OBJECT CONDENSATION
FOR GNN-BASED PARTICLE TRACKING

GAGE DEZOORT

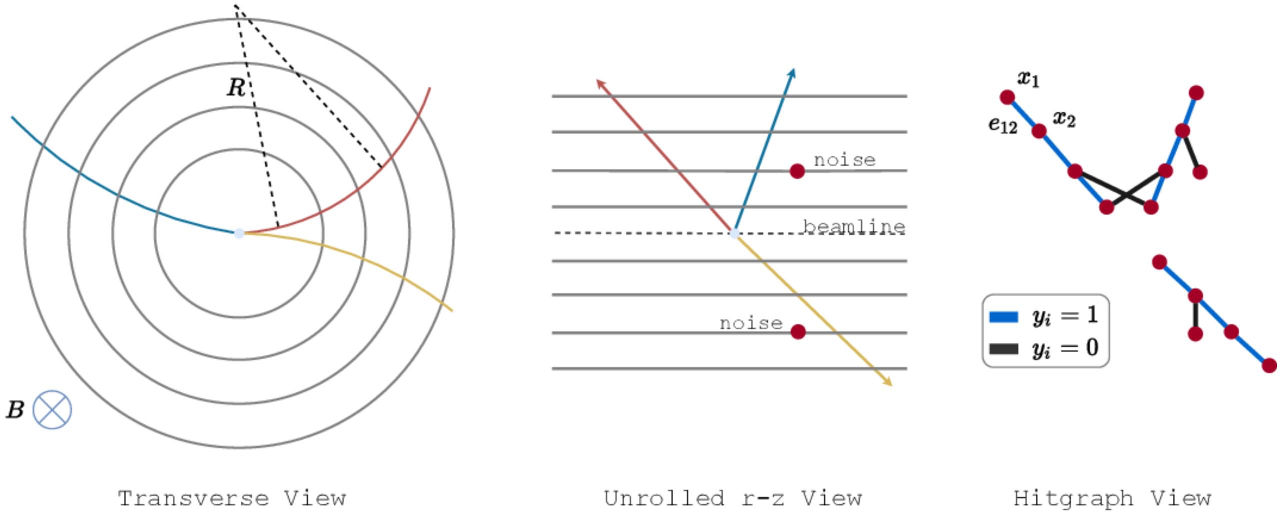
06/1/2022

GNN TRACKING

EDGE CLASSIFICATION PARADIGM

Edge Classification Task

- Draw edges to hypothesize various particle trajectories, train a GNN to classify edges

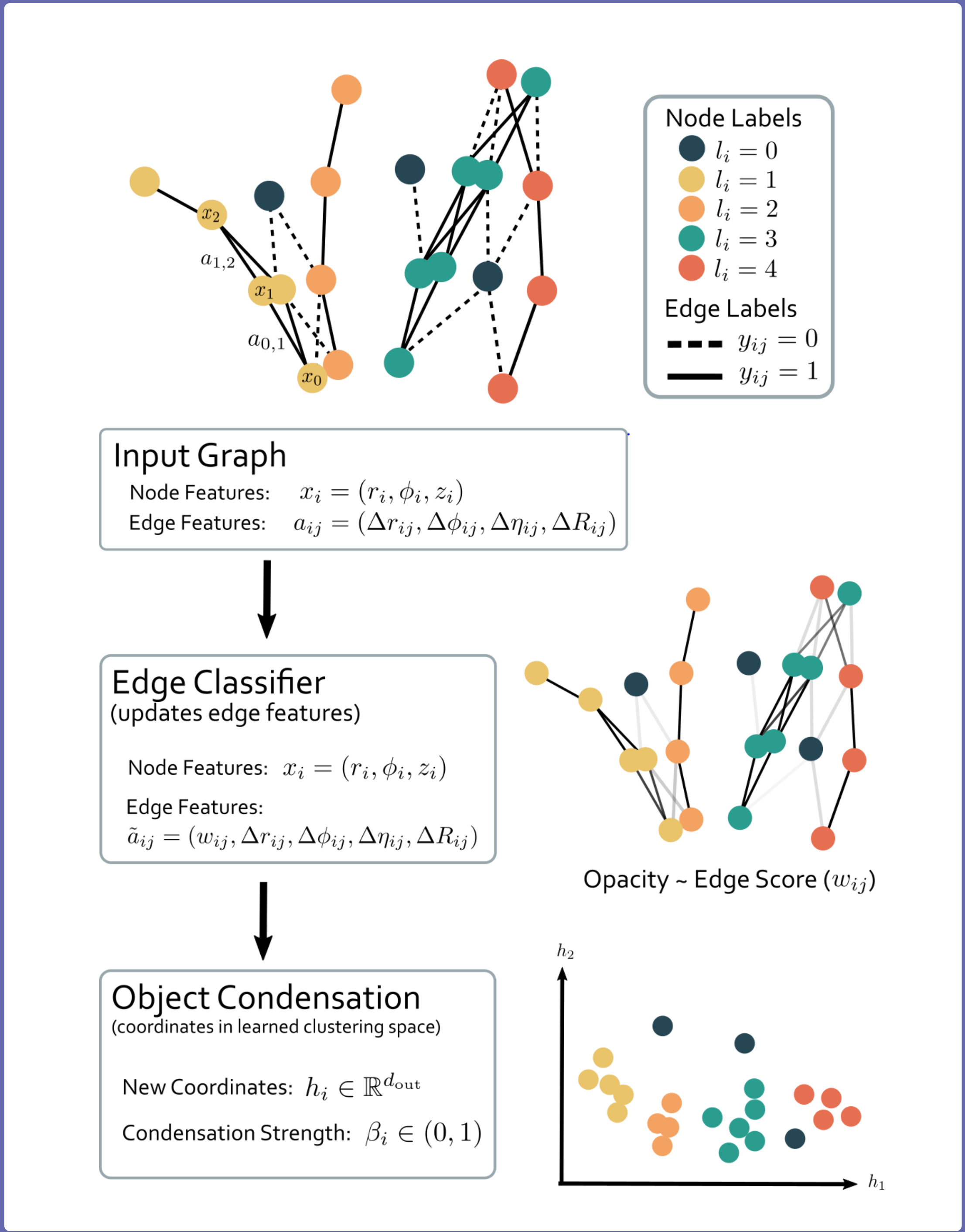


- Use edge weights to produce tracks (i.e. apply a threshold to produce disjointed subgraphs)

- **Key steps** (general to many GNN workflows)
  1) Graph construction from underlying data
  2) GNN-based inference
  3) *Post processing to form tracks*

EDGE CLASSIFICATION / OBJECT CONDENSATION

STRATEGY OVERVIEW



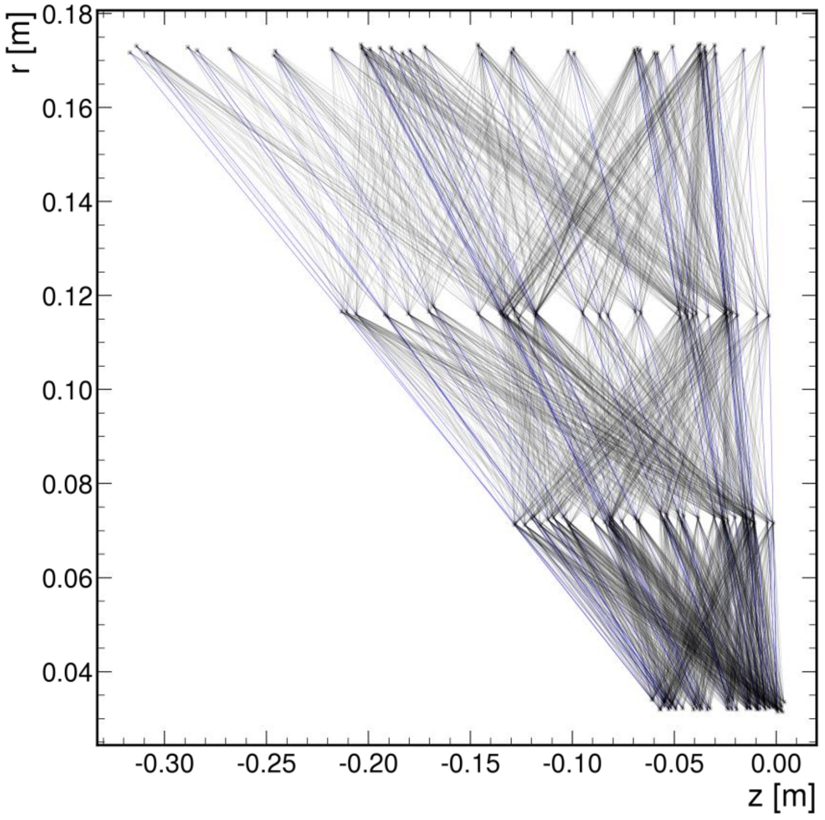
GRAPH CONSTRUCTION
PER-SECTOR BREAKDOWN

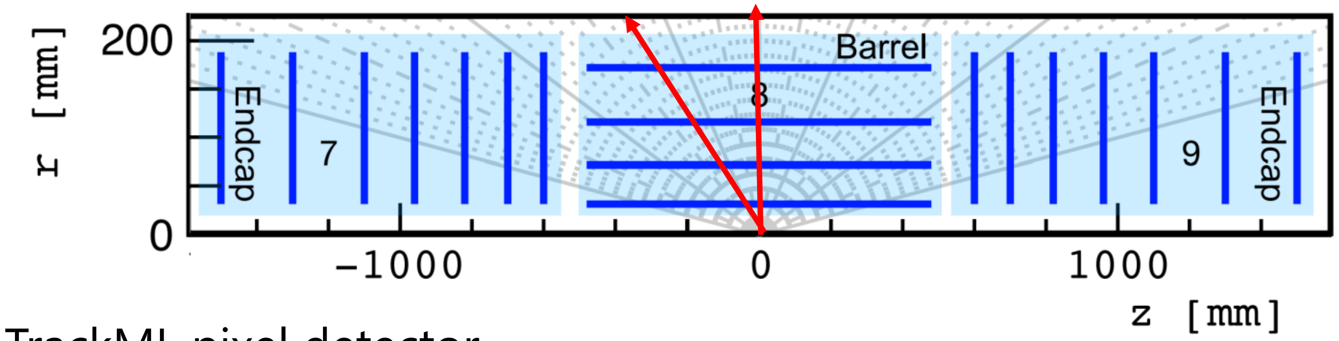
- Track p_T > 1.0 GeV
- phi_slope < 0.007
- z0 < 350 mm
- n_phi_sectors: 8
- n_eta_sectors: 8
- phi sector overlap: 0.08
- eta sector overlap: 0.125
- remove_noise: true

