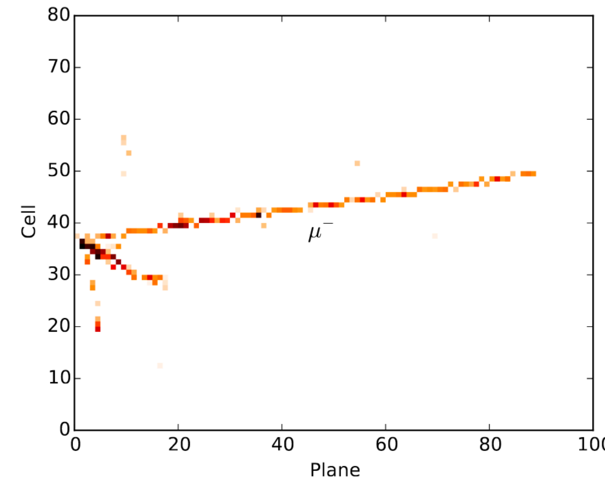
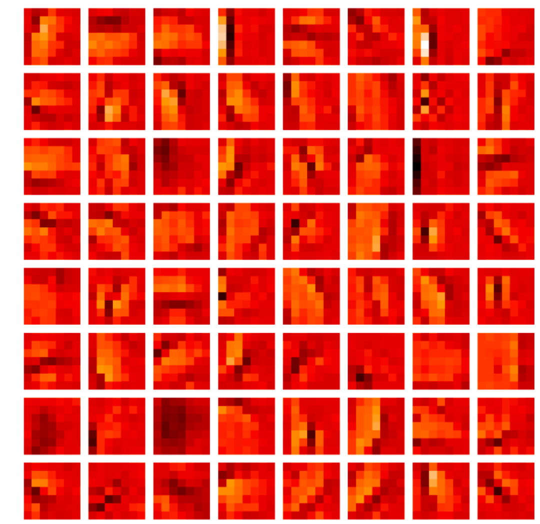
Graph Neural Networks for reconstruction in DUNE

Jeremy Hewes
DUNE FD simulation & reconstruction meeting
10th June 2019

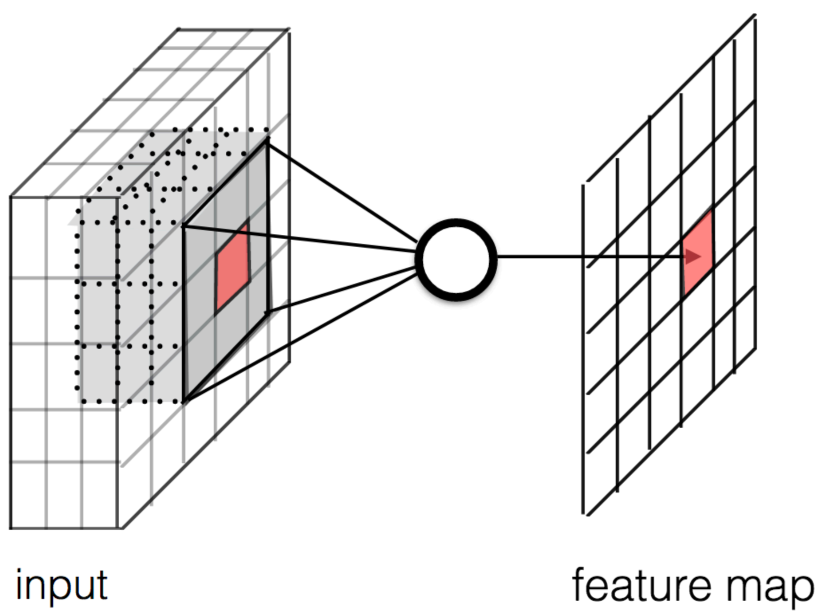
Convolutional Neural Networks

- Convolutional neural networks show great promise in image classification over the past decade.
- Most neutrino detector technologies naturally provide pixel maps which can be classified using CNNs.
- Examples: NOvA, MicroBooNE, DUNE.





