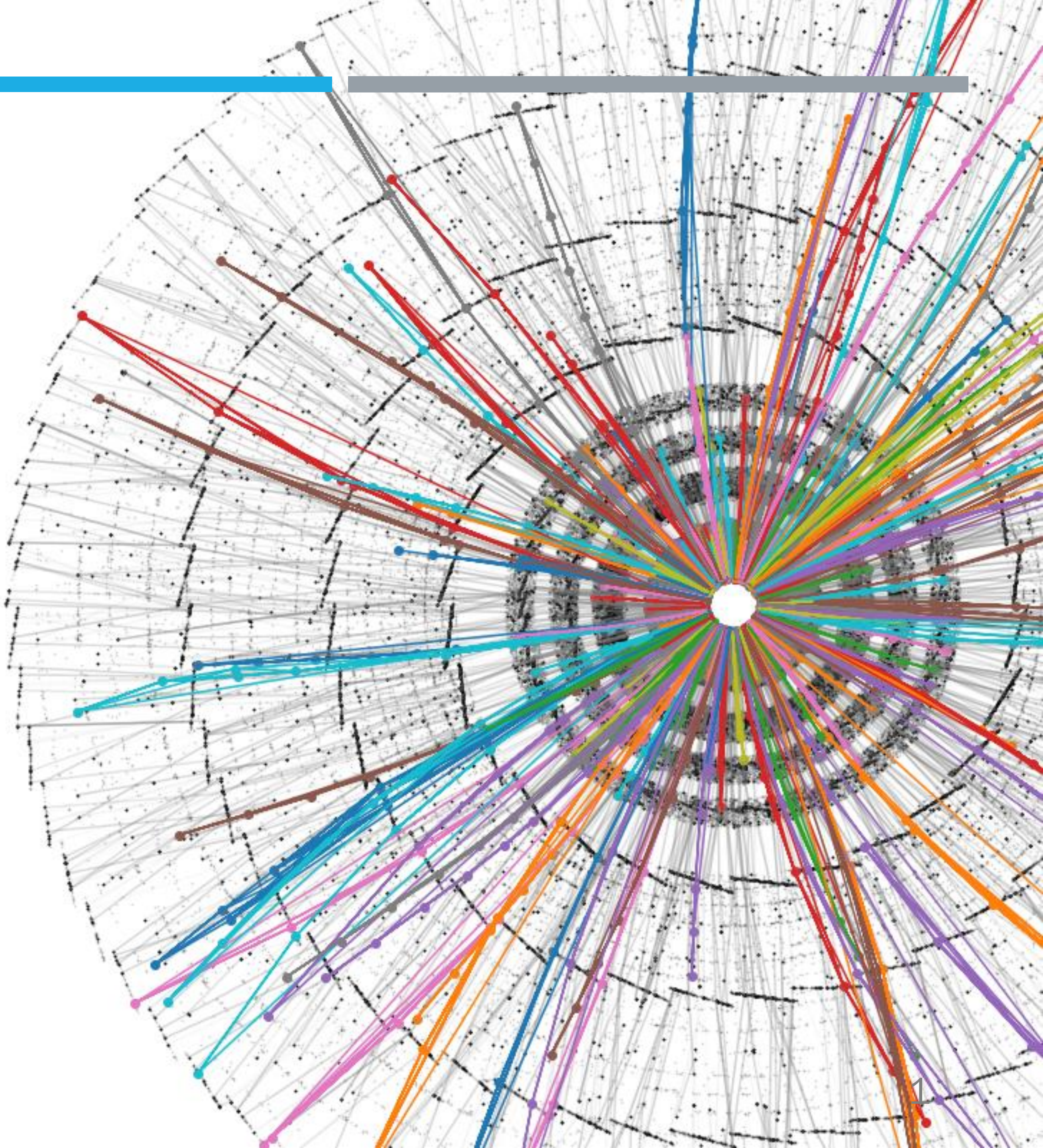
HETEROGENEOUS GRAPH NEURAL NETWORK FOR HI-LUMI LHC

CONNECTING THE DOTS MINI-WORKSHOP
3RD JUNE, 2022, PRINCETON

DANIEL MURNANE & SYLVAIN CAILLOU

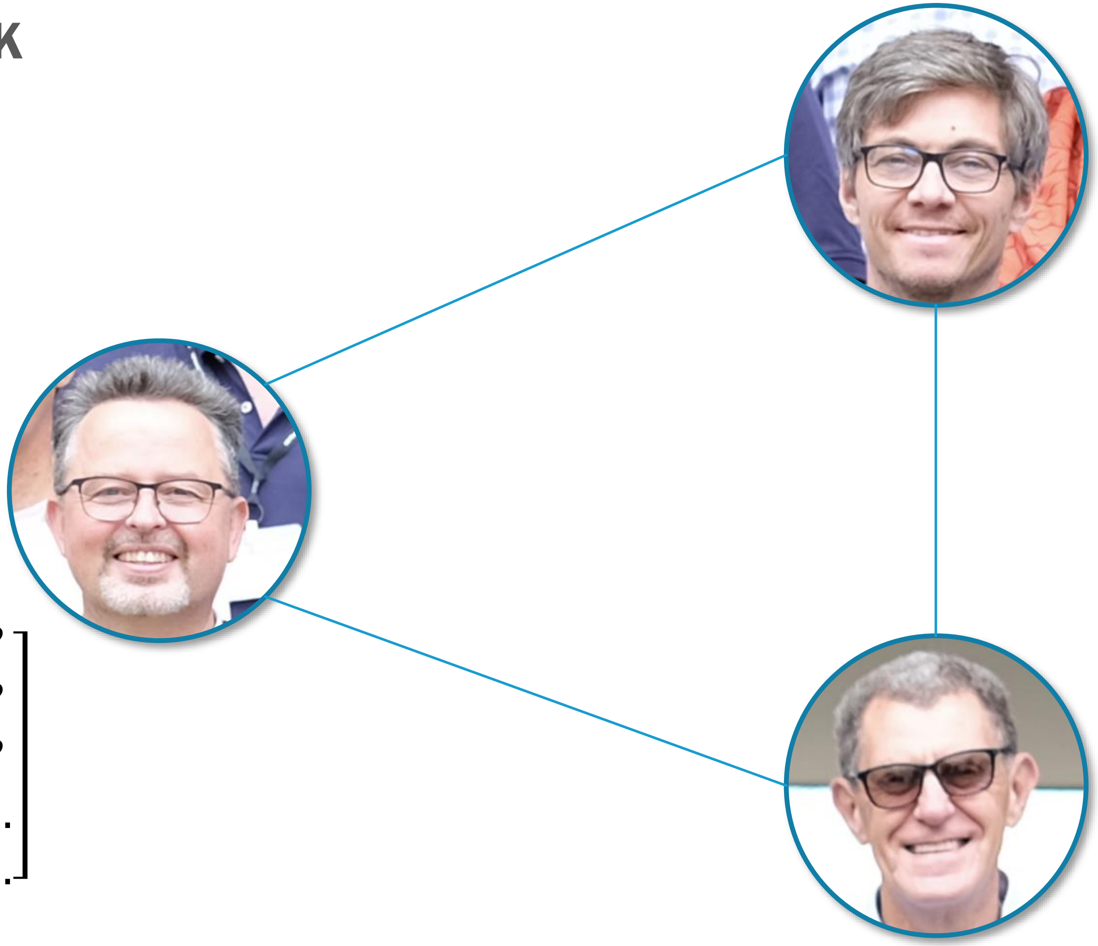
ON BEHALF OF THE **EXATRKX AND L2IT PROJECTS**



ExaTrkX L2IT BERKELEY LAB cnrs

HETEROGENEOUS GRAPH NEURAL NETWORKS

A TYPICAL FRIEND NETWORK



$$\begin{bmatrix} 0.7 \\ 0.3 \\ 0.9 \\ \dots \\ \dots \end{bmatrix}$$

$$Likes = \begin{bmatrix} Beer \\ Pizza \\ Karaoke \\ \dots \\ \dots \end{bmatrix}$$

