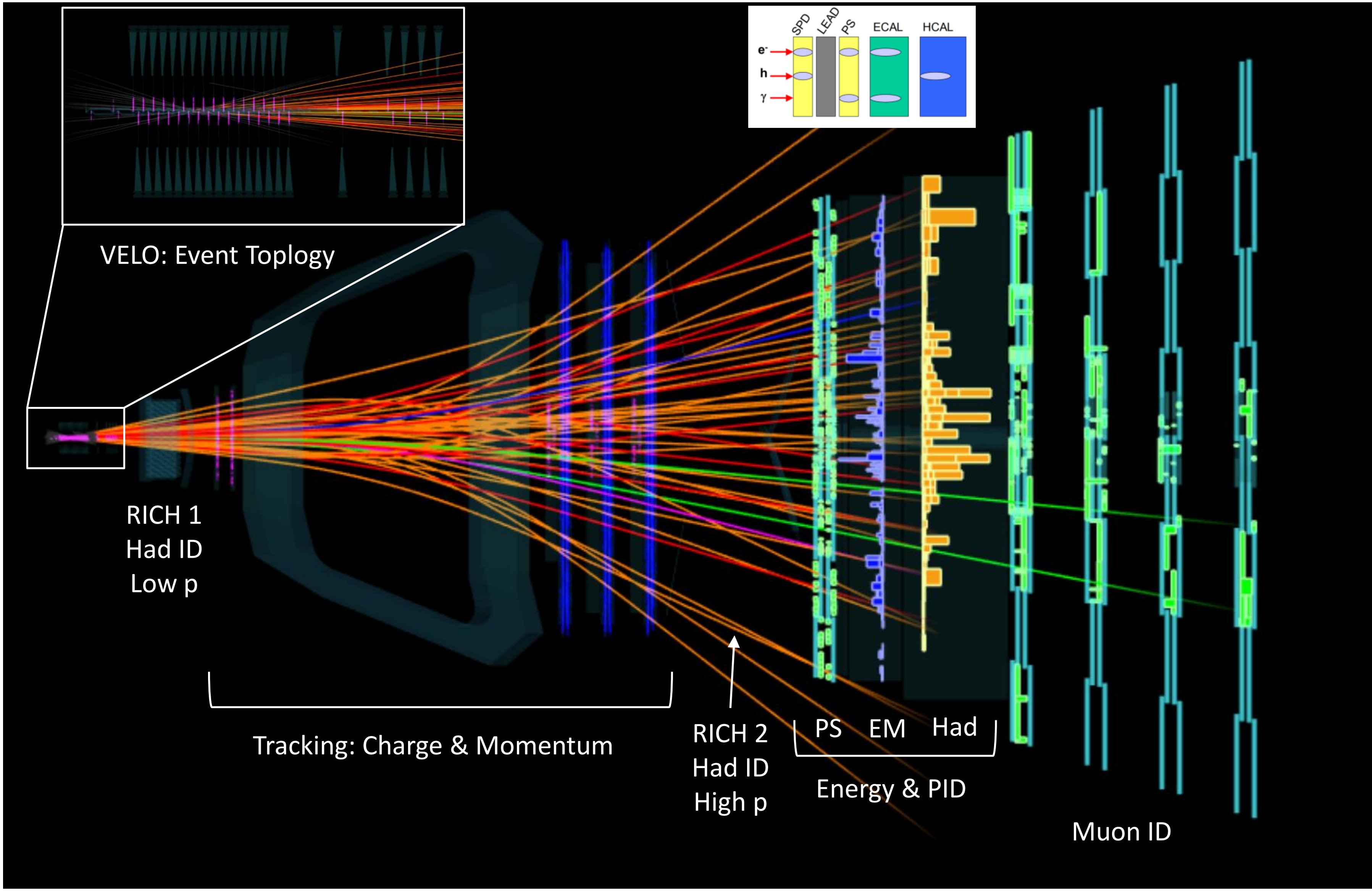
Calorimeter reconstruction with computer vision at LHCb

João Coelho in Collaboration with

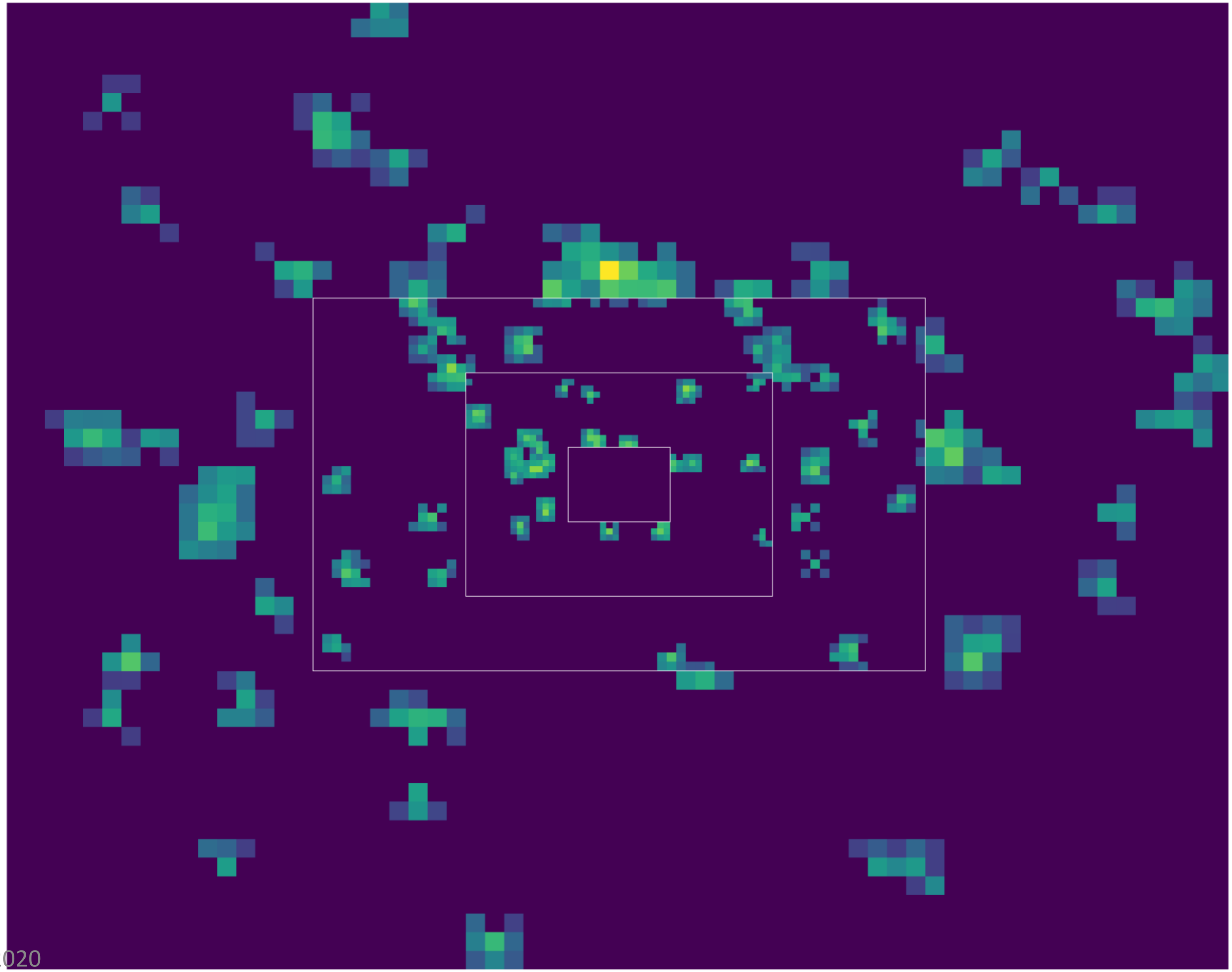
B. Delaney (Cambridge) and M. Mazurek (NCBJ)