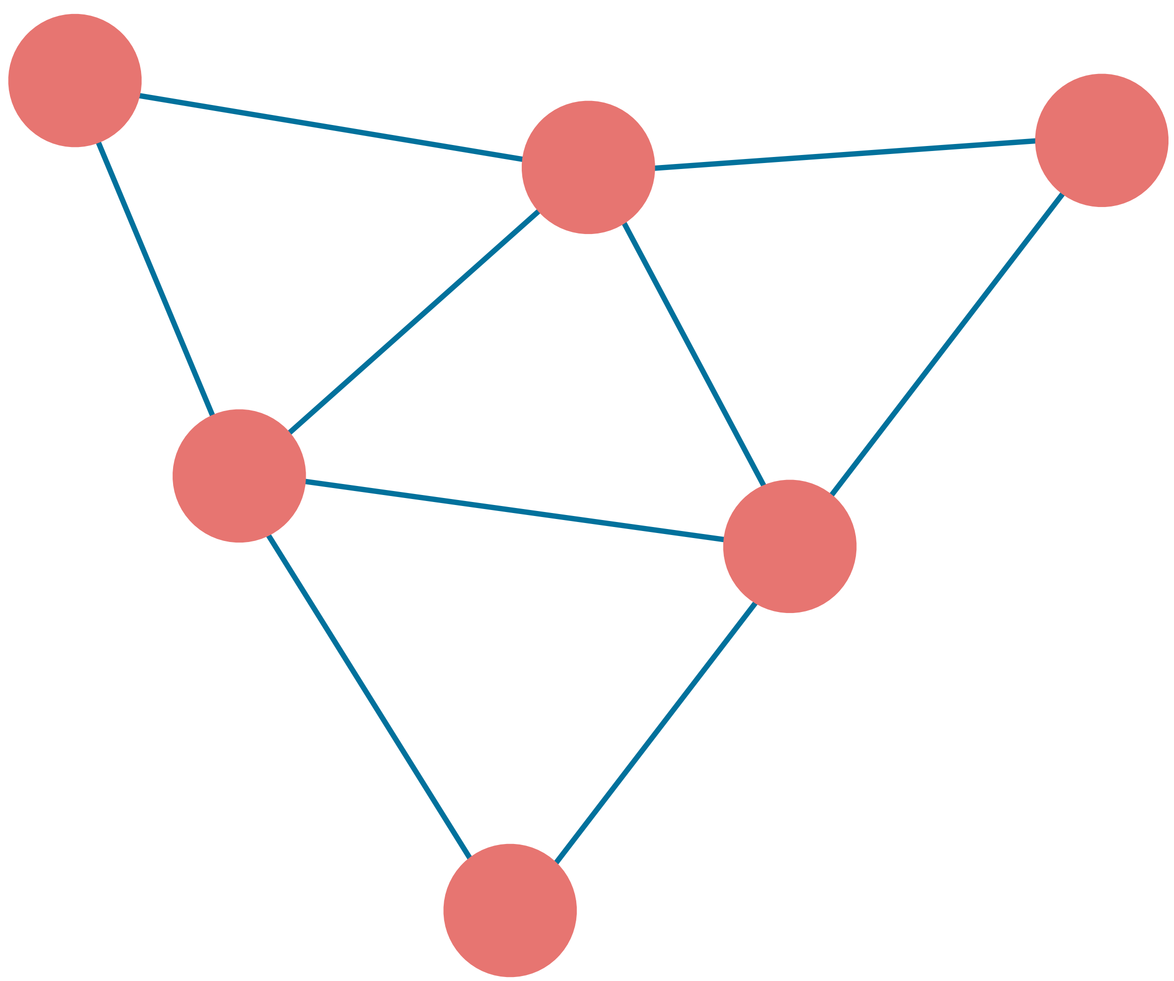
arXiv:1604.01444



- Issues with this approach:
    - **Dense** representation of **sparse** data.
    - Operate over mostly empty space!
- Some ways around this: ie. Sparse submanifold representation (arxiv: 1903.05663).
- Alternatively: reframe problem entirely!

