Improving Computational Performance of a GNN Track Reconstruction Pipeline for ATLAS

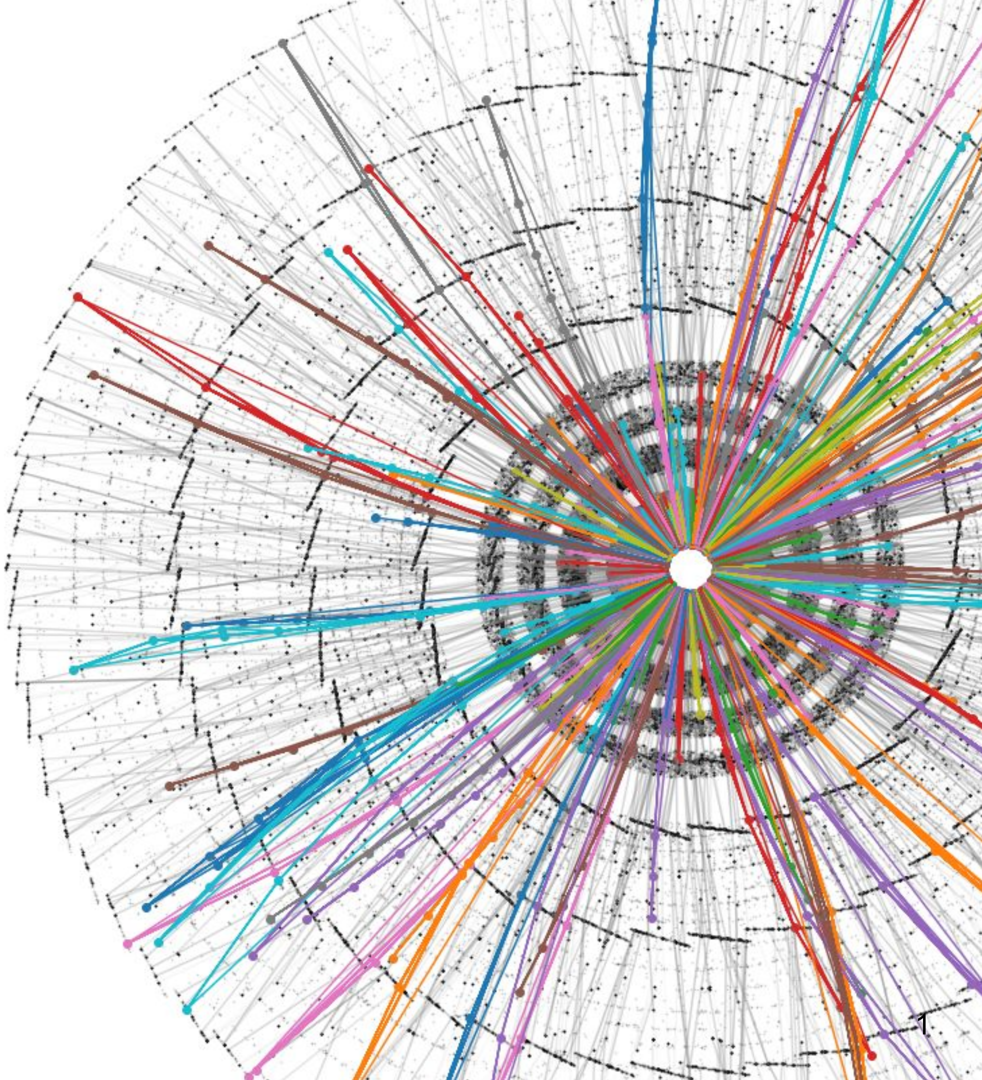
Daniel Murnane

On behalf of the ATLAS Collaboration

UNIVERSITY OF COPENHAGEN

BERKELEY LAB

ATLAS EXPERIMENT

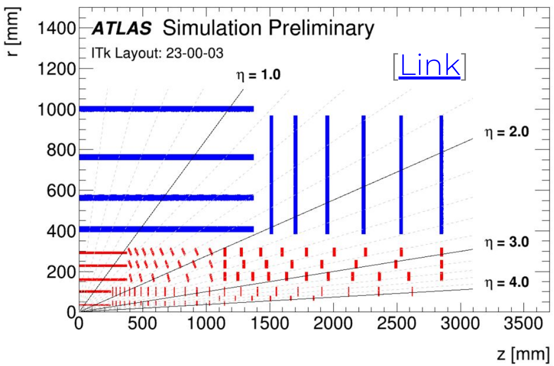


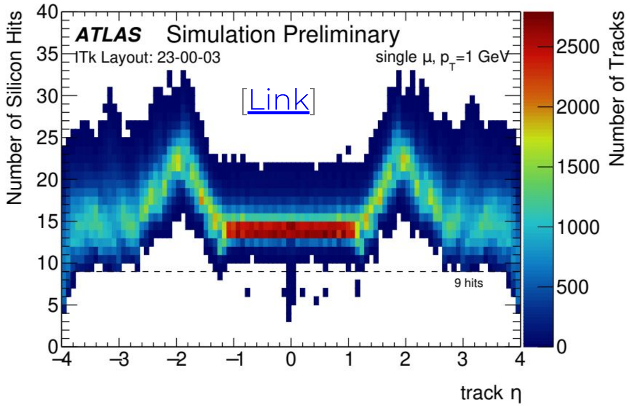
Outline

- Description of current pipeline
- Physics performance
- Acorn training, inference and evaluation framework
- Computational constraints for offline and online tracking
- Optimization research directions

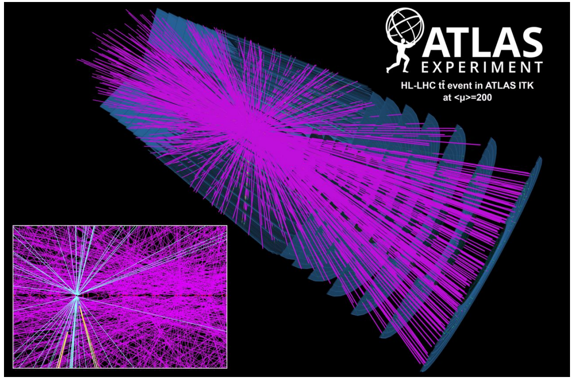
Physics Performance of GNN4ITk Pipeline

Tracking in ATLAS HL-LHC Inner Tracker (ITk)



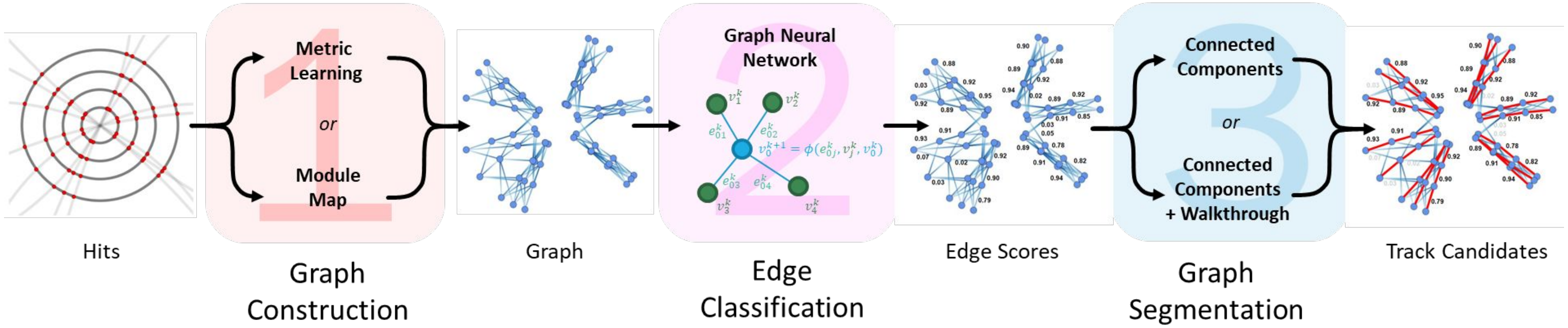


- Track finding requires associating each hit to a track candidate