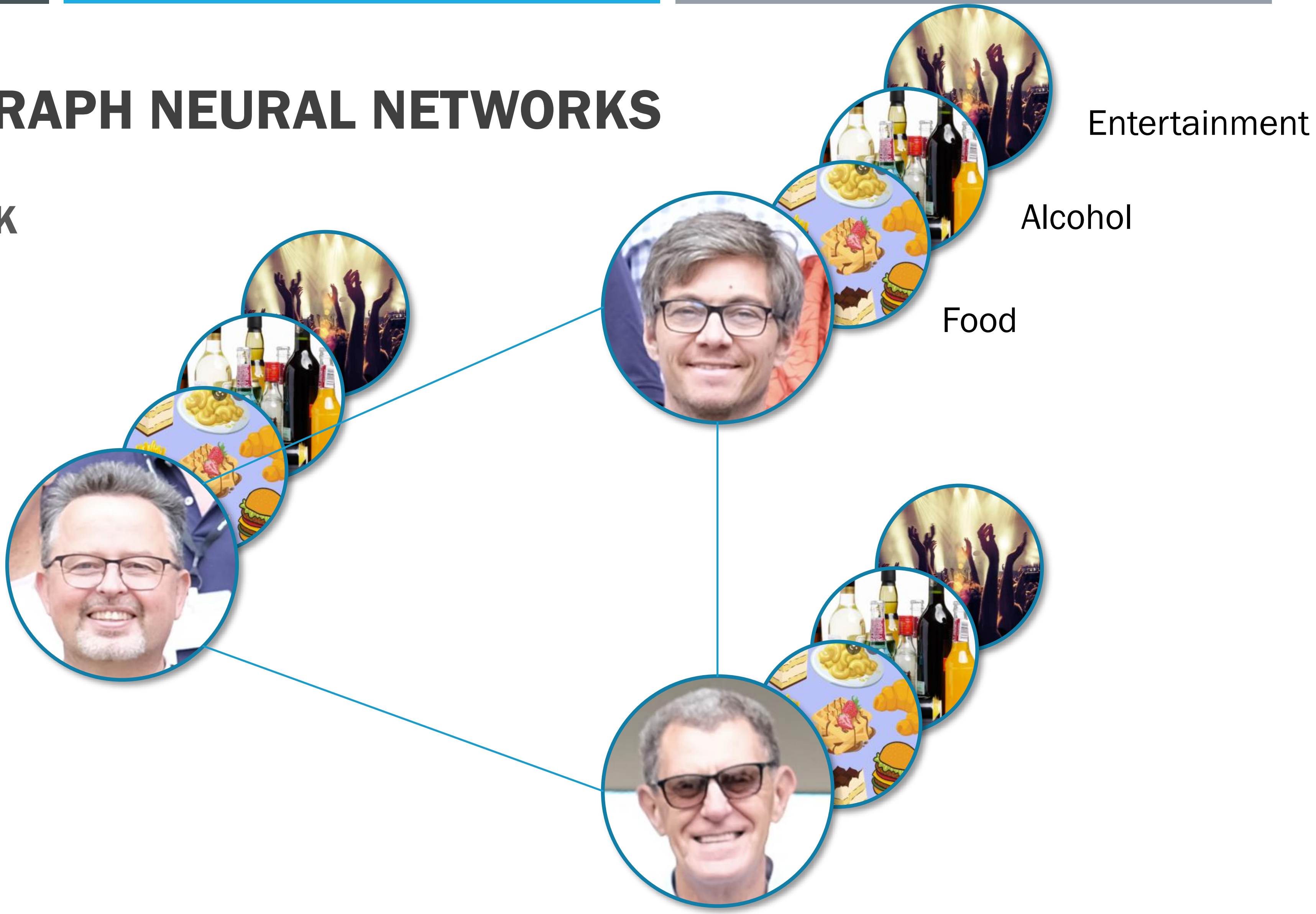
$$\begin{bmatrix} ? \\ ? \\ ? \\ \dots \\ \dots \end{bmatrix}$$