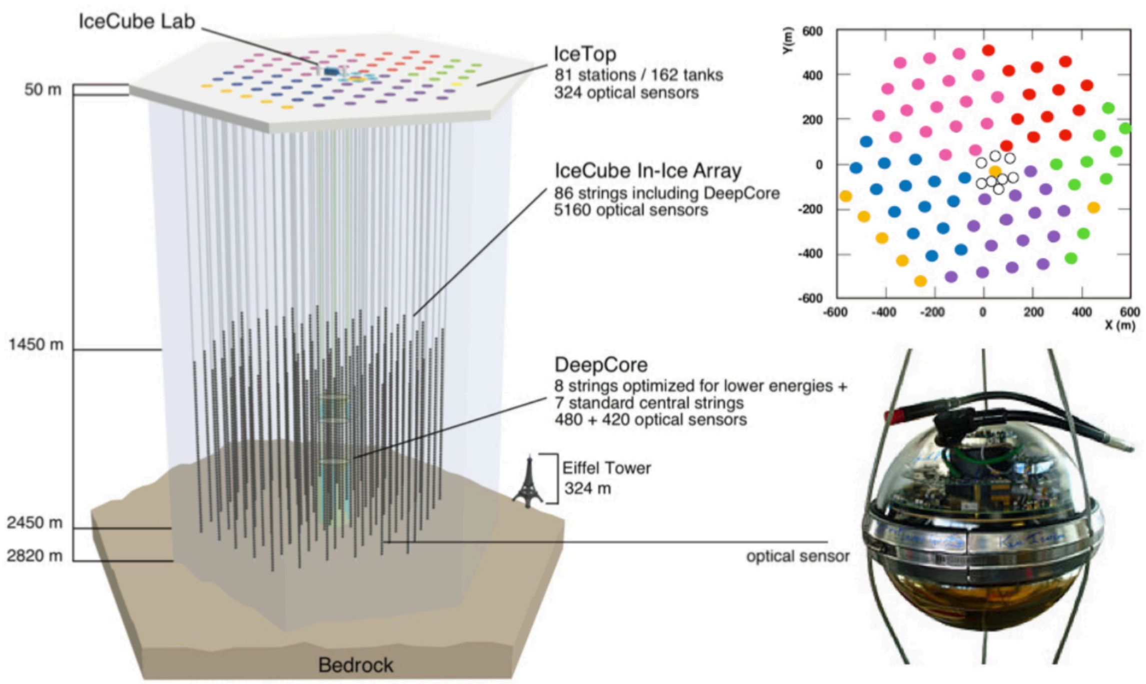
Graph Neural Networks

- Instead, define your data structure as a **graph** represented by **nodes** and **edges**.