- Number of hits per $pp \rightarrow t\bar{t}$ event: 311,000 +/- 35,000
- Number of particles per $pp \rightarrow t\bar{t}$ event: 16,000 +/- 1,700
- Innermost pixel layer 25x100 μm^2, all other pixel layers 50x50 μm^2
- Strip layers are at millimeter resolutions
- We focus on Athena simulation in the following slides



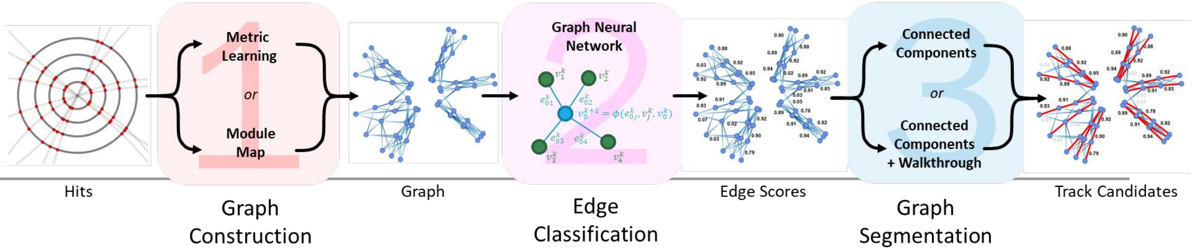
UNSG-2021-57

GNN4ITk Pipeline



- Pipeline receives clusters = collections of energy deposits on silicon. These are associated with 3D spacepoints, to be used as nodes for stage 1 onwards
- Out of stage 3 we obtain a set of track candidates, each is an unordered set of spacepoints
- For processing in Athena track fitting chain, we associate these back to the original clusters, and order in increasing distance from beamspot origin

