Graph Neural Network Acceleration on FPGAs for Fast Inference in Future Muon Triggers at HL-LHC

Martino Errico, Davide Fiacco, Stefano Giagu, Giuliano Gustavino, Valerio Ippolito, Graziella Russo

Istituto Nazionale di Fisica Nucleare

Sapienza Università di Roma

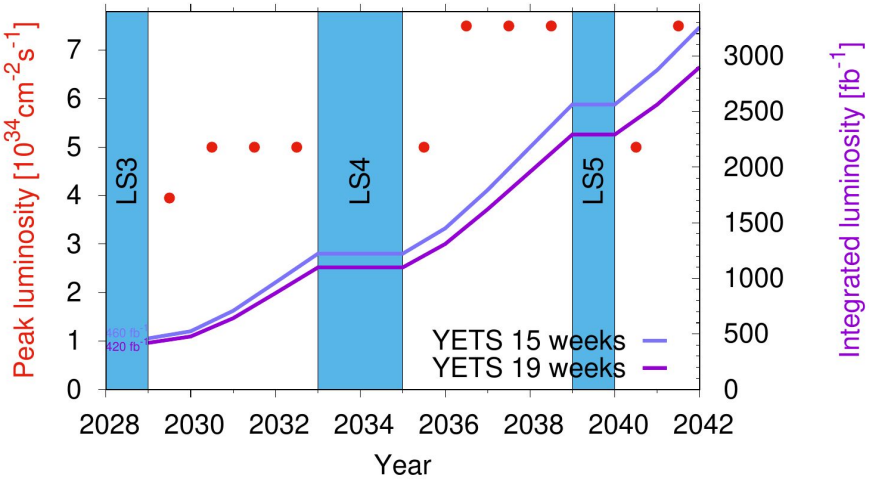
EuCAIFCon 2025
Cagliari - 17/06/2025

Triggers for the high luminosity frontier

Near future colliders, such as HL-LHC, will operate in a high luminosity regime to allow for higher statistics

Such experimental conditions impose demanding constraints on trigger algorithms, which must be able to cope with very high rates and complex events

Machine Learning trigger algorithms with O(100 ns) inference latency offer a compelling alternative to traditional algorithms



Detector geometry

The viability of a ML trigger algorithm is studied on a typical use case: particle tracking and selection in a muon spectrometer

The model for the geometry of the detector is the Phase-II ATLAS Muon Spectrometer, a detector with cylindrical symmetry around the z-axis of the reference frame, and the quantities of interest are:

$$\eta = -\ln \tan \frac{\theta}{2}$$

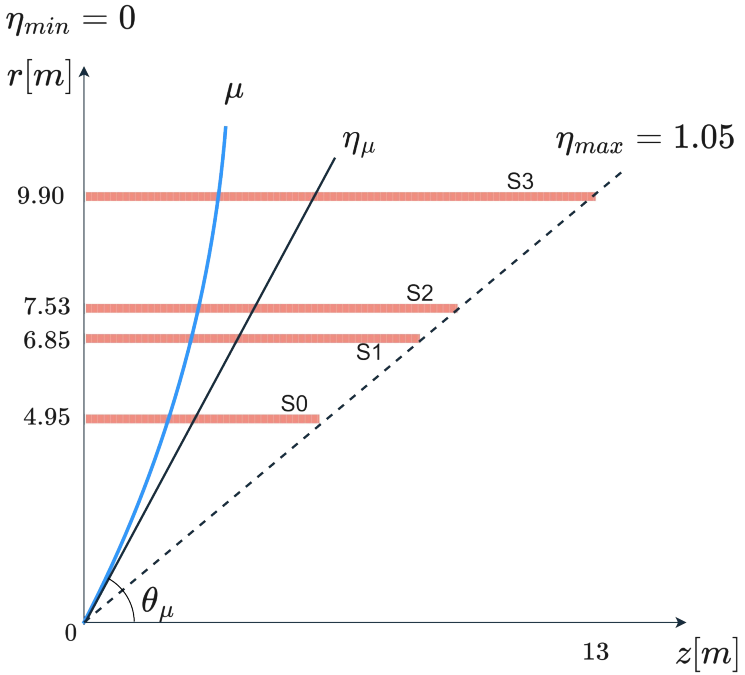
$$r = \sqrt{x^2 + y^2}$$

$$p_T = \sqrt{p_x^2 + p_y^2}$$

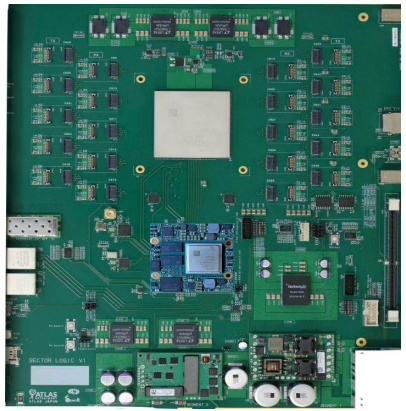
The Muon Spectrometer

- Covers the region:
  {|η| < 1.05} x {5 < r[m] < 10}
- 4 stations of detectors in volume
- Detectors are assumed to have nominal space-time resolution:
  1cm x 1ns
- Independently segmented in η and φ
- Immersed in 0.5 T toroidal field which bends in the (η, r) plane to allow p_T reconstruction
- Detector response fast enough to be used for triggering

We assume one trigger instance per Δη x Δφ
~ 1.05 x 11°



The hardware muon trigger



Implementation studies using a Virtex Ultrascale+ FPGA (XCVU13P) and hls4ml

The selected FPGA is the same used for the ATLAS muon trigger. Typical target resource usage is estimated as follows:

Resource	Total	Typical target occupancy
LUT	432k	25%
FF	864k	25%
BRAM	23.6 Mb	-

Performance criteria

The following criteria are used to judge the viability of the trigger algorithm for a high rate experiment use case

- Implementable on selected FPGA
- Easily maintainable
- Max latency ~ 100 ns
- Output:
  - n of muons in event (0, 1, 2 or 2+)
  - reconstruction (η, p_T, q) of up to 2 leading tracks
- Good p_T reconstruction performance, as evaluated through the efficiency turn-on curve

Dataset

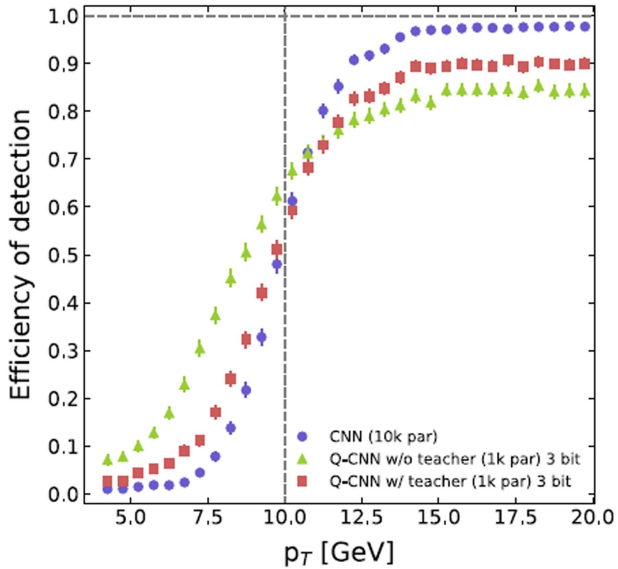
A toy dataset of about 4M events has been produced for the study of the trigger algorithms, according to the geometry, magnetic field and expected resolution of the model detector