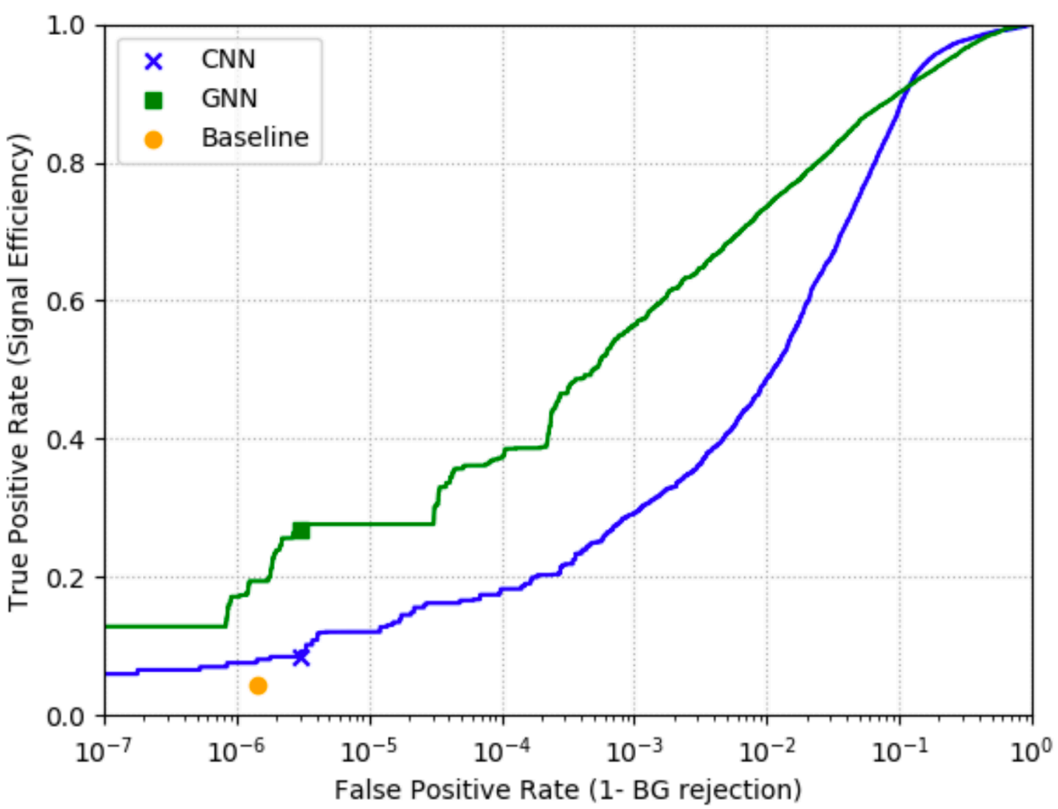


- **Nodes** are generalised as quantised objects with some arbitrary set of **features**.
- **Edges** describe the **relationships** between nodes.
- Perform convolutions on nodes and edges to learn relationships within the graph.
- Output is user-defined:
  - Classify nodes or edges.
  - Classify full graph.
  - Regression outputs.

IceCube graph network



$$\mathrm{GConv}(\mathbf{X}^{(t)}) = [\mathbf{A}\mathbf{X}^{(t)},\ \mathbf{X}^{(t)}](\mathbf{a}^{(t)})^{\top} + b^{(t)}\mathbf{1}$$

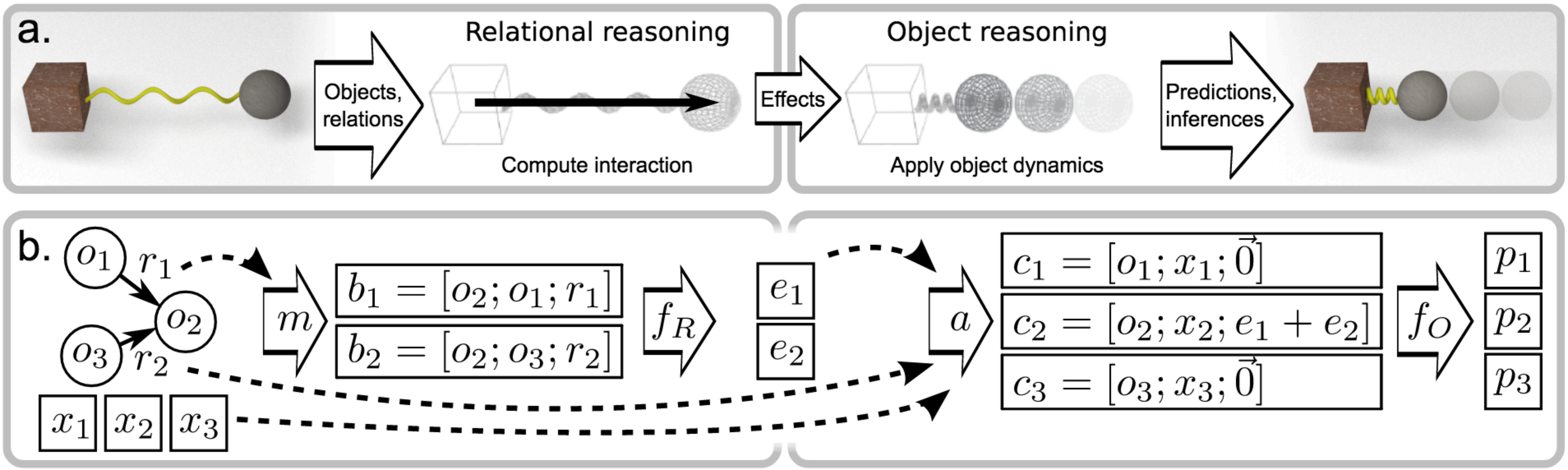


- Search for **astrophysical ν_μ interactions**.
  - Reject cosmic ray μ background.
- Construct fixed-size graph with DOMs as nodes.
- Use GNN for **event classification**.
- Outperforms both baseline reconstruction and 3D CNN approach.

