ML tracking

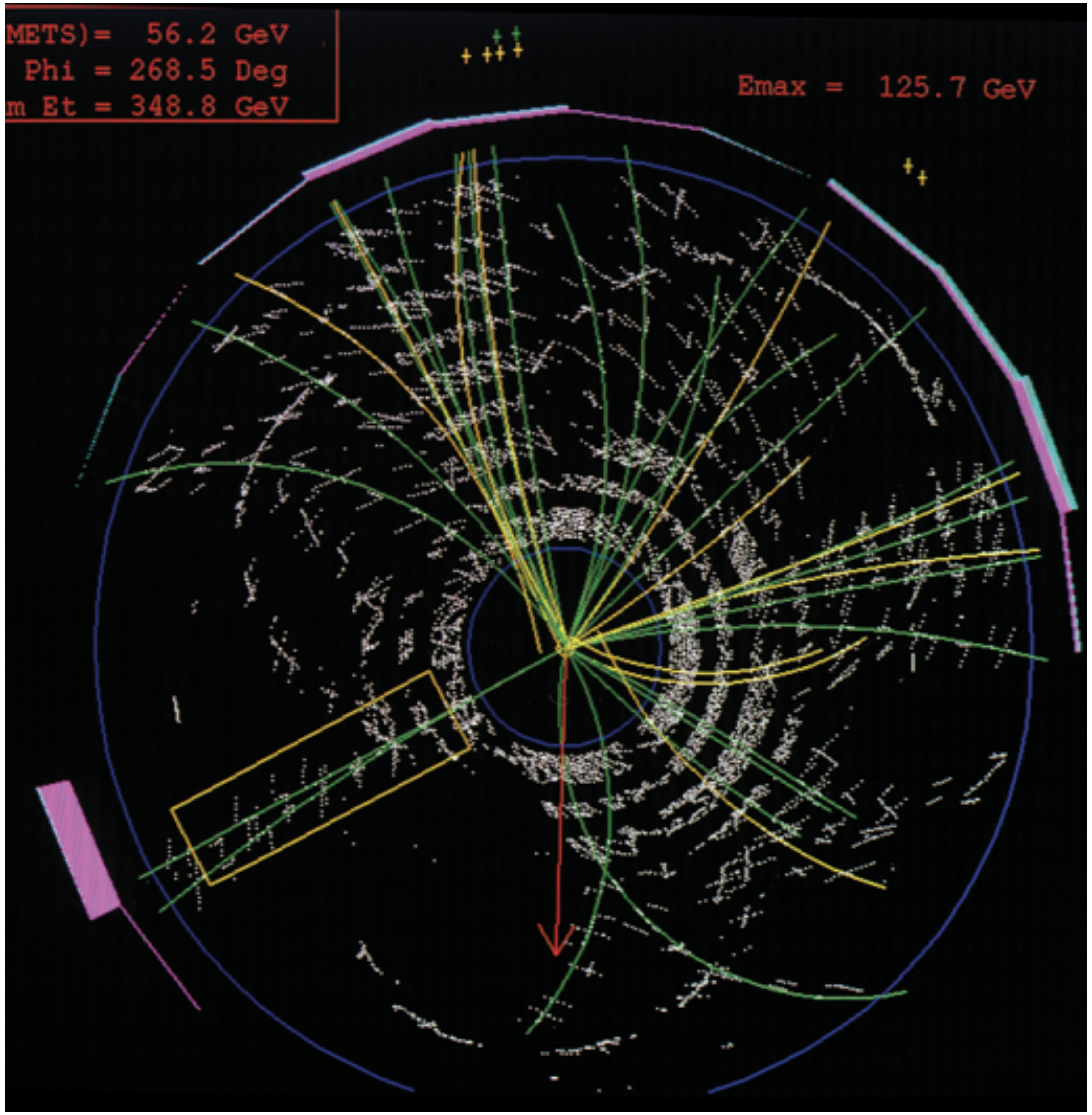
IRIS-HEP fellowship project presentation

Viacheslav Kucherenko, Kyiv Academic University
Mentor: Dantong Yu, New Jersey Institute of Technology

