a) and z_0 (b) as a function of the true p_{T} for primary particles [11].



(a) (b)

Figure 7: (a): Average number of strip hits on a track as a function of the true pseudorapidity η [11]. (b): Relative track p_{T} resolution as a function of the true η for hard scattering primary particles [10].

is a particle passing through the border between two strip modules, firing strip singlets in each of the two neighboring modules, these singlets are not considered by the GNN4ITk, while they are taken into account by the CKF. That is why we observe lower strip cluster counts for the GNN4ITk compared to the CKF, and is why in the track selection we have applied a looser cut on the total number of hits for the GNN4ITk. Despite this difference, we obtain a competitive track p_{T} resolution as shown in Figure 7(b).

5 Conclusion and prospects

We have presented a first look at the physics performance of the ATLAS GNN4ITk tracking chain, for the HL-LHC, together with an apples-to-apples comparison against the Combinatorial Kalman Filter. The GNN4ITk provides competitive tracking efficiency, even in challenging dense environment inside jets, and good track parameter resolution. In addition, it has already been proven that the track reconstruction using a GNN with GPUs provides a promising computing speed [16].

There are still several work items to be completed before the GNN4ITk can be validated for production. They include:

- Further optimization and acceleration of the full chain, including acceleration of graph construction with GPU, the exploration of advanced machine learning architectures for pattern recognition, including heterogeneous and hierarchical GNNs.
- Further study of corrections to the strip-space-point positions, and investigations of the impact of the missing strip-cluster singlets.
- Continue the physics performance studies, including tracking of B-hadron decays and the reconstruction of electron tracks.
- Study robustness against detector effects, like dead modules, mis-alignment of the detector and beam-spot variations.
- Integration of the GNN4ITk software into ACTS [17] and the ATLAS Athena framework.

ACKNOWLEDGEMENTS

We thank our colleagues at the IN2P3 computing centre (CC-IN2P3) in Lyon (Villeurbanne) for the smooth operation of their GPU production platform, and for the successful deployment of a new experimental platform dedicated to machine learning developments that require large amounts of memory. Without these resources, the present studies would not have been possible.

This research was supported in part by the U.S. Department of Energy's Office of Science, Office of High Energy Physics, under Contracts No. DE-AC02-05CH11231 (CompHEP Exa.TrkX) and No. KA2102021 (US ATLAS Operations), and by the Exascale Computing Project (17-SC-20-SC), a joint project of DOE's Office of Science and the National Nuclear Security Administration. This research used resources from the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231.

References

[1] ATLAS Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, JINST **3** (2008) S08003 (cit. on p. 1).

[2] ATLAS Collaboration, *ATLAS Inner Tracker Strip Detector: Technical Design Report*, ATLAS-TDR-025; CERN-LHCC-2017-005, 2017, URL: https://cds.cern.ch/record/2257755 (cit. on p. 1).

[3] ATLAS Collaboration, *ATLAS Inner Tracker Pixel Detector: Technical Design Report*, ATLAS-TDR-030; CERN-LHCC-2017-021, 2017, URL: https://cds.cern.ch/record/2285585 (cit. on p. 1).

[4] *ATLAS Software and Computing HL-LHC Roadmap*, Geneva, 2022, URL: http://cds.cern.ch/record/2802918 (cit. on p. 1).

[5] S. Farrell et al., *Novel deep learning methods for track reconstruction*, 2018, arXiv: 1810.06111 [hep-ex] (cit. on p. 1).

[6] C. Biscarat et al., *Towards a realistic track reconstruction algorithm based on graph neural networks for the HL-LHC*, EPJ Web Conf. **251** (2021) 03047, URL: https://doi.org/10.1051/epjconf/202125103047 (cit. on p. 1).

[7] X. Ju et al., *Performance of a geometric deep learning pipeline for HL-LHC particle tracking*, The European Physical Journal C **81** (2021) 876, URL: https://doi.org/10.1140/epjc/s10052-021-09675-8 (cit. on p. 1).

[8] ATLAS Collaboration, *Track finding performance plots for a Graph Neural Network pipeline on ATLAS ITk Simulated Data*, IDTR-2022-01, 2022, URL: https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PLOTS/IDTR-2022-01/ (cit. on pp. 1–3).

[9] C. Rougier et al. on behalf of the ATLAS Collaboration, *CTD2022: ATLAS ITk Track Reconstruction with a GNN-based Pipeline*, 2023, URL: https://doi.org/10.5281/zenodo.8187248 (cit. on pp. 1, 4).

[10] ATLAS Collaboration, *Update of track finding performance plots for a Graph Neural Network pipeline on ATLAS ITk simulated data*, IDTR-2023-01, 2023, URL: https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PLOTS/IDTR-2023-01/ (cit. on pp. 1, 3, 4, 7).

[11] ATLAS Collaboration, *Track finding performance plots for a Graph Neural Network pipeline on ATLAS ITk simulated data*, IDTR-2023-06, 2023, URL: https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PLOTS/IDTR-2023-06/ (cit. on pp. 1, 5–7).

[12] 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, URL: https://cds.cern.ch/record/2776651 (cit. on pp. 1, 5).

[13] GNN4ITk Team, *A Charged-particle geOmetric Reconstruction Network (ACORN)*, 2023, URL: https://gitlab.cern.ch/gnn4itkteam/commonframework (cit. on p. 1).

[14] P. W. Battaglia et al., *Interaction Networks for Learning about Objects, Relations and Physics*, 2016, arXiv: 1612.00222 [cs.AI] (cit. on p. 2).

[15] S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, 2015, arXiv: 1502.03167 [cs.LG] (cit. on p. 2).

[16] A. Lazar et al., *Accelerating the Inference of the Exa.TrkX Pipeline*, Journal of Physics: Conference Series **2438** (2023) 012008, URL: https://doi.org/10.1088%2F1742-6596%2F2438%2F1%2F012008 (cit. on p. 8).

[17] X. Ai et al., *A Common Tracking Software (ACTS)*, 2023, URL: https://acts.readthedocs.io/en/latest/ (cit. on p. 8).