GNN Tracking

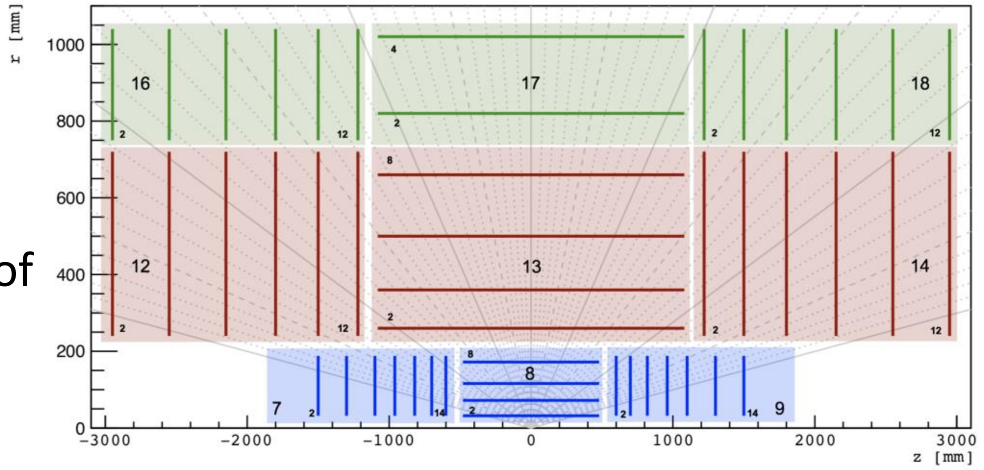
Graph Construction and Network Architectures

IRIS-HEP Topical Meeting
10/21/20

Markus Atkinson
Gage DeZoort, Javier Duarte, Lindsey Gray, Aneesh Heintz, Isobel Ojalvo, Mark Neubauer, Vesal Razavimaleki, and Savannah Thais

Basic Procedure / Outline

- Pre-Processing
  - Construct the graph using hit coordinates as nodes
    - Truth level Pt cuts applied at this stage to control size of graphs, remove noise hits
  - Edges are selected based on geometric cuts
  - Data Augmentation techniques applied
- Process with GNN to get all edge probabilities
  - Edge Classifiers
  - Interaction Network
- Post-Processing
  - Tracking type algorithms
    - Calculate Track Parameters directly
    - Use for Tracklet seeding



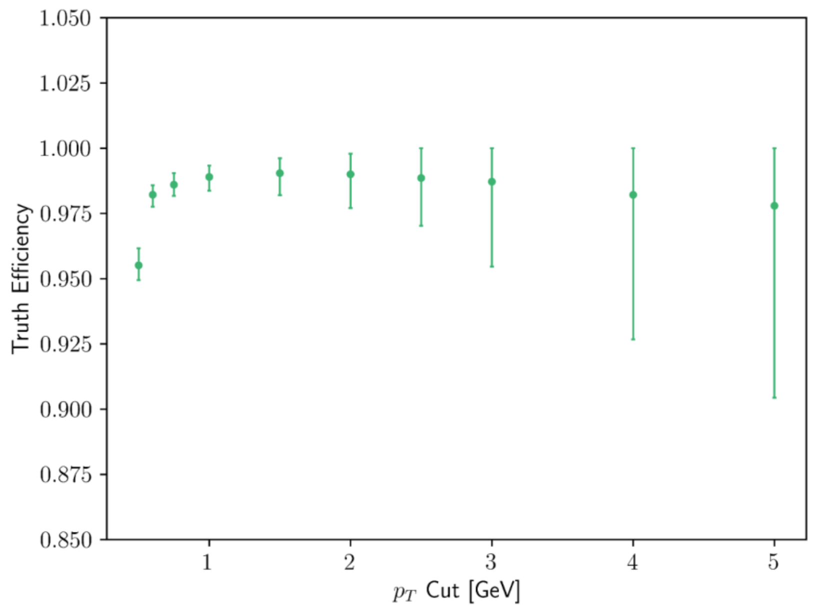
- Work shown here uses TrackML dataset
  - Open, experiment agnostic
  - Has 'score' functionality to compare models
- Many places to improve/innovate
  - Other ways of augmenting the data?
  - Exploration of many GNN architectures with varying parameters
  - Tracking work is very preliminary