$\eta \in (-1.25, 0)$

n_{nodes} = 181+/-48
n_{edges} = 3380+/-1828

purity = 0.056+/-0.017
efficiency = 0.999+/-0.001





TrackML pixel detector

GRAPH CONSTRUCTION

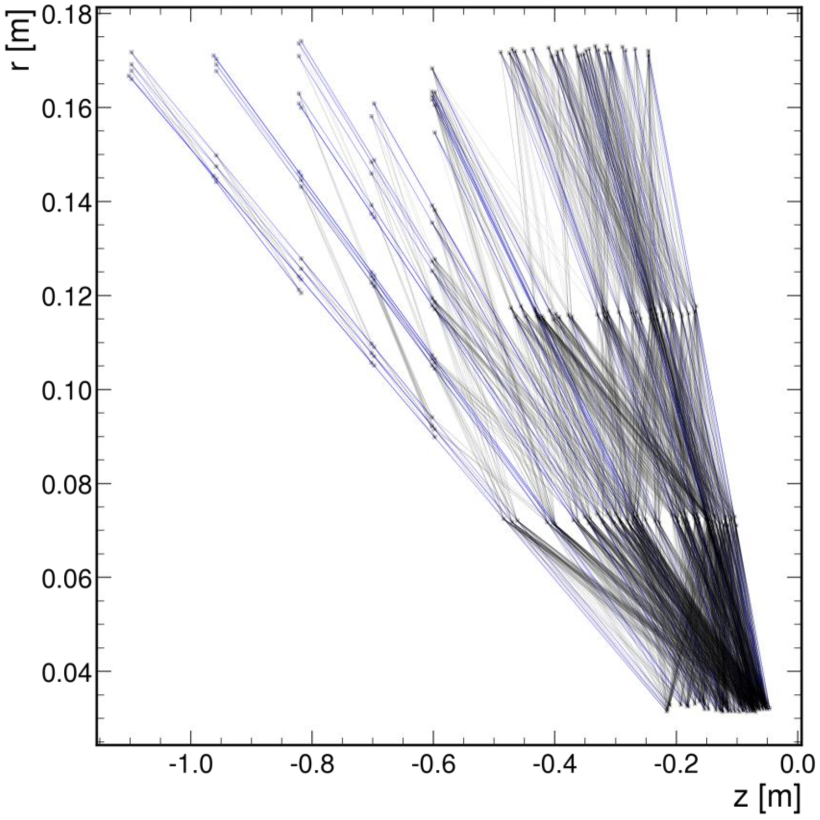
PER-SECTOR BREAKDOWN

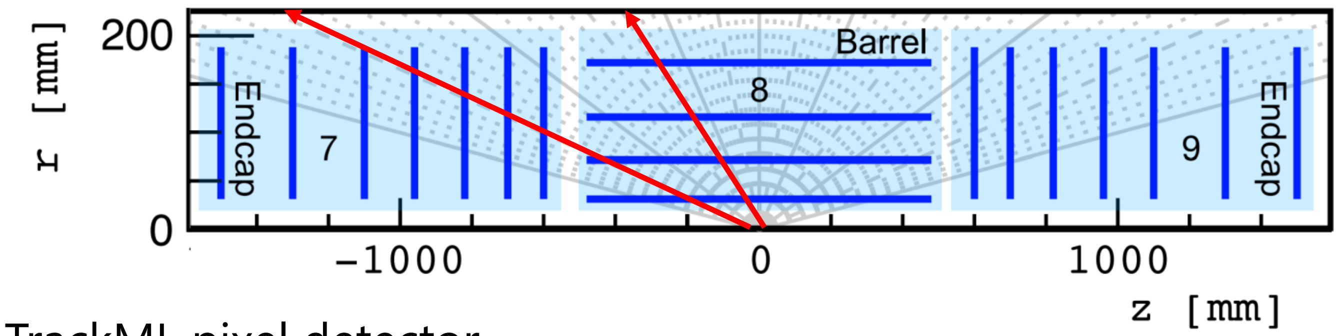
- Track p_T > 1.0 GeV
- phi_slope < 0.007
- z0 < 350 mm
- n_phi_sectors: 8
- n_eta_sectors: 8
- phi sector overlap: 0.08
- eta sector overlap: 0.125
- remove_noise: true

$\eta \in (-2.5, -1.25)$

n_{nodes} = 178+/-45
n_{edges} = 1923+/-947

purity = 0.098+/-0.029
efficiency = 0.996+/-0.007





TrackML pixel detector

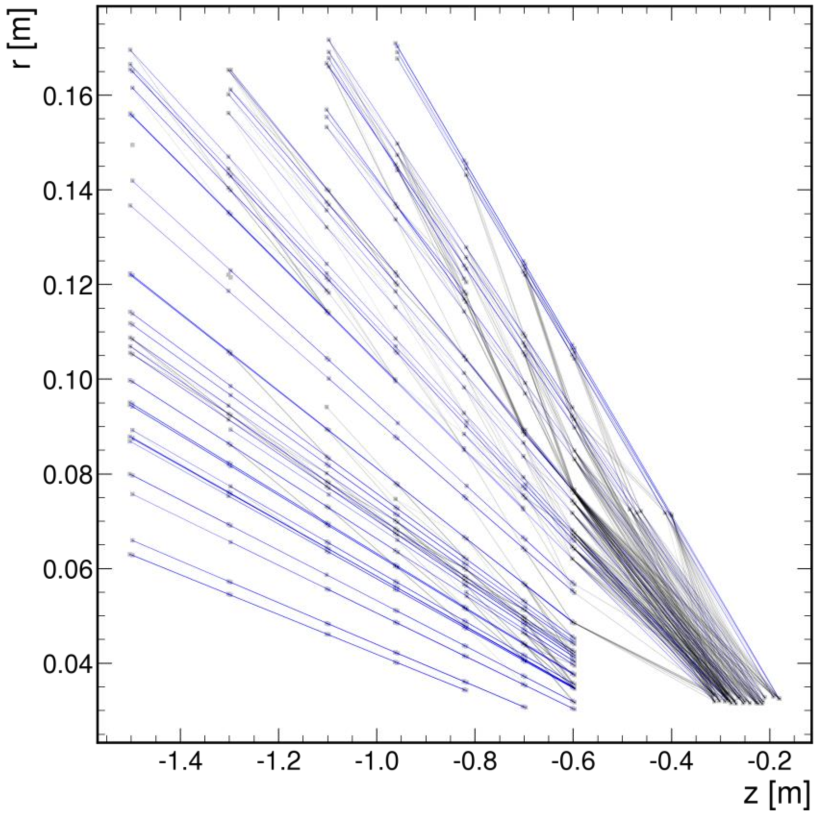
GRAPH CONSTRUCTION
PER-SECTOR BREAKDOWN

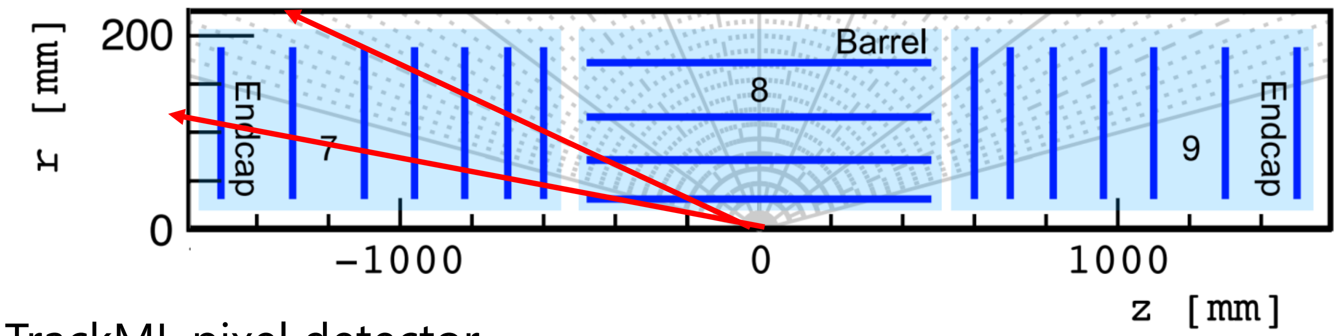
- Track p_T > 1.0 GeV
- phi_slope < 0.007
- z0 < 350 mm
- n_phi_sectors: 8
- n_eta_sectors: 8
- phi sector overlap: 0.08
- eta sector overlap: 0.125
- remove_noise: true

$\eta \in (-3.75, -2.5)$

n_{nodes} = 272+/-16
n_{edges} = 743+/-313

purity = 0.555+/-0.104
efficiency = 0.996+/-0.009





TrackML pixel detector

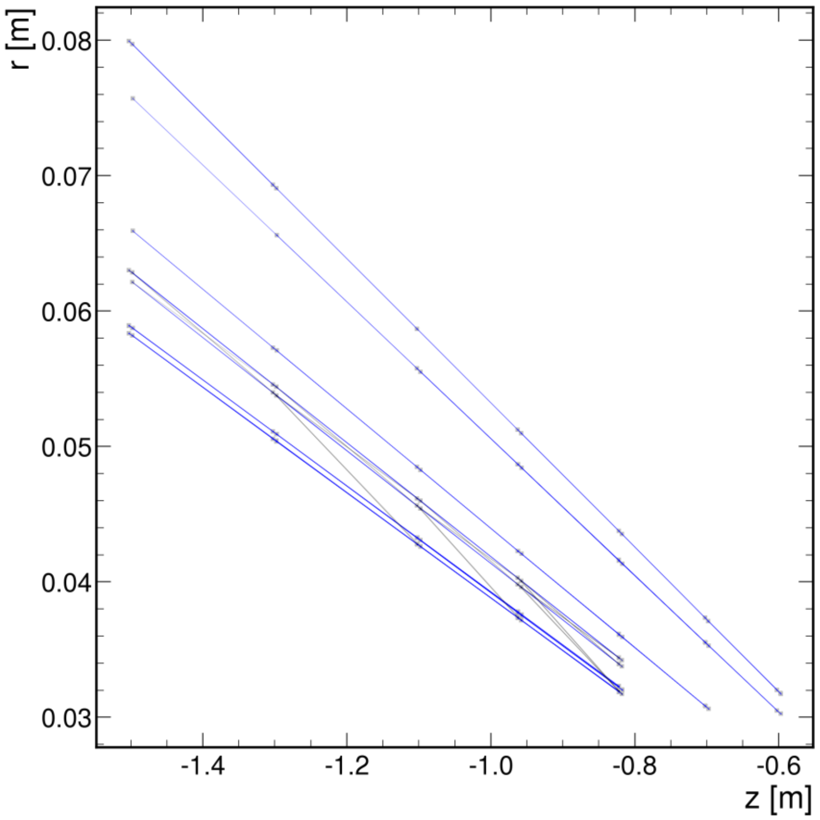
GRAPH CONSTRUCTION
PER-SECTOR BREAKDOWN

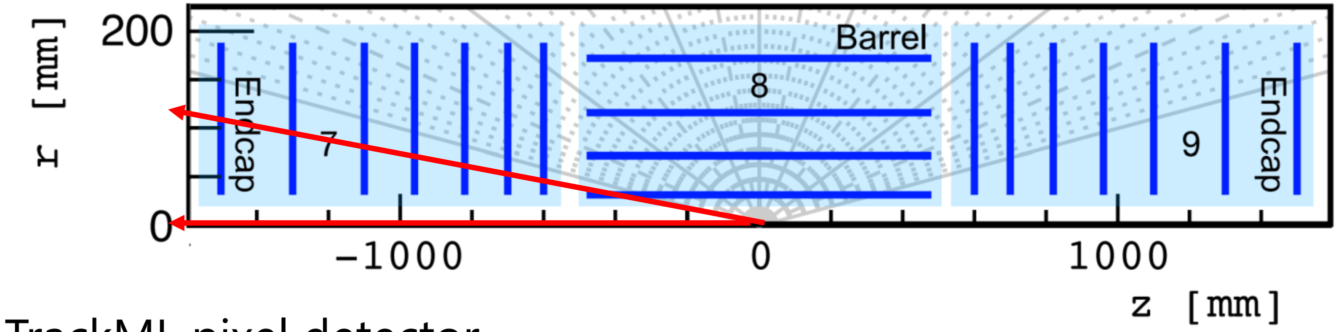
- Track p_T > 1.0 GeV
- phi_slope < 0.007
- z0 < 350 mm
- n_phi_sectors: 8
- n_eta_sectors: 8
- phi sector overlap: 0.08
- eta sector overlap: 0.125
- remove_noise: true

