Faster Dynamic GNNs with GPUs

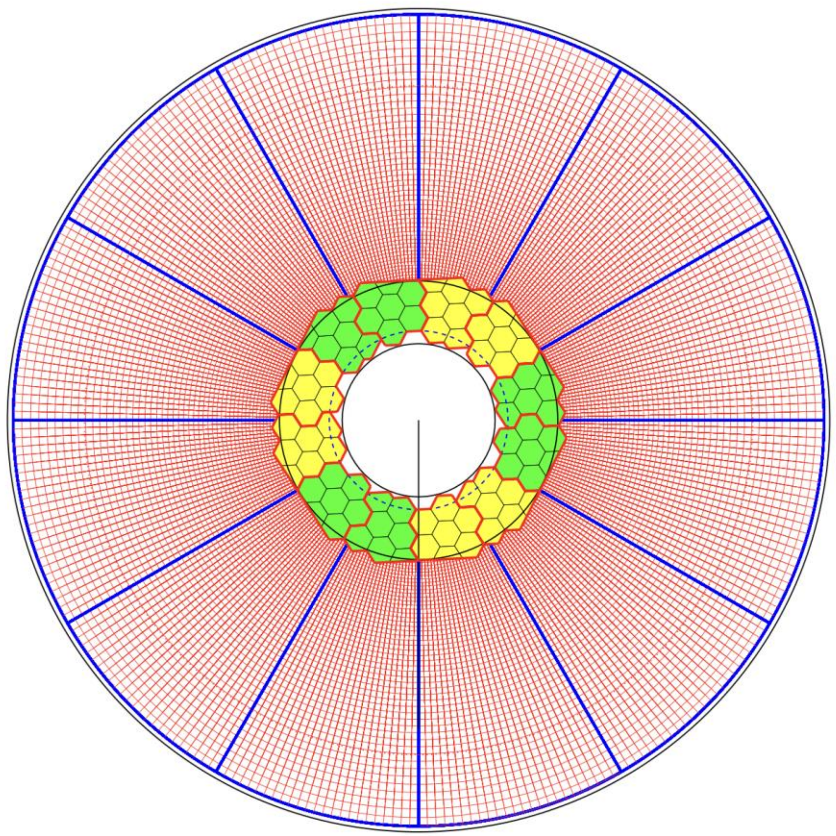
Shah Rukh Qasim (University of Zurich)
Jan Kieseler (Karlsruhe Institute of Technology)
jan.kieseler@cern.ch

Fabian Riemers (Karlsruhe Institute of Technology)
Ayman Ratey (Karlsruhe Institute of Technology)
Nico Serra (University of Zurich)

24.10.2024

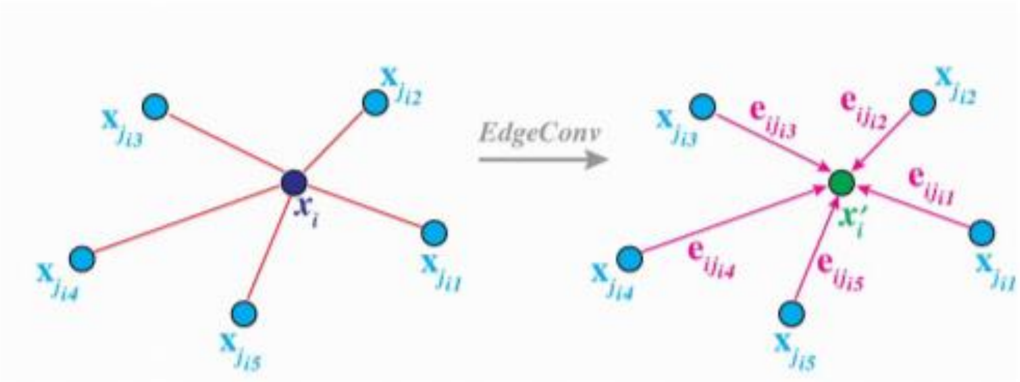
GNNs for reconstruction and other tasks

- Challenges considering more traditional machine learning approaches
  - Sparse data
  - Non-regular shape
- GNNs are a natural fit
  - We don’t have a graph structure but only a point cloud
  - So we learn the graph structure with what we call dynamic GNNs