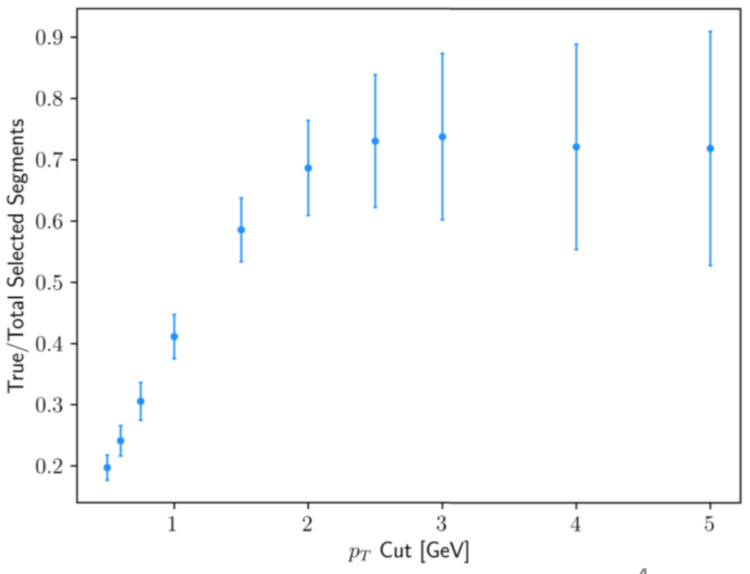
Graph Construction

Graph Construction

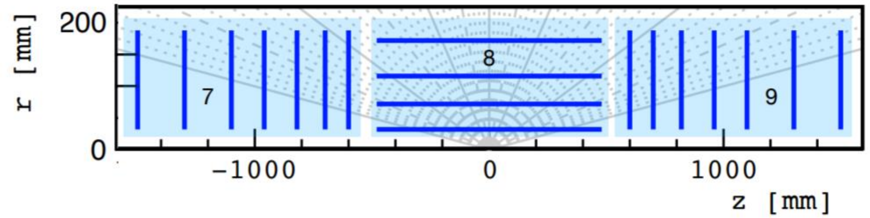
- Contribute to ExaTrkX code base (numpy)
  - Forms edges between nodes of adjacent layers
  - Added Endcaps
  - Added some data augmentation abilities
  - Fully converted into a pytorch-geometric [datset]

- Useful quantities
  - True edge efficiency
    true edges in graph / all possible true edges
  - True edge purity
    true edges in graph / all edges in graph





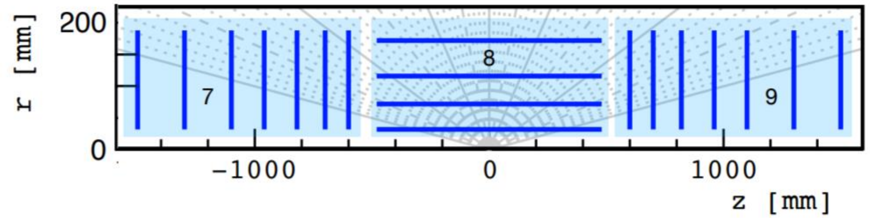
Graph Construction - Node Cuts



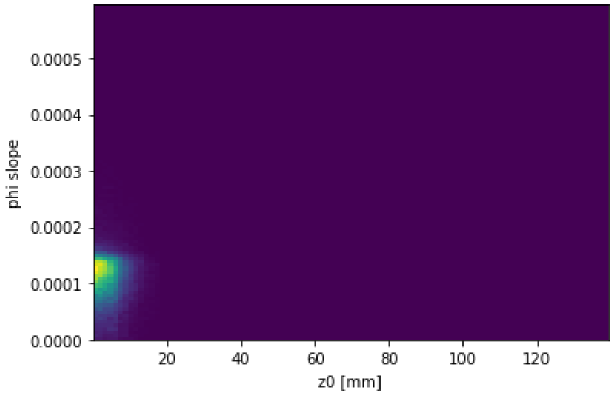
- Layers Selected
  - Studies focused on the Inner Detector (4 barrel layers, 7 endcap layers per side)
- Eta based cuts : [-5, 5]
- Truth based cuts
  - Remove noise hits – (To do: add the ability to keep these)
  - p_T based cuts: p_T > 2.0 GeV
  - Remove duplicate hits within same layer from same particle
    - There is an option to disable this for a mode that allows edges within same layer

Graph Construction - Edge Cuts

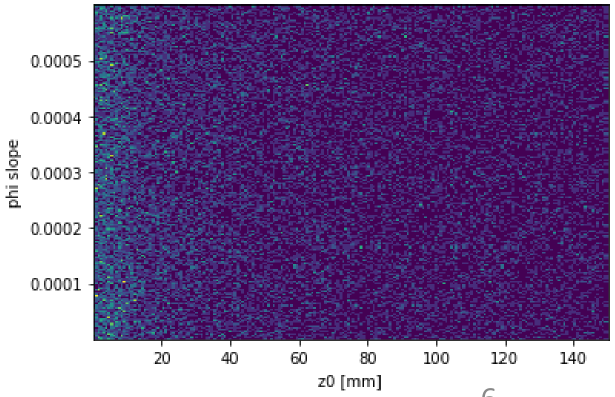
- Two Modes
  - Layer Pairs: Allow adjacent layers to connect
  - Layer Pairs plus: Also allow edges within same layer
- Geometric cuts
  - $\Delta\phi/\Delta r < .000262$
  - $z_0 < 15$ cm
  - Remove edges that intersect intermediate barrel layers when connecting barrel to endcap