- Roughly 2M muons:
  - muons from primary vertex
  - uniformly distributed: η in (0, 1.05), p_T in (3, 30) GeV
  - labelled with target q, η and p_T
- Roughly 2M events of simulated bg:
  - random bg, no correlation with signal
  - clusters, uniformly distributed
  - labelled with zeros
- Random bg also added to muon events
- Hits are given by indices/coordinates of the relative η bins (384 per layer)

CNN model

Strategy outlined in [Eur. Phys. J. C 81, 969 (2021)]:

- Train large Teacher model
- Train smaller Student model using Teacher as guide (Knowledge Distillation, KD)
- Quantize the Student model using Quantization-Aware Training (QAT)
- Promising results for single-track implementation:
  - good reconstruction performance
  - FPGA occupation < 1% (trigger only)
  - 440 ns latency
  - later implementation with 84 ns latency

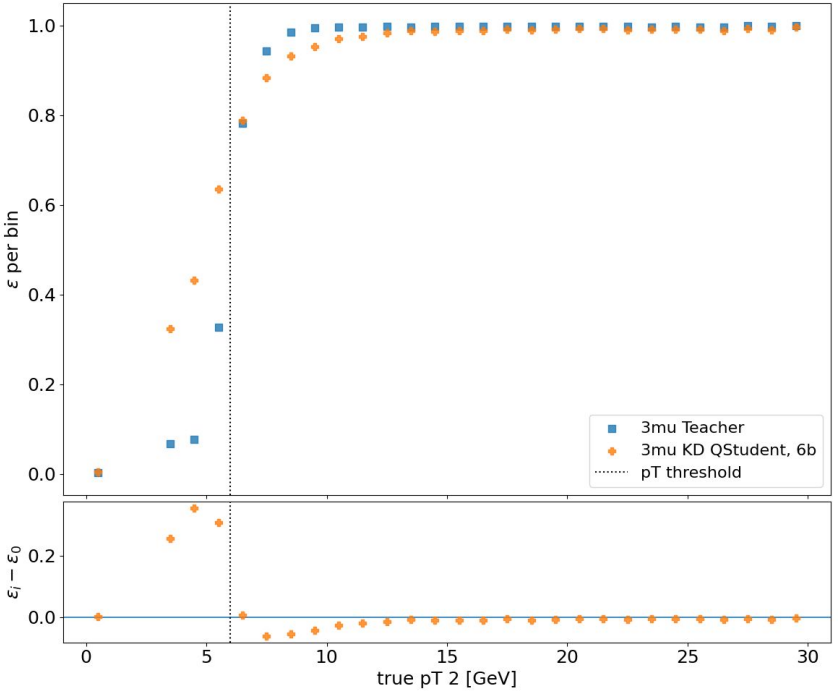


CNN model

Good single track performance, but not directly scalable to multi-track case:

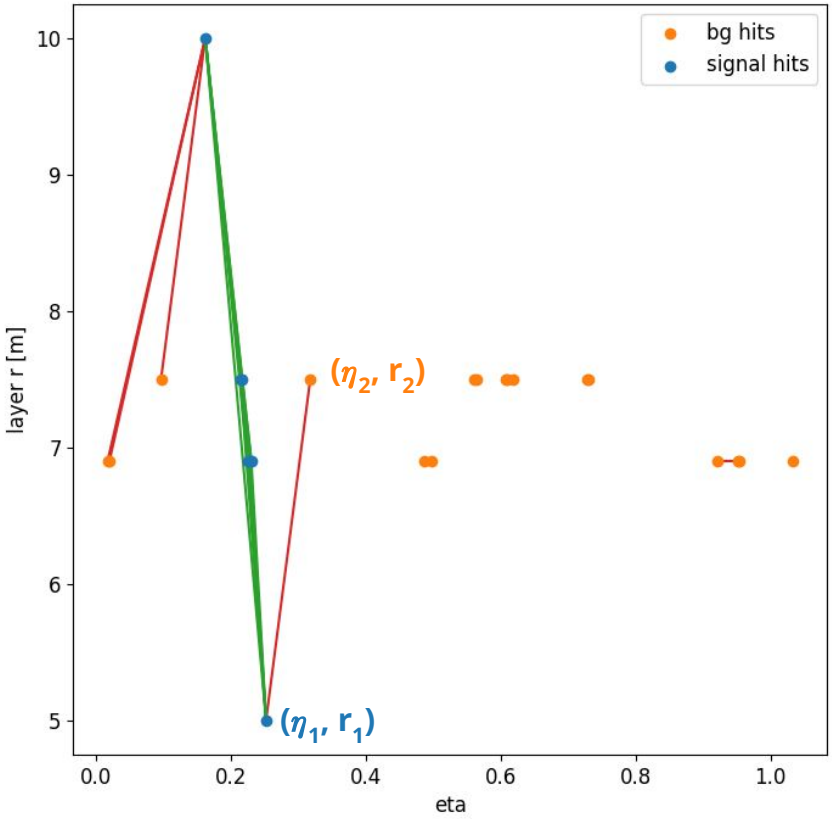
1. less effective on events with tracks not sufficiently resolved in η (|Δη|< 0.04)
2. suboptimal rejection of subleading muons below lower thresholds
3. good end-to-end latency (<250 ns), but hard to meet FPGA constraints

IDEA: Study effectiveness of Graph Neural Network proof-of-concept



Resource	Required
LUT	214%
FF	36%

GNN - graph building



General idea: build static graphs starting only from the hits, select track segment candidates and reconstruct the muons

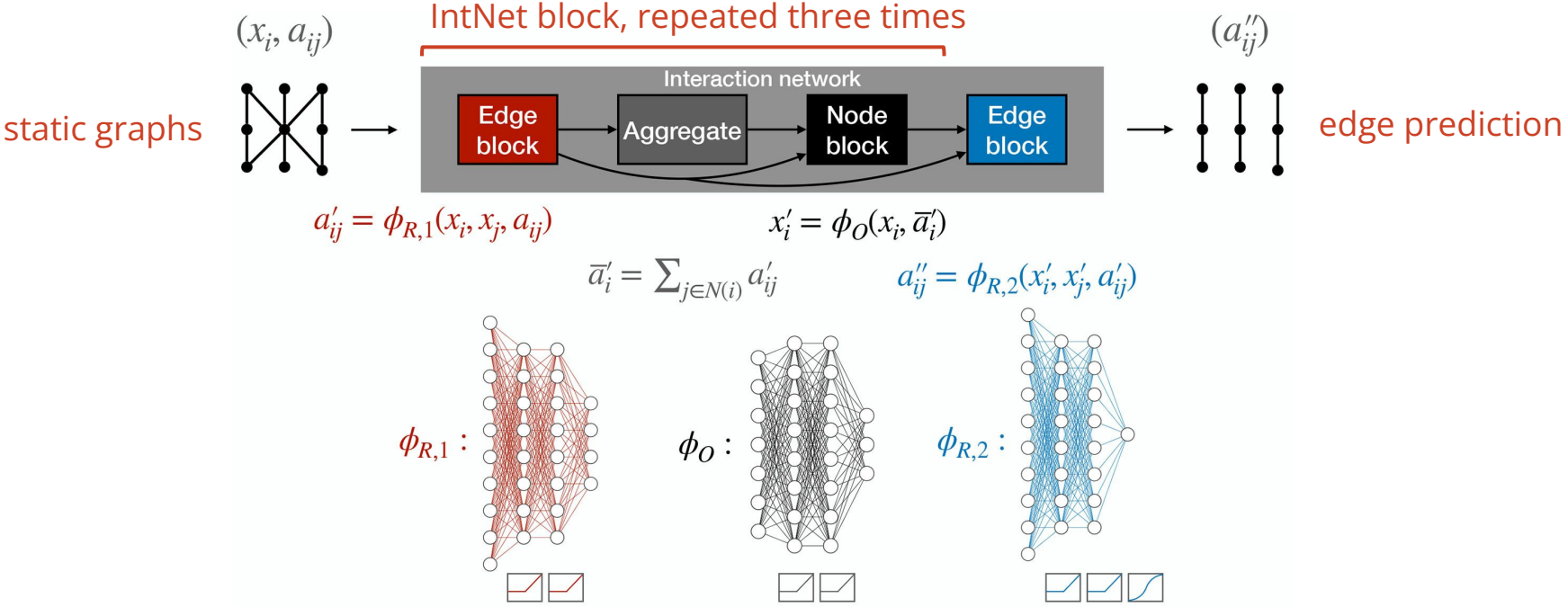
Connect nodes if (assuming minimum curvature radius R):