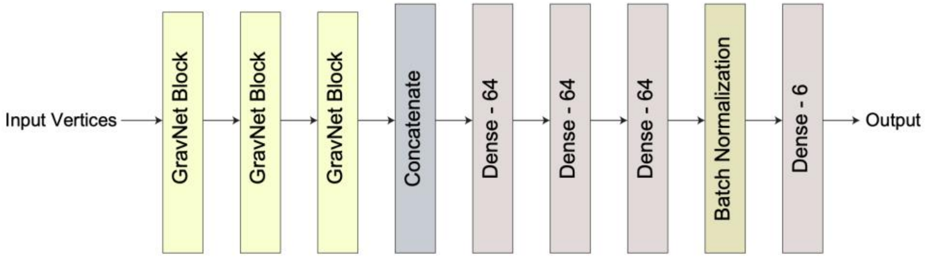


Dynamic GNNs

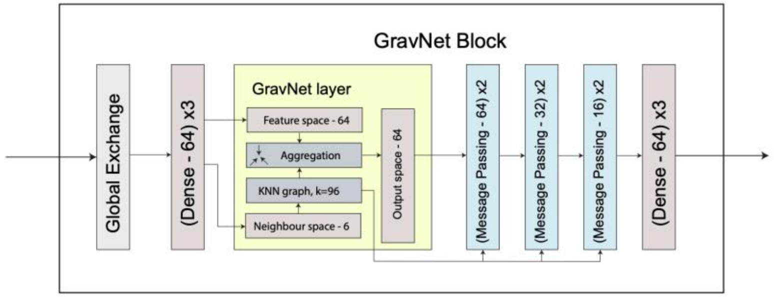
- Feature vector for every point
- You apply a neural network to feature vectors of every point to get feature space F_s
- Build a graph as KNN of every node using euclidean distance in F_s
  - So you have N*K edges
  - Perform message passing with permutation invariant aggregation functions
  - Dynamic GNN:
    - The KNN graph isn't built in the original space but in a learned feature space