Method	Signal	Background	Signal:Noise
	# events per year		
Physics Baseline	0.922	0.934	0.987
3D CNN	1.815	1.937	0.937
GNN	**5.772**	1.937	**2.980**

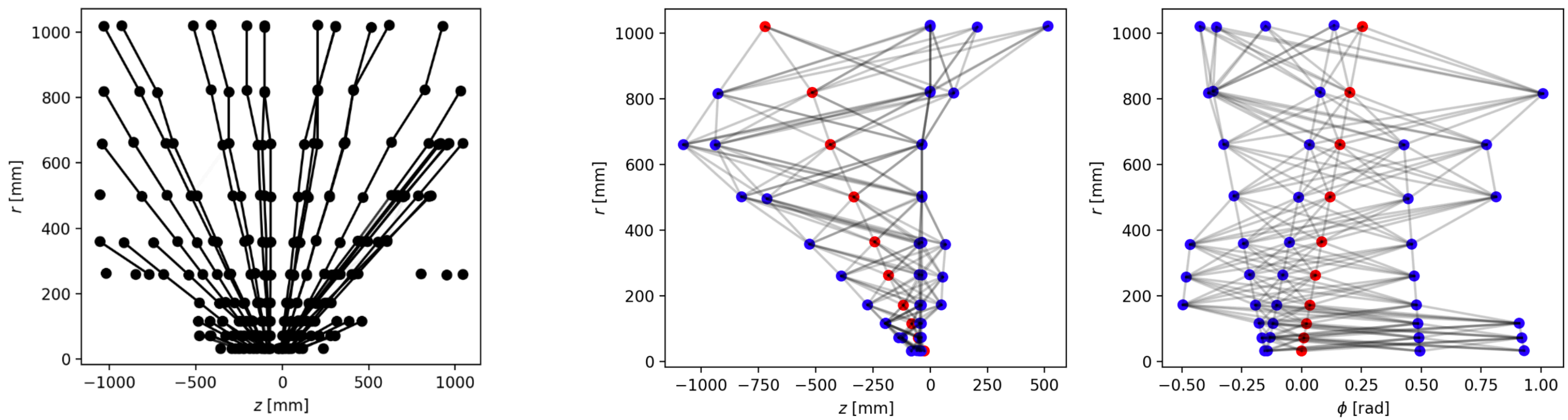
arXiv:1809.06166

Interaction networks



- Interaction network developed by Google DeepMind (arxiv) performs convolutions on graphs.
- Using input **objects o_i** and **relations r_i** (ie. nodes and edges) to perform **edge classification e_i** and **node classification p_i**.
- Independent models for node and edge classification.
- Apply the same model iteratively, allowing neighbour information to propagate outwards.