$$|\tanh\eta_2 - \tanh\eta_1| = k\frac{\Delta r_{12}}{R}$$

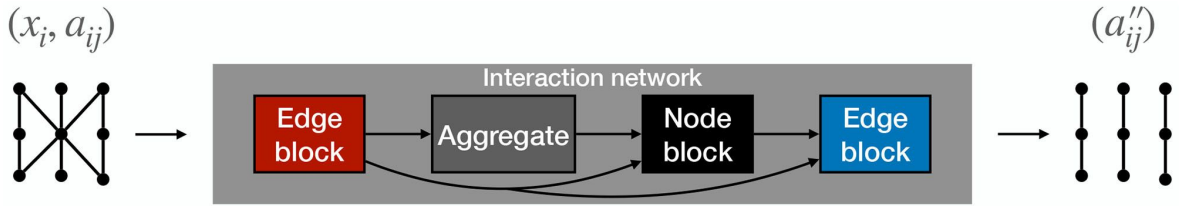
Edges connecting two signal nodes are labelled as signal edges; all other edges are labelled as background edges.

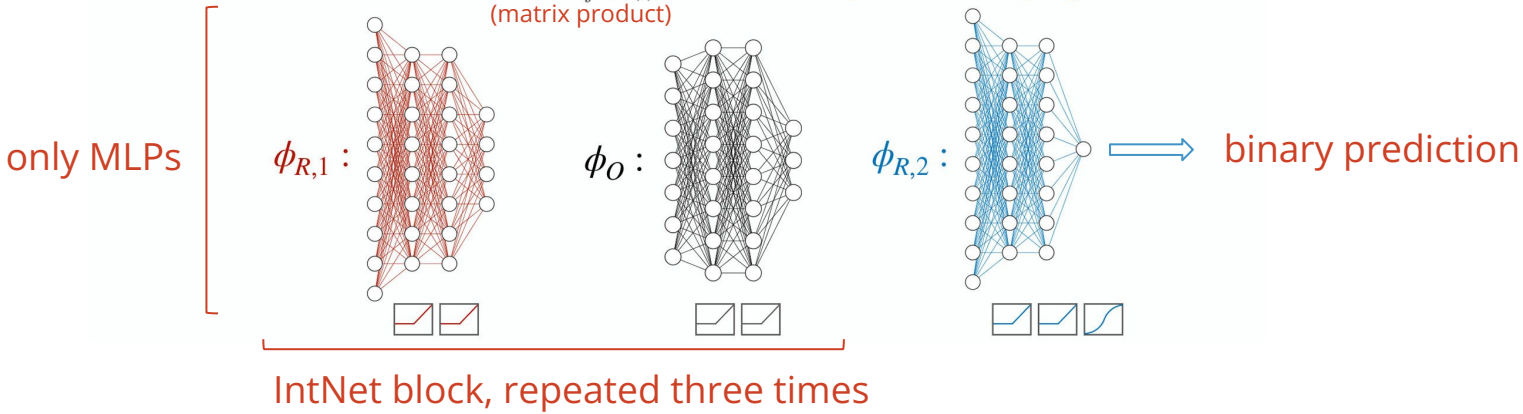
Currently only 0 and 1 muon events in the dataset, 99.7% with 50 nodes or fewer