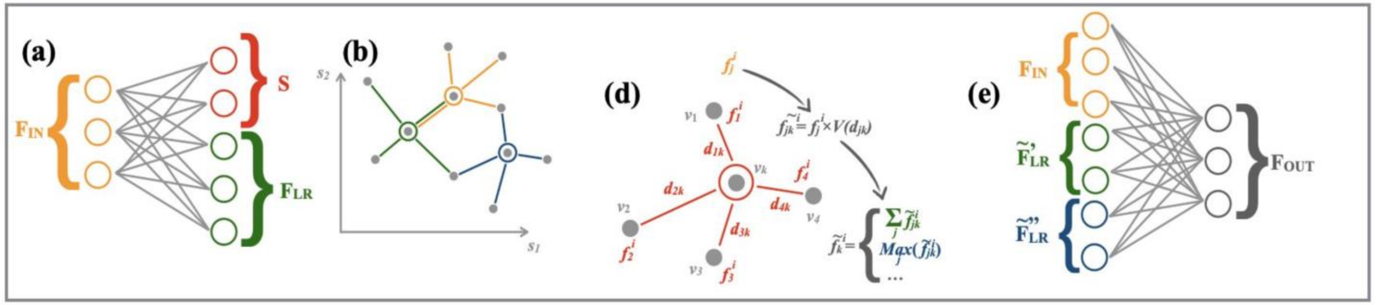


Dynamic GNNs



- One is DGCNN [2]
- GravNet has similar performance but is muc faster





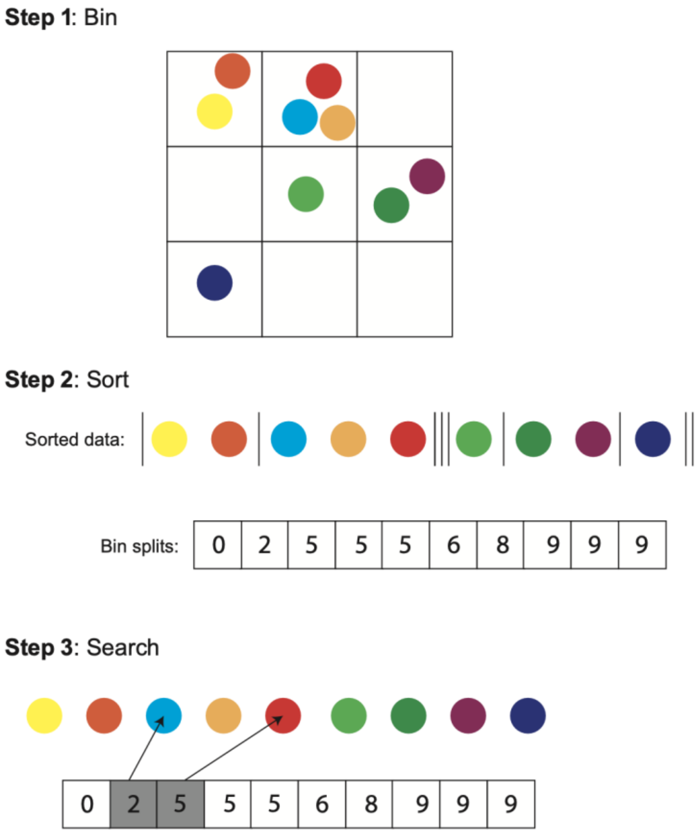
[1] arXiv:1902.07987
[2] arXiv:1801.07829

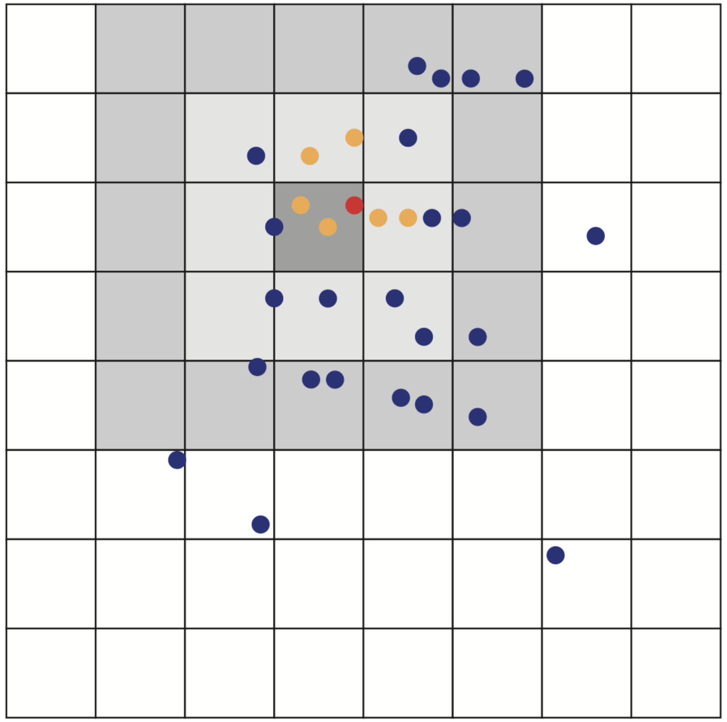
GNNs are expensive

- The CMS HGCAL can have up to 200k nodes
- Number of hits in trackers are also high
- KNN is the bottleneck
  - Among some others