$$\begin{bmatrix} 0.8 \\ 0.7 \\ 0.4 \\ \dots \\ \dots \end{bmatrix}$$

HETEROGENEOUS GRAPH NEURAL NETWORKS

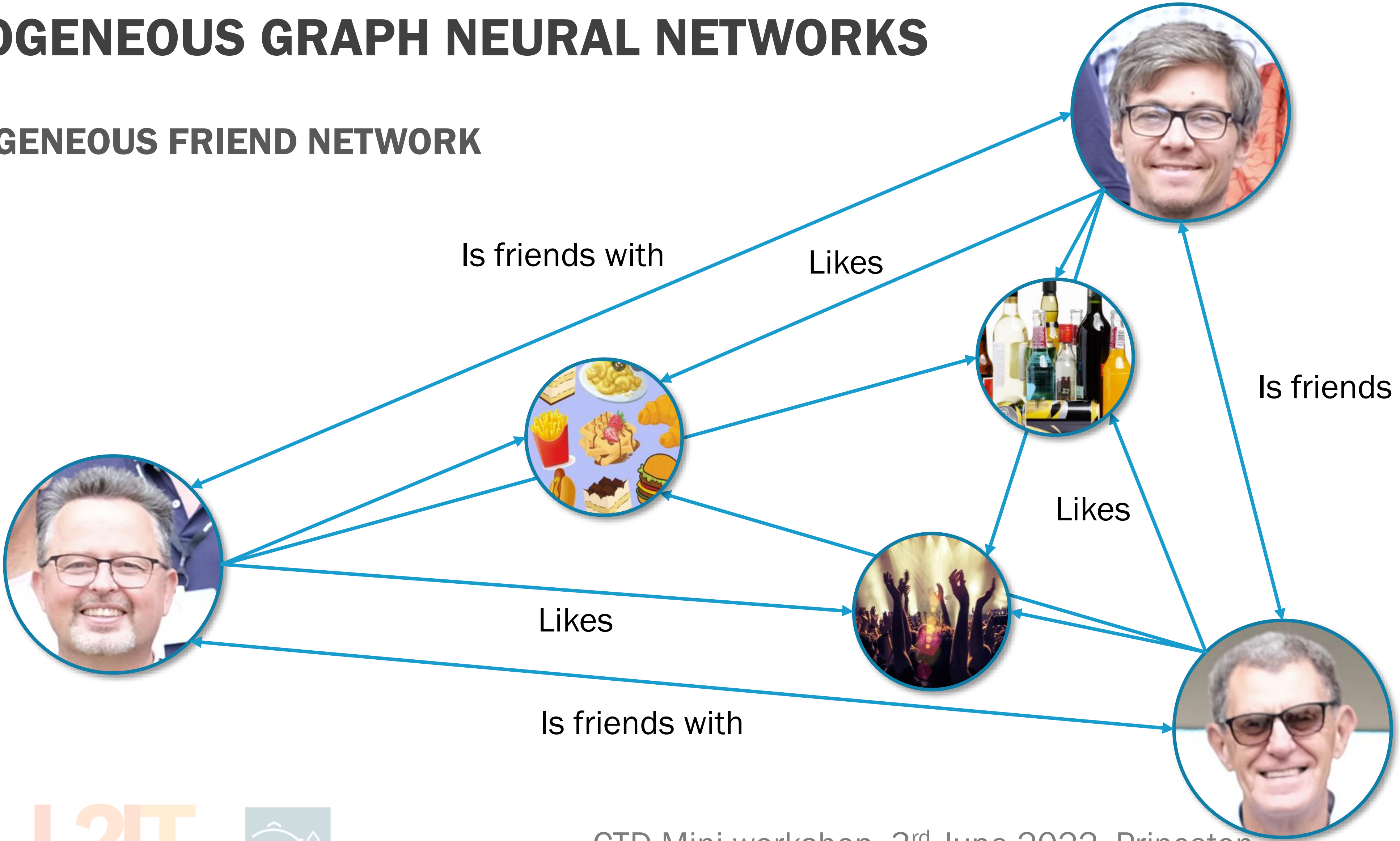
A TYPICAL FRIEND NETWORK

$Food \in R^N$
$Alcohol \in R^M$
$Entertainment \in R^O$



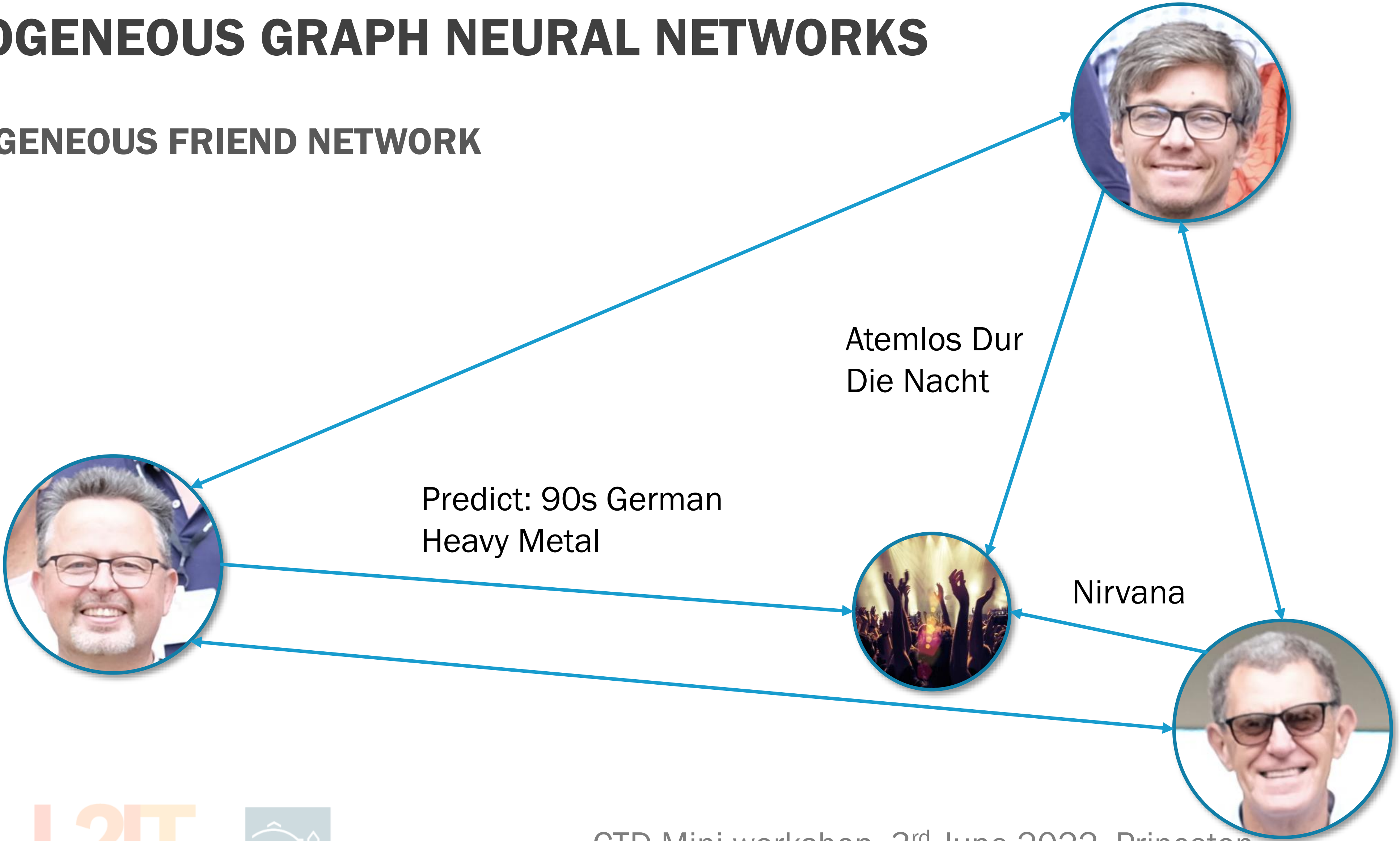
HETEROGENEOUS GRAPH NEURAL NETWORKS

A HETEROGENEOUS FRIEND NETWORK



HETEROGENEOUS GRAPH NEURAL NETWORKS

A HETEROGENEOUS FRIEND NETWORK



HETEROGENEOUS GRAPH NEURAL NETWORKS

- Can do heterogeneity with padding, long one-hot encodings, etc. using homogeneous GNN
- It is hard to reproduce comparisons between homoGNNs and heteroGNNs, but Zhang et al did exactly that
- Showed their model *HetGNN* outperformed homoGNNs on most tasks (involving different node/edge types)
- There are now tools* that handle heteroGNN natively, which can simplify implementation
- The results we show *don't* use a library, so could be optimized

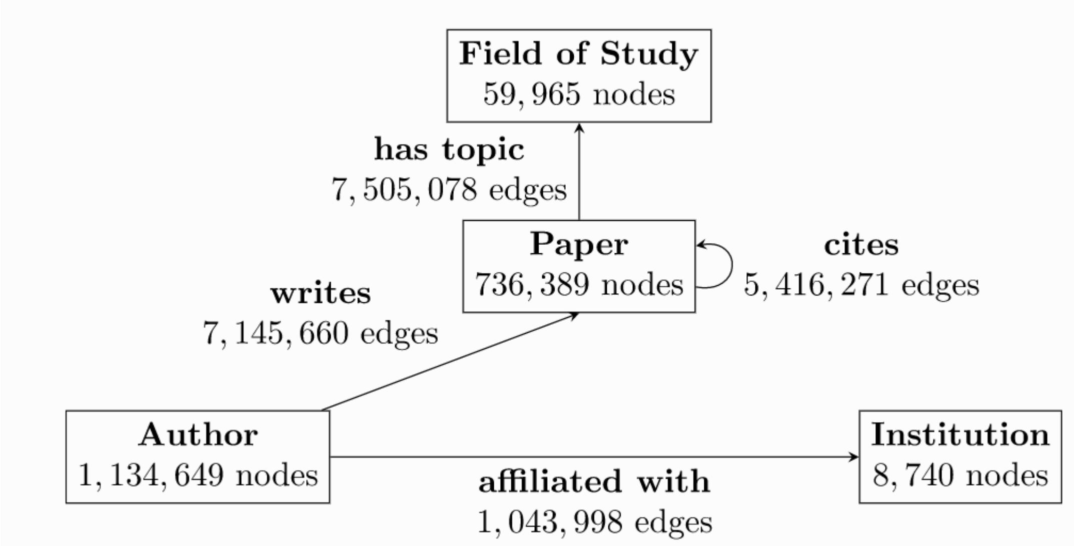
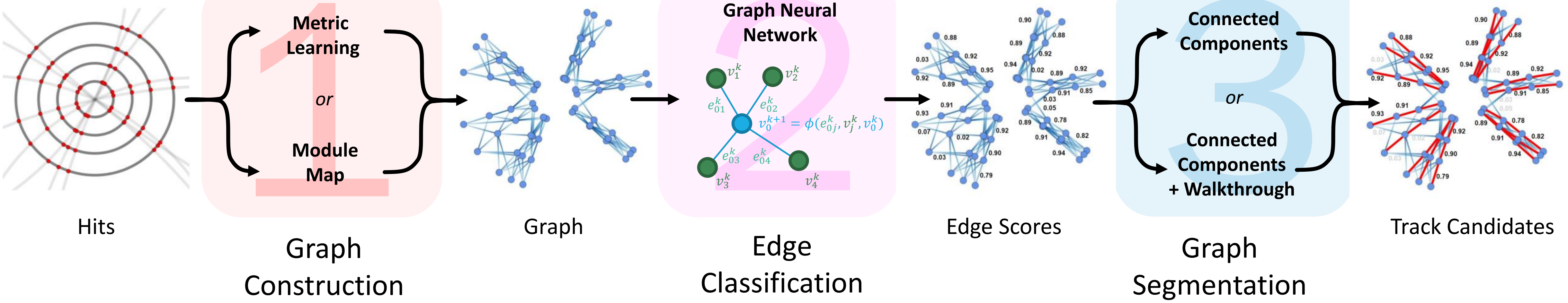


Figure from Pytorch Geometric documentation – represents ogbn-mag dataset

*Pytorch Geometric HeteroData, DGL HeteroGraph, new kid on the block GNNKeras?

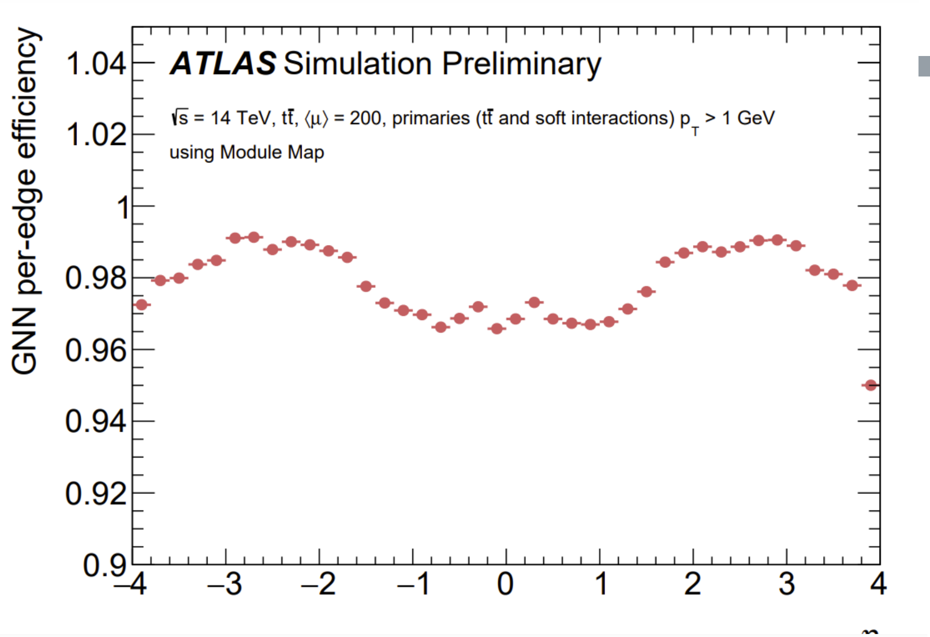
GOAL & PIPELINE OVERVIEW

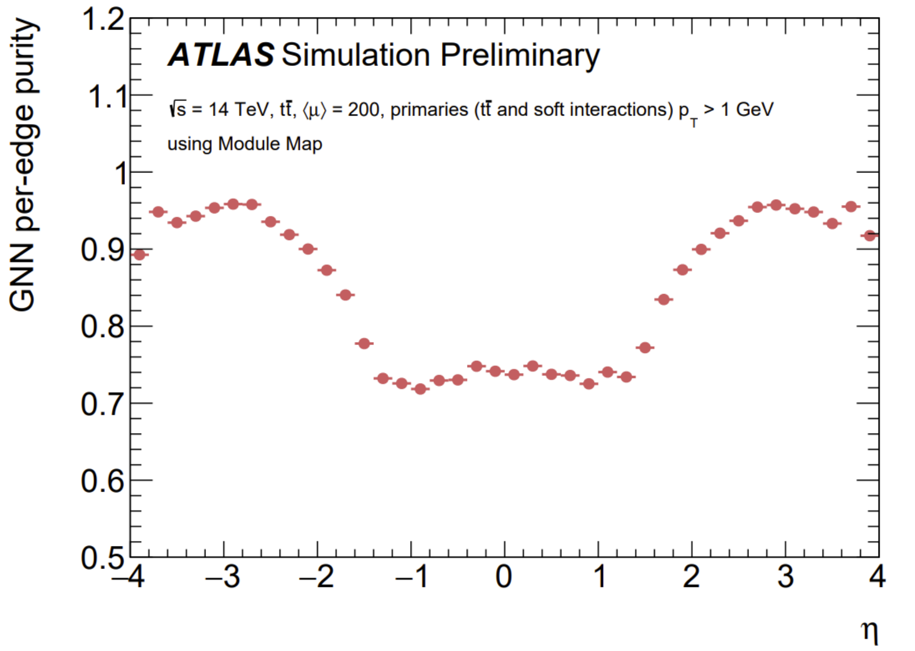
- **Goal:** From a list of spacepoints, produce a list of track candidates, where each candidate is a list of spacepoints
- Current pipeline of the L2IT-Exatrkx collaborative effort
- Each stage offers multiple independent choices, depending on hardware and time constraints