True Edges

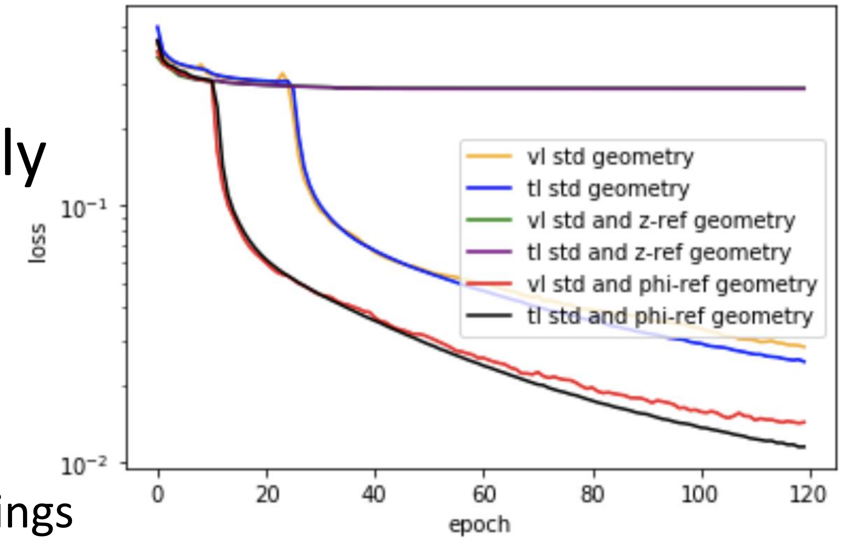


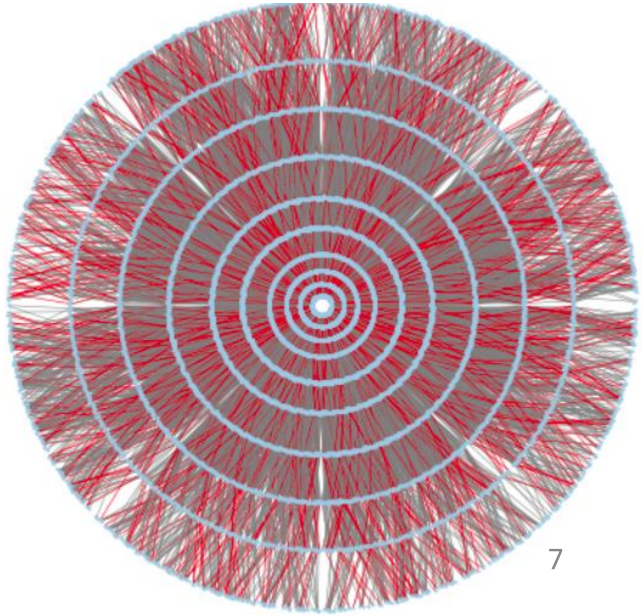
False Edges



Graph Construction – Data Augmentation

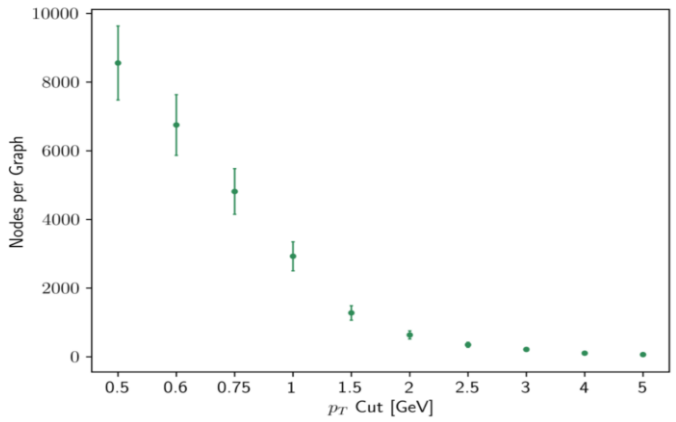
- Augmenting the graph in various ways can drastically affect the training
- Tricks that help in various ways
  - Segmenting the graph to reduce size (subsections of the detector)
    - Special care needed in post processing to handle stitching things back together
  - Making a copy of the graph reflected through phi
    - This flips the handedness of the tracks, effectively making the charge conjugate version of the graph
- Failed attempts
  - Reflecting across z (magnetic field symmetry not preserved)
  - Coordinate transforms – Cylindrical/Spherical Inversions

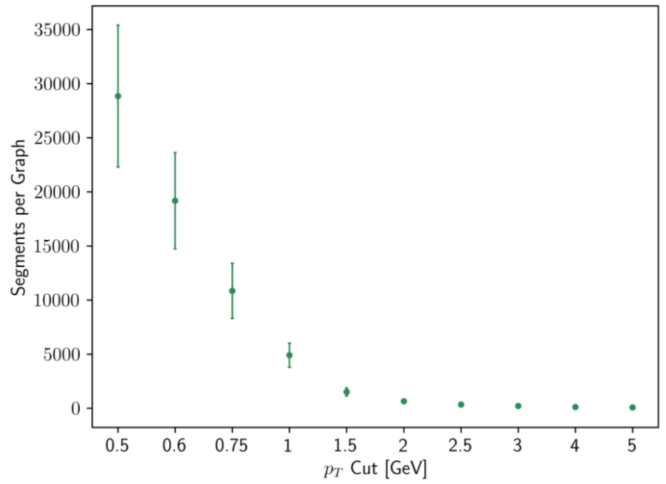




Graph Construction – Graph Size

- In addition to trying to maximize edge and tracking efficiencies, there is also concern about the size of the graphs
- Segmenting the graphs
- Can we get similar results using alternating layers to reduce edge/node count?
- Do we need all the endcap layers?