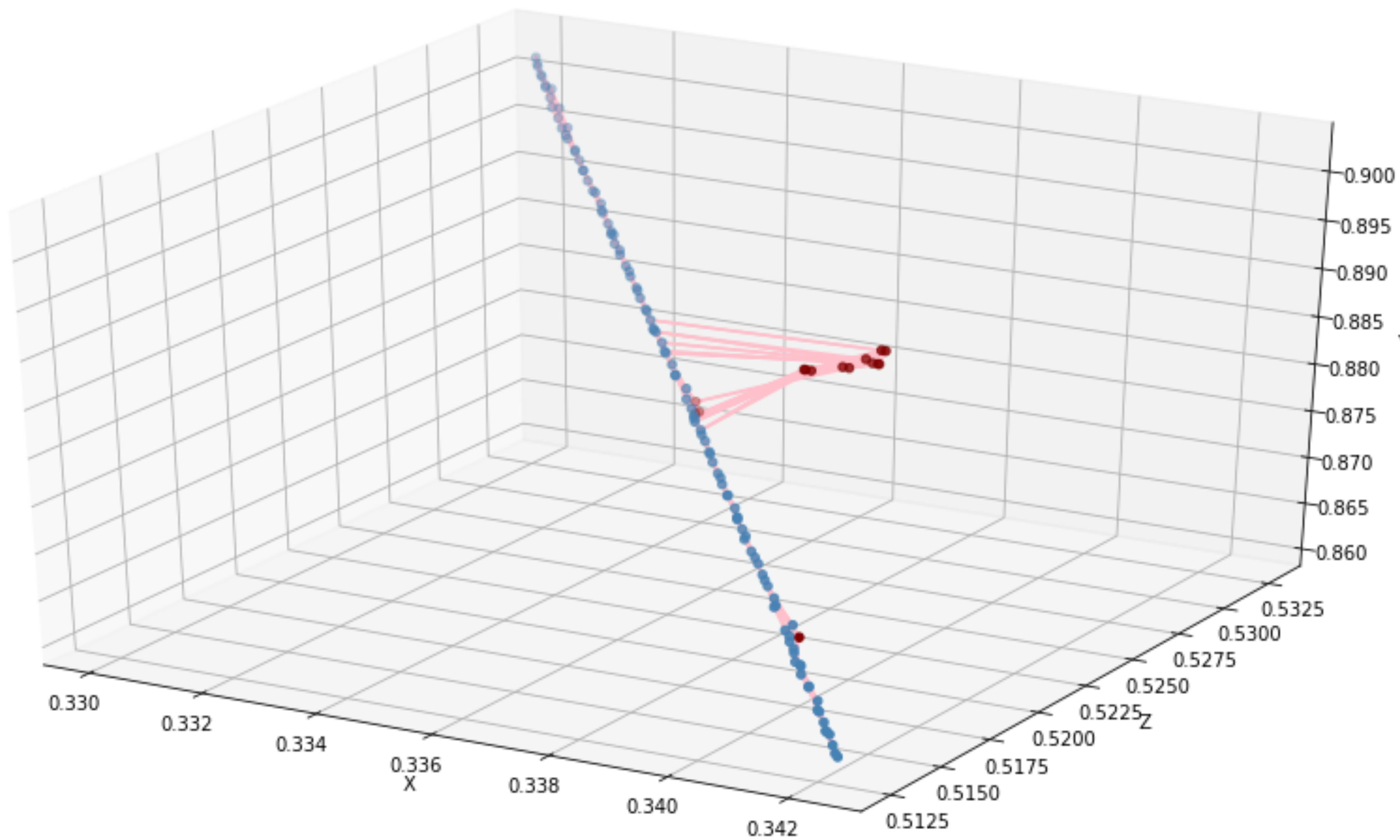
HEP.Trkx graph network



arXiv:1810.06111

- Use GNNs for **track reconstruction** in HL-LHC.
- Construct graphs with detector hits as nodes, and edges linked across adjacent detector layers.
- Alternate between **node classifier** and **edge classifier** networks.
- Whether the network ultimately classifies nodes or edges simply depends on which comes last in the chain.
- Hit classification model achieves **99.2% purity**, **97.9% efficiency** and **99.4% accuracy**.

Applications in DUNE



- Use **3D WireCell spacepoints** as input to network.
- Train network to **group 3D spacepoints into clusters** for downstream processing.
- Toy study: use Pandora 3D spacepoints from ProtoDUNE MC as input.