$\eta \in (-5, -3.75)$

n_{nodes} = 77+/-89
n_{edges} = 157+/-96

purity = 0.873+/-0.164
efficiency = 0.982+/-0.121

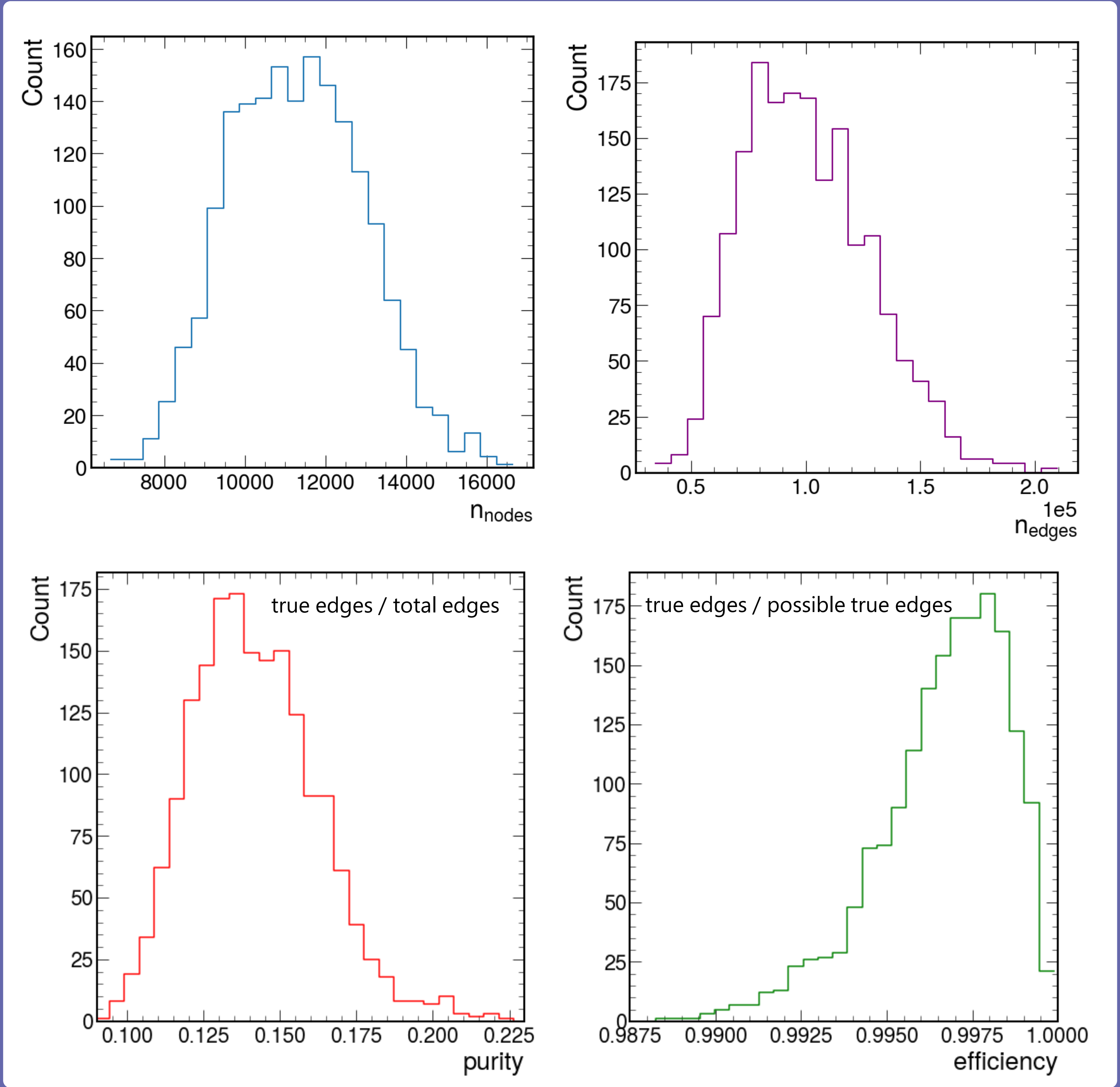




TrackML pixel detector

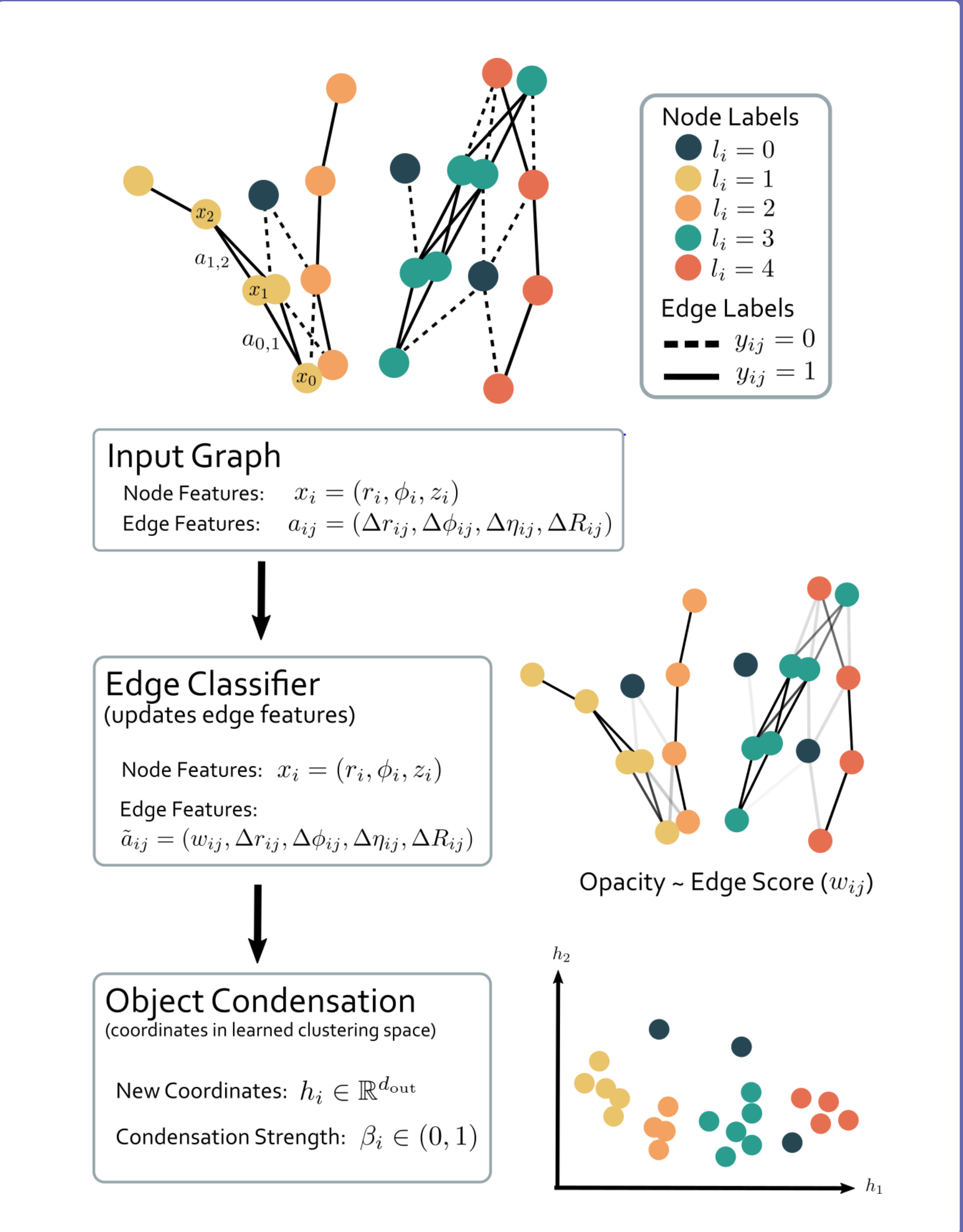
GRAPH CONSTRUCTION
PARAMETERS AND MEASUREMENTS

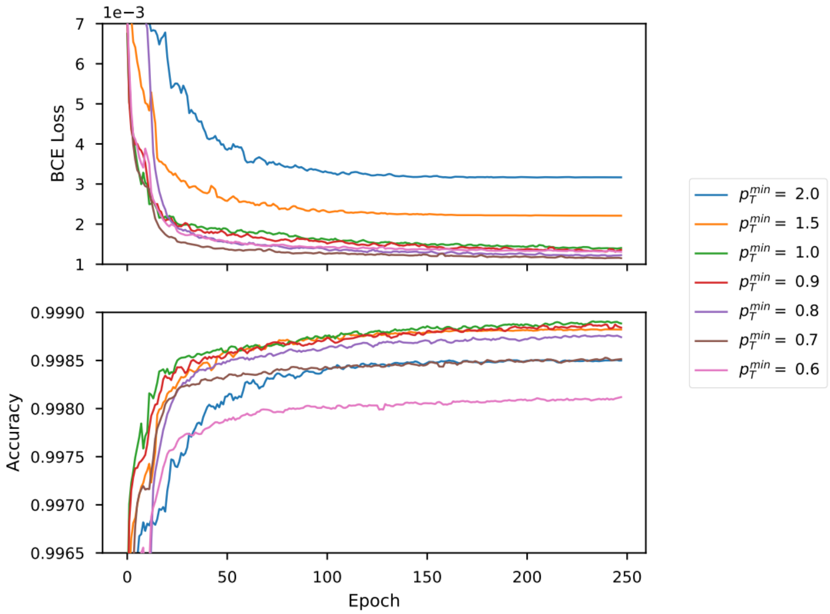
- Truth cuts
  - *track p_T > 1.0 GeV*
  - *remove_noise: true*
- Geometric edge selections:
  - phi_slope < 0.007
  - z0 < 350 mm
  - n_phi_sectors: 8
  - n_eta_sectors: 8
  - phi sector overlap: 0.08
  - eta sector overlap: 0.125



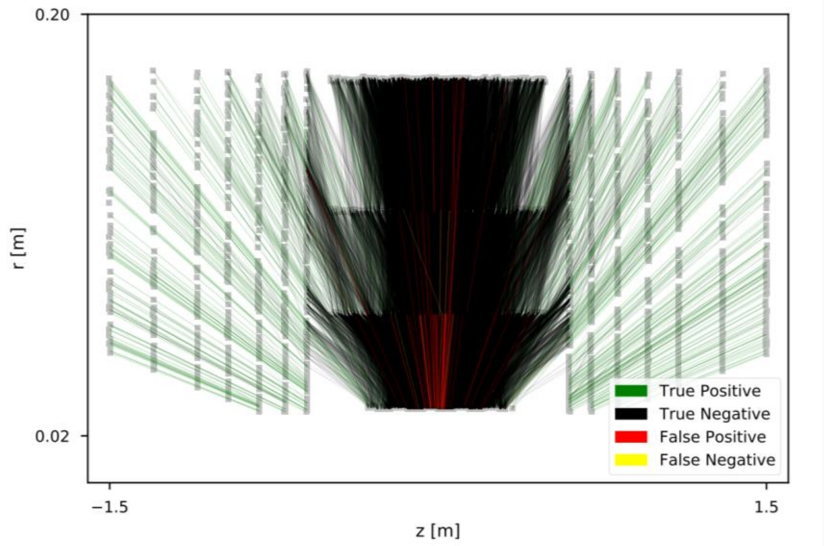
EDGE CLASSIFICATION / OBJECT CONDENSATION

STRATEGY OVERVIEW





loss/accuracy training curves on a range of graph sizes



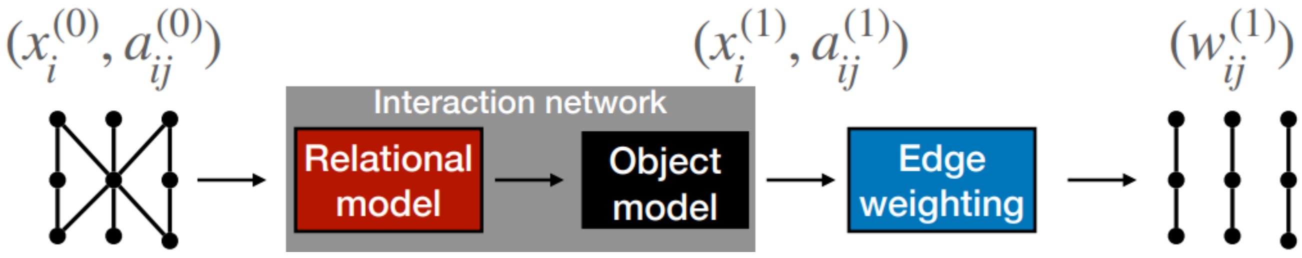
edge classification performance on a single graph

Interaction Networks:

[1612.00222] Interaction Networks for Learning about Objects, Relations and Physics (arxiv.org)

Even a single interaction network layer (depth-1 GNN) can achieve excellent edge classification accuracy

- (**Edge Block**) compute an interaction between two entities
- (**Node Block**) use the interaction to update the state of the receiving node



simple architecture explored in 2103.16701.pdf (arxiv.org)

EDGE CLASSIFICATION
BINARY CROSS ENTROPY

Edge Classifier
(updates edge features)