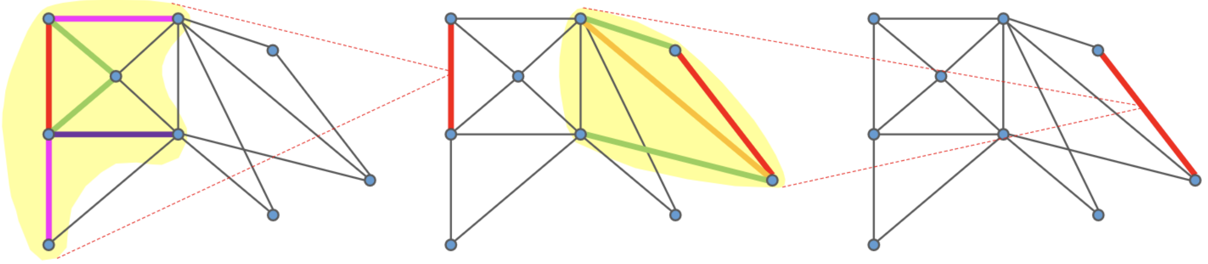


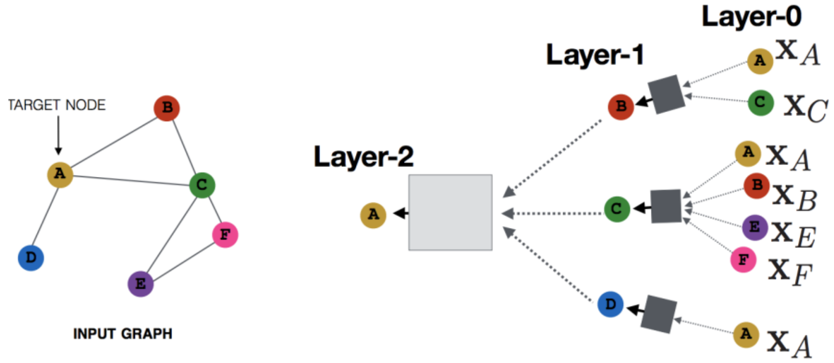


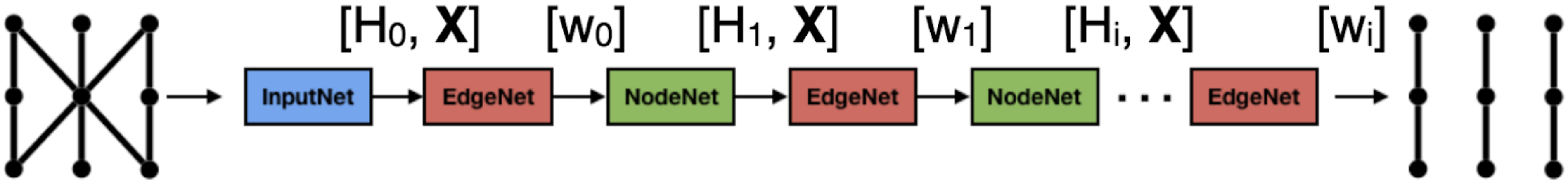
GNN Architectures

Edge Classifier 1

- A Graph Module combines an edge network and a node network
- Entire architecture is feed forward
- Parameters: 101249
- 6 Graph Modules, 128 Hidden Dimensions
- BCE Loss and .001 learning rate

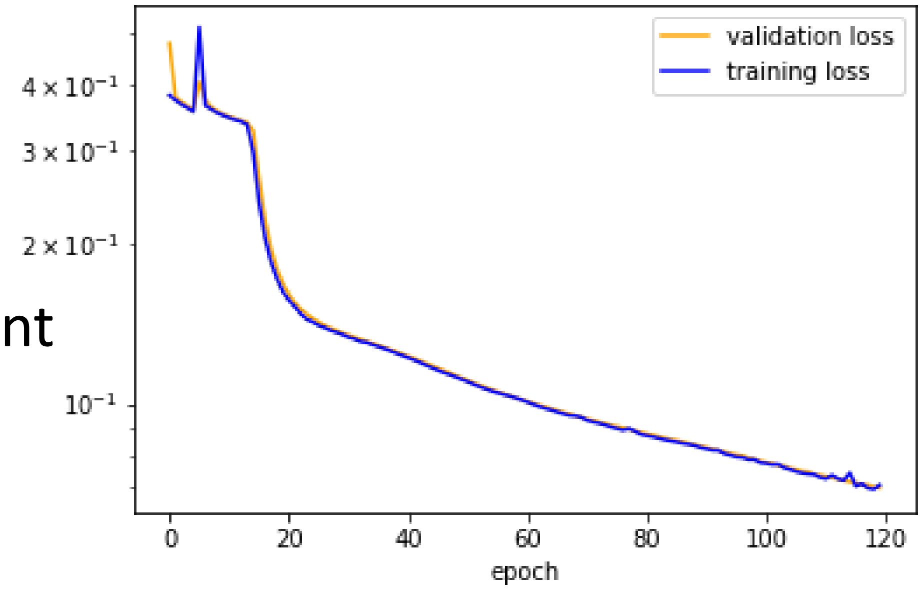






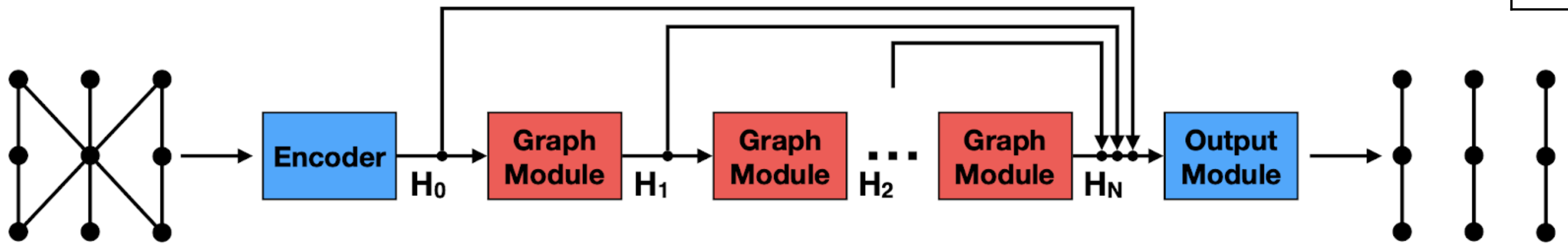
Edge Classifier 2

- Graph modules are recursively connected
  - Allows aggregation of progressively more distant information
  - Weights can be shared across modules
- Parameters: 259075
- 6 Graph Modules, with 64 hidden dimensions
- NLL Loss and .0001 learning rate