CURRENT PIPELINE PERFORMANCE

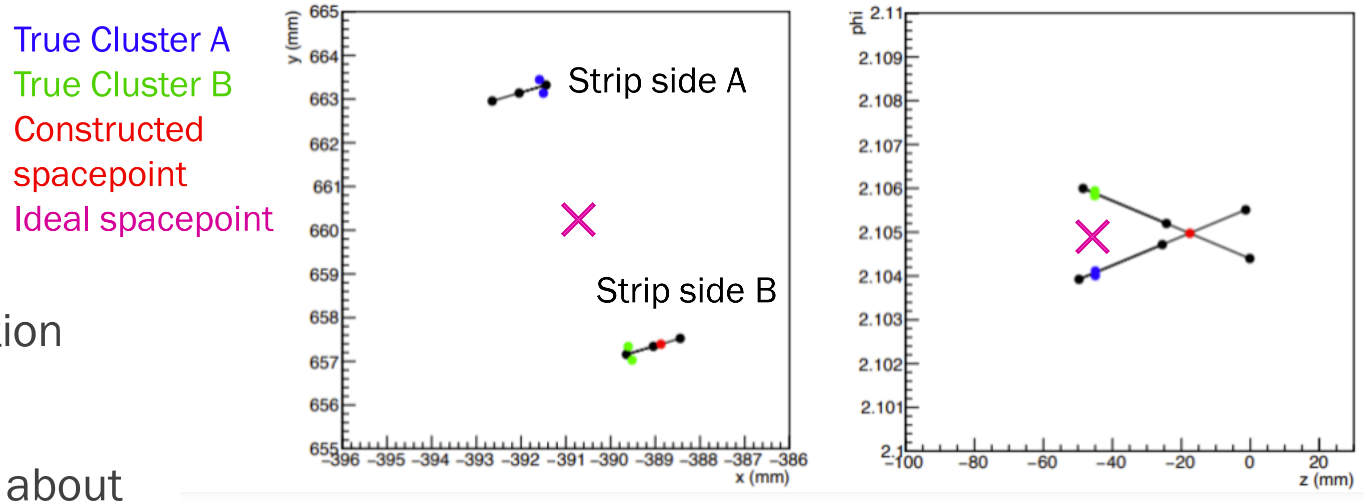
- Consider GNN performance on edge classification across pseudorapidity η
- Drop in performance at low η – what is special about this region?



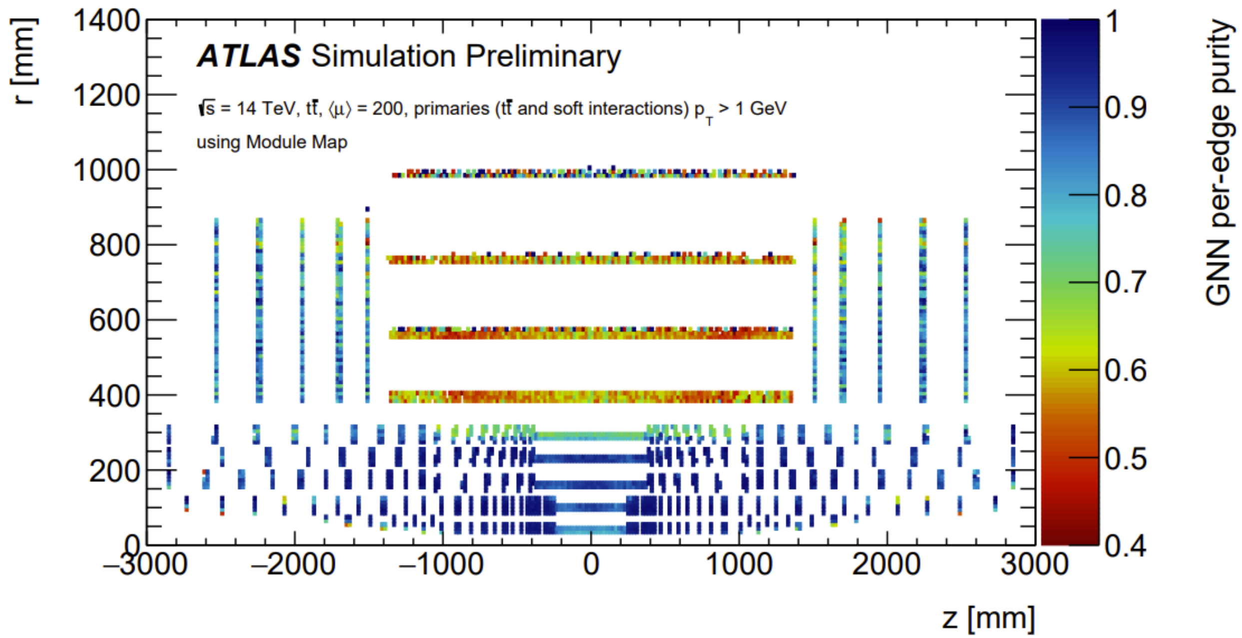


See Charline's talk!

CURRENT PIPELINE PERFORMANCE



- Consider GNN performance on edge classification across pseudorapidity η
- Drop in performance at low η – what is special about this region?
- Low performance in barrel strips, where spacepoints are built from two strip clusters
- Spacepoint position may be far from “ideal” position – i.e. midpoint between ground truth clusters
- How can we attach these two sets of cluster features? Pixel spacepoints only have one set of cluster features…



MESSAGE PASSING MECHANISM

For each node neighborhood:

a) Pass node channels through a multi-layer perceptron (MLP) encoder

b) Pass encoded channels along each edge to the central node of the neighborhood

At each node:

Sum all messages

Repeat

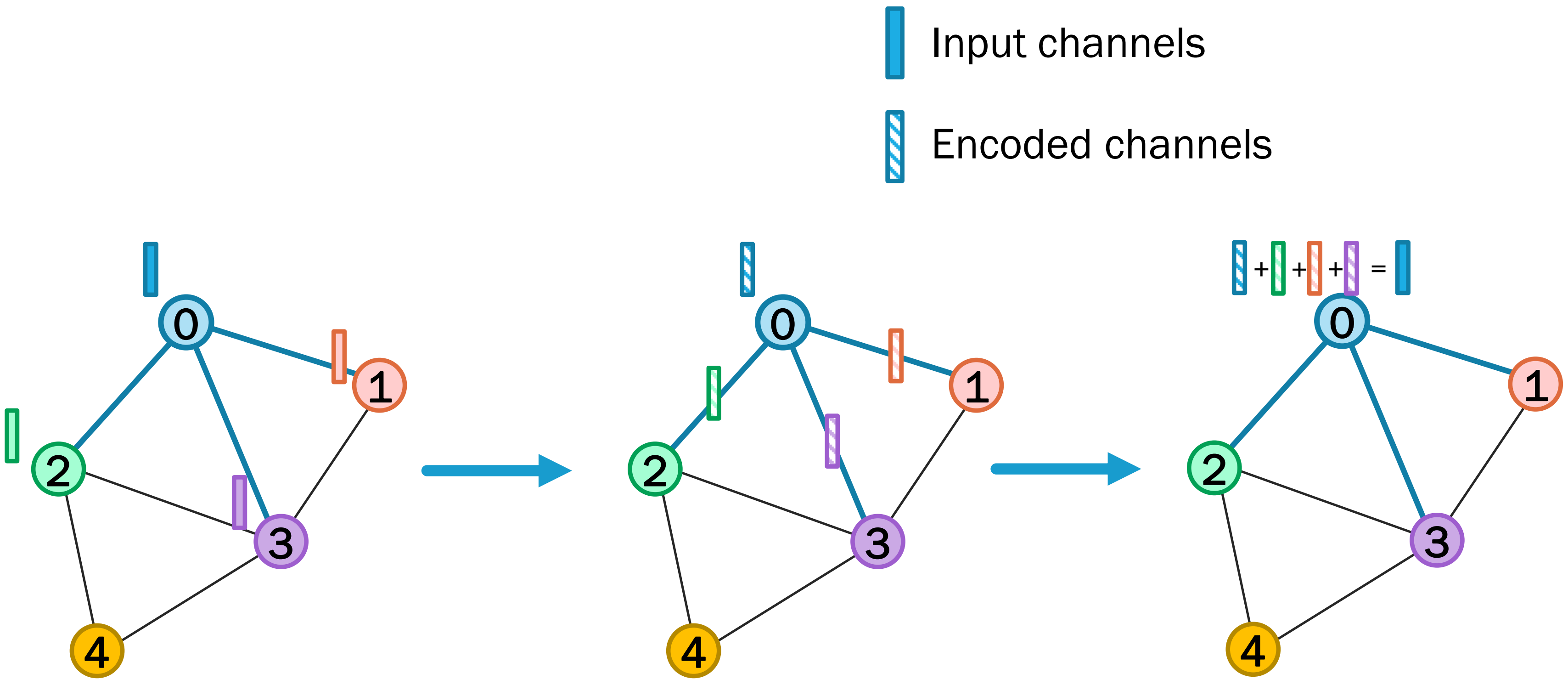
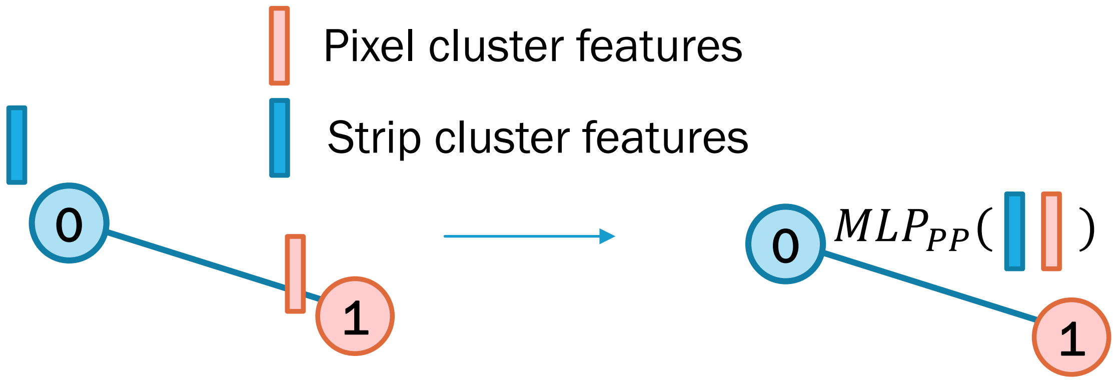
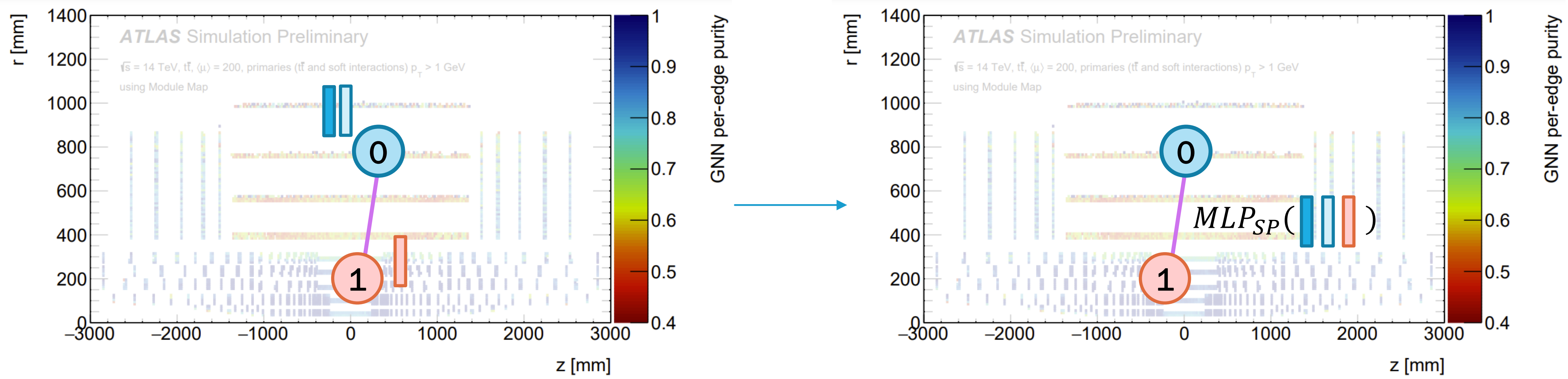


Figure inspired by [Koshi et. al.]

MINIMAL HETEROGENEITY: EDGE MLP

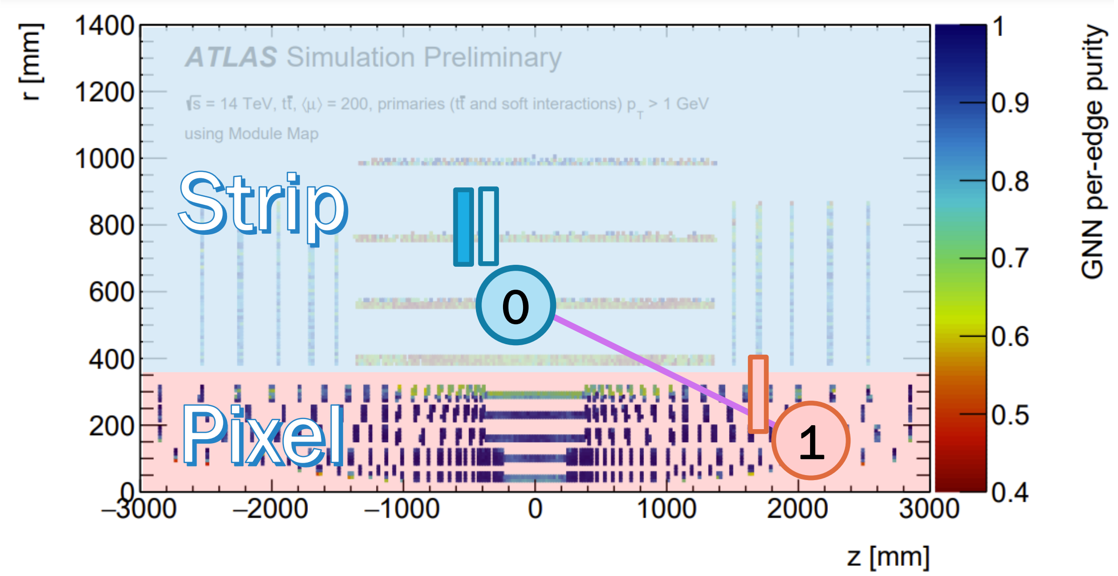


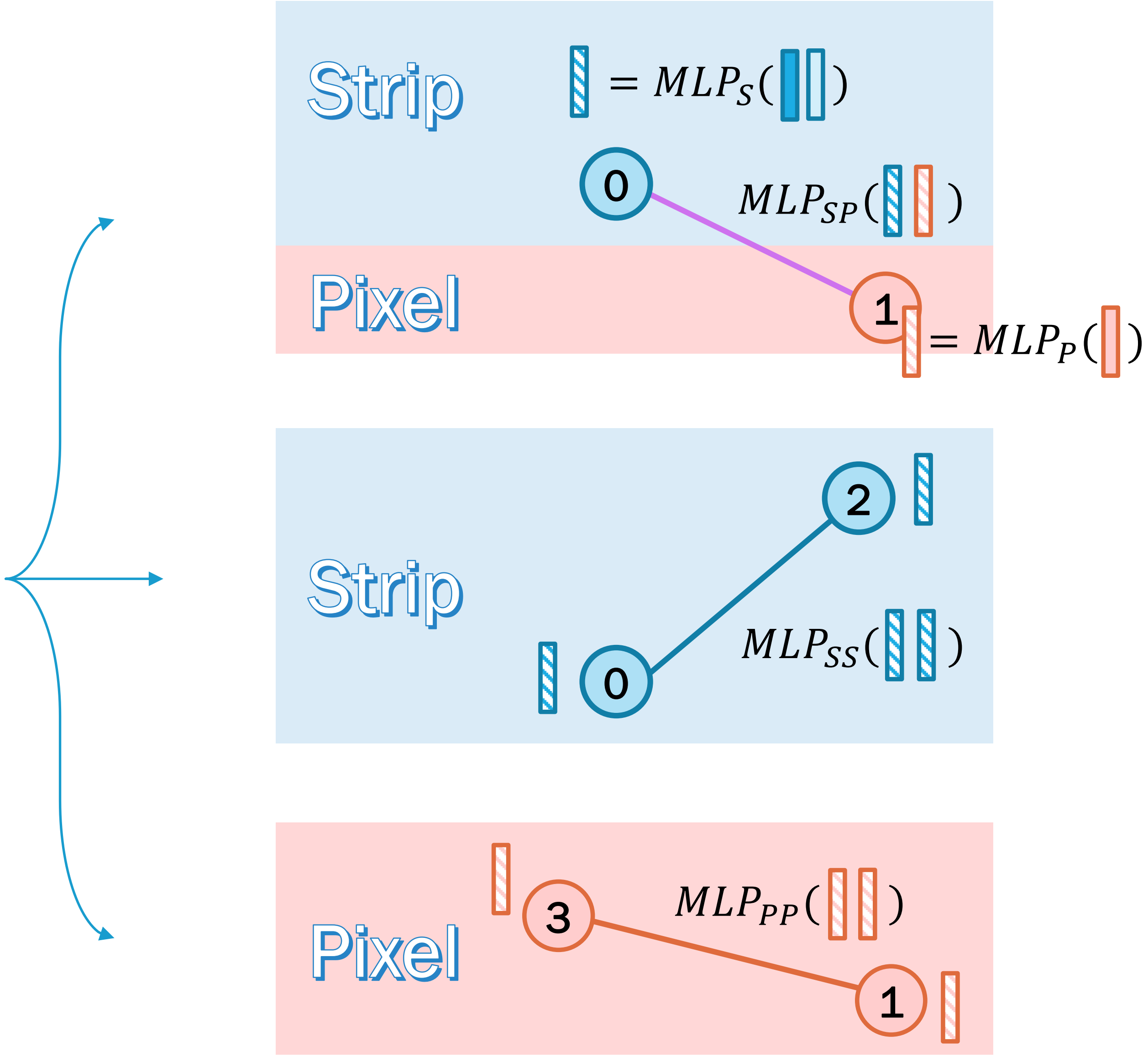
- To get intuition, consider simple edge classifier MLP applied to two pixel nodes:
- To apply a filter MLP to a pixel (single cluster) and strip (double cluster) node combination, need a *different* MLP:



- Already gives better than homogeneous filter MLP (~2x construction purity)

MINIMAL HETEROGENEITY: EDGE CLASSIFIER GNN

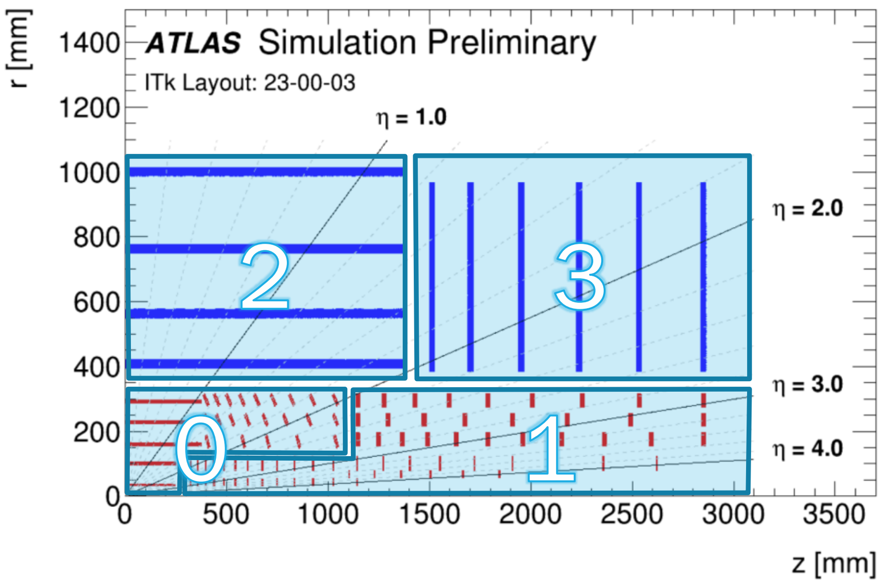




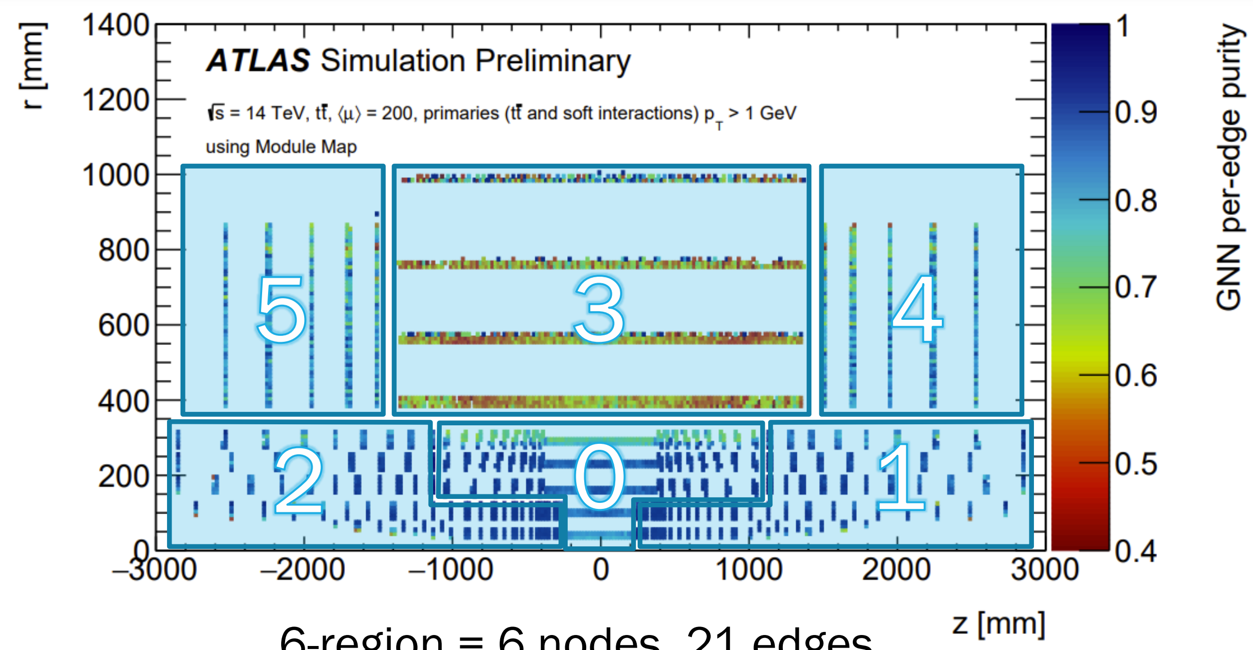
- Node strip encoder and node pixel encoder
- Edge strip-strip encoder, strip-pixel encoder and pixel-pixel encoder

NON-MINIMAL EXTENSIONS: MULTIPLE NODE TYPES

- Can extend logic to all distinct hardware regions in detector
- For a four-region heterogeneous GNN, we have four node encoders/networks (N_0, N_1, N_2, N_3) and ten edge encoders/networks $(E_{00}, E_{01}, E_{02}, E_{03}, E_{11}, \ldots, E_{34}, E_{44})$
- Larger model and takes longer to train
- Note: Could have heterogeneous (i.e. different, dedicated) models with the same node features
- **For each edge and node type, we need a dedicated MLP model**



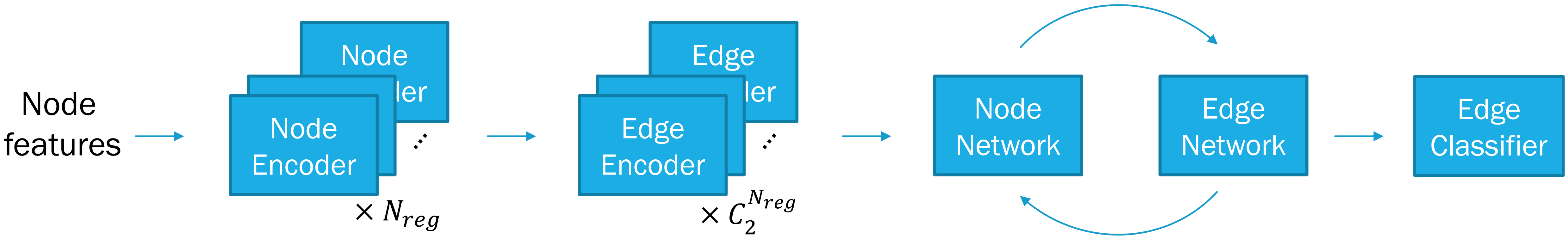
4-region Symmetrical = 4 nodes, 10 edges