Training Details



Dataset

- Run 4 ATLAS simulation, ttbar <μ>=200 pileup, ITk geometry

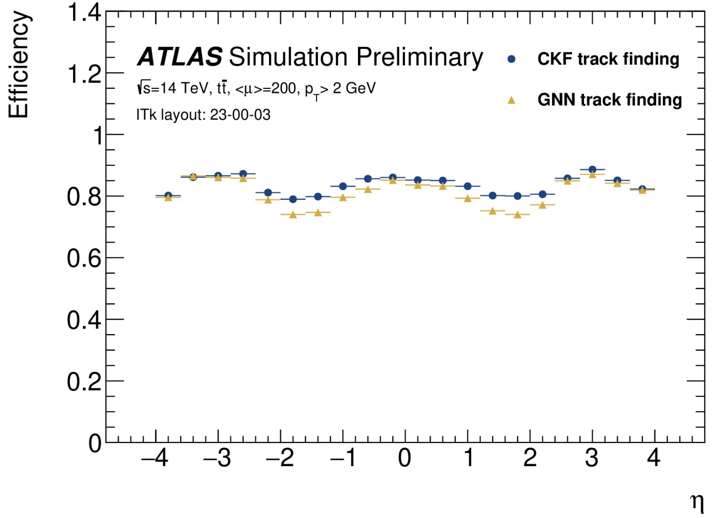
Truth

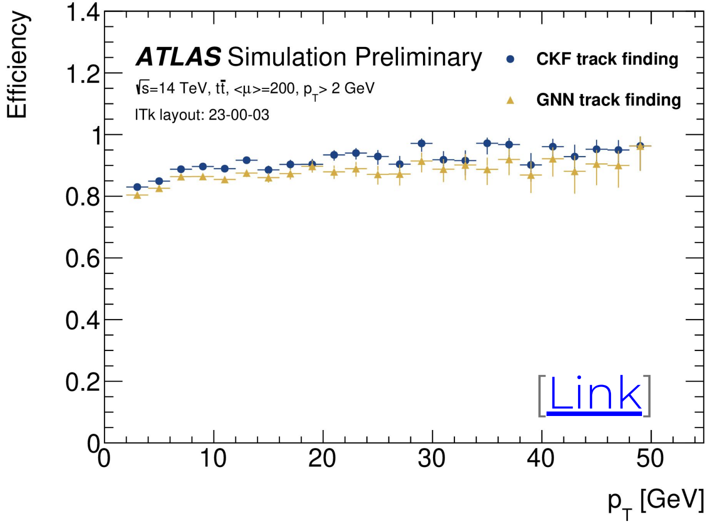
- Pairwise connections between *sequential* hits in *target tracks* treated as true
- A target track is primary, non-electron, p_T>1GeV, and has at least 3 hits
- All other connections between all other tracks (or noise) considered fake

Training Strategies

- Data-driven adjacency matrix and geometric cuts for module map
- Contrastive hinge loss for metric learning
- Binary cross entropy for edge classification (GNN and edge filter)