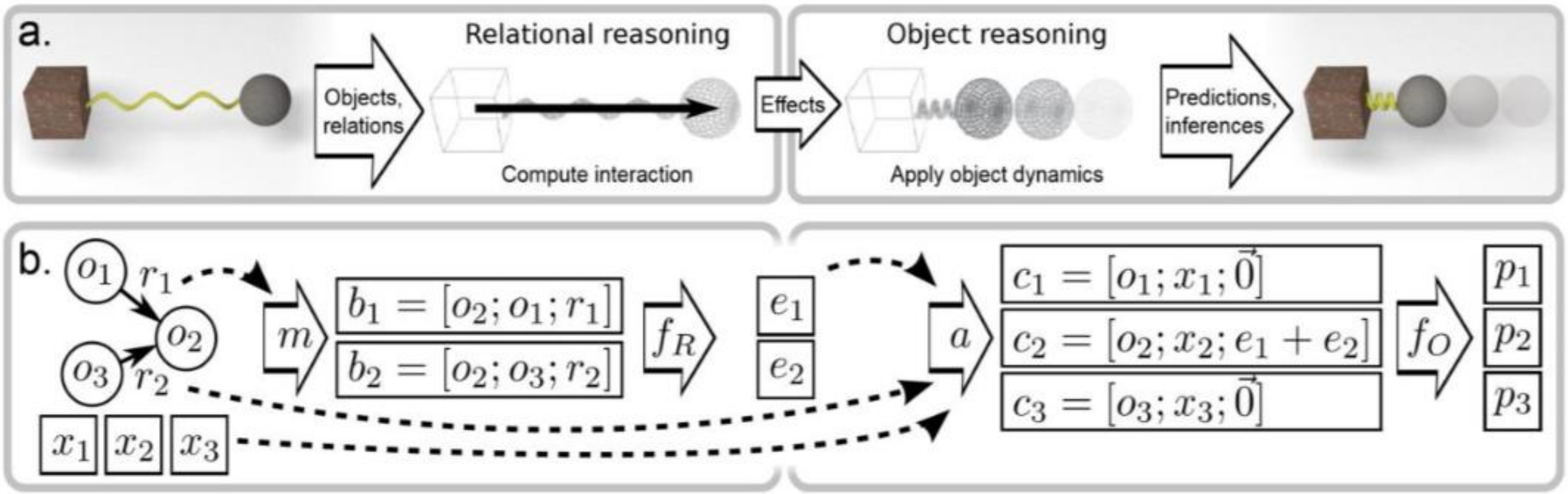
Confusion Matrix

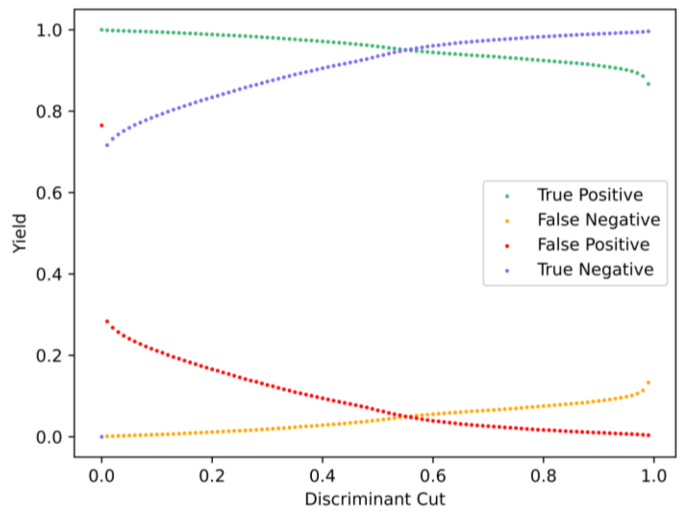
.999581	**.001837**
.000419	.998173

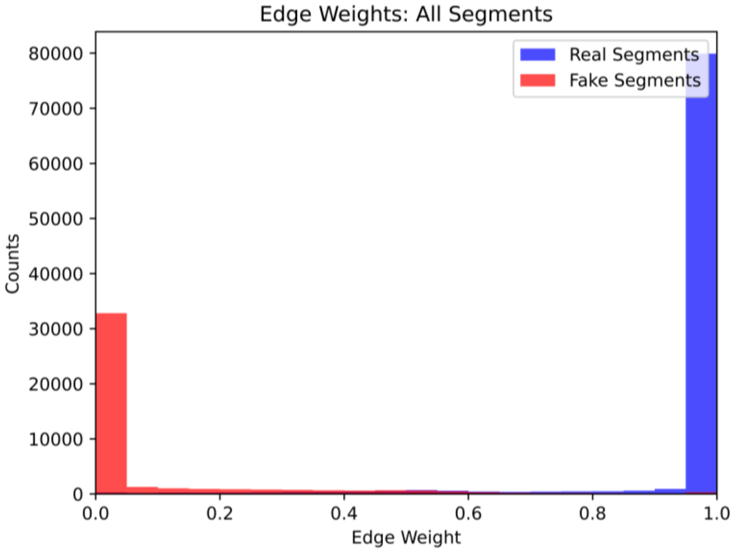


Interaction Networks

- Applies relational and object models in stages to infer abstract interactions and object dynamics
  - Relation and object models are FCNs
  - Total of 89400 parameters
  - 95% Edge efficiency