22 January 2020

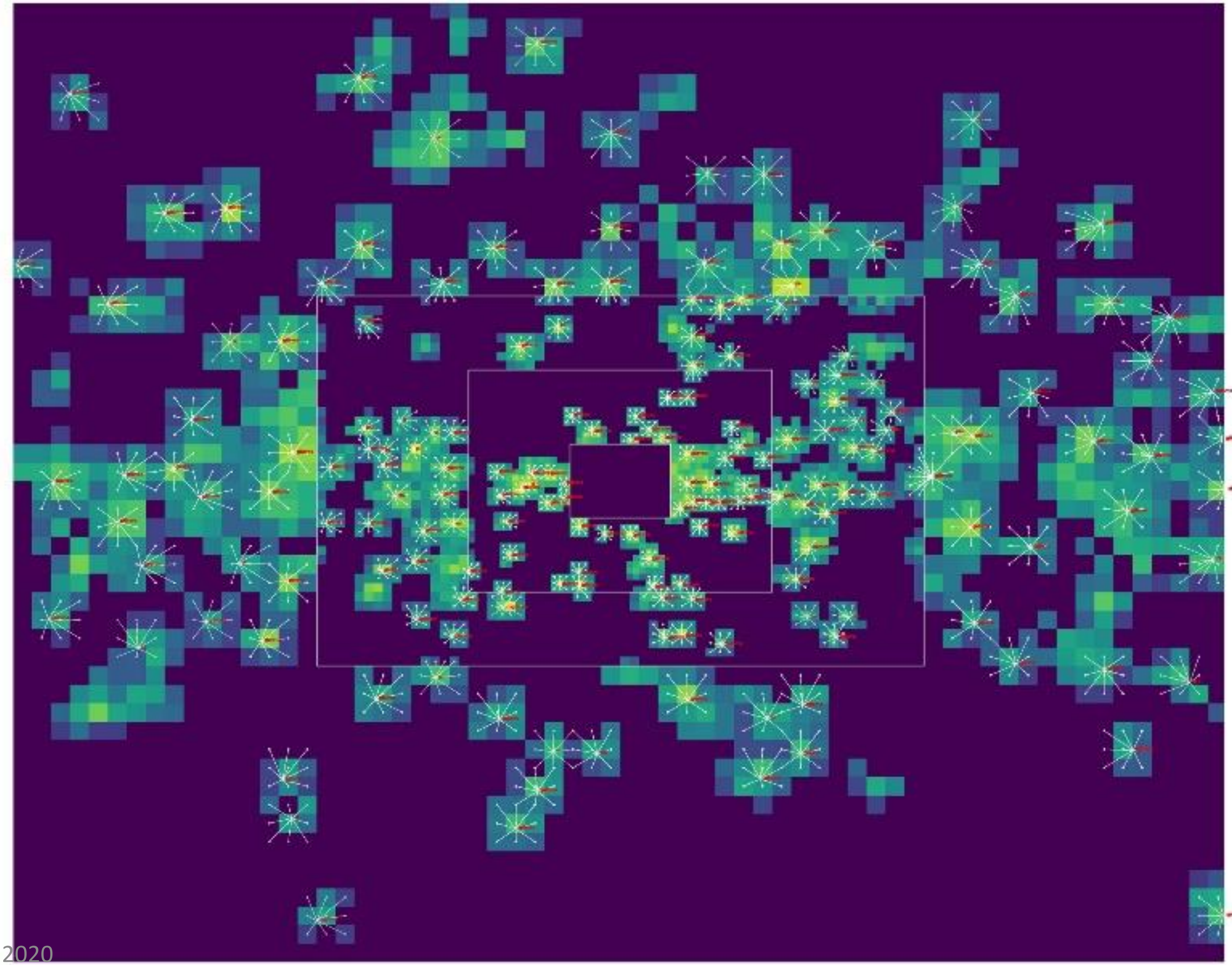
The LHCb Detector



A Calorimeter Event

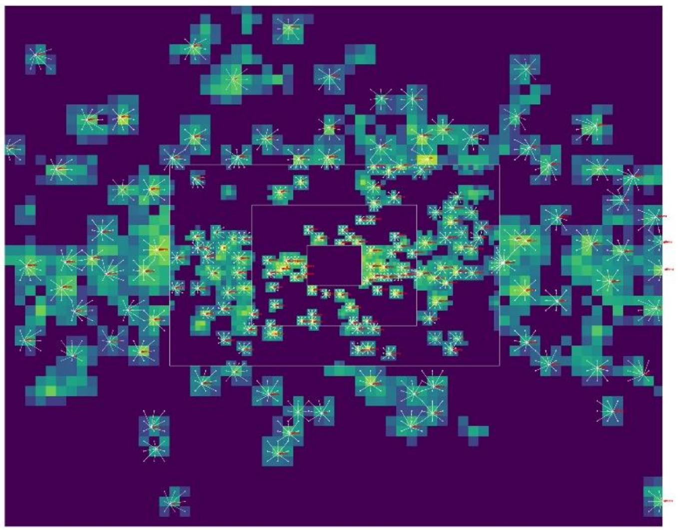


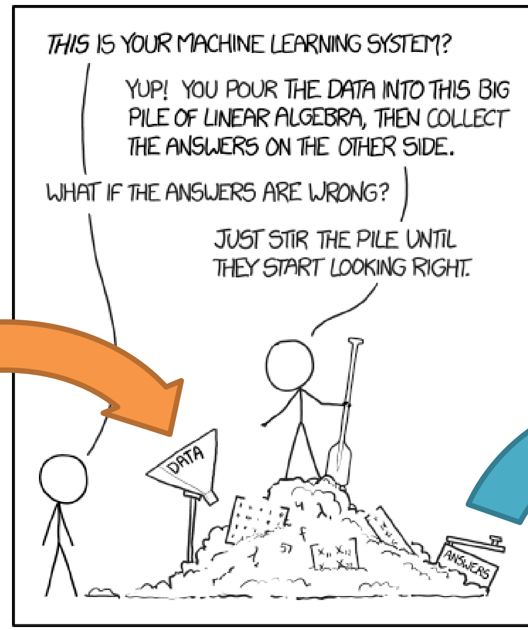
Things can get busy!



The Goal

- Our goal is to create a neural network architecture to process calorimeter images and output a set of clusters with position and energy information

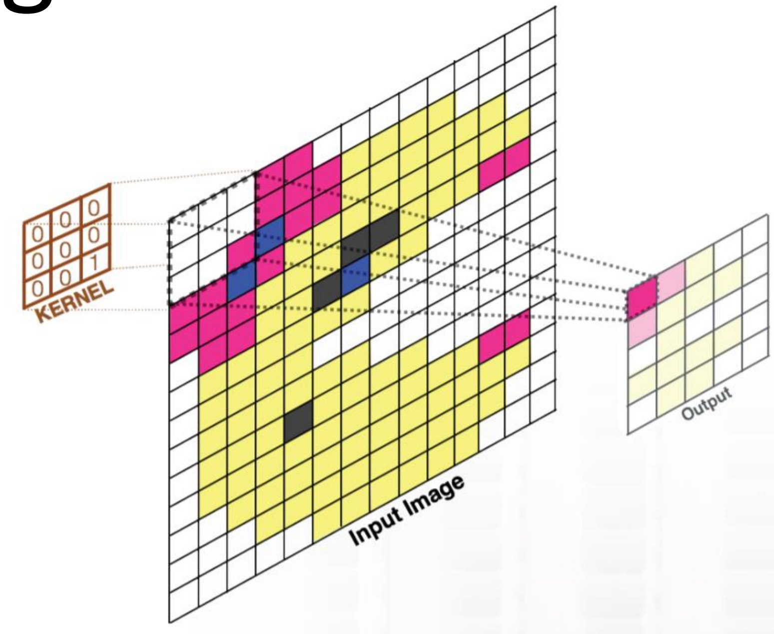


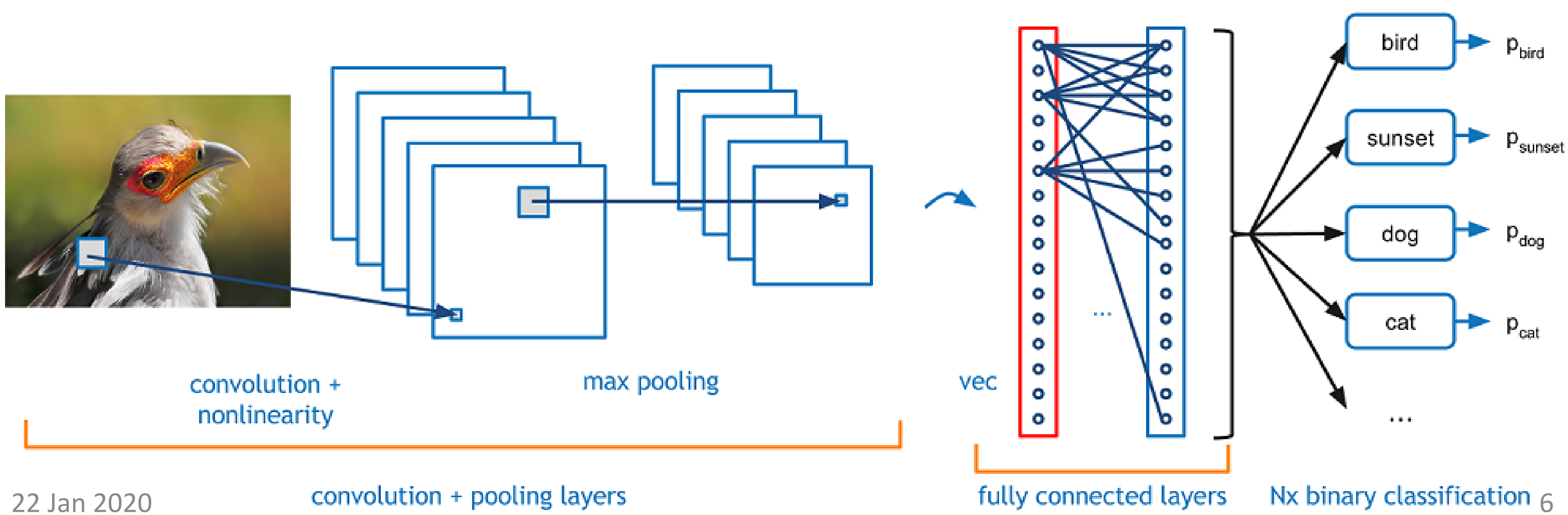


x	y	E
0.54	−0.88	1.24
−0.39	0.02	1.54
0.65	0.16	1.23
−0.09	0.84	0.48
−0.44	0.29	1.71
	.	
	.	
	.	