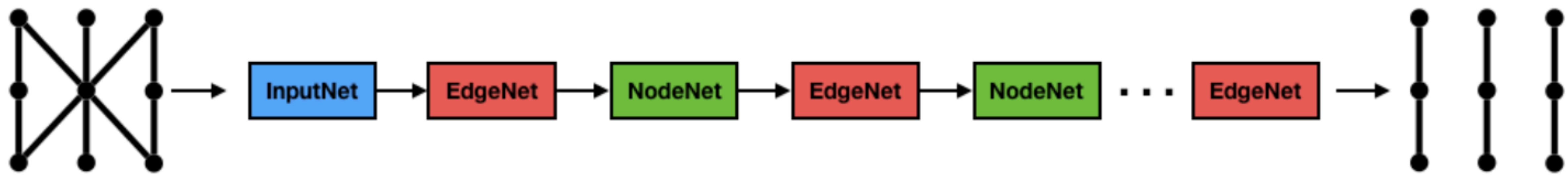
Spacepoint classifier

- Testing an interaction network adapted from the HEP.TrkX variant.
- First test: ProtoDUNE MC readout window has **~50k Pandora spacepoints**.
    - Some thought required to construct a sensibly defined graph structure.
- First pass: construct **cluster-wise** graphs from Pandora PFParticles.
    - Construct graph from reconstructed spacepoints in local area, and train a graph network to classify spacepoints.
    - Four features per node: **3D position (x,y,z)** and **ADC**.
    - Filter out oversize events (>10k spacepoints), limit number of incoming & outgoing edges per node to four.
    - Start with **edge network** to strengthen connections between spacepoints that come from the same true particle.

Input preparation

- Utilised **GCN graph maker** module developed in dunetpc by Leigh Whitehead to produce graph objects.
    - Available in **feature/cvnUpdates** branch.
- Wrote a LArSoft module for writing graphs in HDF5 format.
    - Used HighFive headers which trivialise the process of writing .h5 files in C++.
    - Code uncommitted for now due to hacky HDF5 interface.
    - In communication with DUNE computing about getting official support for HDF5 C++ interfaces in the DUNE environment.
- **Future:** Benchmark ROOT file loading times in Python with uproot vs HDF5, see how they compare.
    - uproot's support for jagged arrays (ie. variable graph sizes) should require less time spent packing/unpacking vectors.

Network architecture



arxiv:1810.06111

- **Edge classifier:**
  - Input for each node is the features of incoming and outgoing nodes.
  - Two multi-layer perceptrons, using Tanh and sigmoid activations.
  - Outputs sigmoid score on each edge.
- **Node classifier:**
  - Uses edge score to aggregate each node's features with incoming & outgoing edges as input.
  - Two multi-layer perceptrons with Tanh activation.
  - Produces new features for each node.