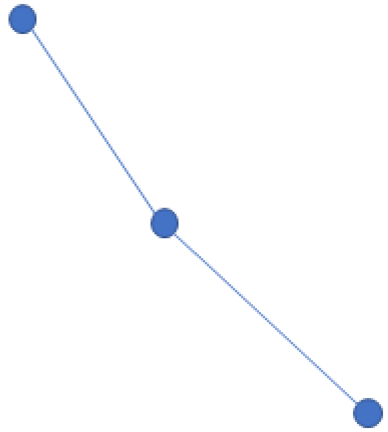




Confusion Matrix: (cut=0.60) [[0.948 0.052]
[0.053 0.947]]

Tracking Studies

Tracking – Union Find



- Using the inferred edges, nodes are grouped into subsets of connected unions
- Track candidates are defined as any union with ≥ 3 nodes
- Truth information for the entire detector is packaged with the graph that is used for track matching algorithms. (Requires pytorch-geometric dataset)
- **Match Criteria**
  - All hits within same union came from same particle

Tracking – Union Find

Track Efficiency

Matched Track Candidates / Total Truth Tracks

Fake Fraction

Unmatched Tracks Candidates / Total Track Candidates

- **Run on Target Graphs**
  - Allows us to quantify the best case scenario (perfect GNN inference)
  - Tracking Efficiency – Maximum achievable with the current graph construction cuts
  - Fake Fraction = 0

- **Run on Inferenced Graphs**
  - Tracking Efficiency for the particular GNN architecture and graph cuts
  - Fake Fraction for that particular GNN Architecture

  - **Calculate Tracking Efficiency Ratio**
    - Inference/Target
    - This is the fraction of unions that were inferenced correctly

Tracking – Union Find Studies

# Endcap Layers (per side)	Input Truth Graphs Track Efficiency	GNN Inferenced Graphs Track Efficiency	Ratio	Fake Fraction
0	.593322+-.037292	.592970+-.037357	.999407	.000599+-.001796
1	.689928+-.028164	.689625+-.028255	.999561	.000442+-.001327
2	.762234+-.026424	.761882+-.026793	.999538	.000478+-.001435
3	.877614+-.018441	.876589+-.018062	.998832	.001157+-.002528
4	.900148+-.020160	.900148+-.020160	1	0
5	.933451+-.016345	.933148+-.016031	.999675	.000317+-.000952
6	.940490+-.017478	.939564+-.016709	.999015	.000967+-.002043
7	.946276+-.019355	.942320+-.019378	.995819	.001010+-.002090

How many endcap layers are needed?

All numbers come from Edge Classifier 2 at p_T < 2.0 GeV

Tracking – Union Find Studies

	Input Truth Graphs Track Efficiency	GNN Inferenced Graphs Track Efficiency	Ratio	Fake Fraction
Inner Barrel	.593322+-.037292	.592970+-.037357	.999407	.000599+-.001796
Full Barrel	.633789+-.037619	.627592+-.037330	.990222	.006133+-.005702
Alternating Barrel	.567440+-.028849	.559576+-.027741	.986141	.008883+-.010534
Alt Barrel (doublets)	.615333+-.037120	.606547+-.035055	.985721	.008579+-.009785

How much efficiency is gained by including the outer barrel?
Remove every other layer?

Current best tracking results, need to do full p_T scan

	Input Truth Graphs Track Efficiency	GNN Inferenced Graphs Track Efficiency	Ratio	Fake Fraction
pT > 2.0 GeV	.946276+-.019355	.942320+-.019378	.995819	.001010+-.002090
pT > 1.0 GeV	.949068+-.008017	.940589+-.010110	.991065	.006211+-.002105