Binning

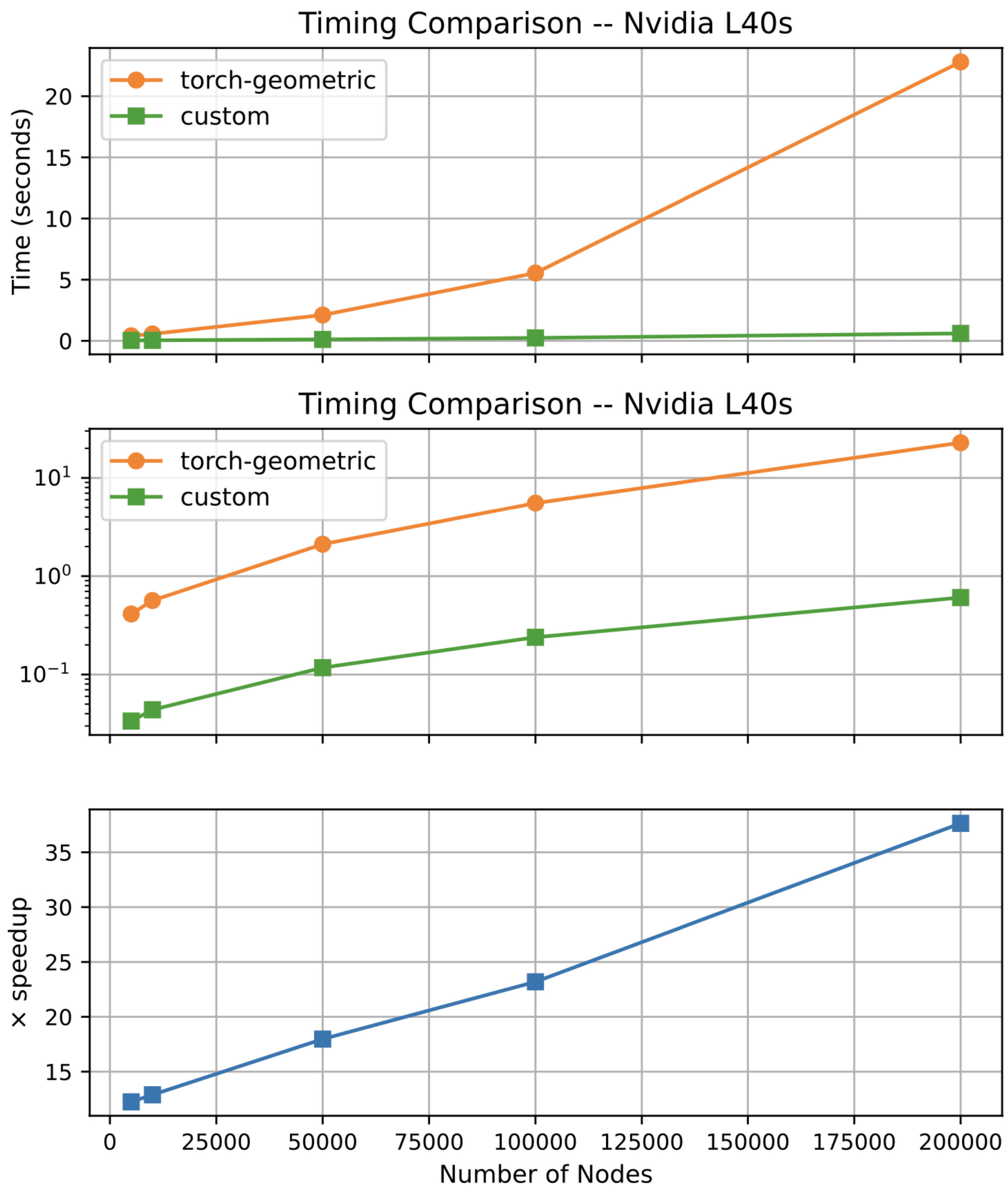
- Implemented in CUDA for PyTorch
- We bin
- And search in onion layers
- Keeping in mind the distance to keep the kNN exact (not approximate)