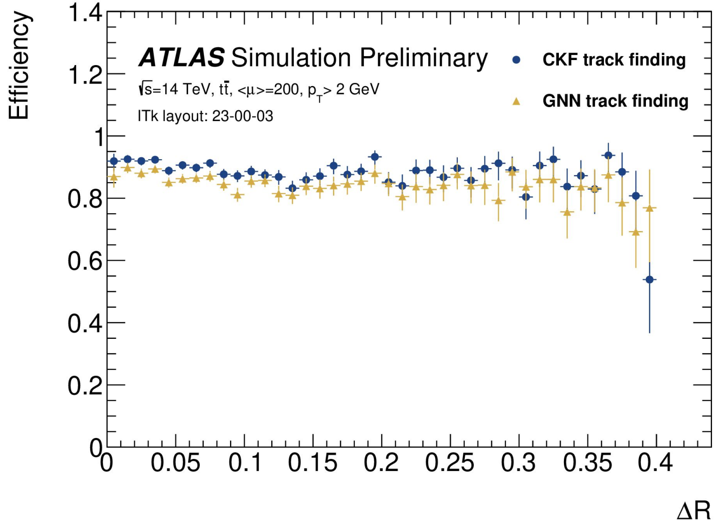
Track Reconstruction Performance

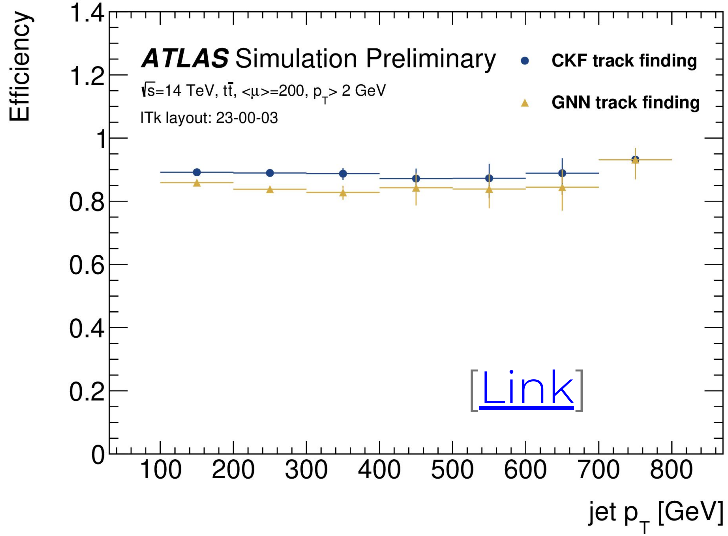




- Tracking efficiency compared with current combinatorial kalman filter (CKF) technique
- Behaviour across **η** and p_T similar to CKF - good sanity check!

Track Reconstruction Performance



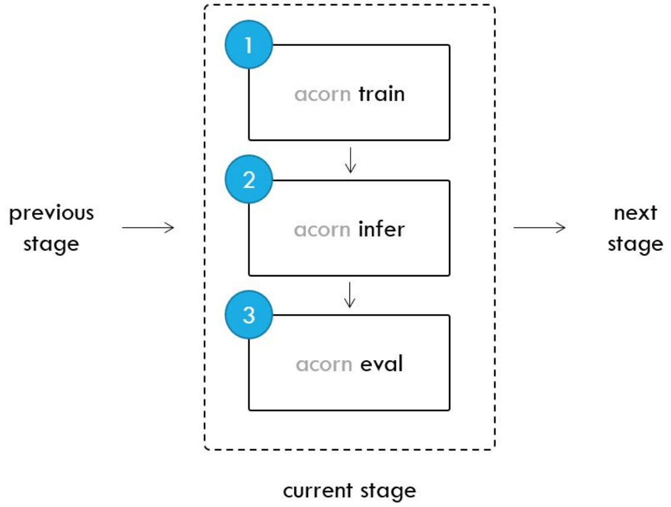


- Again, similar characteristics across ΔR and jet p_T

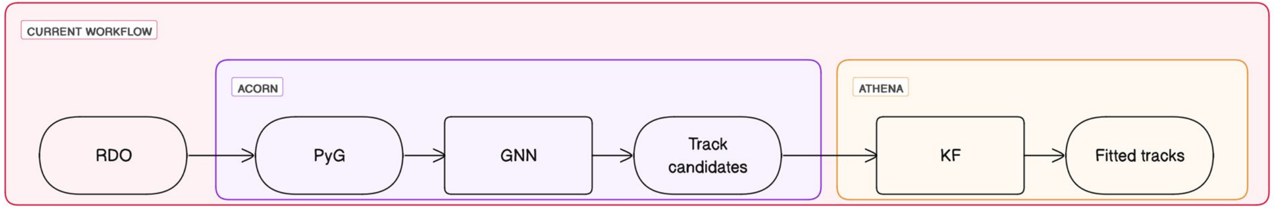
ACORN: **A C**harged **O**bject **R**econstruction **N**etwork

acorn

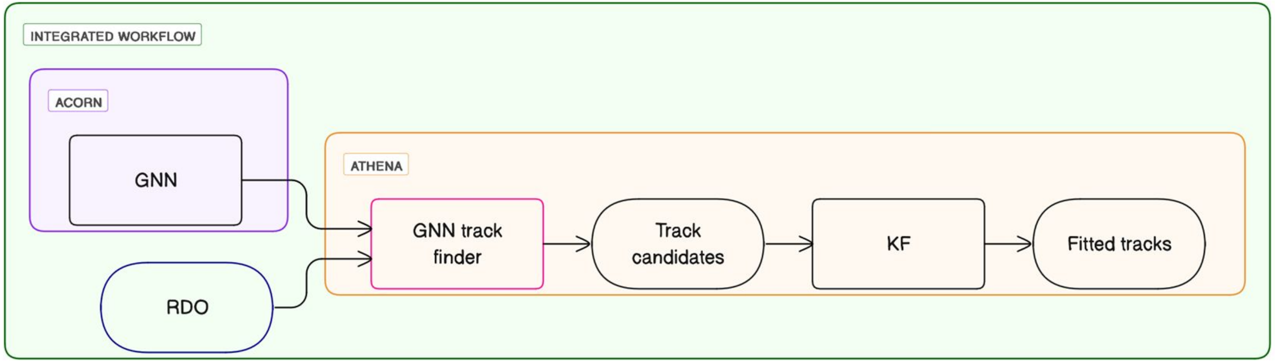
- Framework design & goals [Link]
  - A modular framework for training and R&D of ML-based tracking
  - Runs on pytorch lightning and pytorch geometric
  - Each stage self-contained, run either separately or (newly built) multi-stage inference
  - Approximately 12 active developers across 7 institutes
- Integrations
  - ATLAS ITk
  - ACTS OpenData Detector
  - TrackML