GNN model - edge prediction

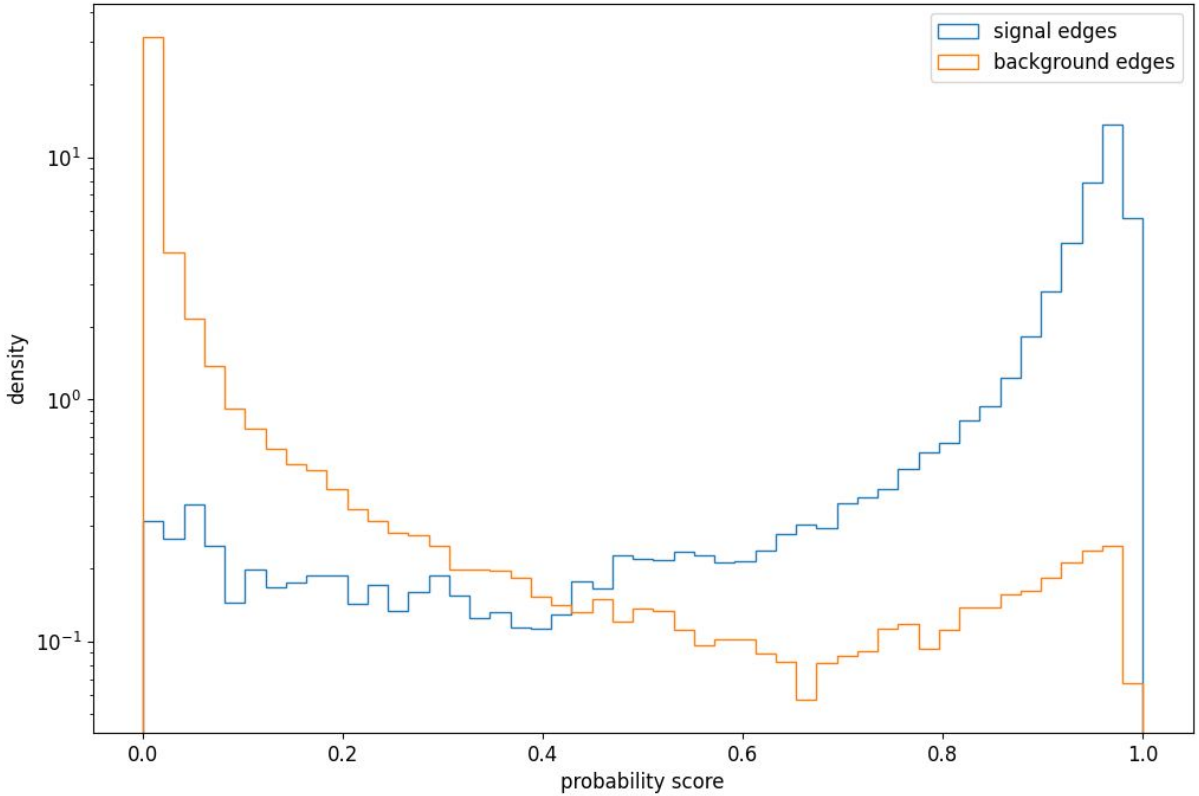


GNN model - edge prediction



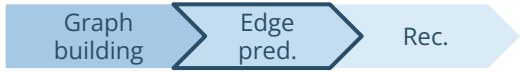


GNN results - edge scores

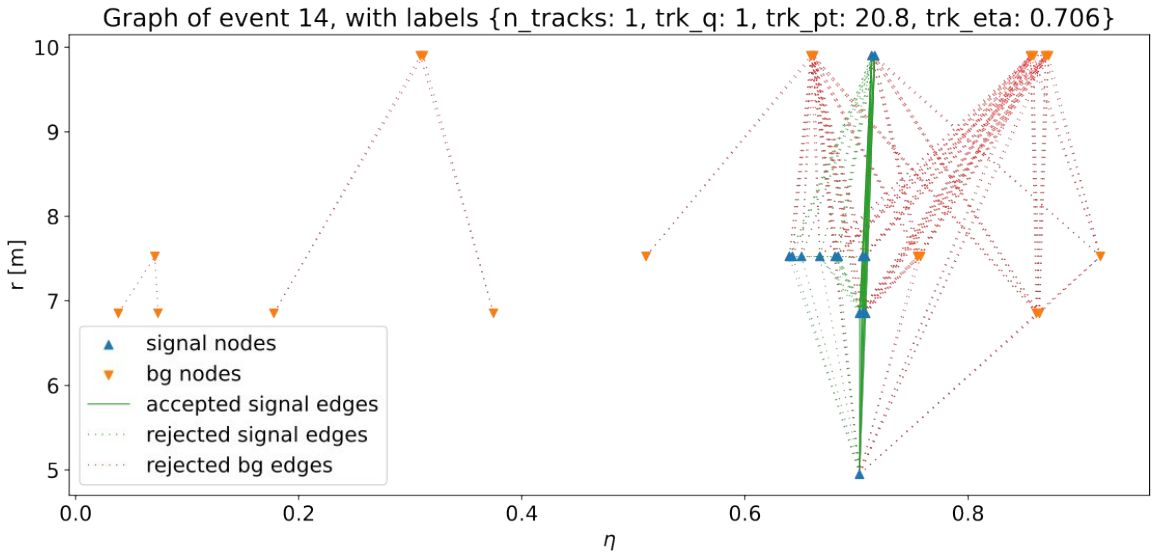


- All edges of all signal events combined
- Good signal/noise separation
- Some misidentified edges, but difficult to quantify effect on reconstruction

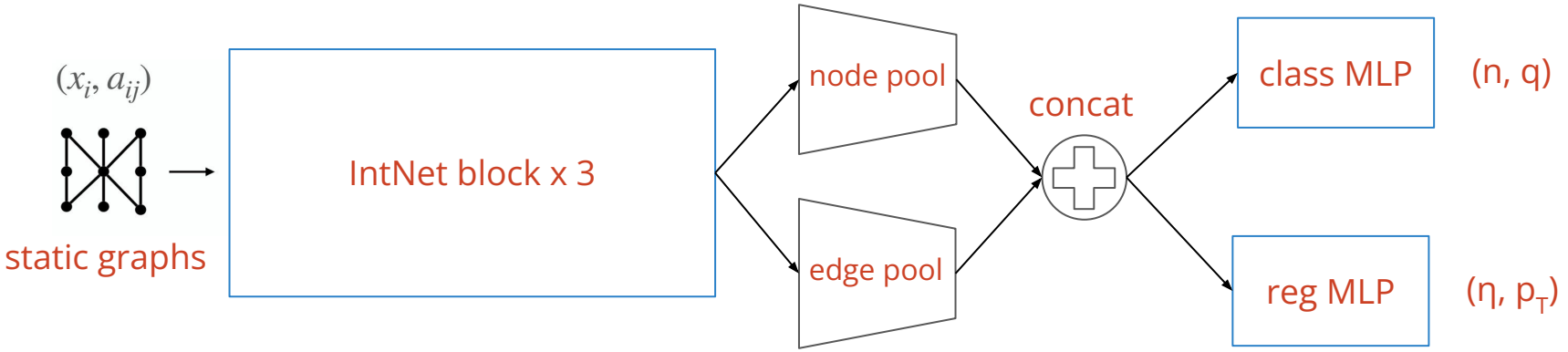
GNN results - edge prediction



- Original hit labels do not differentiate between direct muon hits and other processes
- Thus, rejecting some signal edges could improve reconstruction
- Final word on performance comes from turn-on curves, which require implementing momentum reconstruction

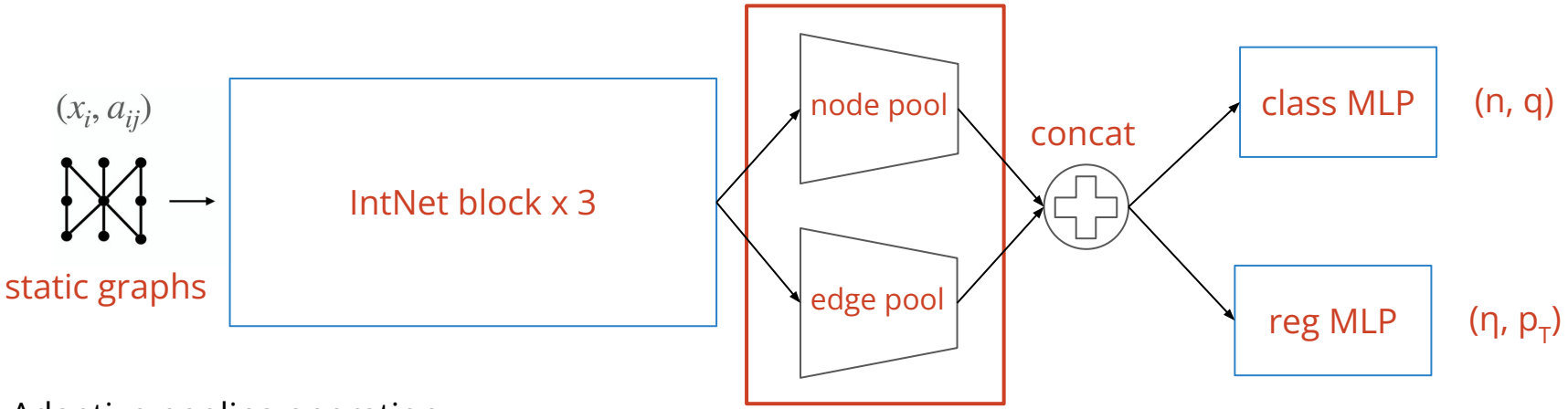


GNN model - track reconstruction



- Include reconstruction in model
- Train over reconstruction labels rather than classifying edges
- Using output representation is more adaptable and better for pipelining

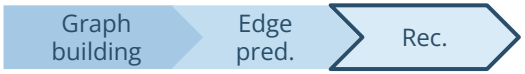
GNN model - track reconstruction



Adaptive pooling operation
(n = nodes, e = edges, F = features)

- Input: ((N_n, F_n), (N_e, F_e))
- Trainable linear layer with M_n sigmoid outs gives M_n scores to each node
- Similarly, M_e scores for each edge
- Features are multiplied by relative scores, summed over the entire input graph and flattened
- Output: (M_n x F_n, M_e x F_e), currently using M_n = M_e = 1

GNN results - efficiency



CNN results:

✔ efficiency > 90% from 14 GeV

✘ minimum efficiency 2% for leading and 10% for subleading track

✔ bg efficiency 0.17%

✘ larger model compared to prototype (8.4k 6 bit par vs 1k 3 bit)

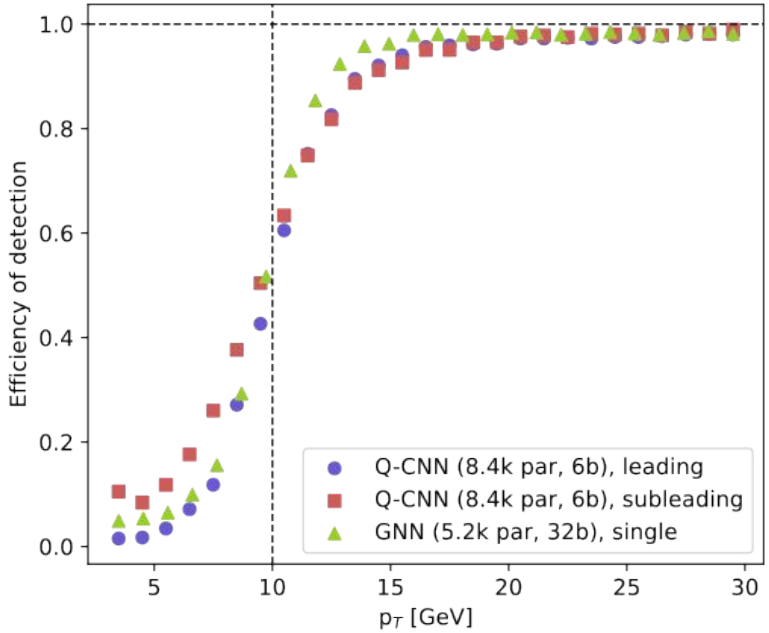
Preliminary GNN results:

✔ efficiency > 90% before CNN

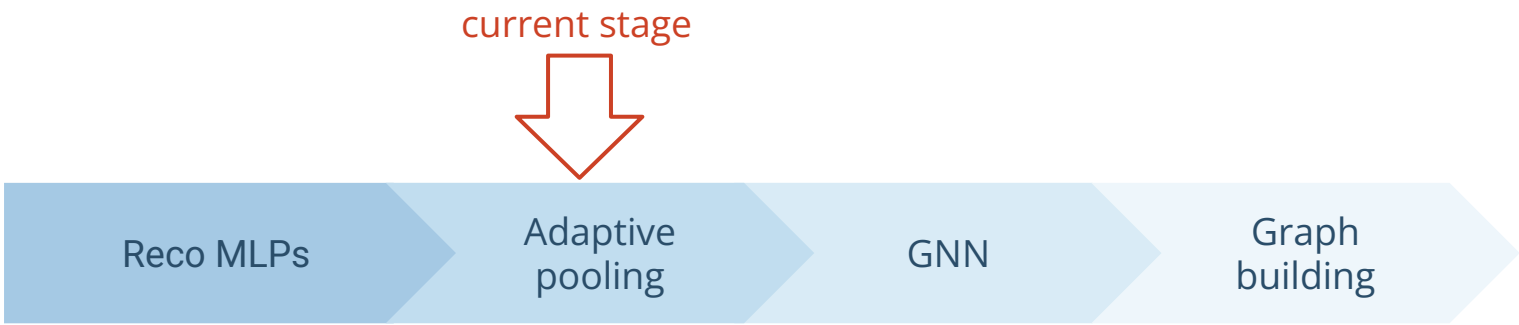
✔ minimum efficiency 5%

✔ fewer parameters (5.2k) than CNN (8.4k)

Conclusion: keep developing GNN architecture



Implementation - status



Reco MLPs	Adaptive pooling	GNN	Graph building
Implemented through hls4ml Latency (preliminary): 59 ns	Custom implementation currently being worked on	MLPs + matrix product, implementable through hls4ml	Custom implementation required

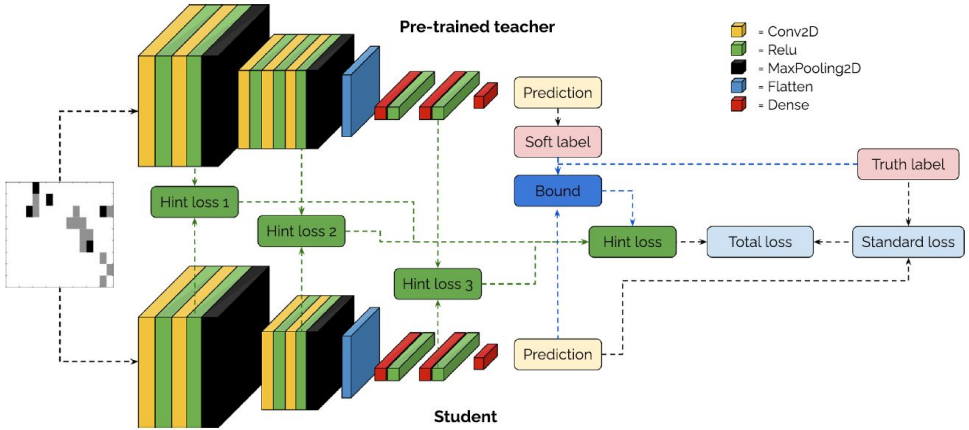
Graph building algorithm inspired by: Graph Neural Networks for Charged Particle Tracking on FPGAs, Real-time Graph Building on FPGAs for Machine Learning Trigger Applications in Particle Physics

Future developments

- Ongoing work on extension of GNN to include classification, as well as multi-track reconstruction
- Inclusion of φ reconstruction
- Inclusion of other input features (such as magnetic field)
- Architecture optimization:
  - ensemble of shallower GNNs
  - quantization-aware training

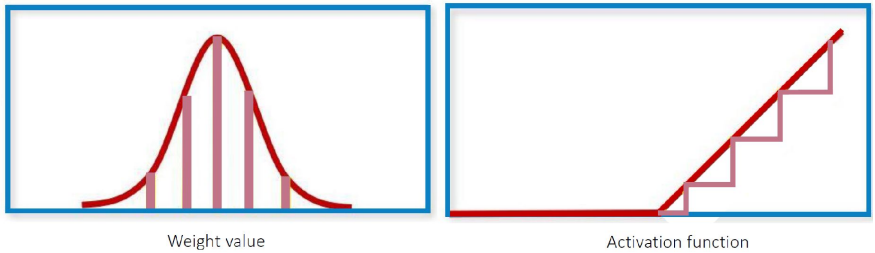
Thank you for your attention

Compression methods

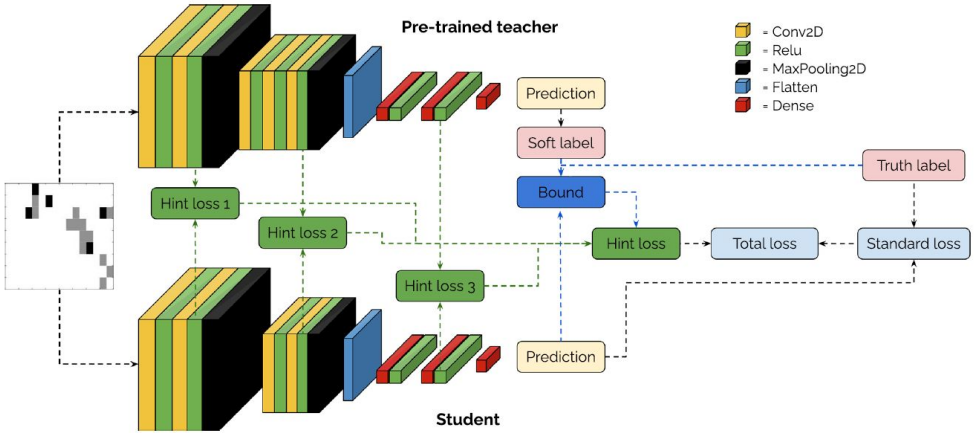


- Knowledge Distillation (KD): transfers learned properties from a large Teacher to a smaller Student
- Compute distance between intermediate representations of Teacher and Student
- Add distance to loss when Student is underperforming