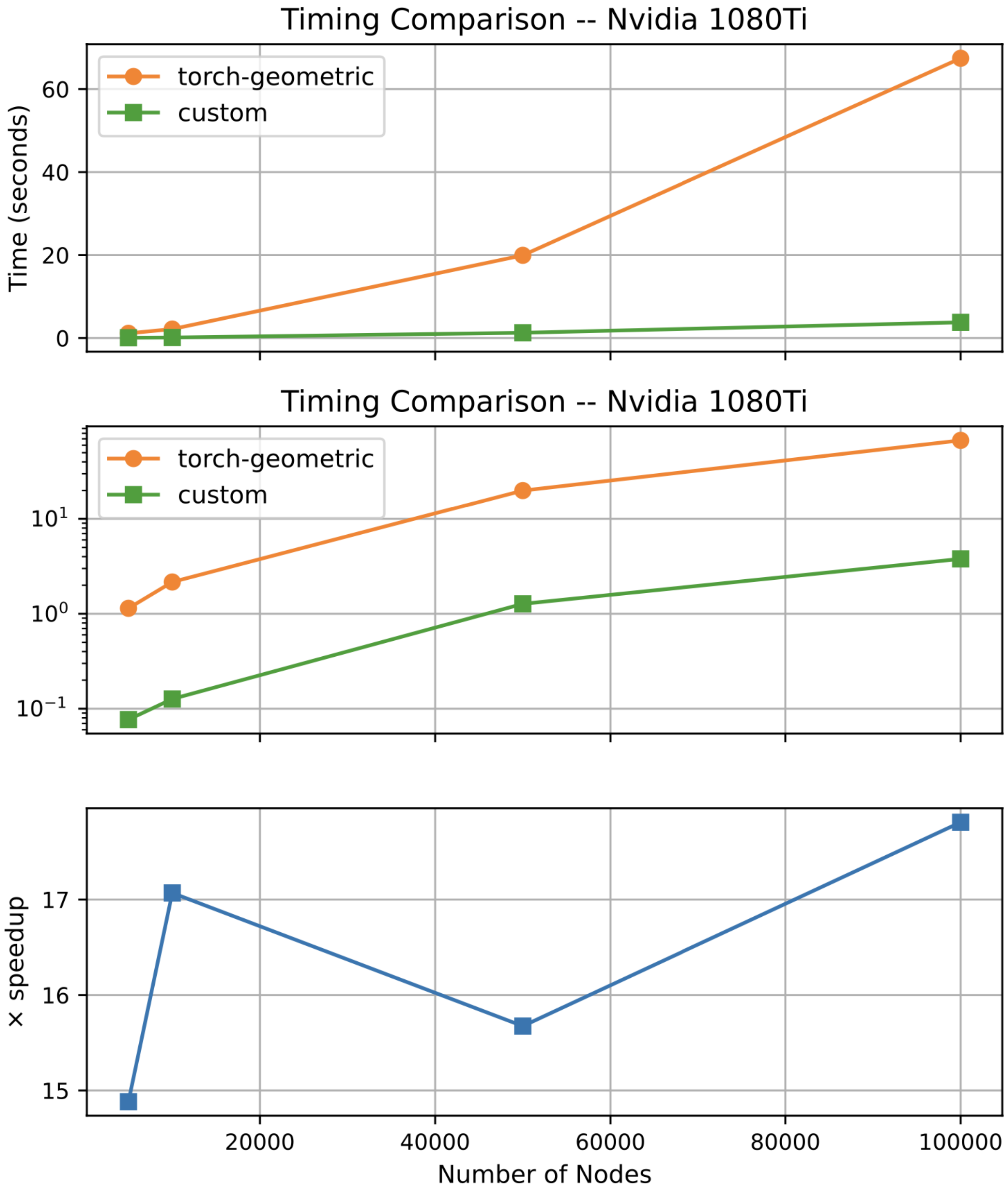
Results

- Inference times of a GravNet model
- DGCNN will be a few times more expensive
  - But much more reasonable
- Nvidia L40s
- No grad
  - But grad overhead is rather minimal