All numbers come from Edge Classifier 2

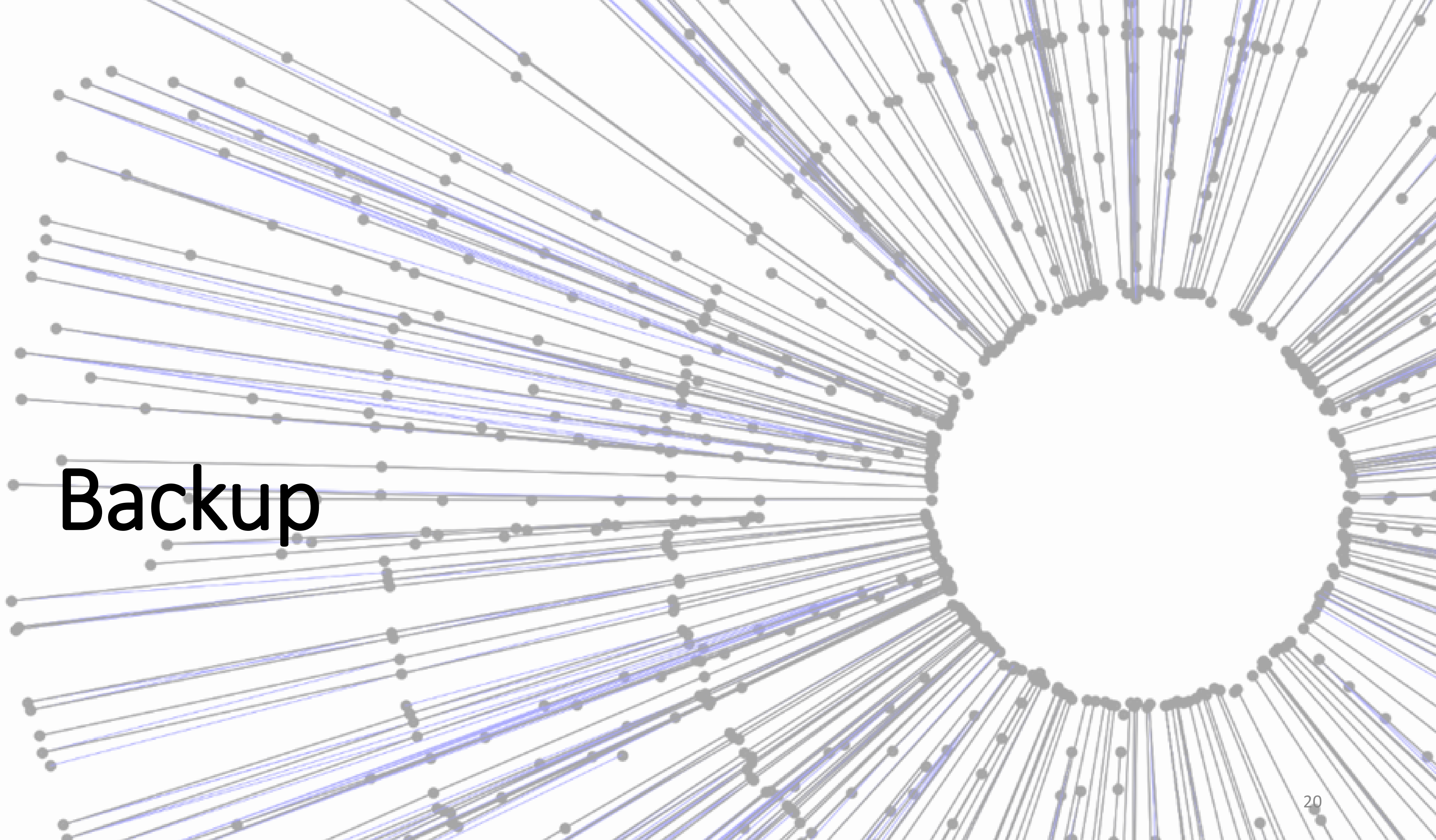
Ongoing Work

- Converting all code into a single useable framework (pytorch geometric)
  - Graph Construction, Edge Classifier 2, and UnionFind code are finished
  - Edge Classifier 1, partially converted
- Continue Optimizing cuts
  - Explore additional cuts
- Improve Existing Architectures
  - Explore additional Architectures
- Additional data augmentation
  - Transforms, embeddings
- Additional Track building algorithms

Summary

- GNNs are a promising method for HL-LHC tracking
  - Geometric data representation with variable number of inputs
- A variety of architectures have been shown to work
  - Focus is now on refining and optimizing
- Graph construction (and embedding) is critical to performance
  - On-going optimization studies
- Working towards accelerating graph algorithms on FPGAs for use at HL-LHC
  - Next Talk