acorn → Athena



RDO = Raw Data Object



- Previously, Acorn used to build tracks, which were passed back into Athena for fitting
- Now, models trained in Acorn, translated to Onnx and TorchScript
- Loaded into Athena Component (c++) as part of tracking chain

Tracking Computational Requirements

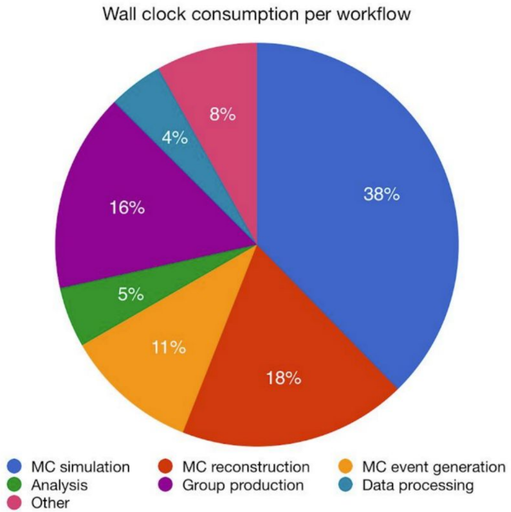
ATLAS Computing Budget

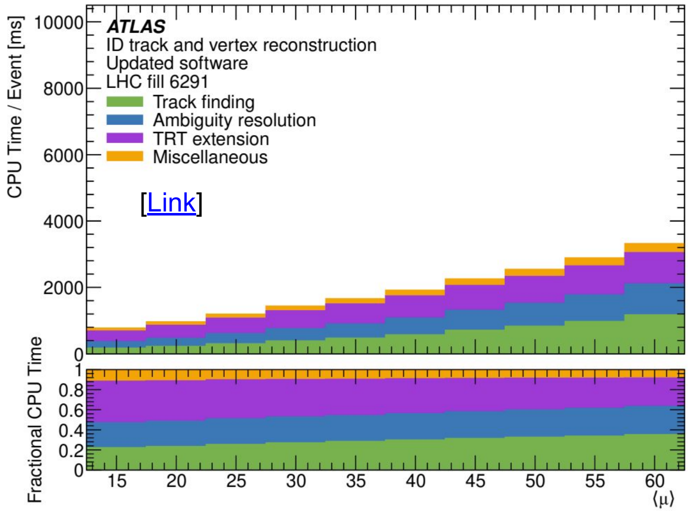
[Link]

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

Seconds/event

- (Top right) Average CPU usage in 2018: Reconstruction significant piece
- (Above) Reconstruction timings for run 2 (seconds): Tracking takes majority of time
- (Right) Run 3 track reconstruction timings: Track finding and ambiguity resolution take ~2s for <μ>=60





HL-LHC Offline & Online Track Reconstruction Needs

	LHC Run 3	**HL-LHC**
L0 trigger accept	100 kHz	**1 MHz**
Event Filter accept	1 kHz	**10 kHz**
Event size	1.5 MB	**4.6 MB**

- Event filter (high level trigger) contains tracking
- Regional tracking @ 1MHz
- Full event tracking @ 150kHz

- Current CPU proposed algorithm is optimized Fast Tracking
- 23.2 s/event single-core CPU, small drop in track efficiency: 1-2% on average, 5% for pT in [1,1.5]GeV

$\langle\mu\rangle$	Tracking	Release	Byte Stream Decoding	Cluster Finding	Space Points	Si Track Finding	Ambiguity Resolution	Total ITk
140	default	21.9	2.2	6.4	3.5	31.6	43.4	87.1
	fast			6.1	1.0	13.4	-	22.7
200	default	21.9	3.2	8.3	4.9	66.1	64.1	146.6
	fast			8.1	1.2	23.2	-	35.7

Fast tracking vs Default tracking timing (s) [Link]