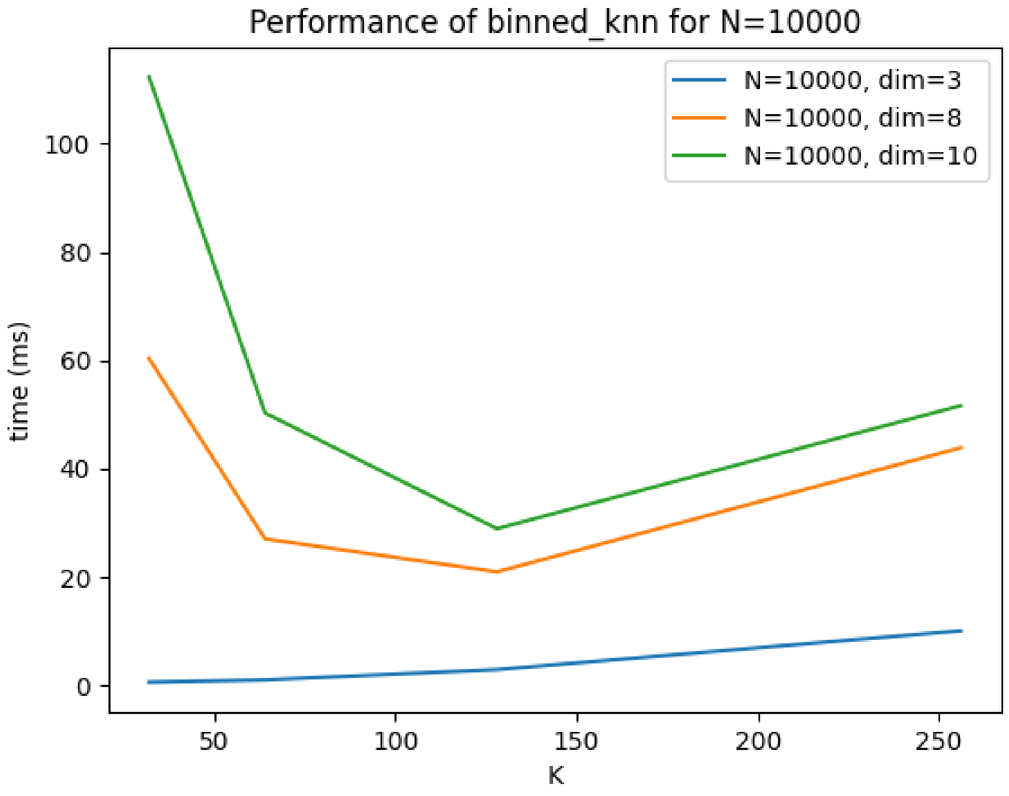


Results

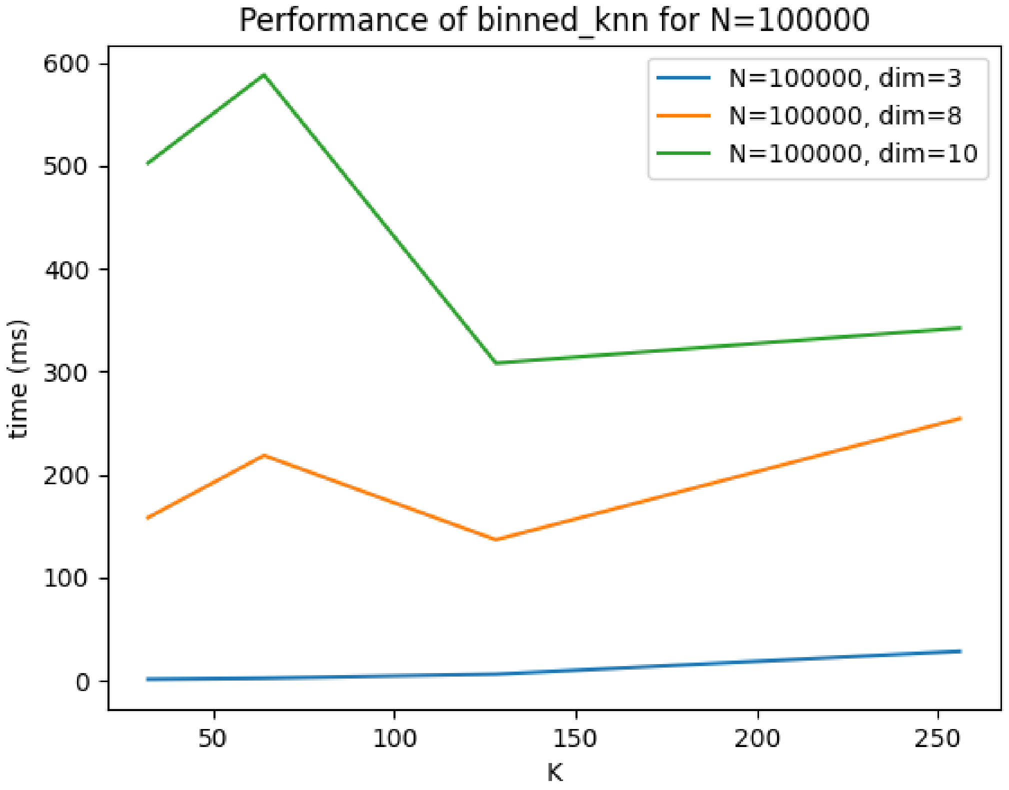
- Inference times of a GravNet model
- DGCNN will be a few times more expensive
  - But much more reasonable
- Nvidia 1080Ti
- No grad
  - But grad overhead is rather minimal