- Reduce weight and activation bit widths
- Quantization Aware Training (QAT) simulates quantization during training
- Learned weights are more easily quantized



Knowledge Distillation



Hint losses: squared L2 distances of the intermediate representations at given depths

Student loss

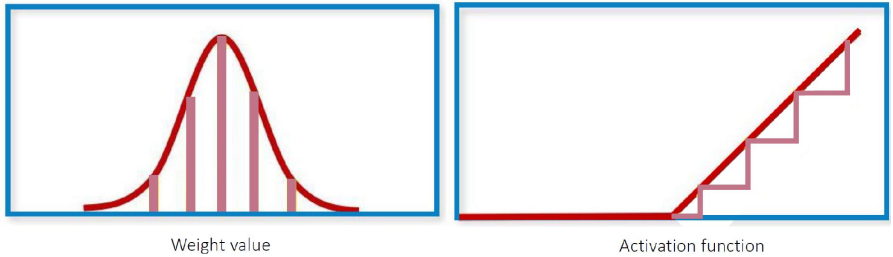
Teacher loss

$$KDL(x) = \begin{cases} S(x) + \sum_{i=1}^{n_{blocks}} \beta_i H_i(x) & \text{if } S(x) < (1+\gamma)T(x) \\ S(x) & \text{otherwise} \end{cases}$$

Eur. Phys. J. C 81, 969 (2021)

Quantization

$$2^{i-b+1}\texttt{clip}\left(\texttt{round}\left(x \cdot 2^{b-i-1}\right), -2^{b-1}, 2^{b-1}-1\right)$$



- Fake quantization applied at each layer during training: precision error is automatically accounted for in the loss
- Straight-through estimator used for fake quantization during backpropagation
- Actual quantization applied after training

CNN - I/O

Input:

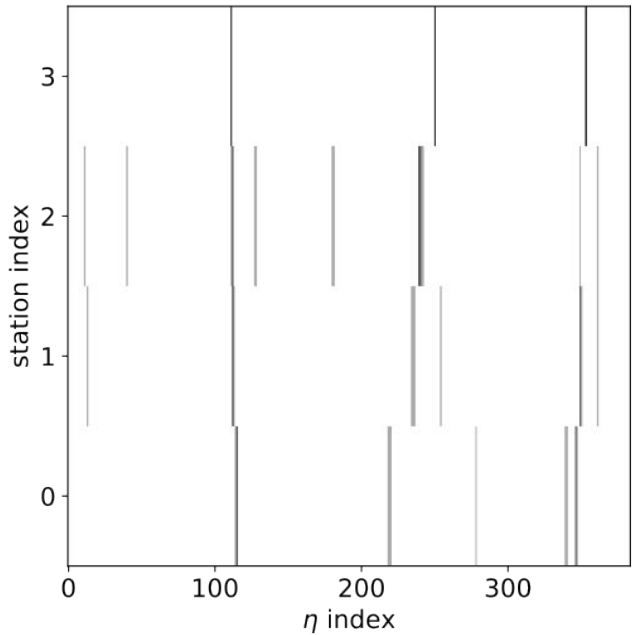
- images with pixels corresponding to hits
- separate single track images ORed to make multi-track images (0, 1, 2 or 3 muons per image)

Output:

- n of muons in event (0, 1, 2 or 2+)
- reconstruction (η, p_T, q) of up to 2 leading tracks
- output tensor:
  $[[n, q_1, p_{T1}, \eta_1], [n, q_2, p_{T2}, \eta_2]]$
- output vectors are zeroed when no corresponding muon is IDed
- φ reconstruction not included

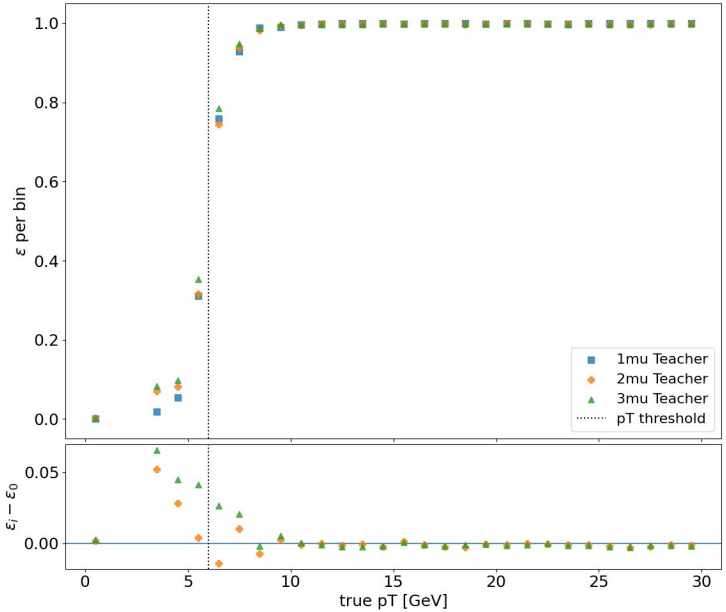
CNN - input

- (station indices) x (discretized η coor.)
- One row of pixels per station (4)
- One column of pixels per η bin (384)
- 4x384 images, reshaped to 4x192x2, one channel with even η indices, one with odd
- Hits obtained by reducing the original 9 layers:
  - 2/3 for S0, OR for other stations
  - pixel value keeps track of number of active pixels reduced (2 or 3 for S0, 1 or 2 for the rest)
- Average density of 3 muon images is 3%