Image Recognition

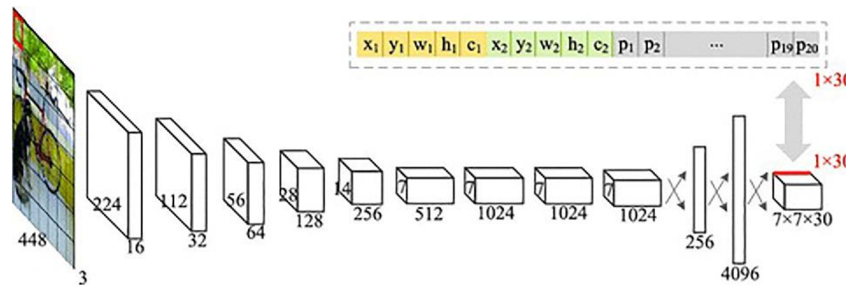
- Convolutional Neural Networks have become a standard method for encoding an image into a vector representation
- The most common use is to take this representation to classify the image
- Method is more general than that and representation can be seen as a compressed form of the original image

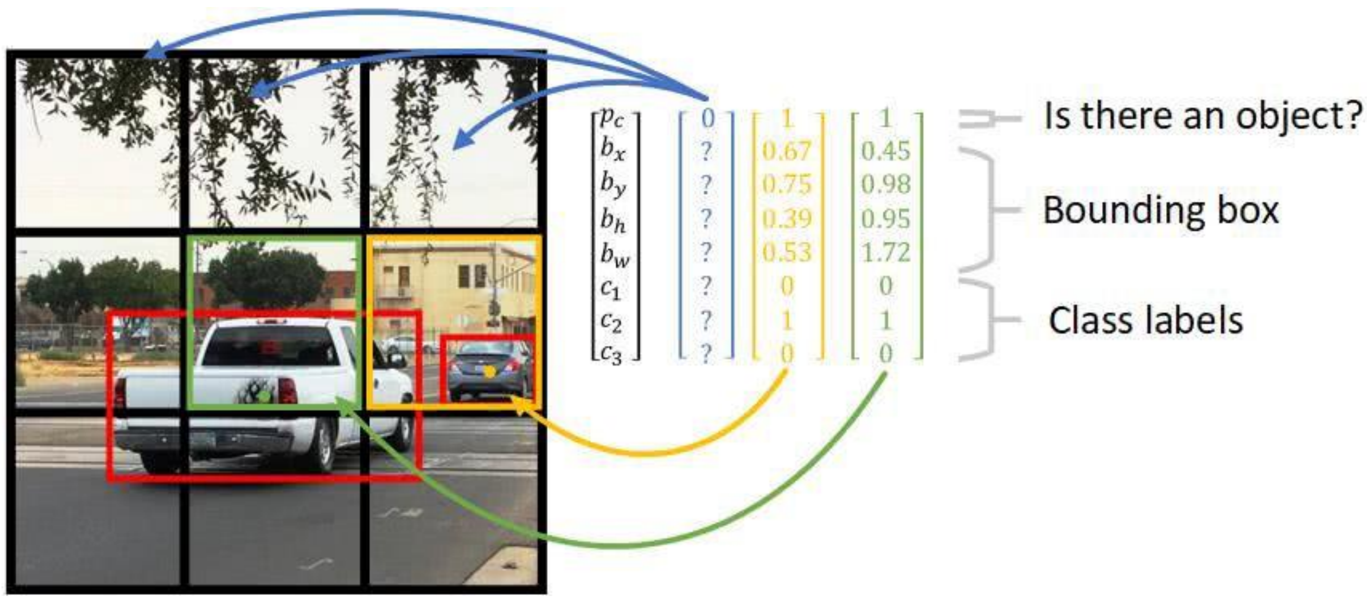




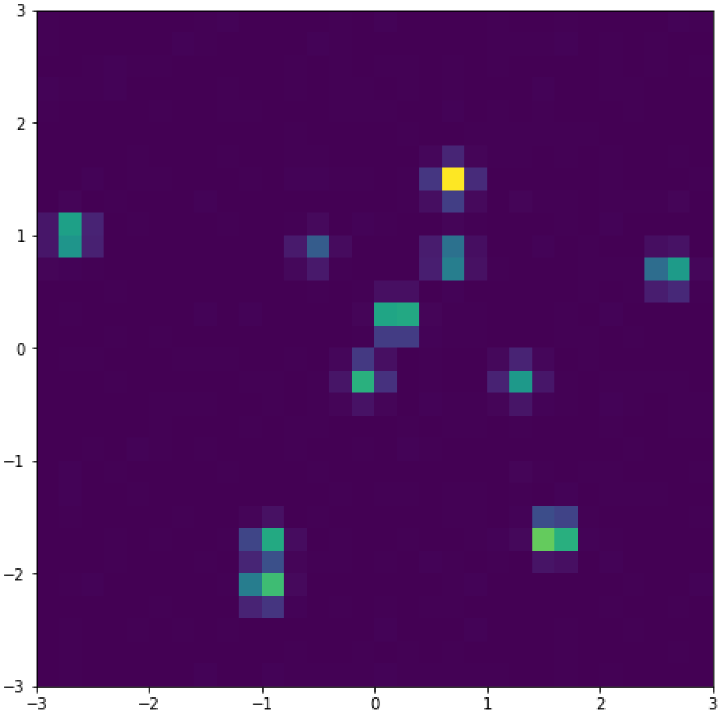
You Only Look Once (YOLO)

- Split the image in a GxG grid of regions
- For each region, predict an object class
- If region not "empty", also predict bounding box and classification score
- Bounding box: x, y, width, height
- Input: NxN image
- Output: GxGxF tensor of box predictions



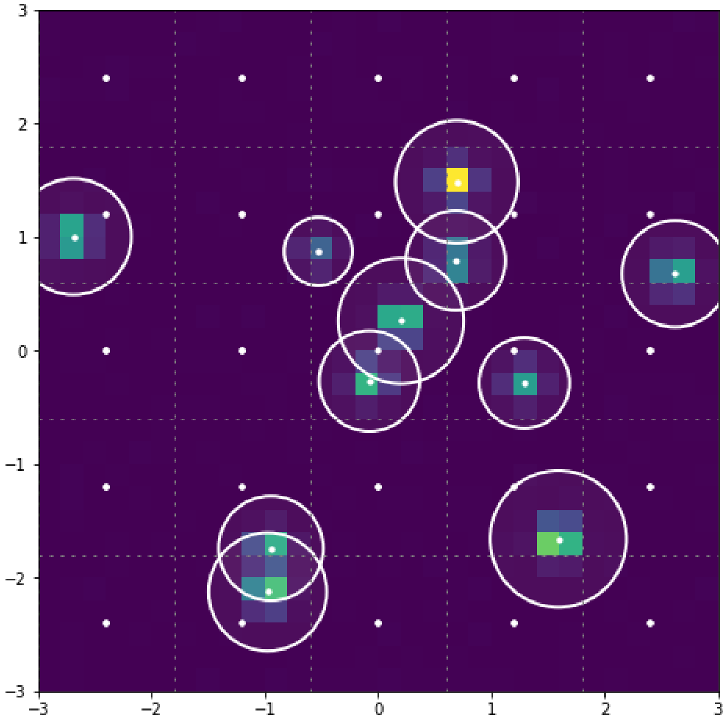


CaloYOLO



- Started with a simple toy dataset
- 30x30 pixel images
- Showers with Gaussian shape
- # of showers: Poisson w/ mean 5
- Position: Gaussian w/ mean 0 & stdv 1
- Energy: Gaussian w/ mean 6 & stdv 2
- Remove showers with E < 0.5