Backup

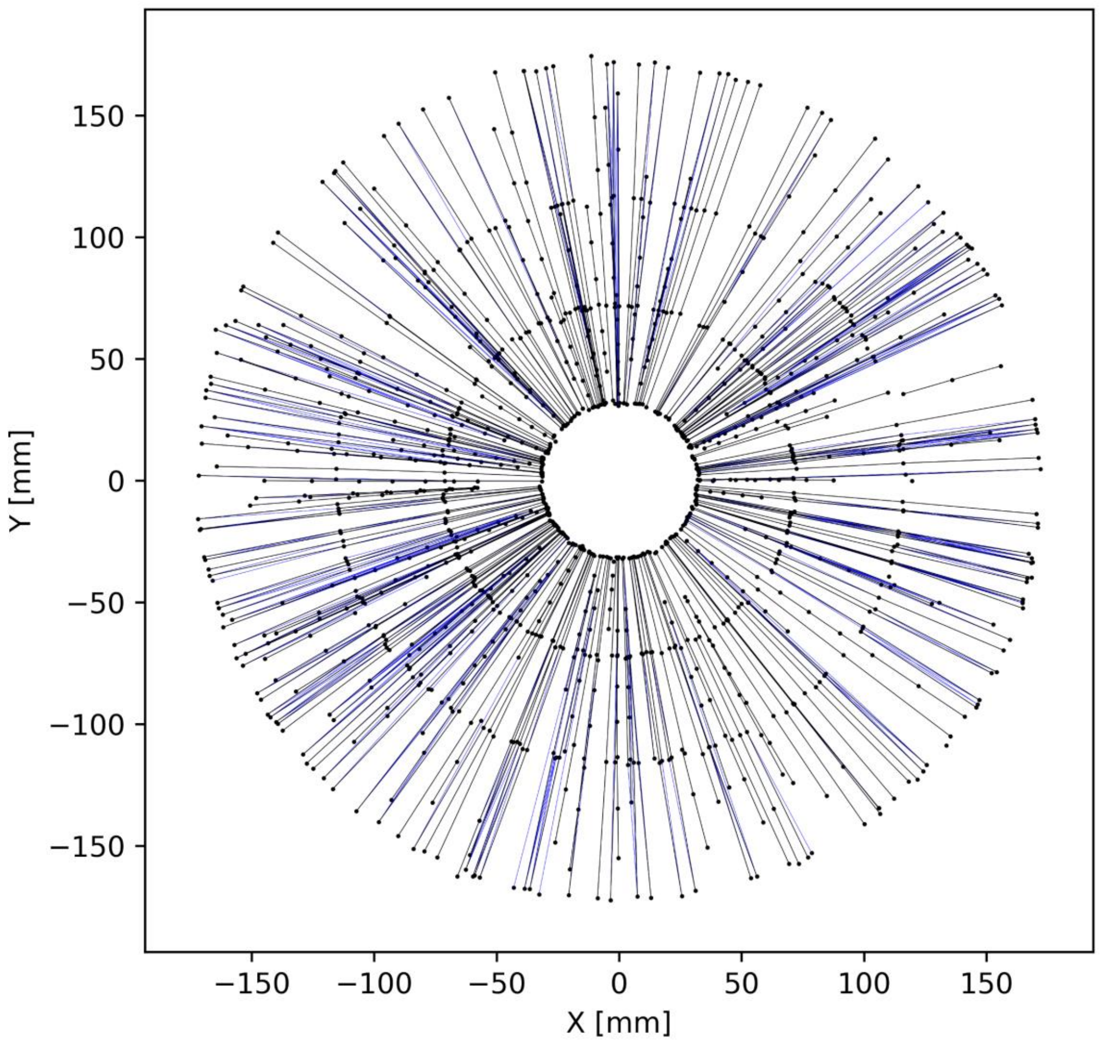


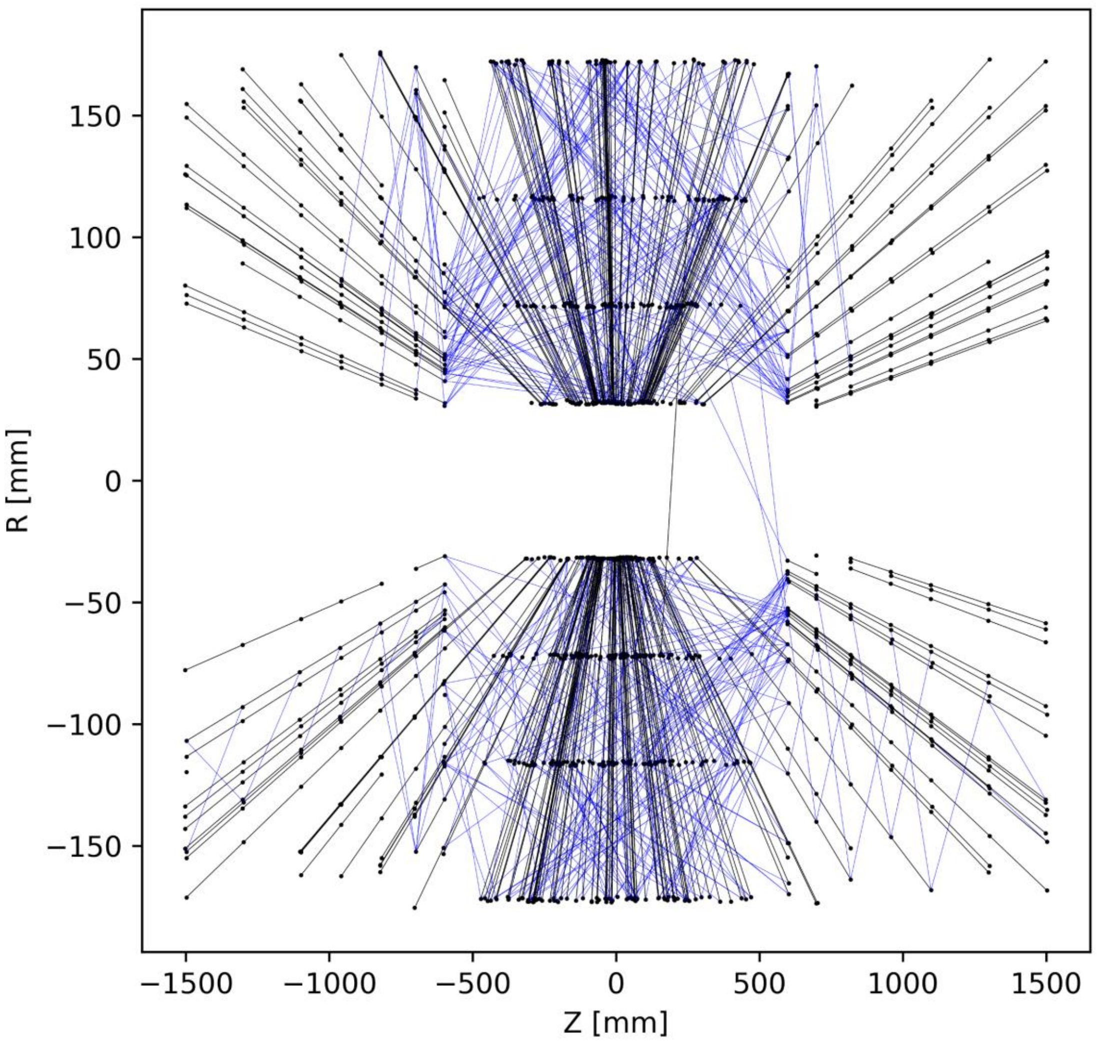
Tracking – DBScan

Graph Construction – Other Algorithms

- Other algorithms being explored
  - Layer Pairs +
  - Dynamic kNN graphs
  - Learned clustering
  - DBScan in eta-phi space

Example Graph pT > 2.0 GeV





Tracking – Union Find