$\langle\mu\rangle$	Tracking	Byte Stream Decoding	Cluster Finding	Space Points	Si Track Finding	Total ITk
140	full-scan	2.2	6.1	1.0	13.4	22.7
	regional	0.33	0.90	0.15	1.11	2.49
200	full-scan	3.2	8.1	1.2	23.2	35.7
	regional	0.48	1.23	0.18	1.92	3.81

Fast tracking timing (s) for regional vs full-scan [Link]

HL-LHC Offline & Online Track Reconstruction Needs

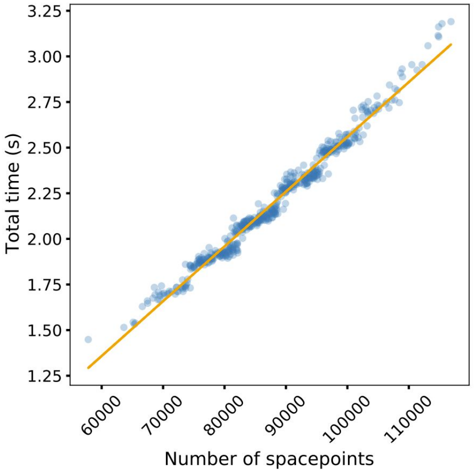
- Goal is to use GNN4ITk pipeline to perform offline tracking in <1s
- Target regional and full event online tracking in 10-100ms
- Starting with right-hand column below (TrackML ~90k hits)
- Optimizing for ITk (~300k hits)
- Need improvements in all stages

TrackML

	Baseline	Faiss	cuGraph	AMP	FRNN
Data Loading	0.0022 ± 0.0003	0.0021 ± 0.0003	0.0023 ± 0.0003	0.0022 ± 0.0003	0.0022 ± 0.0003
Embedding	0.02 ± 0.003	0.02 ± 0.003	0.02 ± 0.003	0.0067 ± 0.0007	0.0067 ± 0.0007
Build Edges	12 ± 2.64	0.54 ± 0.07	0.53 ± 0.07	0.53 ± 0.07	0.04 ± 0.01
Filtering	0.7 ± 0.15	0.7 ± 0.15	0.7 ± 0.15	0.37 ± 0.08	0.37 ± 0.08
GNN	0.17 ± 0.03	0.17 ± 0.03	0.17 ± 0.03	0.17 ± 0.03	0.17 ± 0.03
Labeling	2.2 ± 0.3	2.1 ± 0.3	0.11 ± 0.01	0.09 ± 0.008	0.09 ± 0.008
Total time	$15 \pm 3.$	3.6 ± 0.6	1.6 ± 0.3	1.2 ± 0.2	0.7 ± 0.1

TrackML Inference time - Seconds/event [Link]

TrackML



[Link]

Graph Construction Optimizations

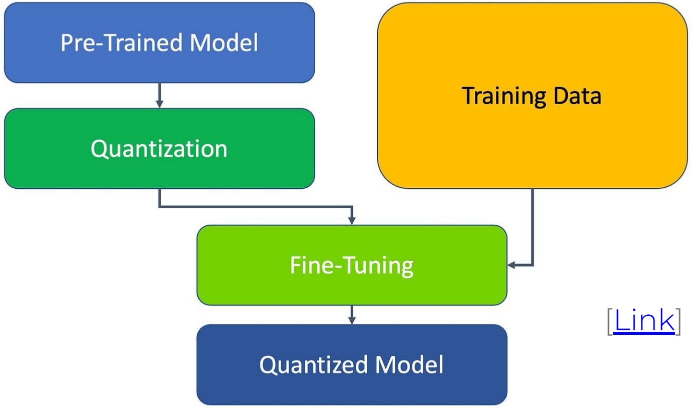
Work by Sebastian Dittmeyer, University of Heidelberg

Quantization and Pruning

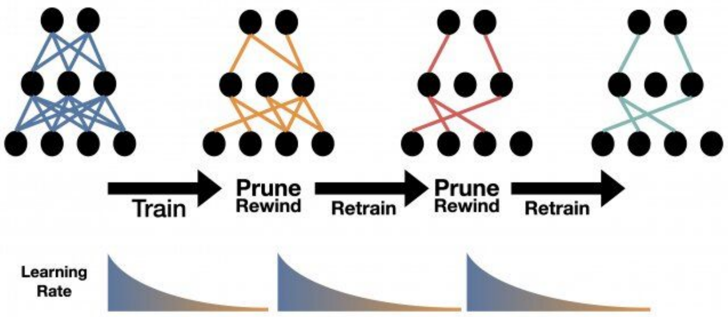
Optimizations for FPGA and GPU studied on embedding (metric learning) stage

1. Quantization Aware Training
   - Fine-tune quantized model with differentiable notion of quantization
   - FPGA can use arbitrary quantization
   - GPU can exploit 8-bit quantization