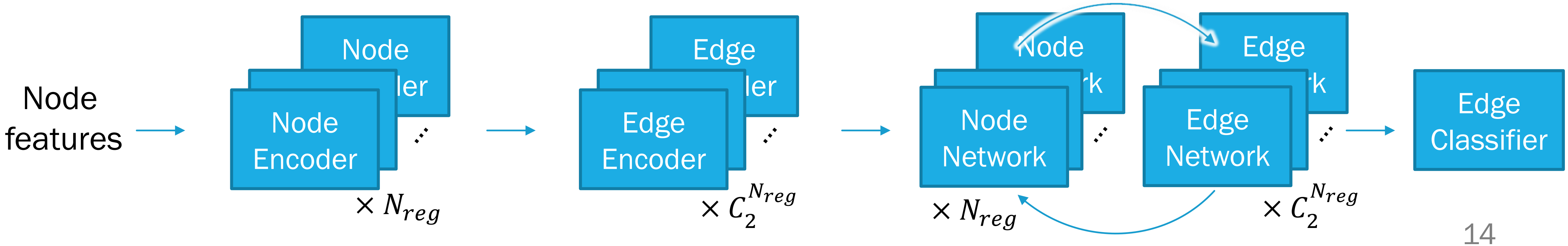
6-region = 6 nodes, 21 edges

NON-MINIMAL EXTENSIONS: HETERO MESSAGE PASSING

Minimal case: Hetero node and edge encoders for N_{reg} regions



Extension: Hetero node and edge networks



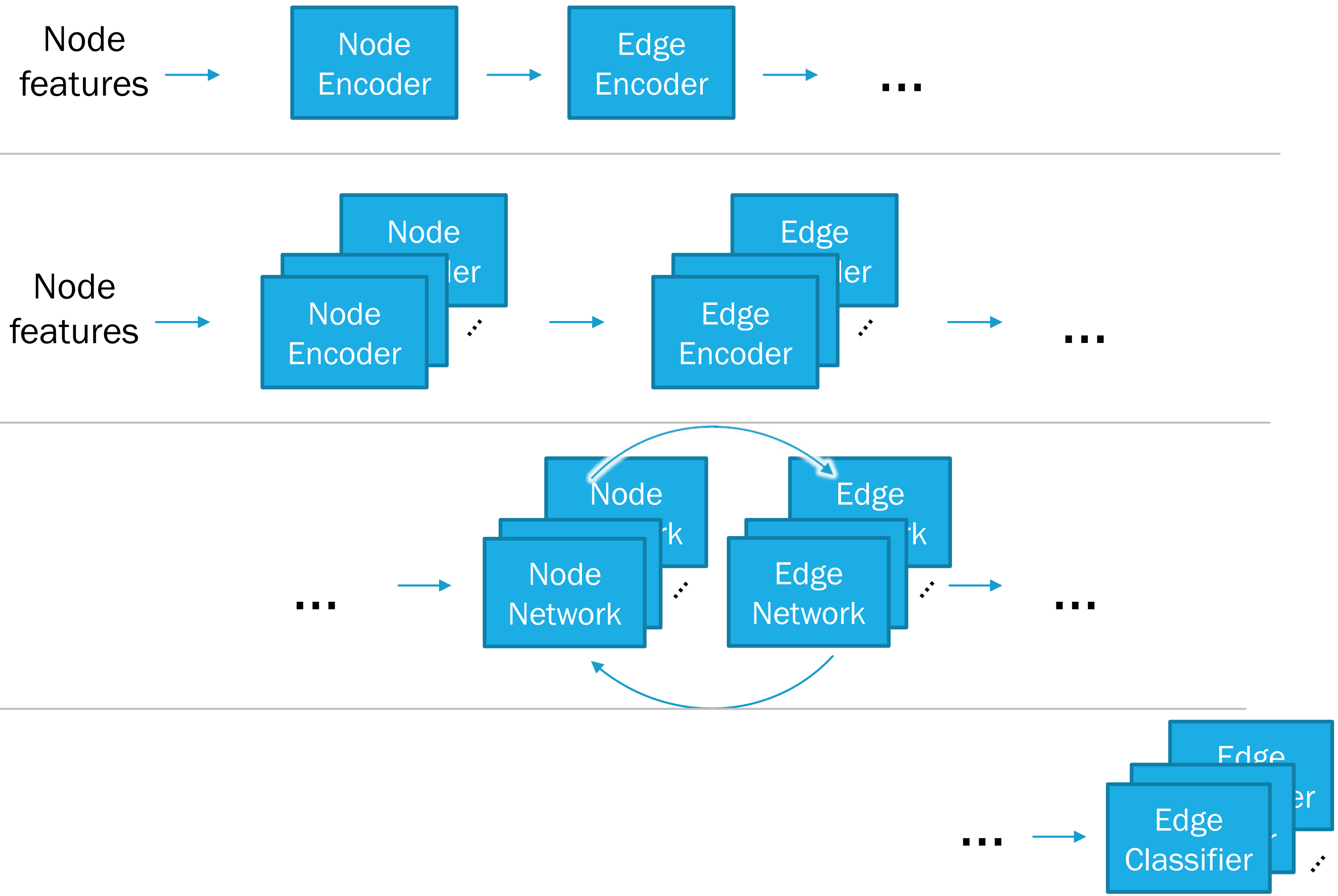
NON-MINIMAL EXTENSIONS: HETERO MESSAGE PASSING

Level 0: Homogeneous

Level 1: HeteroEncoders

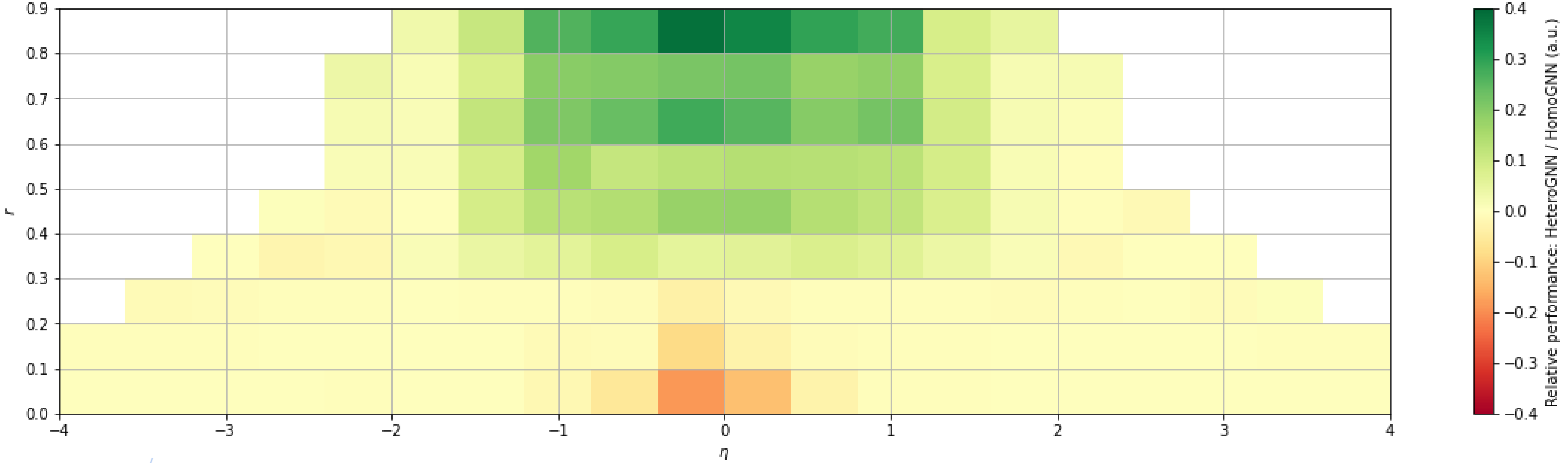
Level 2: HeteroGNN

Level 3: HeteroOutput



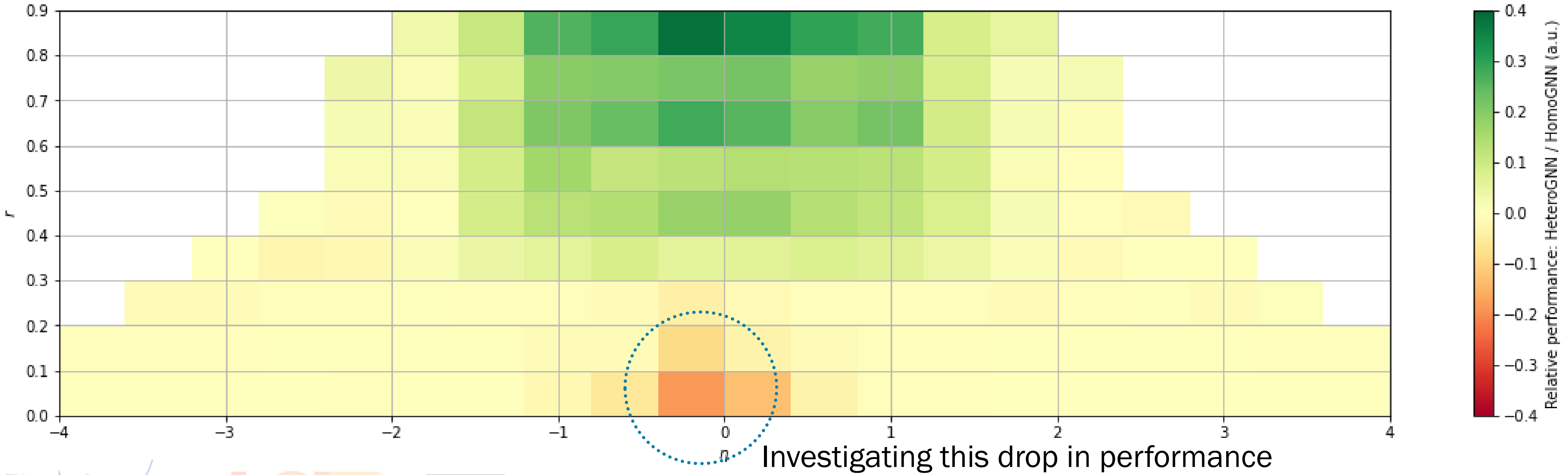
RESULTS

- Apply two models to toy $t\bar{t}, \mu = \langle 200 \rangle$ dataset: homogeneous GNN and best-performing heterogeneous dataset
- HeteroGNN is a level 1 (only heterogeneous encoders), and 3-region (dedicated MLPs for pixel, barrel strip, and endcap strip)
- Compare relative performance across the detector – as expected barrel strip region performance significantly improved