Node Features: $x_i = (r_i, \phi_i, z_i)$

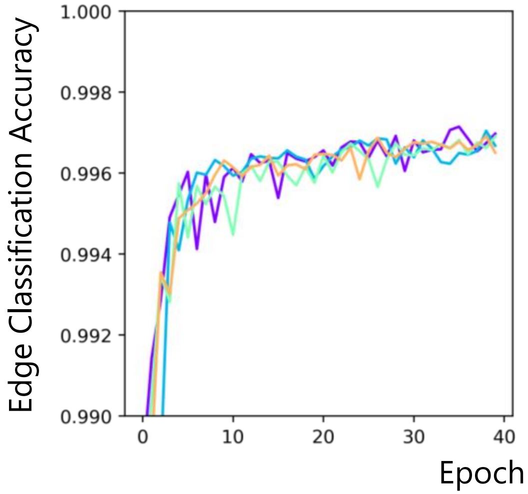
Edge Features:
$\tilde{a}_{ij} = (w_{ij}, \Delta r_{ij}, \Delta\phi_{ij}, \Delta\eta_{ij}, \Delta R_{ij})$

Opacity ~ Edge Score (w_{ij})

- Use two message passing IN layers
- BCE as usual to learn optimal edge weights

$$\mathcal{L}_w(y_j, w_j) = -\sum_{j=1}^{|\mathcal{E}|} \left(y_j \log w_j + (1-y_j)\log(1-w_j)\right)$$

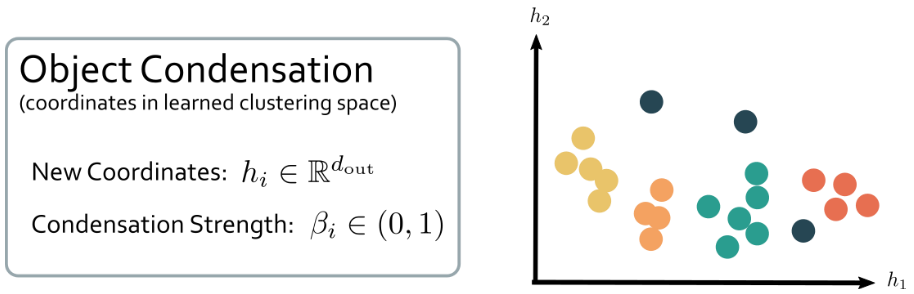
Edge weights converge to high accuracy at intermediate stages of the GNN



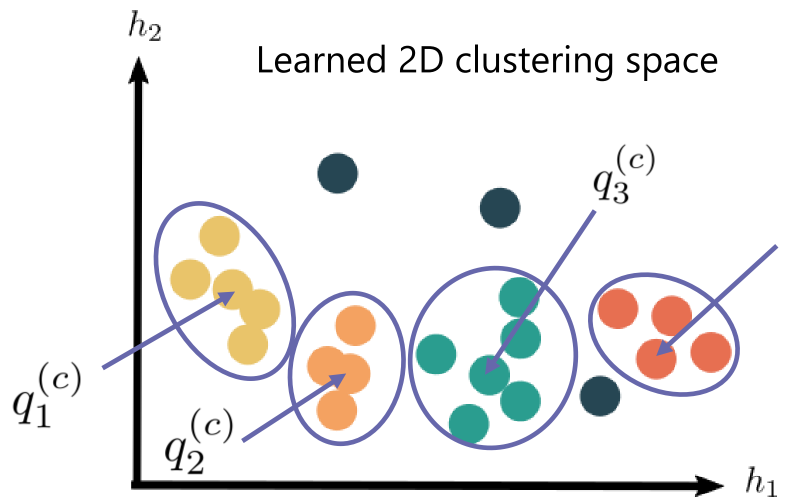
OBJECT CONDENSATION

POTENTIAL LOSS + BACKGROUND SUPPRESSION

(Kieseler 2020) 2002.03605.pdf (arxiv.org)



- **Predict** condensation "likelihood" ($\beta_i \in (0,1)$) and learned clustering coordinates ($h_i \in \mathbb{R}^{d_h}$)
- Train the network to condense hits around condensation points:
  - Define a "charge" per node: $q_i = \text{arctanh}^2 \beta_i + q_{\text{min}}$
  - Condensation points: maximum "charge" hit for each particle, i.e. $q_k^{(c)} = \max_i q_i \, \mathbb{1}_{\{l_i = k\}}$



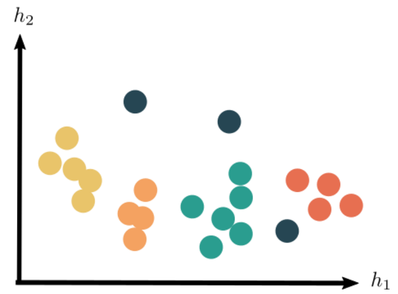
OBJECT CONDENSATION

POTENTIAL LOSS + BACKGROUND SUPPRESSION

Object Condensation
(coordinates in learned clustering space)

New Coordinates: $h_i \in \mathbb{R}^{d_{\text{out}}}$

Condensation Strength: $\beta_i \in (0,1)$

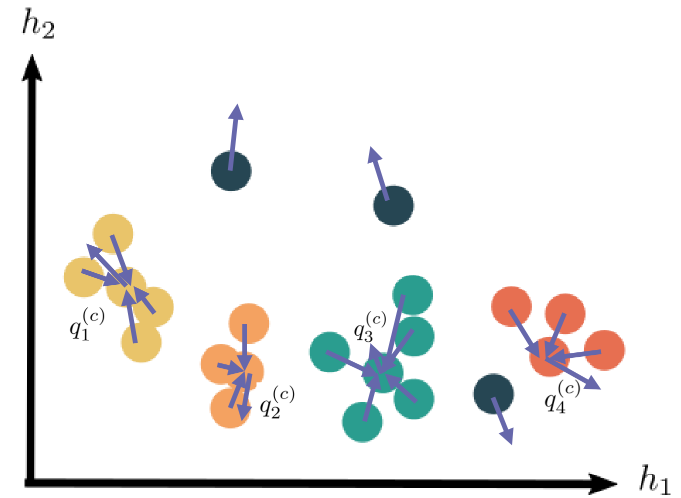


- Condensation points: $q_k^{(c)} = \max_i q_i \, \mathbb{1}_{\{l_i=k\}}$
- Optimize network to attract same-particle hits and repulse different-particle hits:

$$\mathcal{L}_V = \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|} q_i \sum_{k=1}^{K}\left(\mathbb{1}_{(l_i=k)} V_k^{\text{attract}}(h_i) + \left(1-\mathbb{1}_{(l_i=k)}\right) V_k^{\text{repulse}}(h_i)\right)$$

$$V_k^{\text{attract}}(h) = ||h - h_\alpha||_2^2 \, q_k^{(c)} \qquad V_k^{\text{repulse}}(h) = \max(0, 1 - ||h - h_\alpha||) q_k^{(c)}$$

Nodes are attracted to their particle's condensation point and repulsed from other particles'

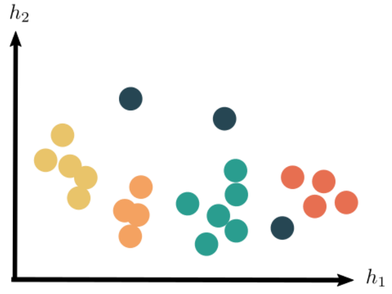


OBJECT CONDENSATION
TOTAL LOSS

Object Condensation
(coordinates in learned clustering space)

New Coordinates: $h_i \in \mathbb{R}^{d_{\text{out}}}$

Condensation Strength: $\beta_i \in (0,1)$



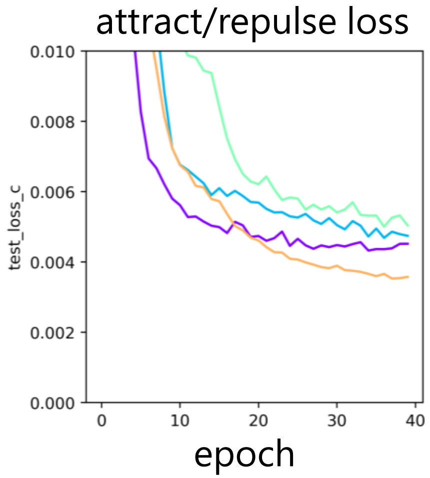
Attraction/Repulsion

$$\mathcal{L}_V = \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|} q_i \sum_{k=1}^{K}\left(\mathbb{1}_{(l_i=k)} V_k^{\text{attract}}(h_i) + (1-\mathbb{1}_{(l_i=k)}) V_k^{\text{repulse}}(h_i)\right)$$

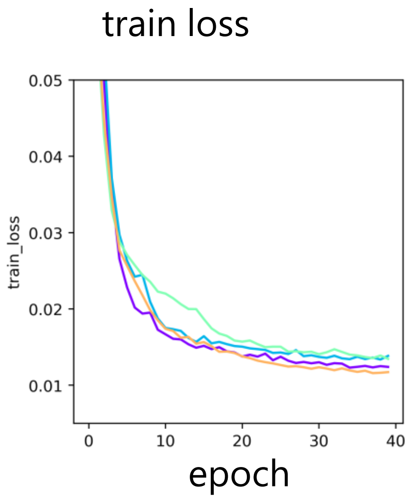
(quadratic) (hinge)

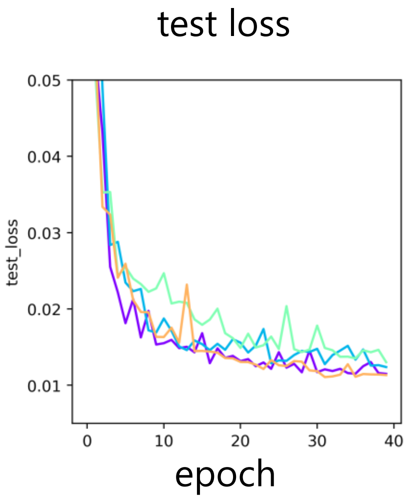
Background Suppression → scale by 2.5×10^{-3}

$$\mathcal{L}_\beta = \frac{1}{K}\sum_k (1-\beta_k^{(c)}) + s_B \frac{\sum_{i=1}^{|\mathcal{V}|} \beta_i \mathbb{1}_{\{l_i=0\}}}{\sum_{i=1}^{|\mathcal{V}|} \mathbb{1}_{\{l_i=0\}}}$$



Architecture: 3 IN layers to re-embed the graph, subsequent MLPs to predict $\boldsymbol{\beta}$ and $\boldsymbol{h}$

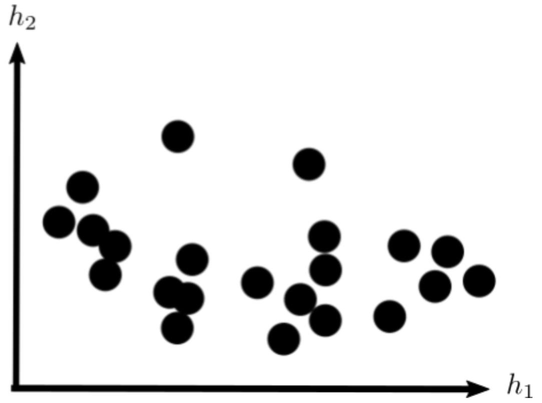




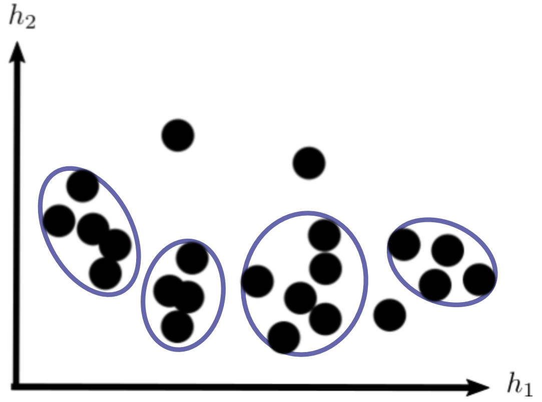
POSTPROCESSING

DBSCAN → TRACK FINDING

- GNN output is the set of hit coordinates in the learned (h_1, h_2) space:



- Need to run DBSCAN to generate cluster labels (clustering parameters are optimized per on graph sector):

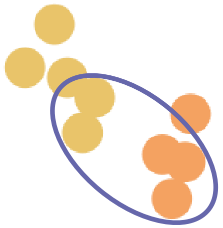


TRACKING EFFICIENCIES
VARIOUS DEFINITIONS

- Perfect Match: fraction of clusters containing every hit associated to a particle and no others