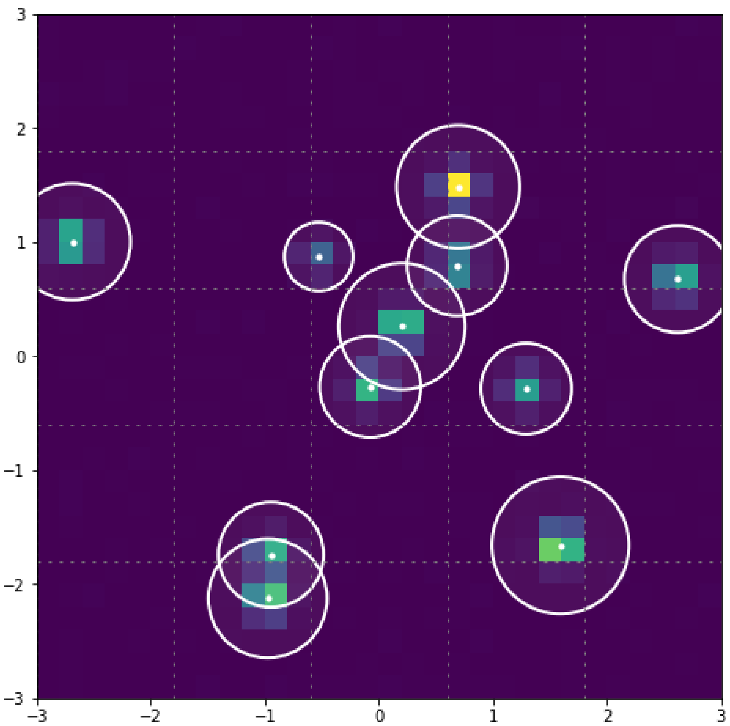
CaloYOLO



- Started with a simple toy dataset
- 30x30 pixel images
- Showers with Gaussian shape
- # of showers: Poisson w/ mean 5
- Position: Gaussian w/ mean 0 & stdv 1
- Energy: Gaussian w/ mean 6 & stdv 2
- Remove showers with E < 0.5

- Truth: GxGxOx3 tensor
  - G: # of grid cells per axis
  - O: Max # of showers per cell
  - For each shower: x, y, E is given
  - Missing showers have x,y,E = 0
- Example: 5x5x3x3

CaloYOLO



Ignore showers with E<0.5

- Started with a simple toy dataset
- 30x30 pixel images
- Showers with Gaussian shape
- # of showers: Poisson w/ mean 5
- Position: Gaussian w/ mean 0 & stdv 1
- Energy: Gaussian w/ mean 6 & stdv 2
- Remove showers with E < 0.5

- Truth: GxGxOx3 tensor
    - G: # of grid cells per axis
    - O: Max # of showers per cell
    - For each shower: x, y, E is given
    - Missing showers have x,y,E = 0
- Example: 5x5x3x3

Loss Function

30x30 input image

Magic Pile of Linear Algebra

Flatten

Fully Connected
G^2*O*3 neurons
Tanh activation

Reshape
GxGxOx3 tensor
Shift x,y,E ranges

GxGxOx3 output tensor prediction

Compare

GxGxOx3 truth tensor

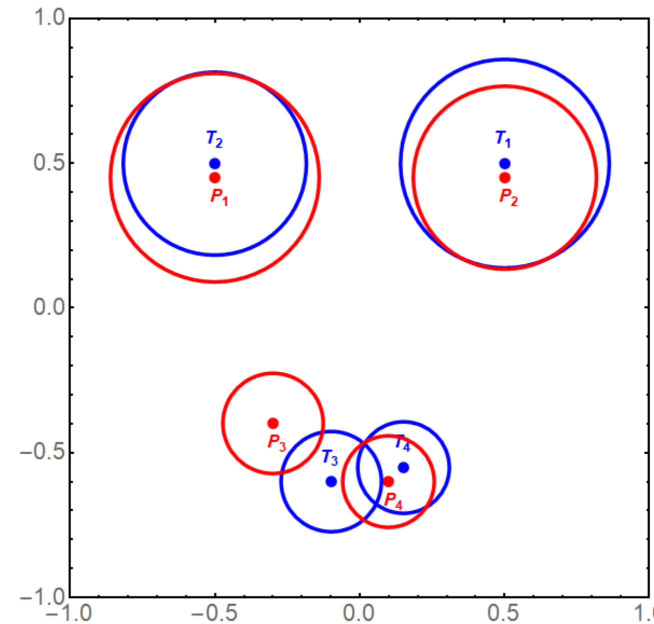
Loss function

- Problem: For each cell, how to define ordering of overlapping showers?
- Known as the Assignment Problem

Predicted

True	P_1	P_2	P_3	P_4
T_1	1.0	0.3	1.6	1.6
T_2	0.3	1.0	1.2	1.5
T_3	1.5	1.4	0.3	0.2
T_4	1.6	1.3	0.5	0.1

Distance Matrix



Loss Function

30x30 input image

Magic Pile of Linear Algebra

Flatten

Fully Connected
G^2*O*3 neurons
Tanh activation

Reshape
GxGxOx3 tensor
Shift x,y,E ranges

GxGxOx3 output tensor prediction

Compare

GxGxOx3 truth tensor

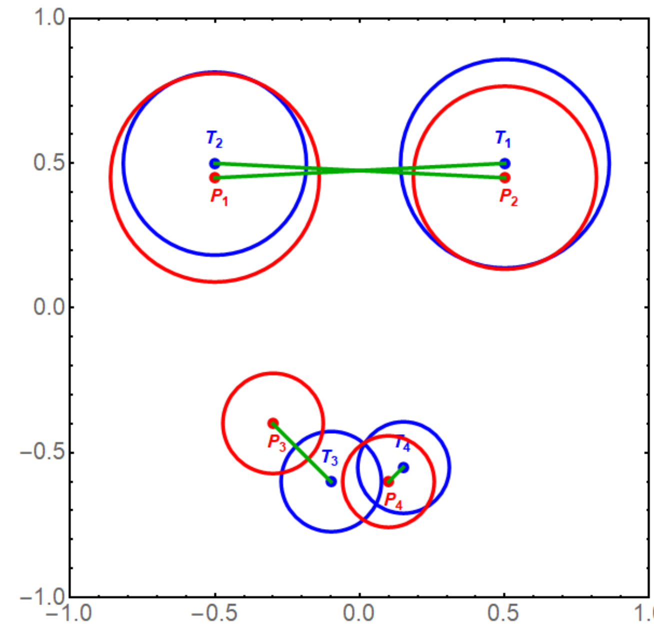
Loss function

- Problem: For each cell, how to define ordering of overlapping showers?
- Known as the Assignment Problem
- Just sort by energy...

Predicted

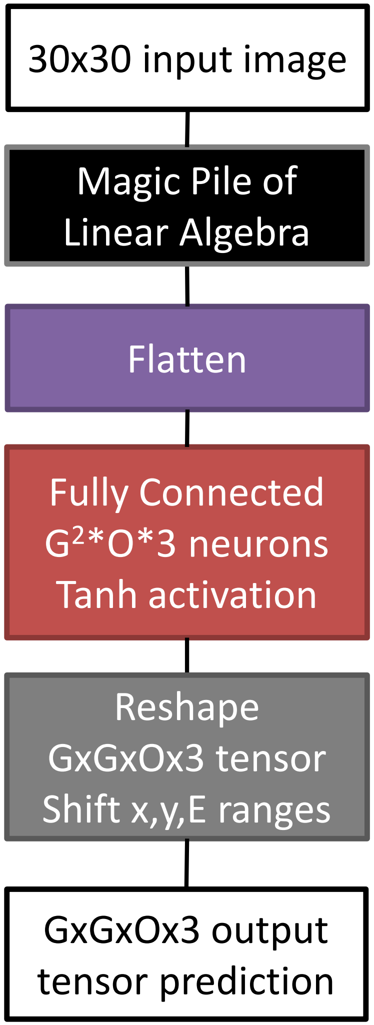
True	P_1	P_2	P_3	P_4
T_1	1.0	0.3	1.6	1.6
T_2	0.3	1.0	1.2	1.5
T_3	1.5	1.4	0.3	0.2
T_4	1.6	1.3	0.5	0.1