RESULTS

- Apply two models to toy $t\bar{t}, \mu = \langle 200 \rangle$ dataset: homogeneous GNN and best-performing heterogeneous dataset
- HeteroGNN is a level 1 (only heterogeneous encoders), and 3-region (dedicated MLPs for pixel, barrel strip, and endcap strip)
- Compare relative performance across the detector – as expected barrel strip region performance significantly improved



NEXT STEPS

- Reproduce the whole pipeline up to approved plots with full ITk dataset, including track reconstruction performance
- Study improvement to track reconstruction
- Understand *what* is giving the improvement – using different models, using all the cluster features, or both?
- Balancing LR / weighting between regions
- Insert cluster shape / energy deposit features
- Investigate other architectures applied to hetero structure

CONCLUSION

- Heterogeneous GNNs are straightforward to implement by hand
- Dedicated libraries are being produced that can handle even this small amount of data management automatically
- If you have physically/conceptually different node types, or extra features, don't use padding – use dedicated MLPs for each node and edge type
- Heterogeneous encoders coupled with homogeneous node/edge networks may offer the best bang for buck: Handle separate input features but maintain common message passing space

DO YOU HAVE HETEROGENEOUS DATA? CHIME IN!

Links

ExaTrkx website ● L2IT website ● ExaTrkx paper ● L2IT paper ● Codebase

TrkX

BERKELEY LAB

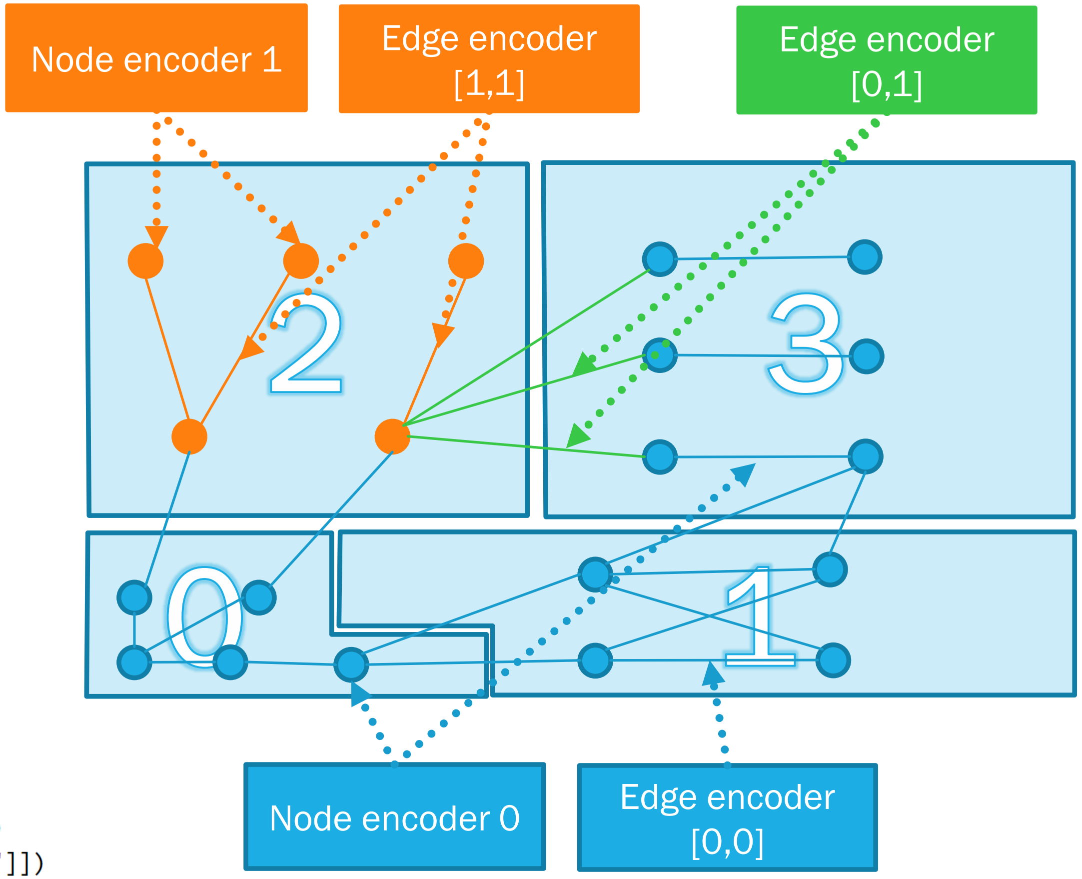
BACKUP

HOW TO IMPLEMENT HETEROGENEOUS ARCHITECTURE

Now consider a node encoder 1 specific to the barrel strip volume, and a node encoder 0 for all other nodes. Message passing proceeds as:

1. Pass node features through the node encoder that belongs to that volume ID. That is, if `volume_id` ∈ $[0, 1, 3]$ then pass (r^s, ϕ^s, z^s) through encoder 0. If `volume_id` ∈ $[2]$ then pass $(r^s, \phi^s, z^s, r^{c_1}, \phi^{c_1}, z^{c_1}, r^{c_2}, \phi^{c_2}, z^{c_2})$ through encoder 1

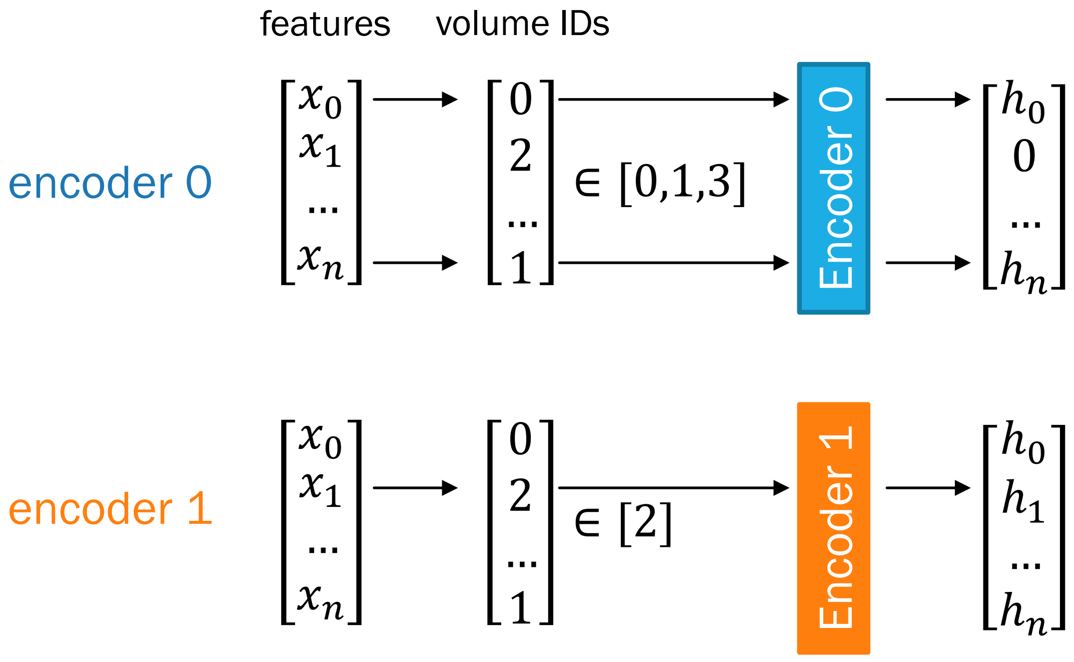
```
encoded_nodes = torch.empty((x.shape[0], self.hparams["hidden"])).to(self.device)
for encoder, model in zip(self.node_encoders, self.hparams["model_ids"]):
    node_id_mask = torch.isin(volume_id, torch.tensor(model["volume_ids"]).to(self.device))
    encoded_nodes[node_id_mask] = checkpoint(encoder, x[node_id_mask, :model["num_features"]])
```



HOW TO IMPLEMENT HETEROGENEOUS ARCHITECTURE

Now consider a node encoder 1 specific to the barrel strip volume, and a node encoder 0 for all other nodes. Message passing proceeds as:

1. Pass node features through the node encoder that belongs to that volume ID. That is, if `volume_id` $\in$ $[0, 1, 3]$ then pass (r^S, ϕ^S, z^S) through encoder 0. If `volume_id` $\in$ $[2]$ then pass $(r^S, \phi^S, z^S, r^{C_1}, \phi^{C_1}, z^{C_1}, r^{C_2}, \phi^{C_2}, z^{C_2})$ through encoder 1




```
encoded_nodes = torch.empty((x.shape[0], self.hparams["hidden"])).to(self.device)
for encoder, model in zip(self.node_encoders, self.hparams["model_ids"]):
    node_id_mask = torch.isin(volume_id, torch.tensor(model["volume_ids"]).to(self.device))
    encoded_nodes[node_id_mask] = checkpoint(encoder, x[node_id_mask, :model["num_features"]])
```