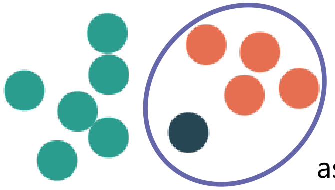
assigned to green particle

- Double Majority: fraction of clusters comprised of >50% of same-particle hits and containing >50% of that particle's hits

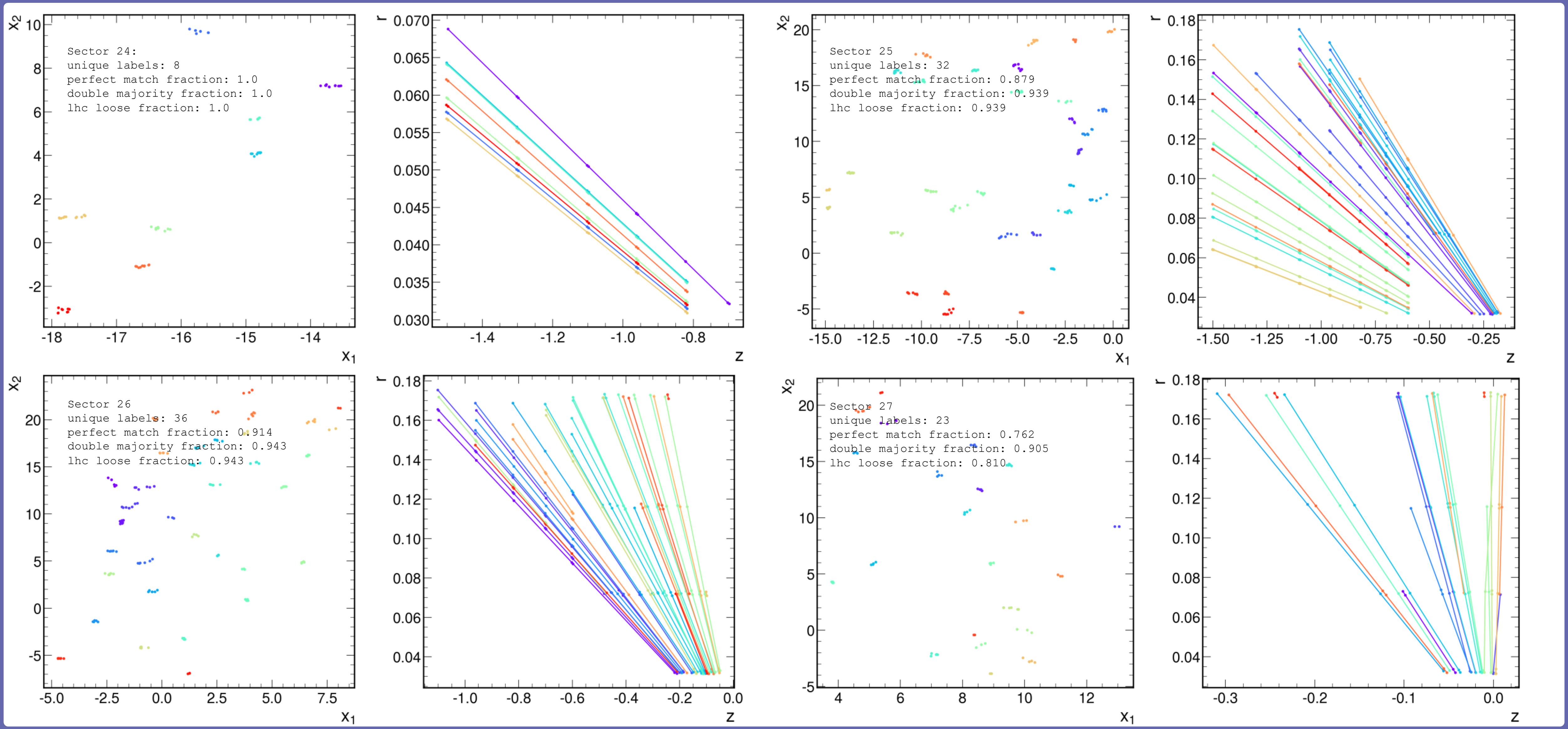


assigned to orange particle

- LHC Loose Match: fraction of clusters comprised of >75% same-particle hits



assigned to red particle

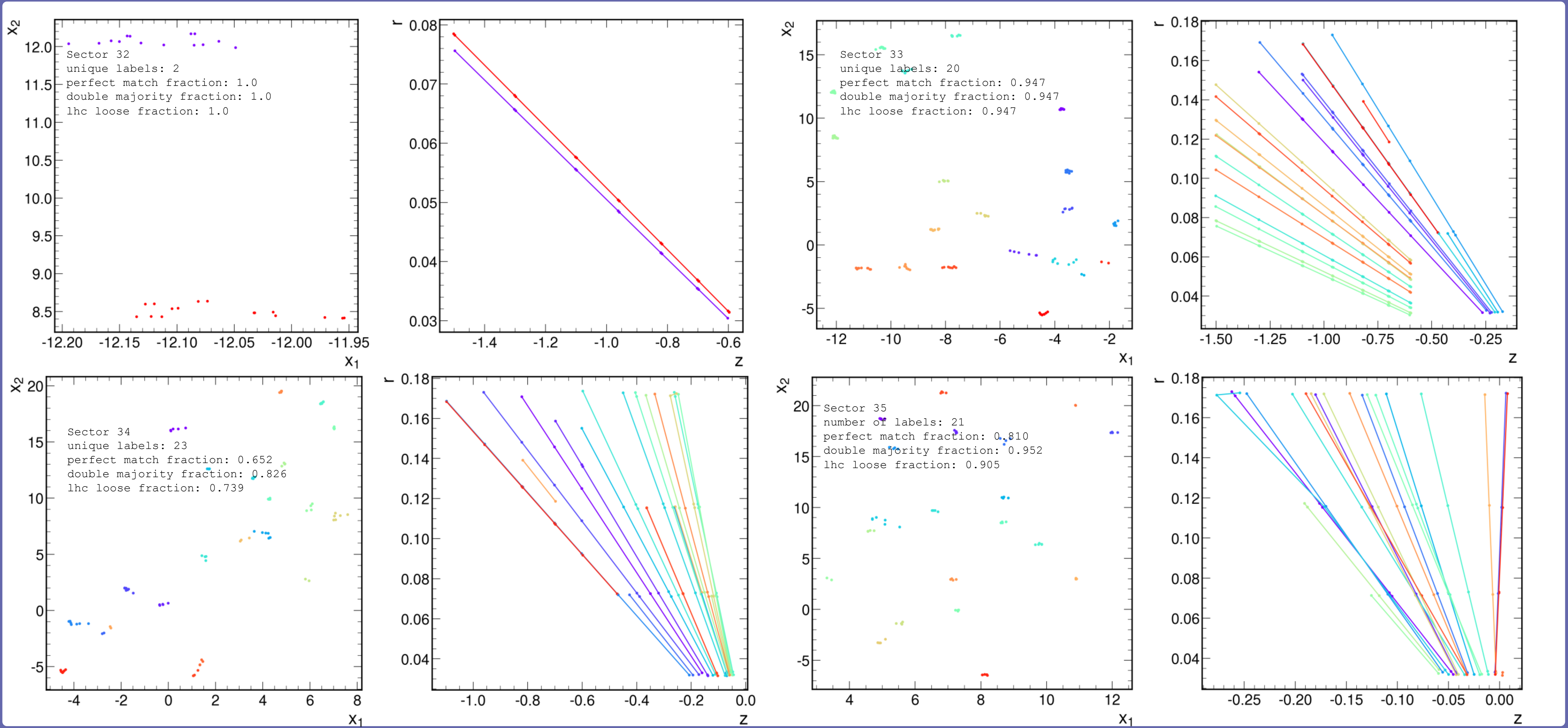


EXAMPLE: EVENT #1127

MODEL 10

```
summary of full event
perfect match fraction: 0.862
double majority fraction: 0.945
lhc loose fraction: 0.906
```

NOTE: Cluster colors are DBSCAN labels, not truth labels!



EXAMPLE: EVENT #1823

MODEL 10

summary of full event
perfect match fraction: 0.860
double majority fraction: 0.939
lhc loose fraction: 0.909

NOTE: Cluster colors are DBSCAN labels, not truth labels!

TRACKING EFFICIENCIES

AVERAGED ACROSS ~10^4 GRAPHS

- Per-graph summary
  - Perfect Match Fraction: 0.827
  - Double Majority Fraction: 0.932
  - LHC Loose Fraction: 0.890

- Per eta-range:
  - Performance decreases with graph construction purity (decreasing eta)

$\|\eta\|$	LHC Loose Match	Double Majority	Perfect Match
$(0, 1.25)$	0.851 +/- 0.070	0.905 +/- 0.058	0.779 +/- 0.099
$(1.25, 2.5)$	0.895 +/- 0.062	0.934 +/- 0.051	0.842 +/- 0.087
$(2.5, 3.75)$	0.939 +/- 0.053	0.966 +/- 0.044	0.884 +/- 0.079
$(3.75, 5)$	0.986 +/- 0.083	0.997 +/- 0.075	0.969 +/- 0.106

Graph construction purity/efficiency isn't consistent among the eta ranges!

CONCLUSIONS
AND FUTURE STEPS

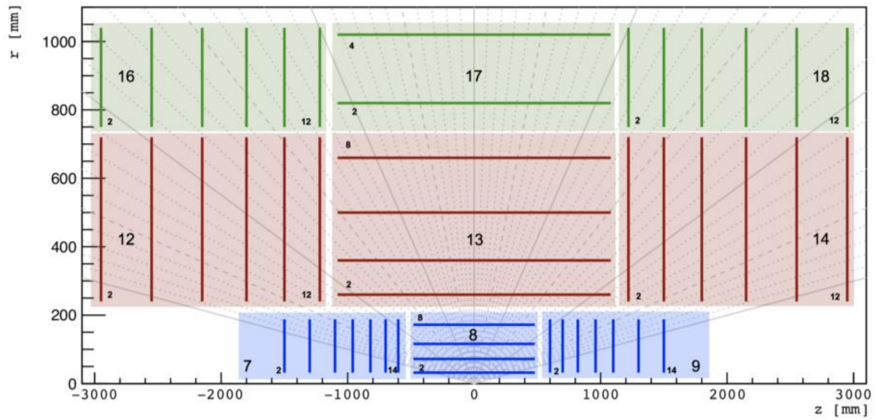
- GNN-based tracking typically involves
  1) graph construction
  2) GNN inference (edge classification, object condensation)
  3) postprocessing (track finding)
- Example GNN pipeline based on edge classification and object condensation
  - Object condensation also accommodates track property predictions! → next step
- Future work:
  - Improve graph construction in central barrel region
  - Relax the truth cuts (re-impose noise, zero the p_T cut)
  - Incorporate track parameter predictions
  - Explore dynamic graph construction techniques like GravNet (no edge classification)
  - Full hyperparameter scan over network size/structure

BACKUP

INPUT DATA
TRACKML DATASET

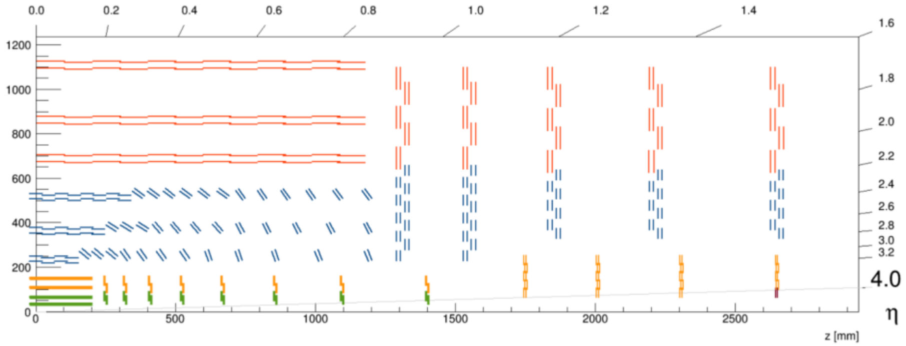
TrackML Dataset

Generic Tracker
is inspired by the
the geometry of the
Phase 2 CMS/ATLAS
trackers:



- Simulated tracker events
  - ttbar events with 200 pileup
  - Includes tracker hits with truth labels indicating which particle generated them
- Public dataset:
  [TrackML Particle Tracking Challenge | Kaggle]
  [CodaLab - Competition]

CMS Phase-2
Tracker geometry

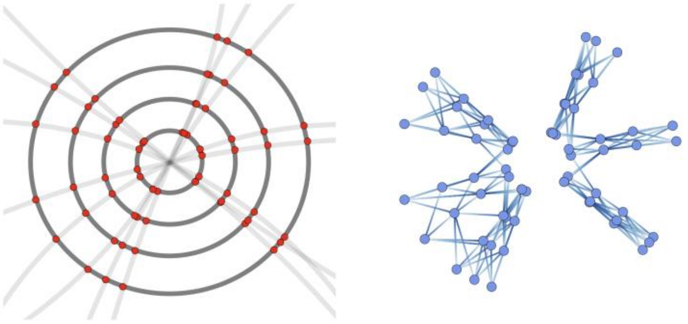


GNNS AT THE LHC
COMMON APPLICATIONS

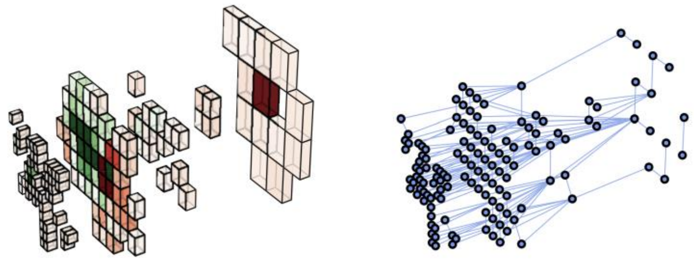
Why GNNs?