Distance Matrix



Total Distance: 2.4

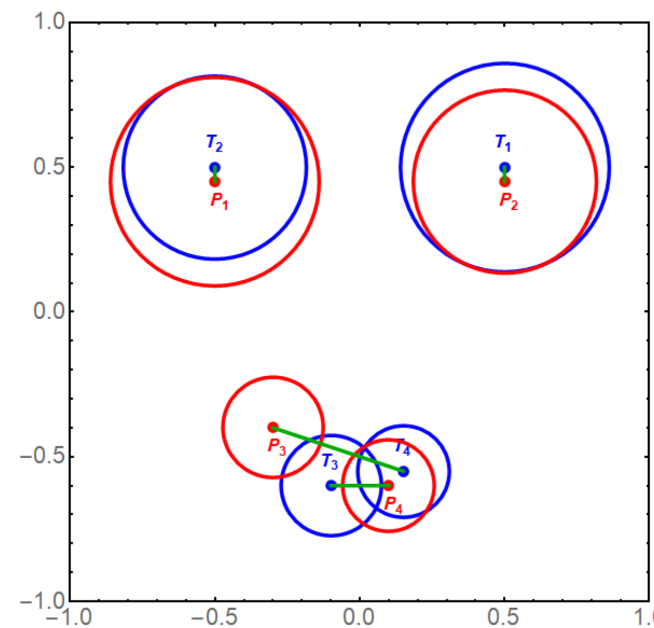
Loss Function



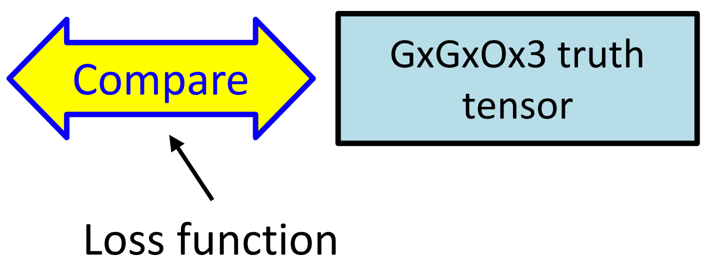
- Problem: For each cell, how to define ordering of overlapping showers?
- Known as the Assignment Problem
- Just take the closest match for each true cluster…

True \ Predicted	P_1	P_2	P_3	P_4
T_1	1.0	0.3	1.6	1.6
T_2	0.3	1.0	1.2	1.5
T_3	1.5	1.4	0.3	0.2
T_4	1.6	1.3	0.5	0.1

Distance Matrix



Total Distance: 1.3



Loss Function

30x30 input image

Magic Pile of Linear Algebra

Flatten

Fully Connected
$G^2 * O * 3$ neurons
Tanh activation

Reshape
GxGxOx3 tensor
Shift x,y,E ranges

GxGxOx3 output tensor prediction

Compare

GxGxOx3 truth tensor

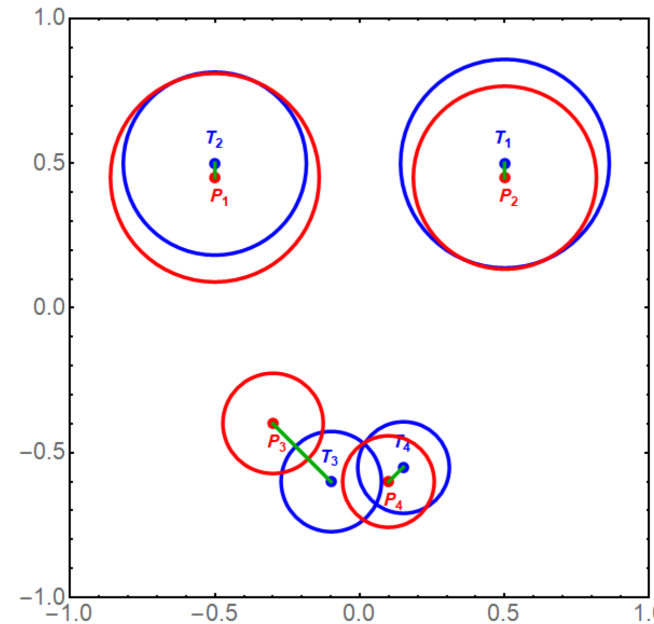
Loss function

- Problem: For each cell, how to define ordering of overlapping showers?
- Known as the Assignment Problem
- Optimal solution: Hungarian Algorithm

Predicted

True	P_1	P_2	P_3	P_4
T_1	1.0	0.3	1.6	1.6
T_2	0.3	1.0	1.2	1.5
T_3	1.5	1.4	0.3	0.2
T_4	1.6	1.3	0.5	0.1

Distance Matrix



Total Distance: 1.0

More Loss Function

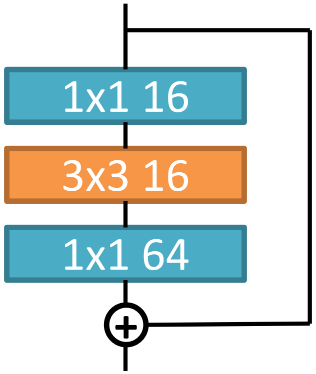
- What to do with "empty" tensor elements:
  - Both true and pred: Matched showers
  - Pred but no true: Ghost prediction
  - True but no pred: Missing prediction

$$Loss(y_{pred}, y_{true}) = \lambda_{match} < (y_{pred}^{match} - y_{true}^{match})^2 > +$$
$$\lambda_{miss} < (E_{pred}^{miss} - E_{true}^{miss})^2 > + \lambda_{ghost} < (E_{pred}^{ghost})^2 > +$$
$$\lambda_{all} \times < N_{ghost} > \times < N_{miss} > \times < (y_{pred} - y_{true})^2 >$$

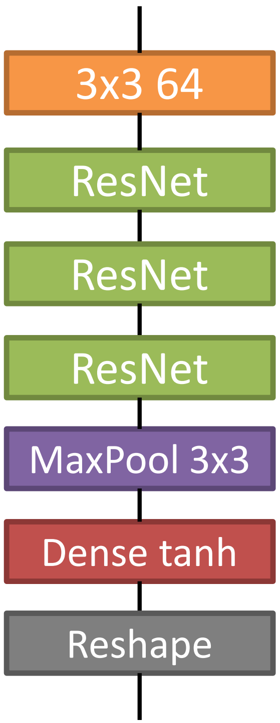
- λ's are hyperparameters to be tuned
- Importance weighting for ghosts and misses
- Not clear whether last term is needed, but it may help in the beginning of training