2. Iterative (Learning Rate Rewind) Pruning
   - During training, iteratively prune model
   - After each iteration, restart learning rate

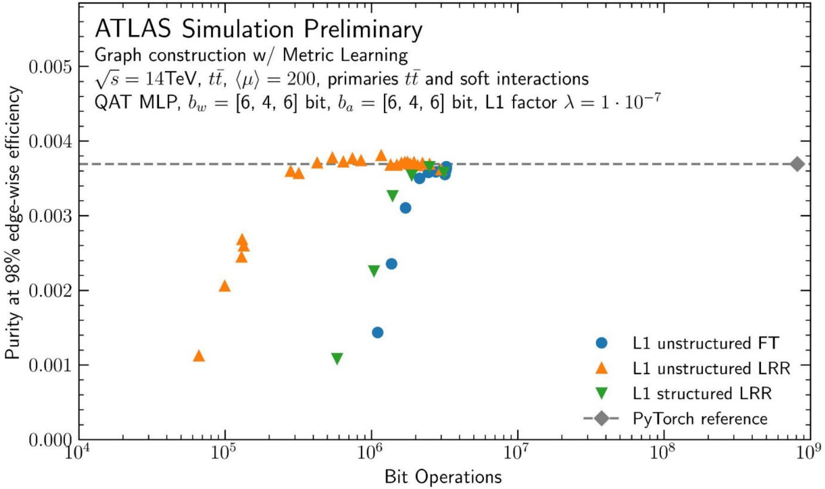


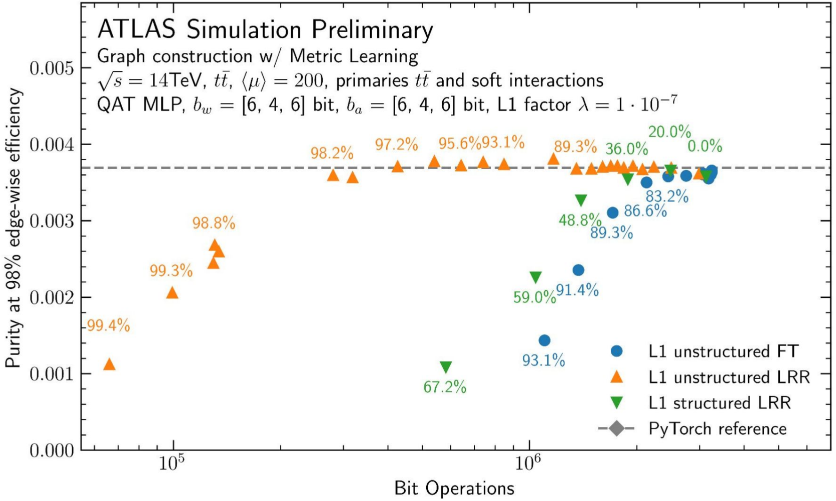
[Link]



[Link]

QAT & Iterative Pruning Results





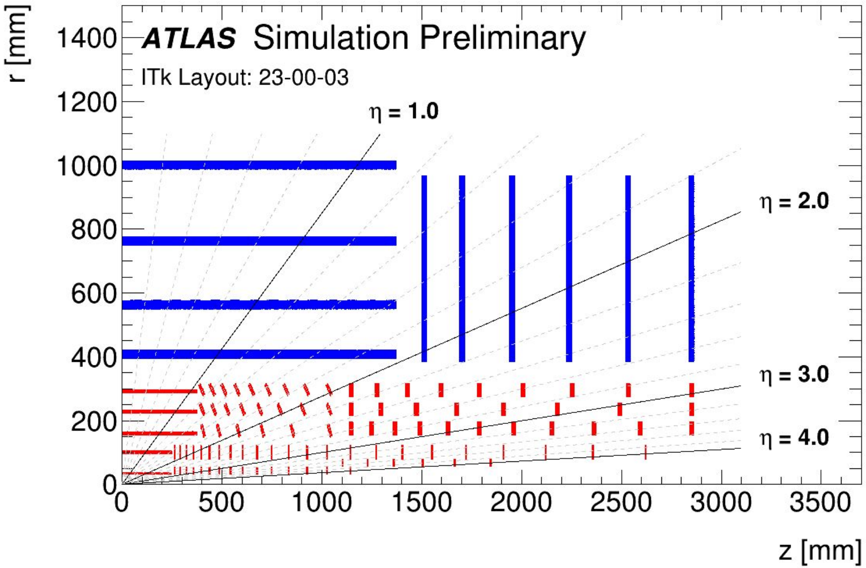
- Can prune model to 1/56 the size and maintain purity at fixed efficiency, using learning rate rewind training (LRR)

[Link]