Sole KNN



Performance of binned_knn for N=10000
N=10000, dim=3
N=10000, dim=8
N=10000, dim=10
time (ms)
0
20
40
60
80
100
50
100
150
200
250
K



Performance of binned_knn for N=100000
N=100000, dim=3
N=100000, dim=8
N=100000, dim=10
time (ms)
0
100
200
300
400
500
600
50
100
150
200
250
K

Conclusion

- We want to finish a few more tests
  - Check its working outside our container
  - Add DGCNN
  - Maybe choose a different name
  - To be full ready with the proceedings / paper
- But if you are in a hurry, it is actually ready
  - Give us the feedback too
- Find here:
  - https://github.com/jkiesele/ml4reco_modules
- In the longer run, we'll try to put it into `pytorch / pyg / torch-cluster` so it remains sustainably maintainable

Backup slides

Backup slides

Some more information about the model

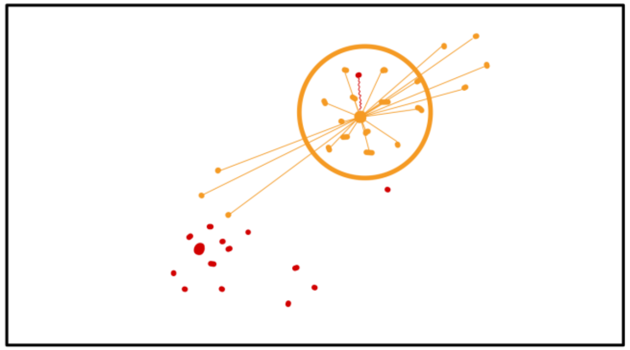
- 3 dense nets in every gravnet block too