Some Results

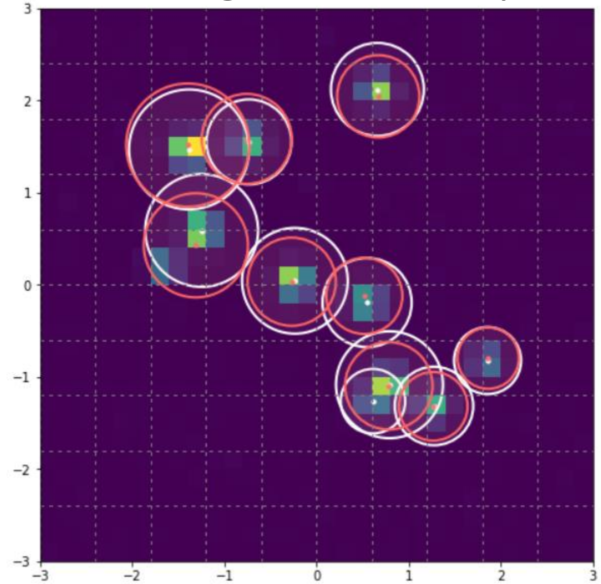
ResNet Block



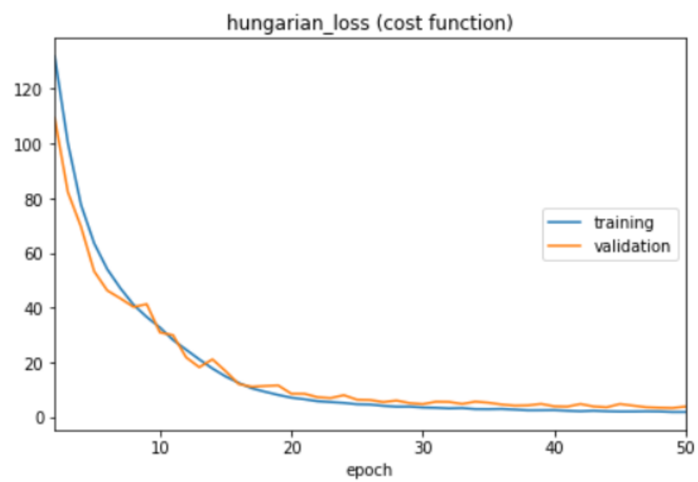
Test Architecture



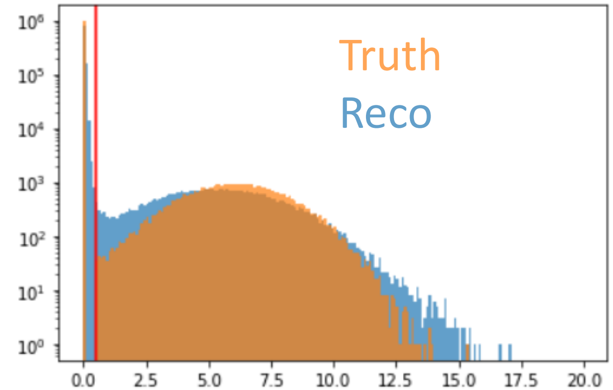
10x10 grid, no overlap

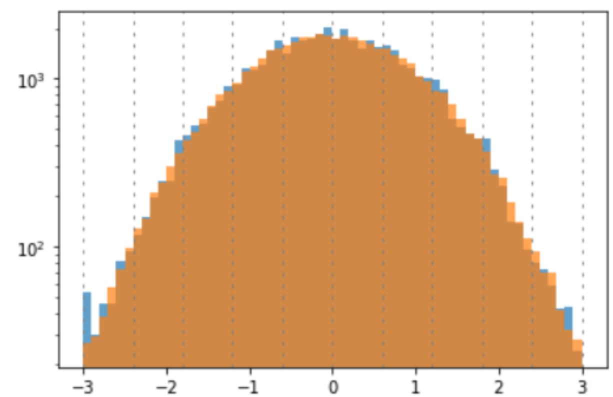


Training on 8k images



Energy Distribution



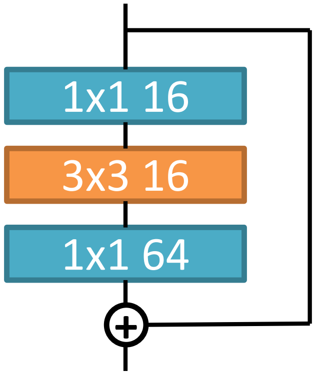


X Distribution

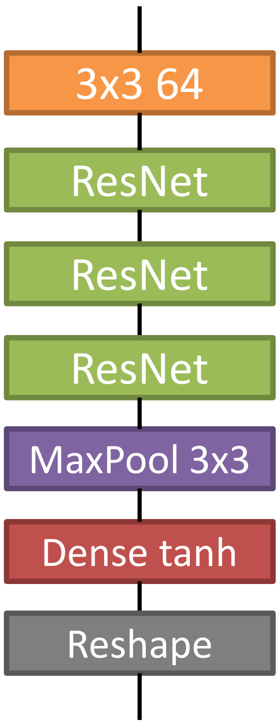
Averages
0.07 ghosts / image
0.06 missed / image

Some Results

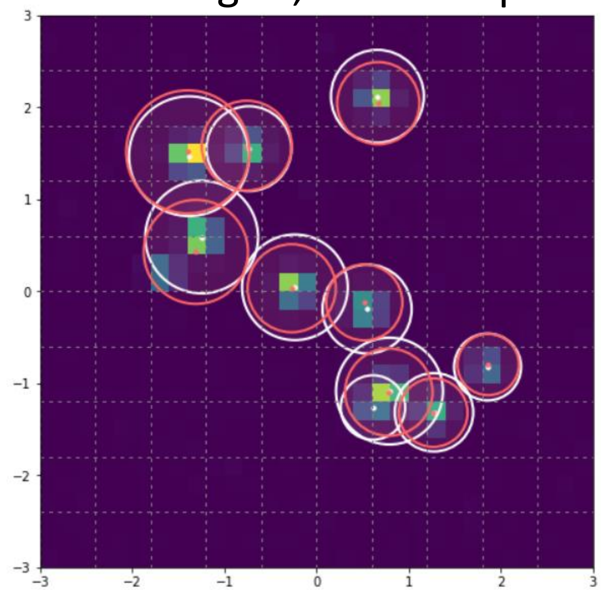
ResNet Block



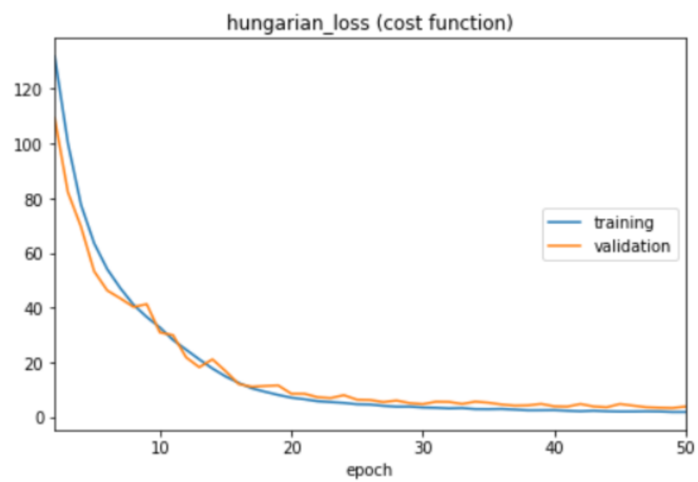
Test Architecture



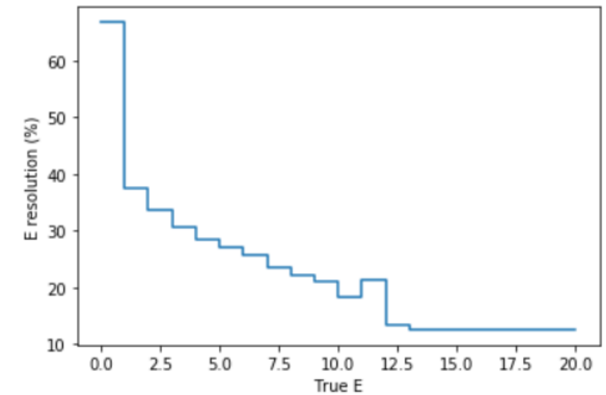
10x10 grid, no overlap

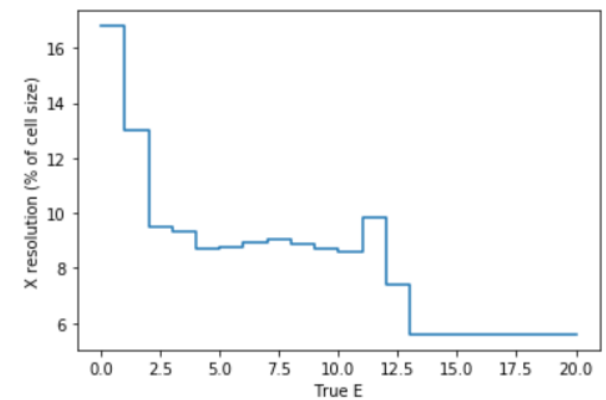


Training on 8k images



Energy Resolution





X Resolution

Averages

0.07 ghosts / image

0.06 missed / image

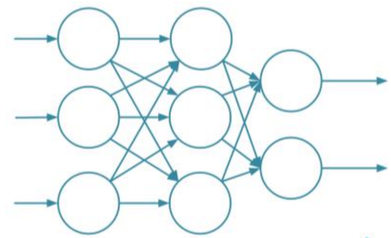
Other Ideas...