- Graphs represent *unordered*, *relational* information → natural for CMS data
  - CMS data is sparse and variably sized
  - CMS data is heterogeneous; recorded from multiple subdetectors, different types of particles, etc.
- Excellent performance
  - Relational inductive bias
  - Message passing leverages low-level detector info in addition to global (or otherwise human-devised) info

Low-level, hit-based tasks

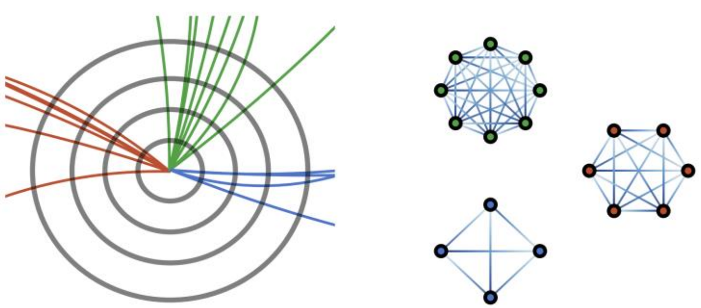


Track Reconstruction

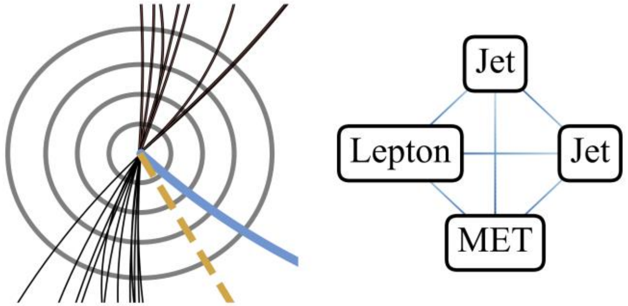


Calorimeter Segmentation

High-level, particle-based tasks



Jet Identification

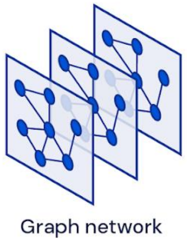


Event Classification

[2007.13681] Graph Neural Networks in Particle Physics (arxiv.org)

GRAPH NEURAL NETWORKS
NEURAL MESSAGE PASSING

Message Passing (MPNN) Layers:
Framework for equivariant graph updates



GNN comprised of multiple message passing layers

At each layer *k*, compute **messages** in each node's neighborhood:

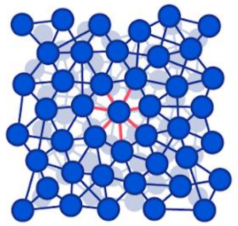
MLP

$$\boldsymbol{m}_{uv}^{(k)} = \psi^{(k)}\left(\boldsymbol{h}_{u}^{(k-1)}, \boldsymbol{h}_{v}^{(k-1)}, \boldsymbol{e}_{uv}^{(k-1)}\right)$$

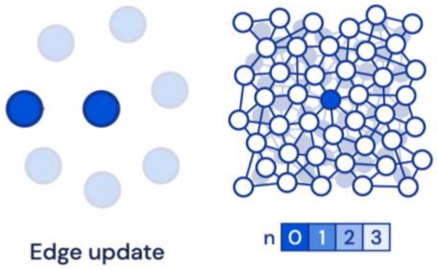
Central node's previous embedding

Neighbor node's previous embedding

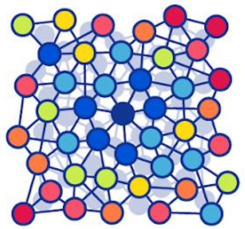
Previous edge features



Input graph



Neural Message Passing



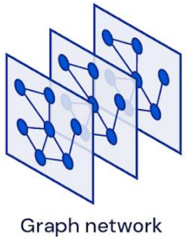
New graph embedding

Figure Source:
https://deepmind.com/blog/article/Towards-understanding-glasses-with-graph-neural-networks

GRAPH NEURAL NETWORKS
NEURAL MESSAGE PASSING

Message Passing (MPNN) Layers:

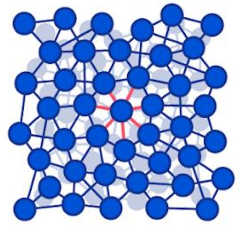
Framework for equivariant graph updates



GNN comprised of multiple message passing layers

At each layer k, compute messages in each node's neighborhood:

$$\boldsymbol{m}_{uv}^{(k)} = \psi^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}, \boldsymbol{e}_{uv}^{(k-1)}\right)$$



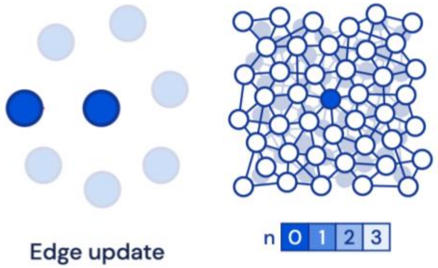
Input graph

Aggregate messages in a permutation-invariant way:

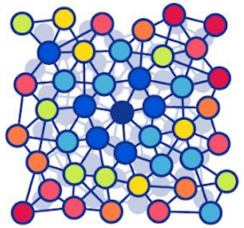
Messages passed only from u's direct neighbors

$$\boldsymbol{a}_u^{(k)} = \bigoplus_{v \in N(u)} \boldsymbol{m}_{uv}^{(k)}$$

Any permutation invariant operation (e.g. sum, mean, max)



Neural Message Passing



New graph embedding

Figure Source: https://deepmind.com/blog/article/Towards-understanding-glasses-with-graph-neural-networks

GRAPH NEURAL NETWORKS
NEURAL MESSAGE PASSING

Message Passing (MPNN) Layers:
Framework for equivariant graph updates

At each layer *k*, compute messages in each node's neighborhood:

$$\boldsymbol{m}_{uv}^{(k)} = \psi^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}, \boldsymbol{e}_{uv}^{(k-1)}\right)$$

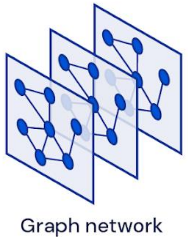
Aggregate messages in a permutation-invariant way:

$$\boldsymbol{a}_u^{(k)} = \bigoplus_{v \in N(u)} \boldsymbol{m}_{uv}^{(k)}$$

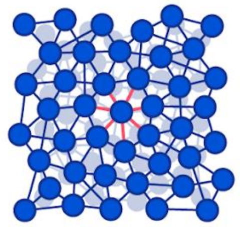
Update the node's state based on the messages it received:

$$\boldsymbol{h}_u^{(k)} = \phi^{(k)}(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{a}_u^{(k)})$$

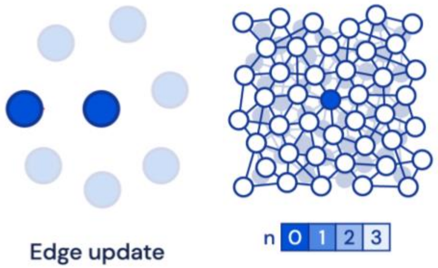
MLP



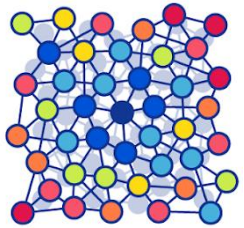
GNN comprised of multiple message passing layers



Input graph



Neural Message Passing



New graph embedding

Figure Source:
https://deepmind.com/blog/article/Towards-understanding-glasses-with-graph-neural-networks

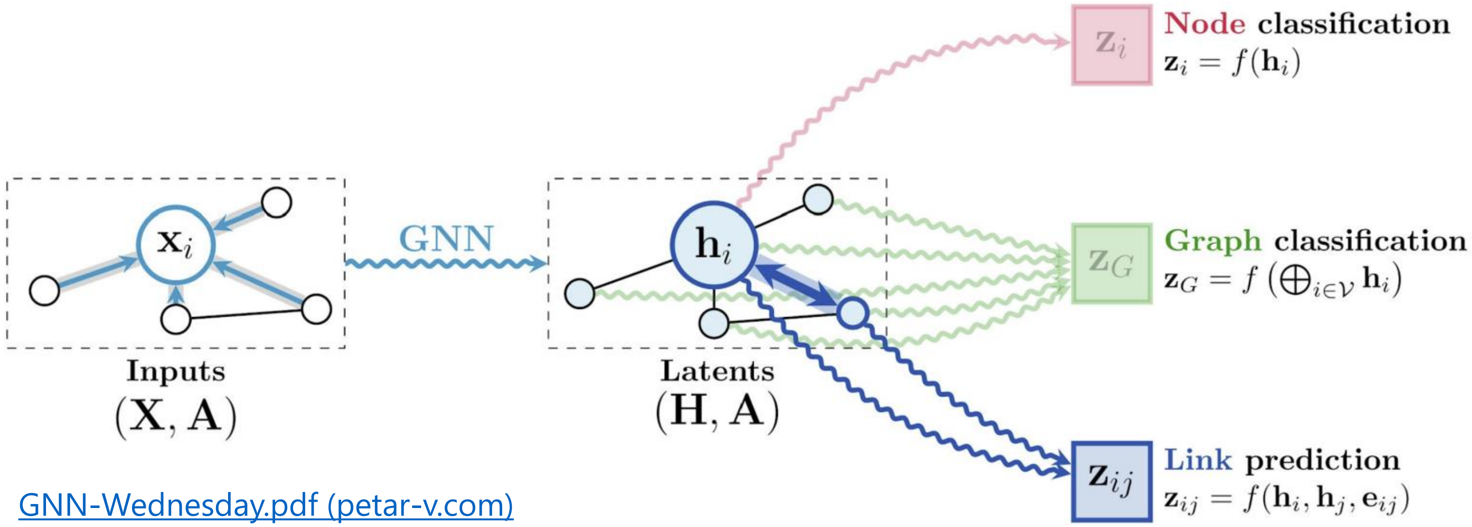
GRAPH NEURAL NETWORKS
REPEATED MESSAGE PASSING

Generic MPNN Layers:

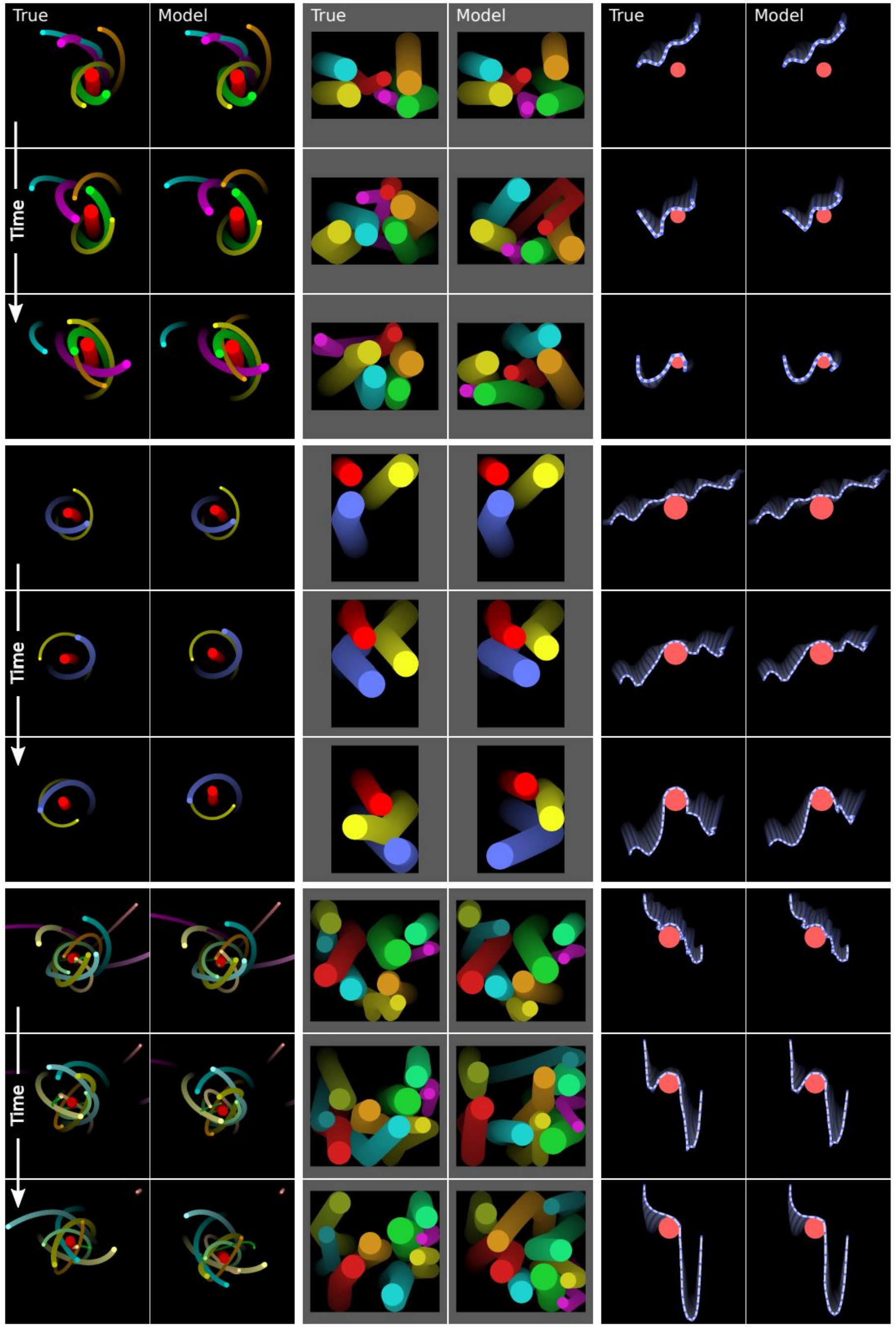
$$\boldsymbol{h}_u^{(k)} = \phi^{(k)}\left[\boldsymbol{h}_u^{(k-1)}, \bigoplus_{v\in N(u)} \psi^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}, \boldsymbol{e}_{uv}^{(k-1)}\right)\right]$$

Node Updates: collecting info from each node's *k*-hop neighborhood at the k^{th} layer

Outputs: node-level, edge-level, or graph-level predictions



GNN-Wednesday.pdf (petar-v.com)



Interaction Networks:

[1612.00222] Interaction Networks for Learning about Objects, Relations and Physics (arxiv.org)

Physics-motivated MPNNs suitable for graphs with pre-constructed edges (originally applied to “next timestep” physics simulations)

- (**Edge Block**) compute an interaction between two entities:

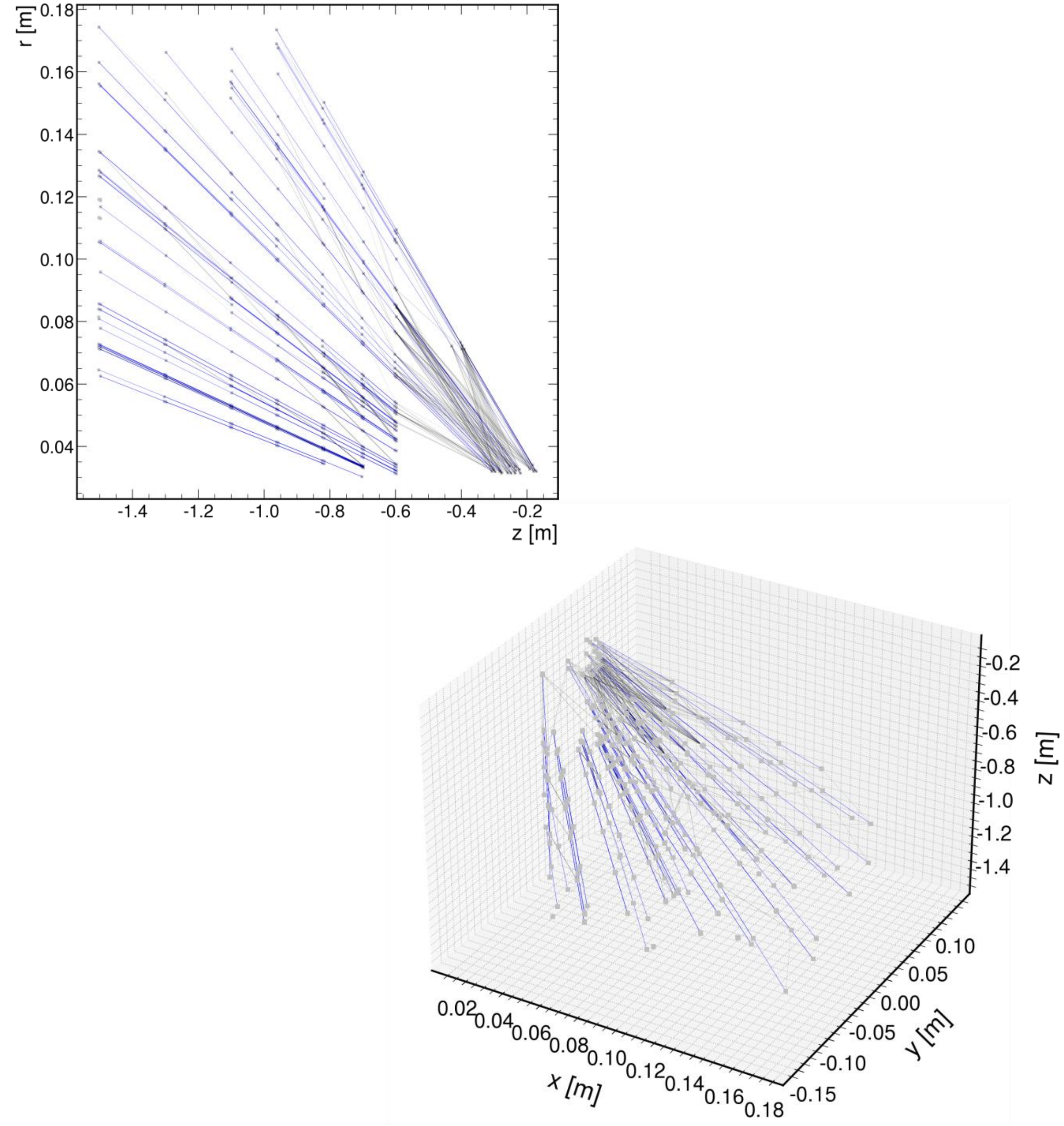
$$e_{uv}^{(k)} = MLP_{\psi}^{(k)}\left(\left[h_u^{(k-1)}, h_v^{(k-1)}, e_{u,v}^{(k-1)}\right]\right)$$

- (**Node Block**) use the interaction to update the state of the receiving node:

$$h_u^{(k)} = MLP_{\phi}^{(k)}\left(h_u^{(k-1)}, \sum_{v\in N(u)} e_{u,v}^{(k)}\right)$$

a common form for many GNN tracking architectures

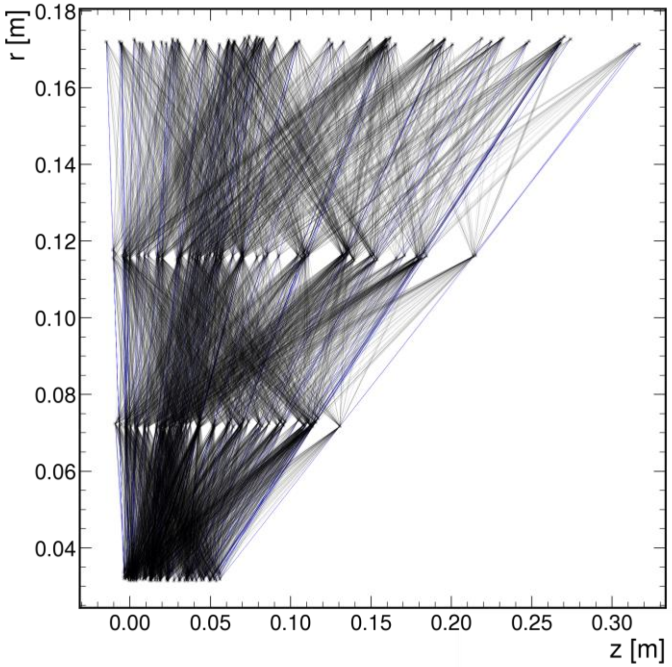
1810.06111.pdf (arxiv.org)

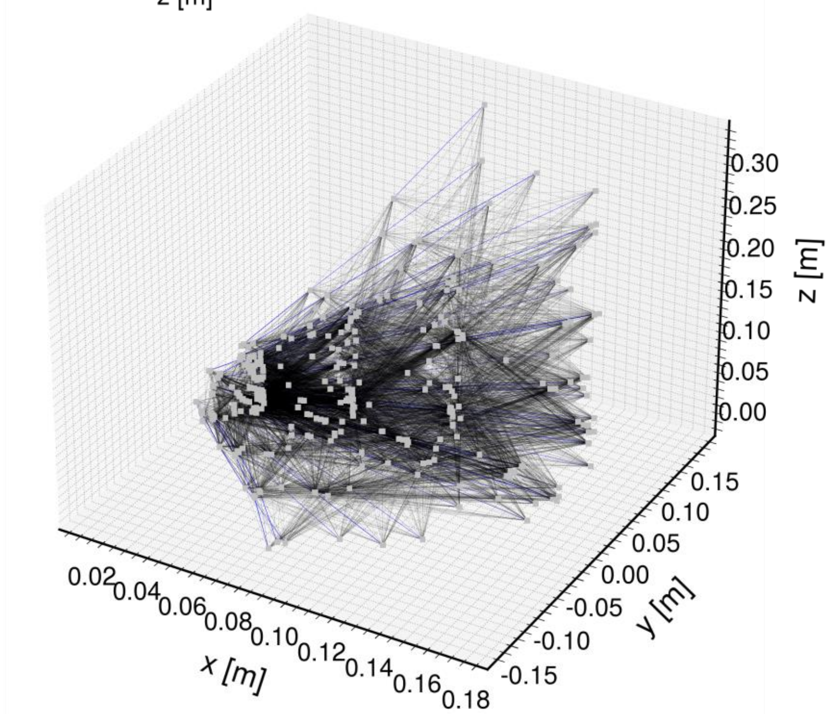


GRAPH CONSTRUCTION

EACH EVENT BROKEN (8X8) PHI-ETA SECTORS

- Truth cuts
  - *track p_T > 1.0 GeV*
  - *remove_noise: true*
- Geometric edge selections:
  - phi_slope < 0.007
  - z0 < 350 mm
  - n_phi_sectors: 8
  - n_eta_sectors: 8
  - phi sector overlap: 0.08
  - eta sector overlap: 0.125