Training parameters

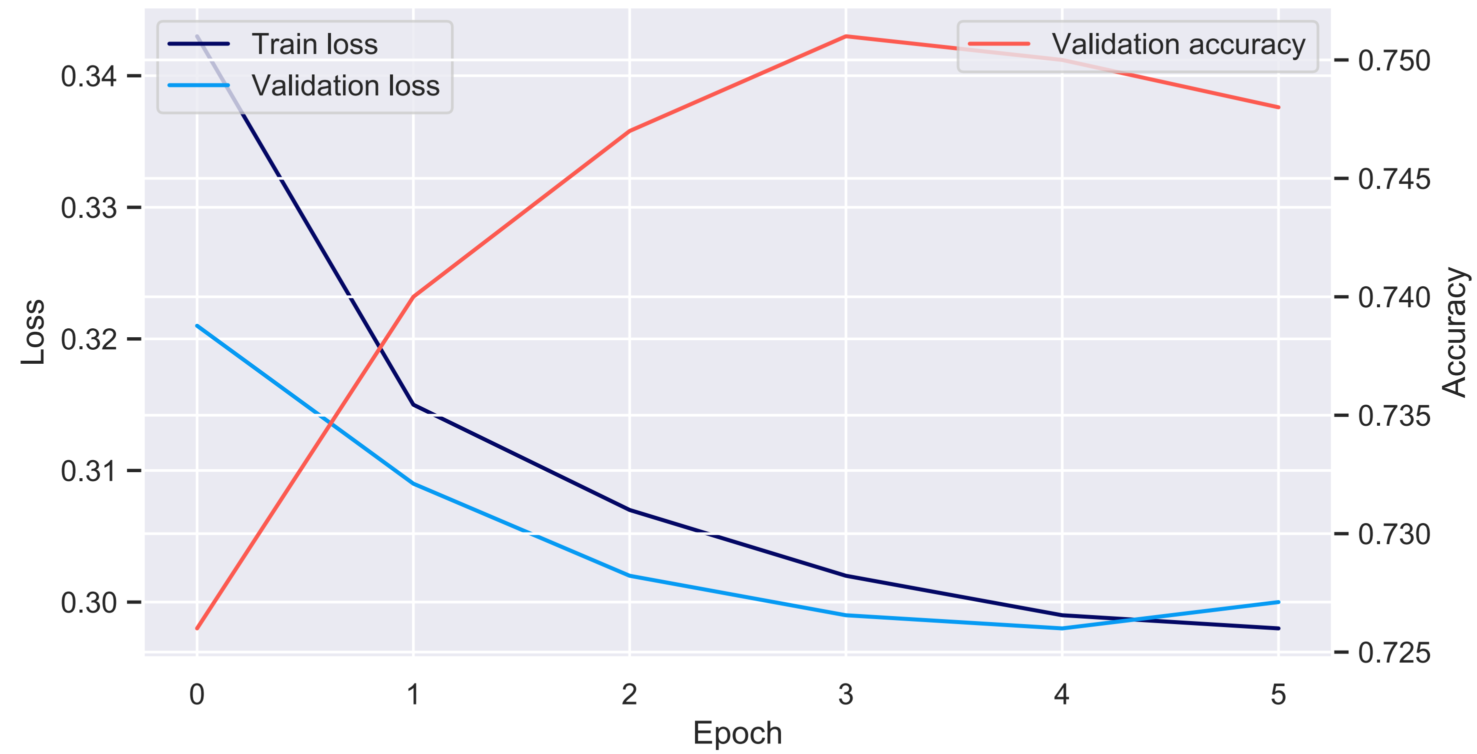
Objective: Binary cross-entropy
Optimiser: Adam
Learning rate: 0.001
7 model iterations
Batch size: 3

Model parameters: 26433

Memory usage: ~16GB

Network performance

- Some small evidence of learning over the first few epochs, but clearly **much room for improvement**!
- Succeeded in initial goal: **constructing workflow** to produce graphs and train networks in TPCs



- **Next steps**:
  - Train on WireCell spacepoints.
    - Less dense point clouds may prevent the need to train on cluster-wise graphs.
  - Strip out some vestigial machinery from the HEP.TrkX network.
  - Investigate node classification in more detail.
  - Explore entirely different graph constructions.
    - Cluster-wise graphs for particle ID?

Current status

- Training in progress – but still iterating on optimal combination of graph definition and network architecture.
- Next steps:
  - **Directionality**: Current network uses a *directed* graph; an undirected approach may be more effective for clustering spacepoints.
  - **Training efficiency**: Batch-processing graphs requires zero-padding to the size of the largest graph in the batch.
    - Intelligently batching inputs of similar sizes should provide yields both in terms of network performance and computational efficiency.
  - **Graph construction**: Anticipate more coarsely distributed spacepoints from wirecell, which may change how graph is constructed.
- Collaborating with Leigh Whitehead and Saul Alonso Monsalve, who are exploring similar graph network ideas.

Graph construction

- Detector geometry in IceCube and LHC provide initial constraints:
  - **Quantised DOM structure** in IceCube limits number of graph nodes.
  - **Layer structure** in LHC provides natural constraint on edges (limited to adjacent detector layers).
- Much of the benefit of graph networks in LArTPCs is reduction of dense information to some sparse representation.
  - A lot of freedom to define our inputs however we want!
  - …which means more thinking up top about how to optimally define our inputs.
  - No one clear solution! Construct a graph, test network architectures and iterate.

Summary

- **Graph Neural Networks** allow for training on **sparsely represented inputs**.
- More efficient and flexible alternative to CNNs (although more assembly required!)
- First goal: construct graph of 3D spacepoints, and train a network to perform clustering.
- Potential applications are broad: defining graphs at **different levels of abstraction**, and with **different outputs**, means potential solutions to **many different problems**.
  - Clustering spacepoints is the first step on a longer path.