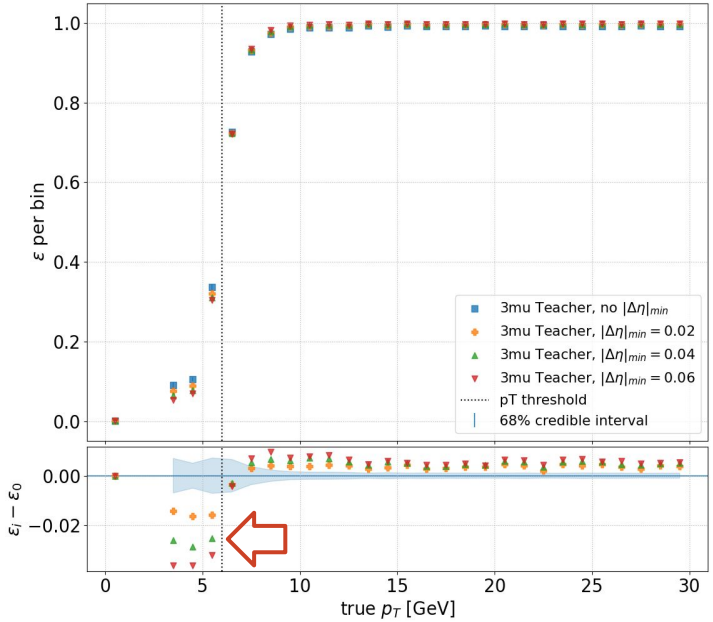
CNN - multi-track reconstruction

Teacher (unconstrained) model scales poorly with max number of muons per event



All CNN results given for $\min|\Delta\eta(\mu_i, \mu_j)| > 0.04$

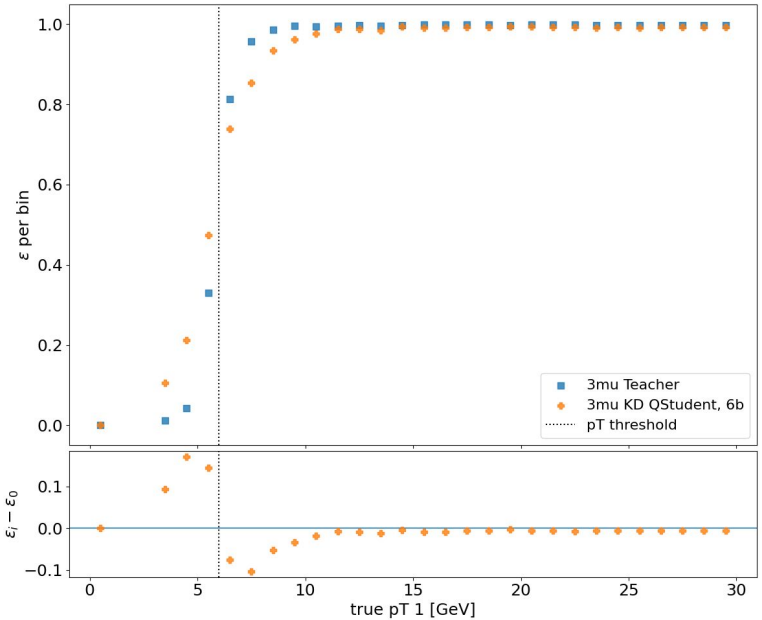
~85% acceptance factor

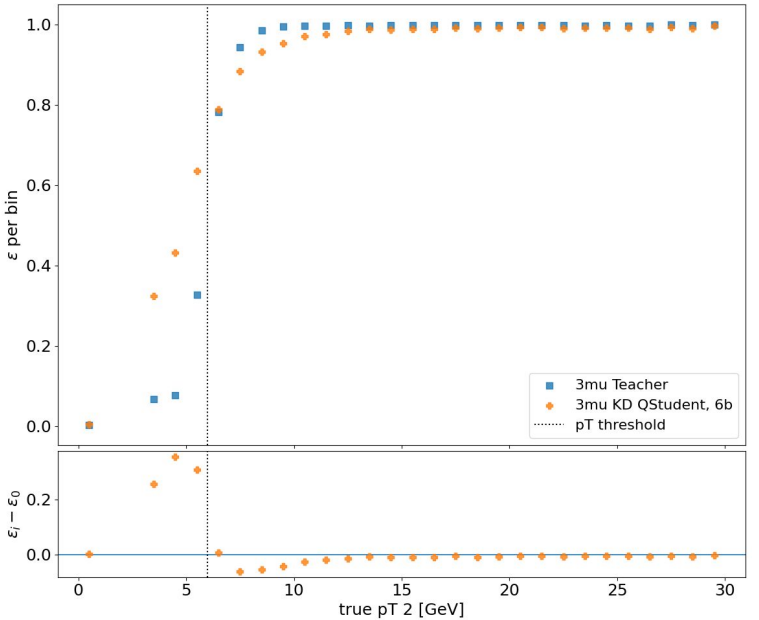


CNN VGG-like architecture not well suited for multi-track reconstruction

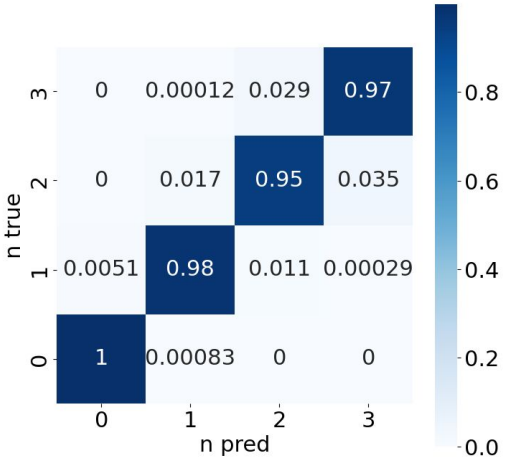
CNN - efficiency

Lowest cut studied (6 GeV) shows that, while performance on leading track reconstruction is good, rejection of subleading muons below threshold is not.





CNN - classification

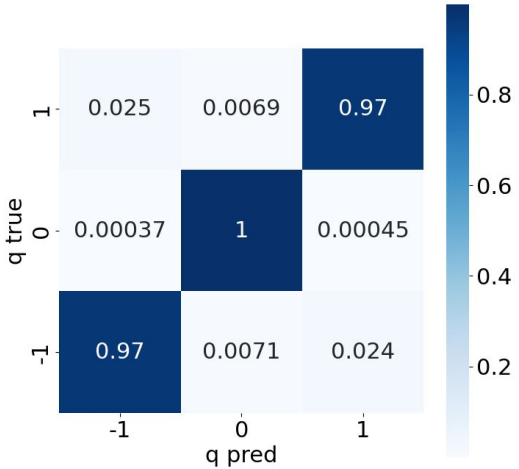


n of tracks in event:

- 0 for bg events
- good accuracy
- used for bg suppression in the reconstruction task

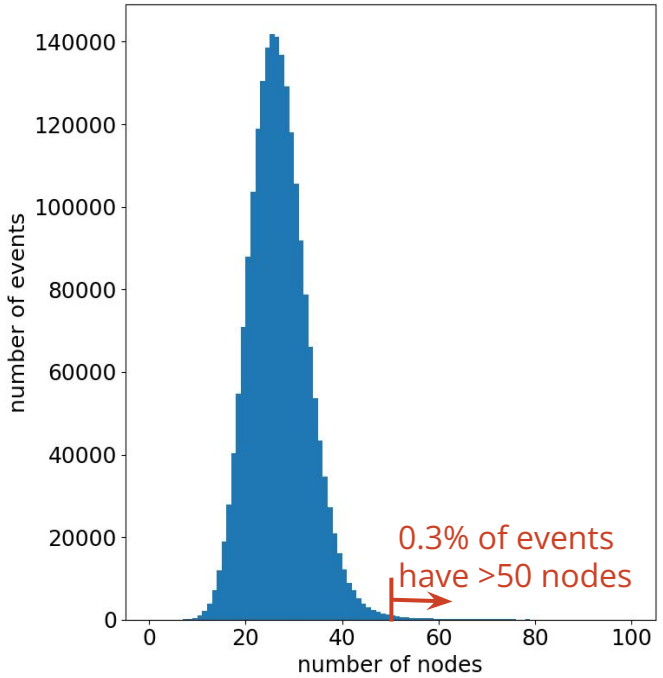
q of leading track:

- 0 for bg events
- good accuracy
- very similar performance on other tracks

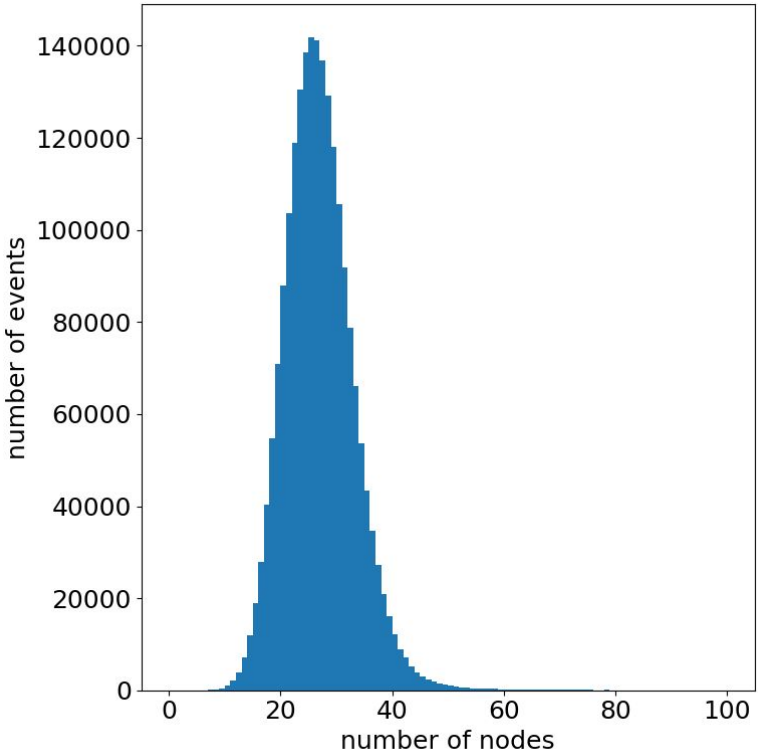


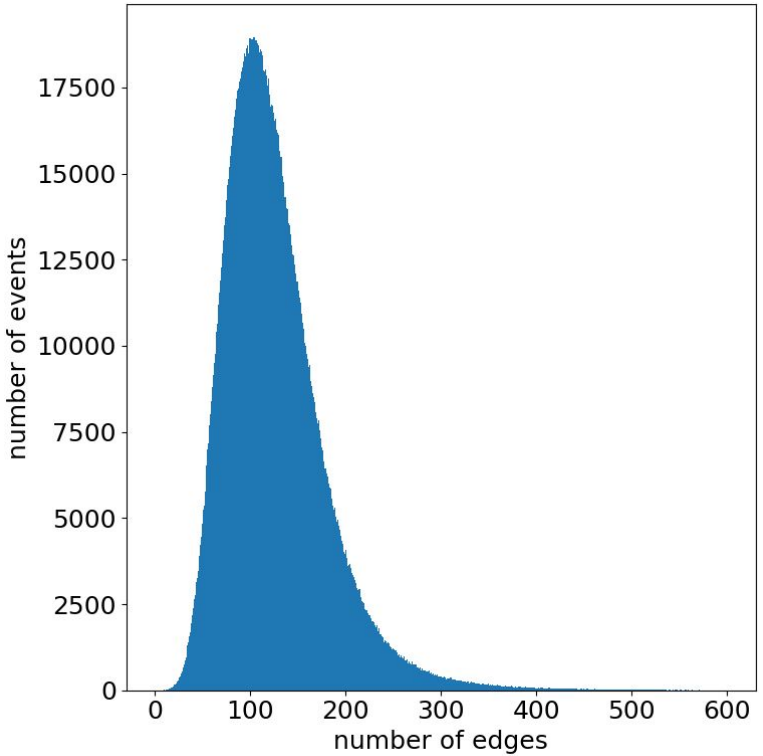
GNN - motivations

- More efficient for sparse data
  (4x384 images vs max 50 nodes)
- Adaptable to complex geometries
  (direct use of strip coordinates)
- Straightforward inclusion of other local features
  (φ measurement, magnetic field)
- Hopefully able to reconstruct different muons
  as separate tracks

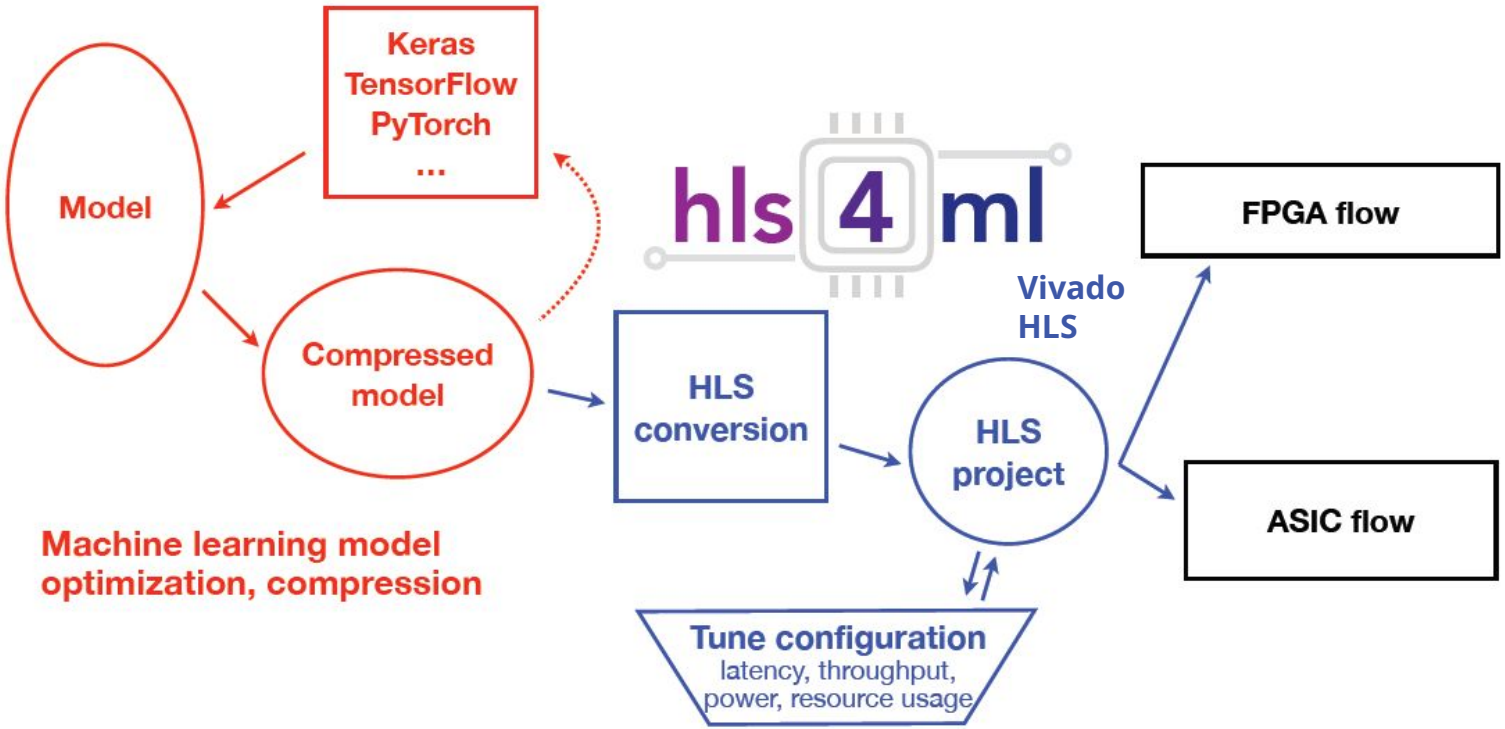


GNN- graph statistics





Development pipeline



https://github.com/fastmachinelearning/hls4ml