GRAPH CONSTRUCTION

EACH EVENT BROKEN (8X8) PHI-ETA SECTORS

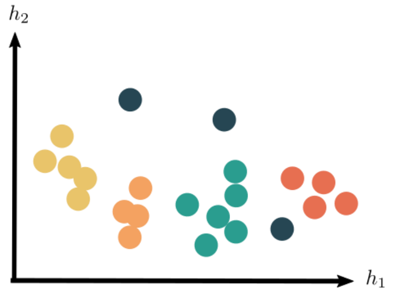
- Truth cuts
  - *track p_T > 1.0 GeV*
  - *remove_noise: true*
- Geometric edge selections:
  - phi_slope < 0.007
  - z0 < 350 mm
  - n_phi_sectors: 8
  - n_eta_sectors: 8
  - phi sector overlap: 0.08
  - eta sector overlap: 0.125

OBJECT CONDENSATION
POTENTIAL LOSS + BACKGROUND SUPPRESSION

Object Condensation
(coordinates in learned clustering space)

New Coordinates: $h_i \in \mathbb{R}^{d_{\mathrm{out}}}$

Condensation Strength: $\beta_i \in (0, 1)$



- **Predict** condensation “likelihood” ($\beta_i \in (0,1)$) and learned clustering coordinates ($h_i \in \mathbb{R}^{d_h}$)
- Define a charge per node: $q_i = \operatorname{arctanh}^2 \beta_i + q_{\min}$
  - Condensation points: $q_k^{(c)} = \max_i q_i \, \mathbb{1}_{\{l_i = k\}}$
- Optimize two terms:
  - Potential Loss:

$$\mathcal{L}_V = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} q_i \sum_{k=1}^{K} \left(\mathbb{1}_{(l_i=k)} V_k^{\mathrm{attract}}(h_i) + \left(1 - \mathbb{1}_{(l_i=k)}\right) V_k^{\mathrm{repulse}}(h_i) \right)$$

$$V_k^{\mathrm{attract}}(h) = ||h - h_\alpha||_2^2 \, q_k^{(c)} \qquad V_k^{\mathrm{repulse}}(h) = \max(0, 1 - ||h - h_\alpha||) q_k^{(c)}$$

  - Background Loss:

$$\mathcal{L}_\beta = \frac{1}{K} \sum_k (1 - \beta_k^{(c)}) + s_B \frac{\sum_{i=1}^{|\mathcal{V}|} \beta_i \mathbb{1}_{\{l_i=0\}}}{\sum_{i=1}^{|\mathcal{V}|} \mathbb{1}_{\{l_i=0\}}}$$

Sum over noise hits (here, any particle hitting only one detector layer – O(20-50) hits)

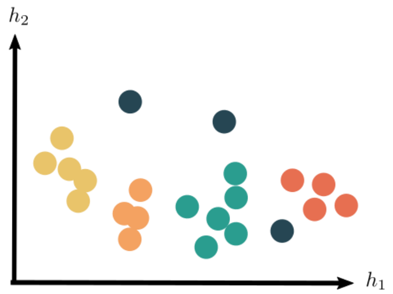
OBJECT CONDENSATION

POTENTIAL LOSS + BACKGROUND SUPPRESSION

Object Condensation
(coordinates in learned clustering space)

New Coordinates: $h_i \in \mathbb{R}^{d_{\text{out}}}$

Condensation Strength: $\beta_i \in (0, 1)$



- Optimize two terms:

$$\mathcal{L}_V = \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|} q_i \sum_{k=1}^{K}\left(\mathbb{1}_{(l_i=k)} V_k^{\text{attract}}(h_i) + \left(1-\mathbb{1}_{(l_i=k)}\right) V_k^{\text{repulse}}(h_i)\right)$$

$$\mathcal{L}_\beta = \frac{1}{K}\sum_k (1-\beta_k^{(c)}) + s_B \frac{\sum_{i=1}^{|\mathcal{V}|}\beta_i \mathbb{1}_{\{l_i=0\}}}{\sum_{i=1}^{|\mathcal{V}|}\mathbb{1}_{\{l_i=0\}}}$$

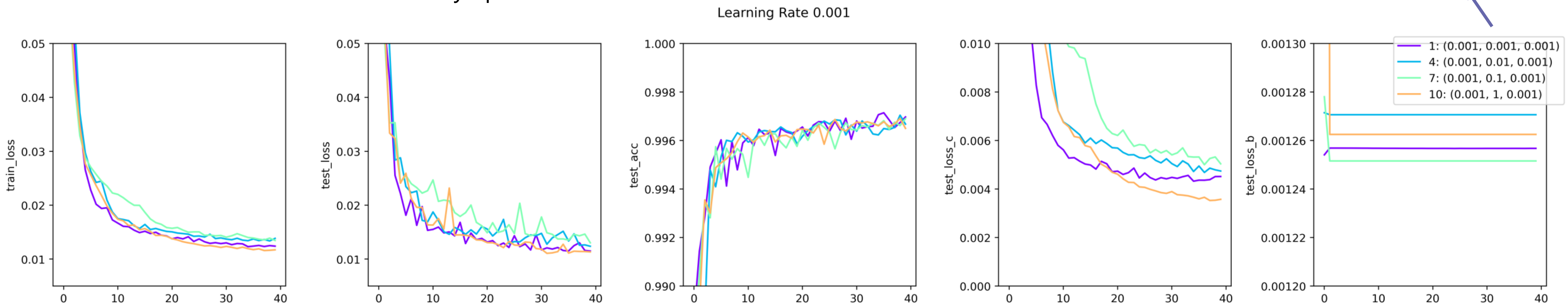
- Architecture:

```
# combine edge weights with original edge features
edge_attr_w = torch.cat([edge_weights,
                         initial_edge_attr], dim=1)
xc1, edge_attr_c1 = self.in_c1(x, edge_index, edge_attr_w)
xc2, edge_attr_c2 = self.in_c2(xc1, edge_index, edge_attr_c1)
xc3, edge_attr_c3 = self.in_c3(xc2, edge_index, edge_attr_c2)
all_xc = torch.cat([x,xc1,xc2,xc3], dim=1)
beta = torch.sigmoid(self.B(all_xc))
xc = self.X(all_xc)
return edge_weights, xc, beta
```

full scan over
$lr \in \{0.0001, 0.001, 0.01\}$
$q_{min} \in \{0.001, 0.01, 0.1, 1\}$
$s_b \in \{0.001, 0.01, 0.1, 1\}$

plot only models achieving test_loss < 0.02 by epoch 40

(q_min, sb, lr)



Train Loss

Test Loss

Edge Classification Accuracy

Attract/Repulse Loss

Background Loss

Smooth convergence, no sign of overtraining

Converges to ~99.7%
(consistent with arXiv:2103.16701)

Learned clustering coordinates exhibit attract/repulse behavior

Background assignment falls into minimum for best solutions

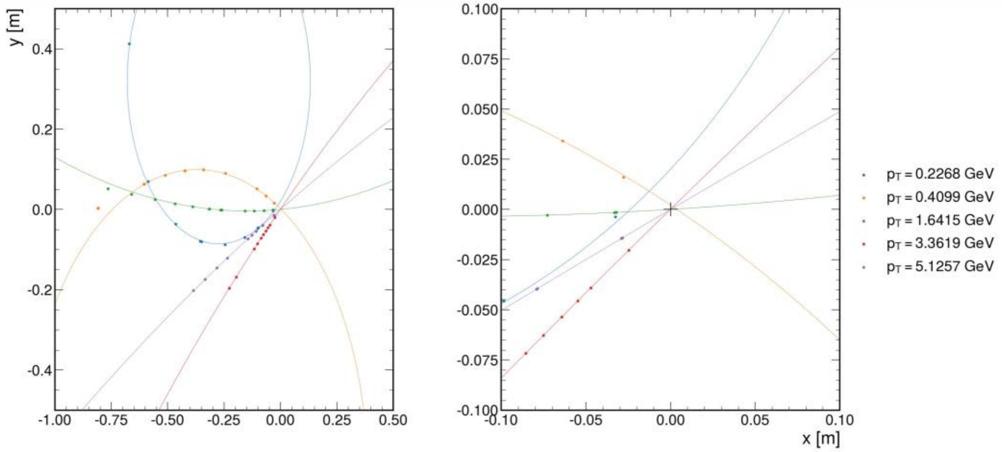
HYPERPARAMETER SCANS

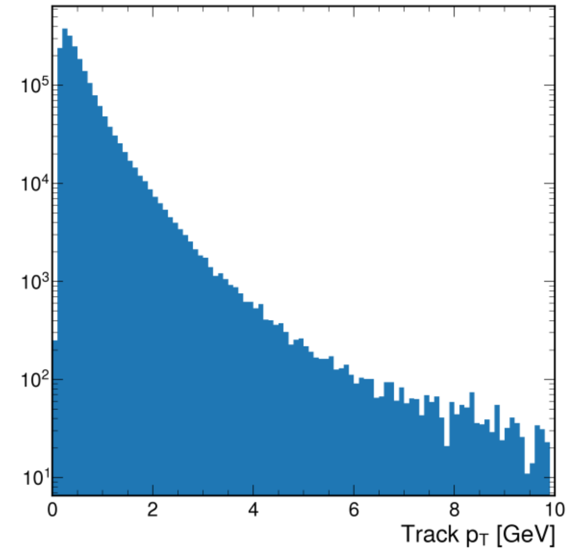
Circle fits in (x,y) space

$$(x - x_c)^2 + (y - y_c)^2 = R^2$$

$$d_0 = \left[R - \sqrt{x_c^2 + y_c^2}\right]$$

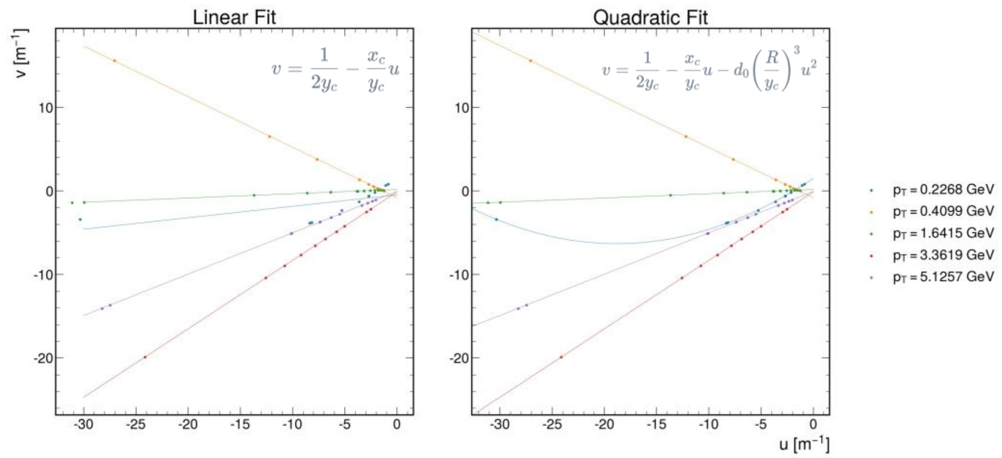
$$p_T = \kappa |q| BR$$

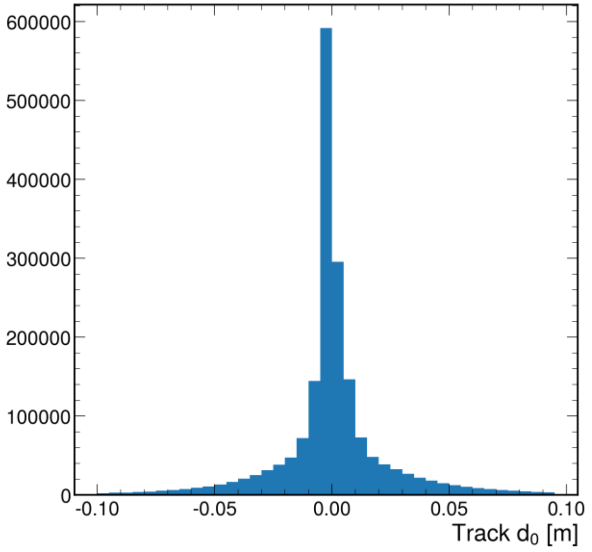




Linear/quadratic fits in conformal (u,v) space

$$u = \frac{x}{x^2 + y^2}, \quad v = \frac{y}{x^2 + y^2}$$





INPUT DATA

TRACKML TRACK FITS