30x30 input image

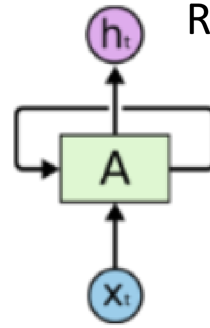
Magic Pile of Linear Algebra

Nx3 output tensor prediction

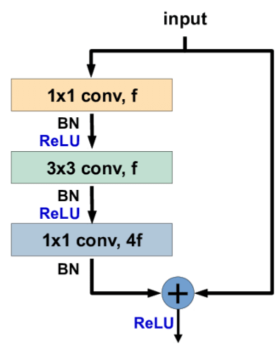
Fully Connected NN



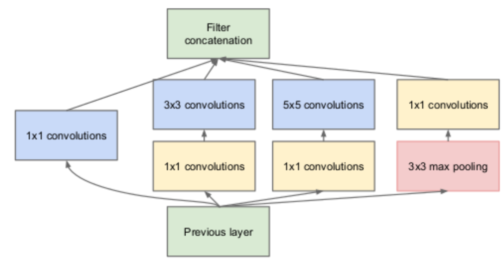
RNN

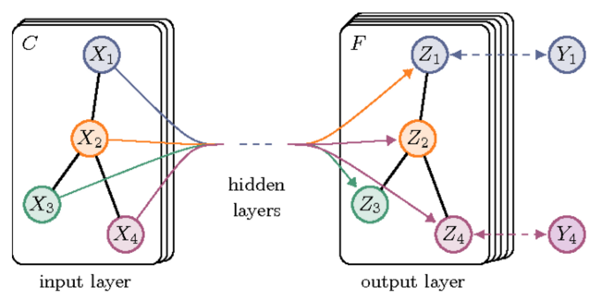


Going Deeper



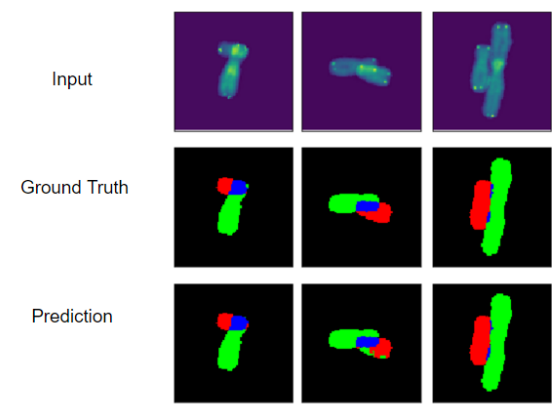
Convolutional NN





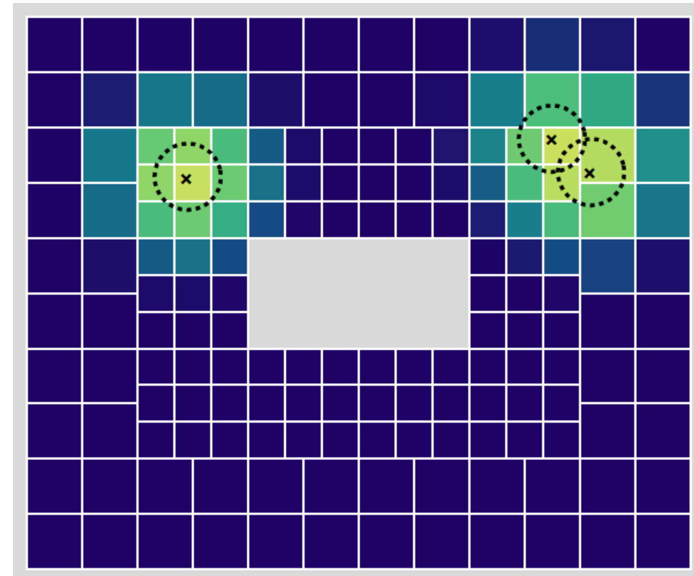
Graph NN

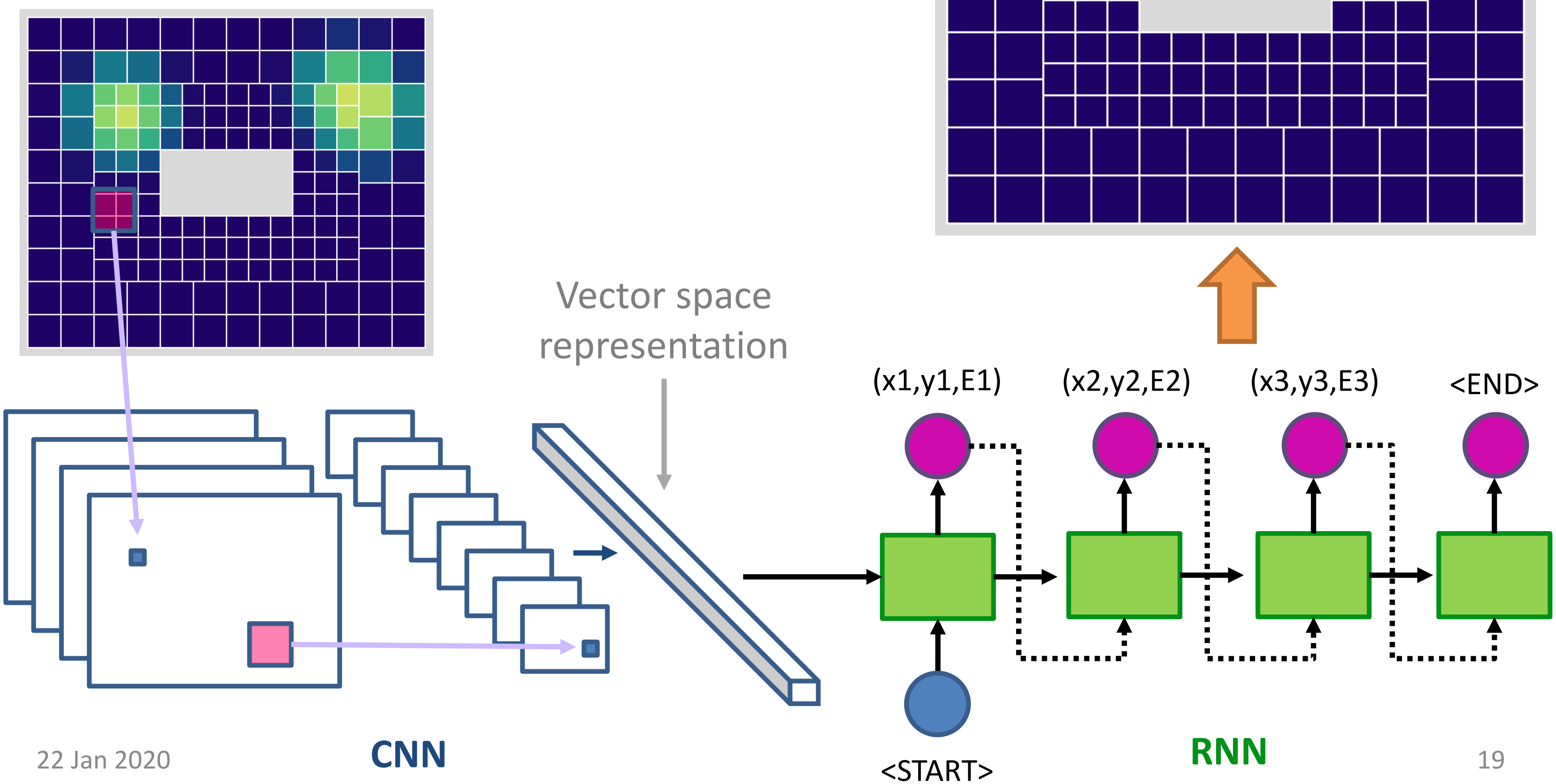
Semantic Segmentation



Calo “Captioning”?

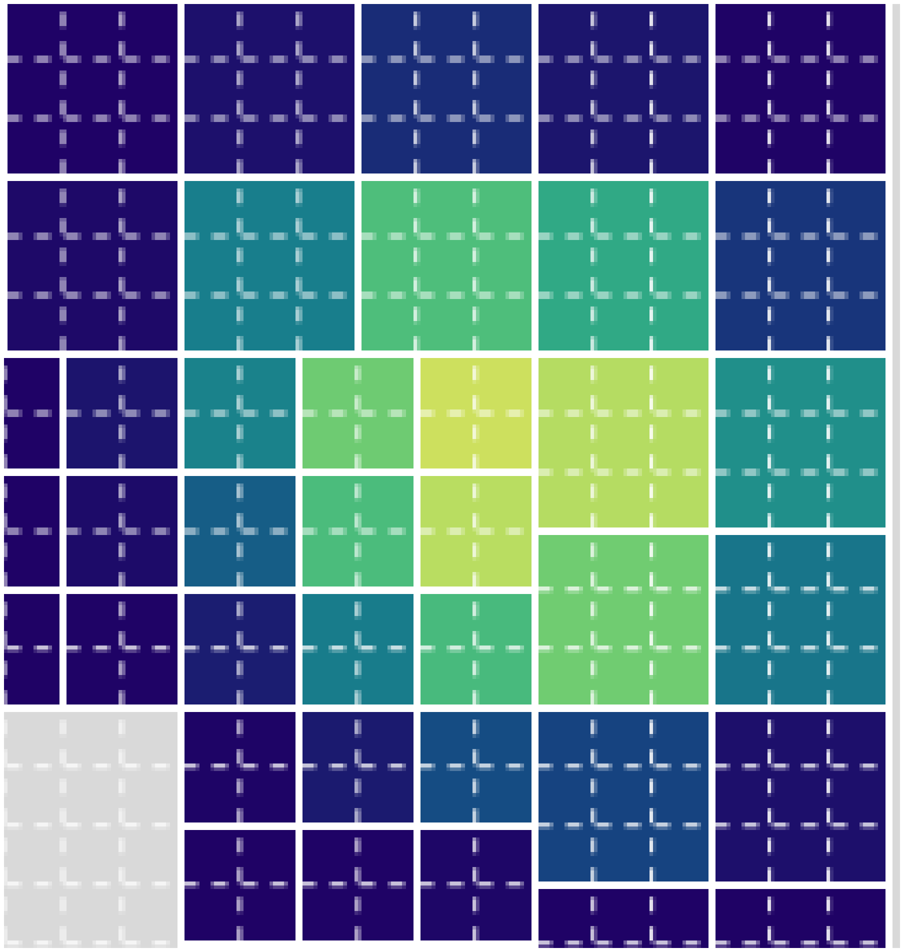
- Idea:
  - Encode calo image into vector space
  - Feed into RNN to output sequence of shower positions and energies