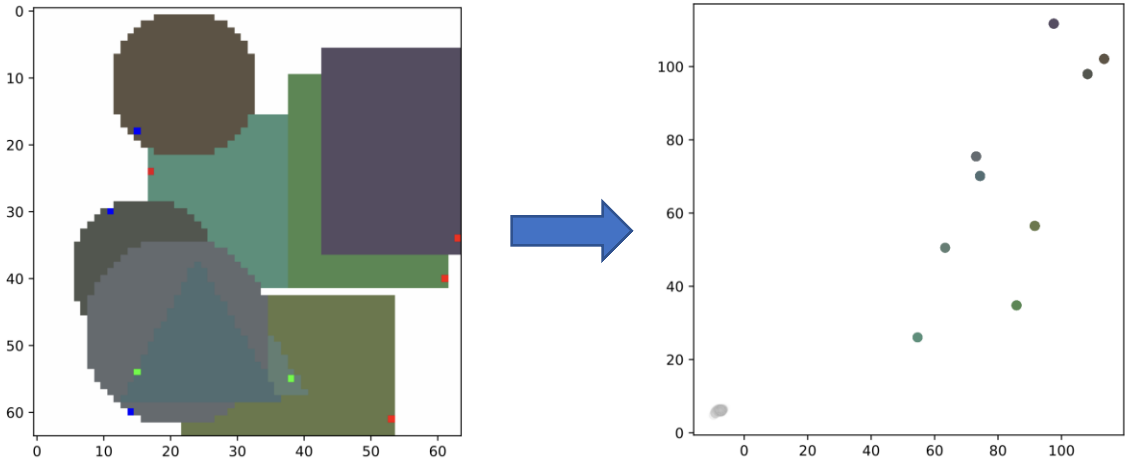
```
in_dim = 64       # Input dimension
prop_dim = 32     # Propagation dimension
space_dim = 4     # Space dimension
k = 64            # Number of neighbors to consider
hidden_dim = 128  # Dimension for the dense layers
output_dim = 10   # Output dimension (you can adjust
according to the task)
```

Multi particle reconstruction

[3] arXiv:2002.03605

- Object condensation [3]
  - Single shot clustering and property-prediction
  - Not exactly the same as contrastive learning
- Learn a clustering space
  - Hits belonging to the same shower are learned to be clustered together in the clustering space
  - For every shower, we also learn a representative hit
  - All object properties are aggregated in this representative hit
- Use attractive and repulsive potentials in the loss

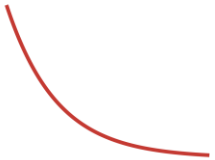




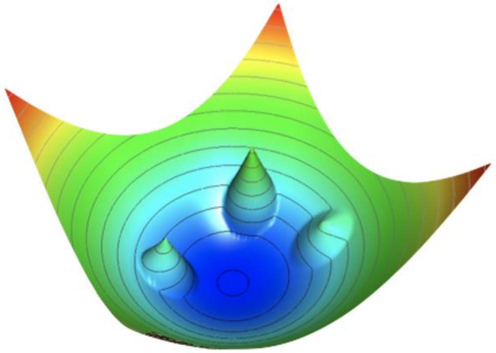
Object condensation

1. $$\check{V}_t(h) = q_{\alpha_t} w_t \ln\left(e \cdot \left(\frac{\|x_h - \overline{x}(t)\|^2}{2\overline{\varphi}_t^2 + \epsilon}\right) + 1\right)$$

2. $$\hat{V}_t(h) = q_{\alpha_t} w_t \cdot \exp\left(-\frac{\|x_h - \overline{x}(t)\|^2}{2\overline{\varphi}_t^2 + \epsilon}\right)$$



3. $$L_V = \frac{1}{|T|}\sum_{t\in T}\left(\frac{1}{|H_t|}\sum_{h\in H_t} q_h \check{V}_t(h) + \frac{1}{|H - H_t|}\sum_{h\in (H-H_t)} q_h \hat{V}_t(h)\right)$$



4. $$L_\beta = \frac{1}{|T|}\sum_{t\in T}(1 - \beta_{\alpha_t}) + s_B \frac{1}{|H_\circ|}\sum_{h\in H_\circ}\beta_h$$

GravNet

$$S = \mathcal{D}_*(X_n)\ ,$$

$$F = \mathcal{D}_*(X_n)\ .$$

$$D = \{\cup\{(x, h, d(x, y))\,\forall x \in \mathrm{KNN}(h)\}\}\ .$$

$$d(x, h) = \exp\left(-\|S_x - S_h\|^2\right)$$

$$G_h = F_h - \mathtt{mean}(\{d(x, h)\underline{F_x\ \forall x \in \mathcal{N}}(h)\})\ ,$$

$$I_h = F_h - \mathtt{max}(\{d(x, h)\underline{F_x\ \forall x \in \mathcal{N}}(h)\})\ .$$