Graph Neural Network Optimizations

Regional Tracking

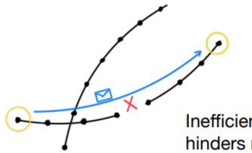
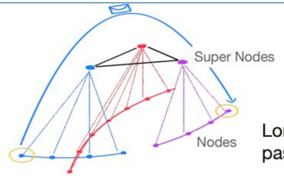
- To handle 150kHz-1MHz EF trigger rate, can parallelize across O(100) regions in event, or reconstruct only specific regions
- For highest flexibility, would like to train *one model* and infer on various topologies
- Initial tests performed very poorly, due to **batch normalization** in model
- Reimplementing with **layer normalization**, recover both original performance, and equal performance in regional track reconstruction



Graph Segmentation Optimizations

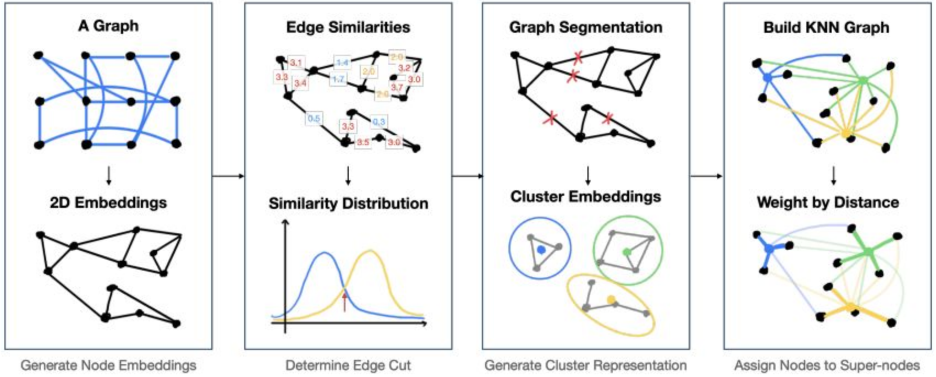
Work by Ryan Liu, UC Berkeley

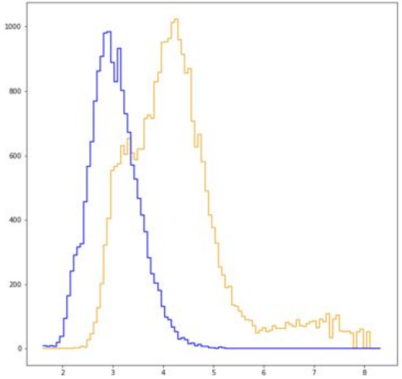
Hierarchical Graph Neural Network: Overview

Current Problem	Proposed Solutions
Performance limited by input graph	Make predictions less graph-dependent
Message passing obstructed by inefficiencies	Construct hierarchical structure
Flat GNN  Inefficient graph construction hinders message passing	Hierarchical GNN  Long distance message passing is possible

Tracking Goal	Feature	DiffPool	SAGPool	EdgePool	GMPool (ours)
Subquadratic scaling	Sparse	✗	✓	✓	✓
End-to-end trainable	Differentiable	✓	✓	✓	✓
Variable event size	Adaptive number of clusters	✗	✗	✓	✓
Many hits to many particles relationship	Soft assignment	✓	✗	✗	✓

How it works:





Hierarchical Graph Neural Network: Results

- The highest physics performance comes from Bipartite Classifier (BC) HGNN with O(1) second inference
- Fastest inference still from connected components
- Latest ITk HGNN model combines both for high efficiency / high throughput
- We see robustness of HGNN to edge construction inefficiencies in earlier stages of pipeline

TrackML

Models	E-GNN	E-HGNN	BC-HGNN	EC-GNN	Truth-CC
Efficiency	94.61%	95.60%	**97.86%**	96.35%	97.75%
Fake Rate	47.31%	47.45%	**36.71%**	55.58 %	57.67%
Time (sec.)	2.17	2.64	1.07	**0.22**	0.07

[Link] to work

Percent Edge Removed	0%	10%	20%	30%	40%	50%
BC Efficiency	98.55%	98.39%	97.68%	96.63%	95.10%	92.79%
BC Fake Rate	1.23%	1.55%	2.13%	3.10%	4.75%	7.31%
Truth-CC Efficiency	98.72%	96.21%	92.31%	85.81%	77.26%	64.81%
Truth-CC Fake Rate	5.87%	15.53%	24.40%	33.48%	42.99%	53.12%

Summary

- GNN4ITk pipeline:
  - Stable and converged
  - Available in open-source via the acorn framework
  - Out-of-the-box (i.e. not yet properly tuned) gives physics performance approaching that of Athena CKF algorithm
- HGNN, quantization, pruning and regional tracking all promising directions that show speed-ups with little/no drop in physics performance
- Also building optimized CPU & CUDA module map algorithm, and faster lightweight GNN for pruning graph