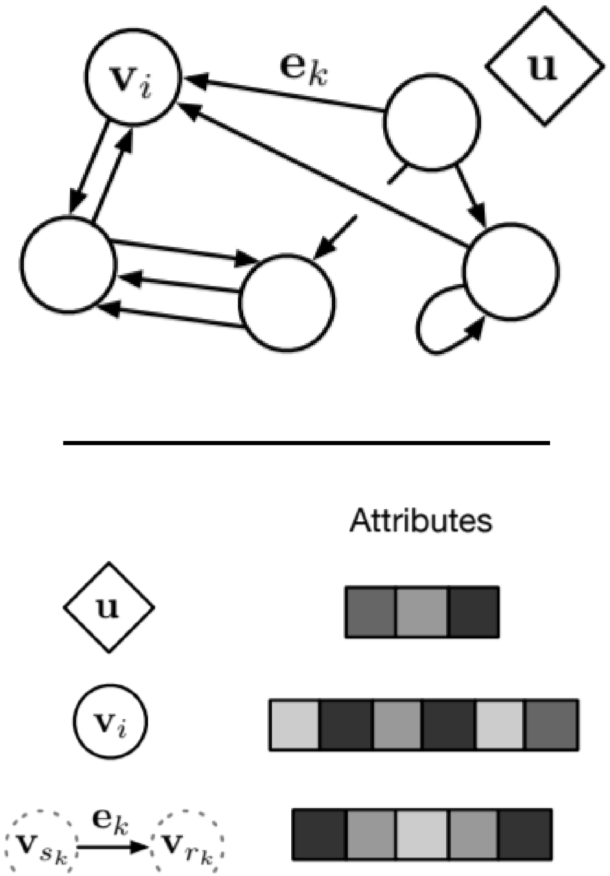
Irregular Geometry Issue

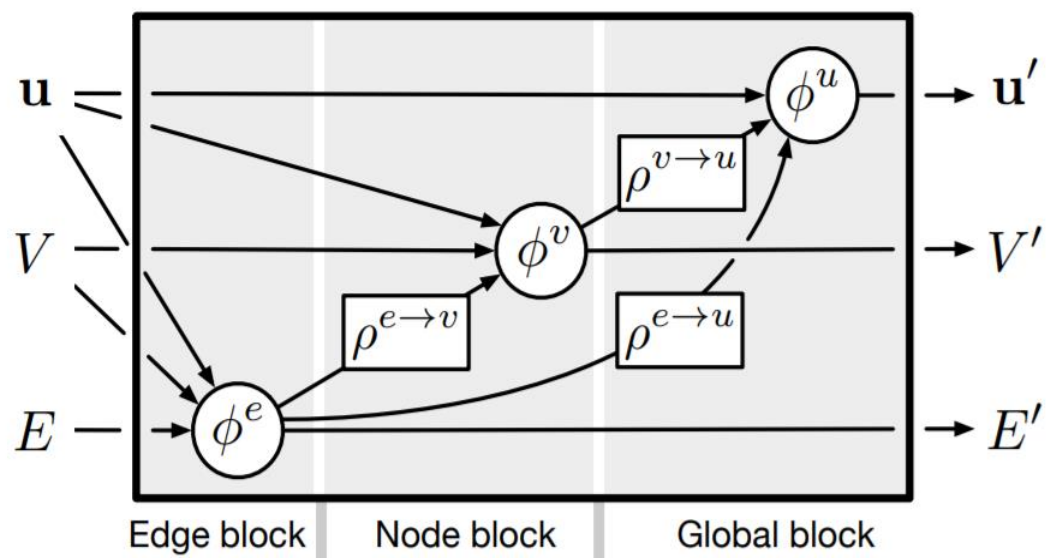
- Calo cells are not all identical
- Coarser granularity on outside cells
- Breaks translation invariance
- How do we apply a kernel in this scenario?
    - Downsampling:
        - Merge all cells in a module?
        - Clearly would degrade resolution
    - Upsampling:
        - Divide every cell into smaller 2x2 cm^2 pixels
        - Predict charge in each pixel
        - Uniform distribution over cell?
        - Deep learned upsampling?



Graph Neural Networks?

- One alternative is to scrap the CNN approach and move into a GNN approach

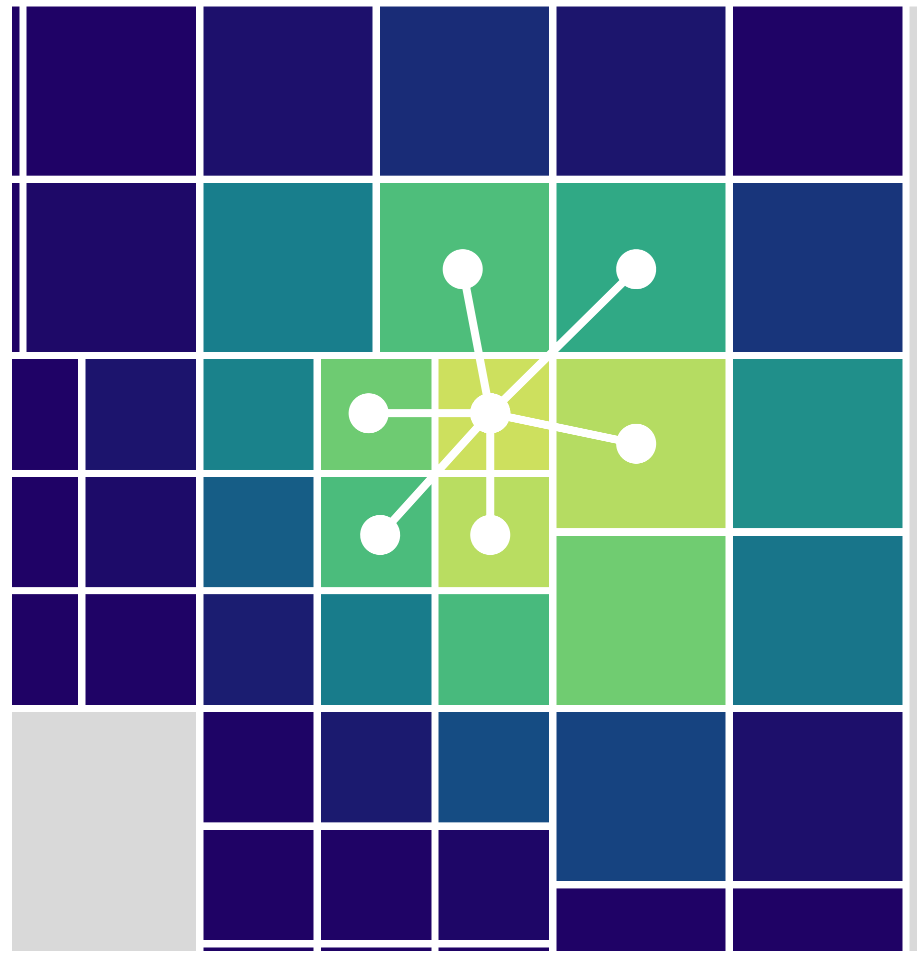




https://arxiv.org/abs/1806.01261

Graph Neural Networks?

- One alternative is to scrap the CNN approach and move into a GNN approach
- Treat each cell as a node in a graph
- Edges connect cells to neighbors characterizing their distance
- Fully connected?
    - Global attributes could encode full graph as input to RNN
- Focus on local graphs?
    - Could be used for upsampling in combination with a CNN



- Many ideas to be explored
- Developing into a student project proposal with the Cambridge group

Summary

- We are exploring deep learning solutions to calorimetry
- Initial tests on easy task gives ok performance
- CNN architecture better than simple NN as expected
- On single GPU (Google Colab), runs at ~7kHz
- Many tests still possible on different datasets:
    - Increase noise
    - Increase pile-up
    - Change energy and position distributions
    - Irregular pixel size (LHCb-like)
- Eventually run this on actual LHCb MC
- Also looking to explore solutions other than CNN