28.09.2022

Agenda

Project description
Graph Neural Networks
Data analysis
Data preprocessing
Train overview
Results
Acknowledgment

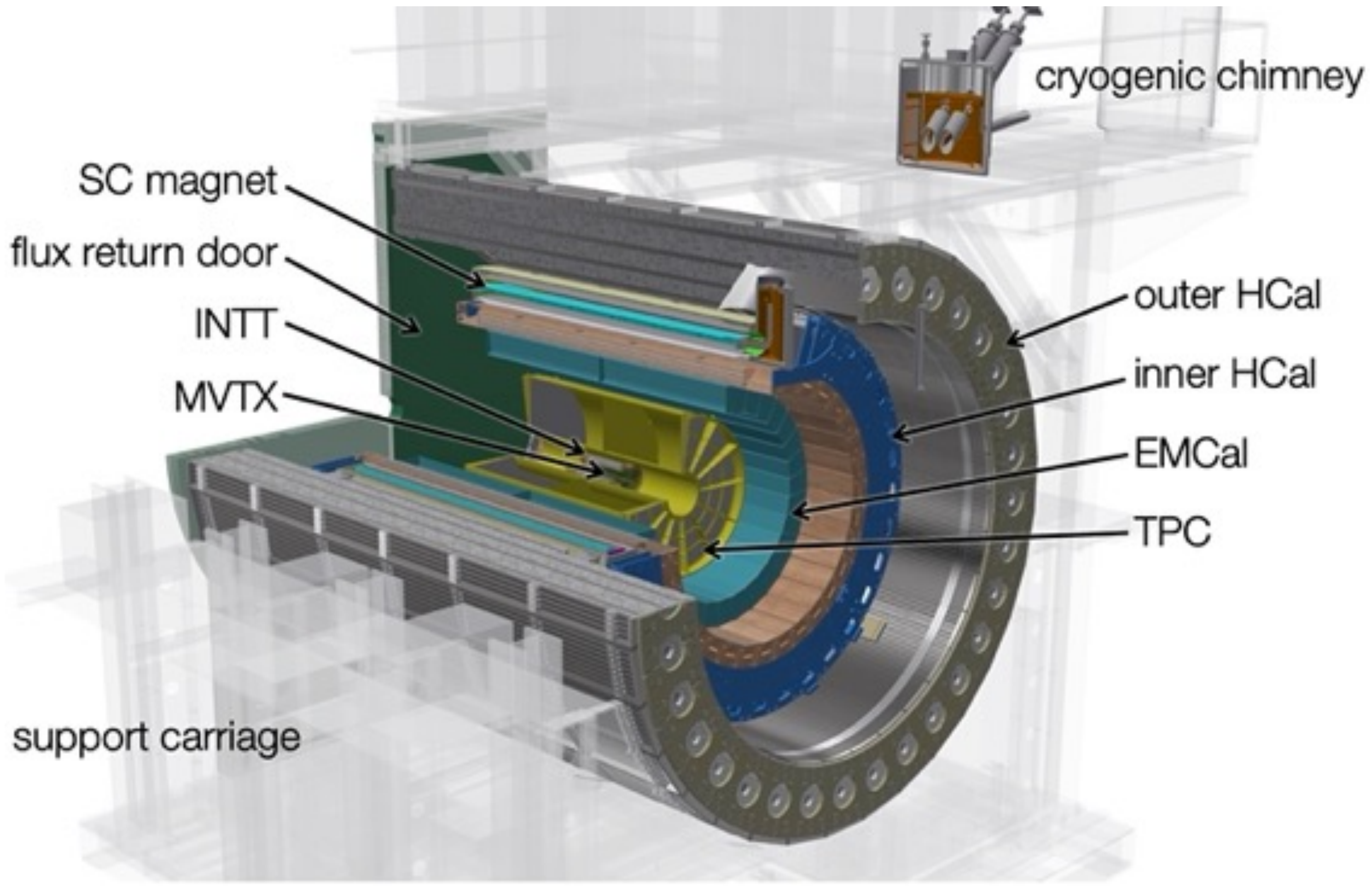


Project description

Particles collision data

Simulated by sPHENIX project

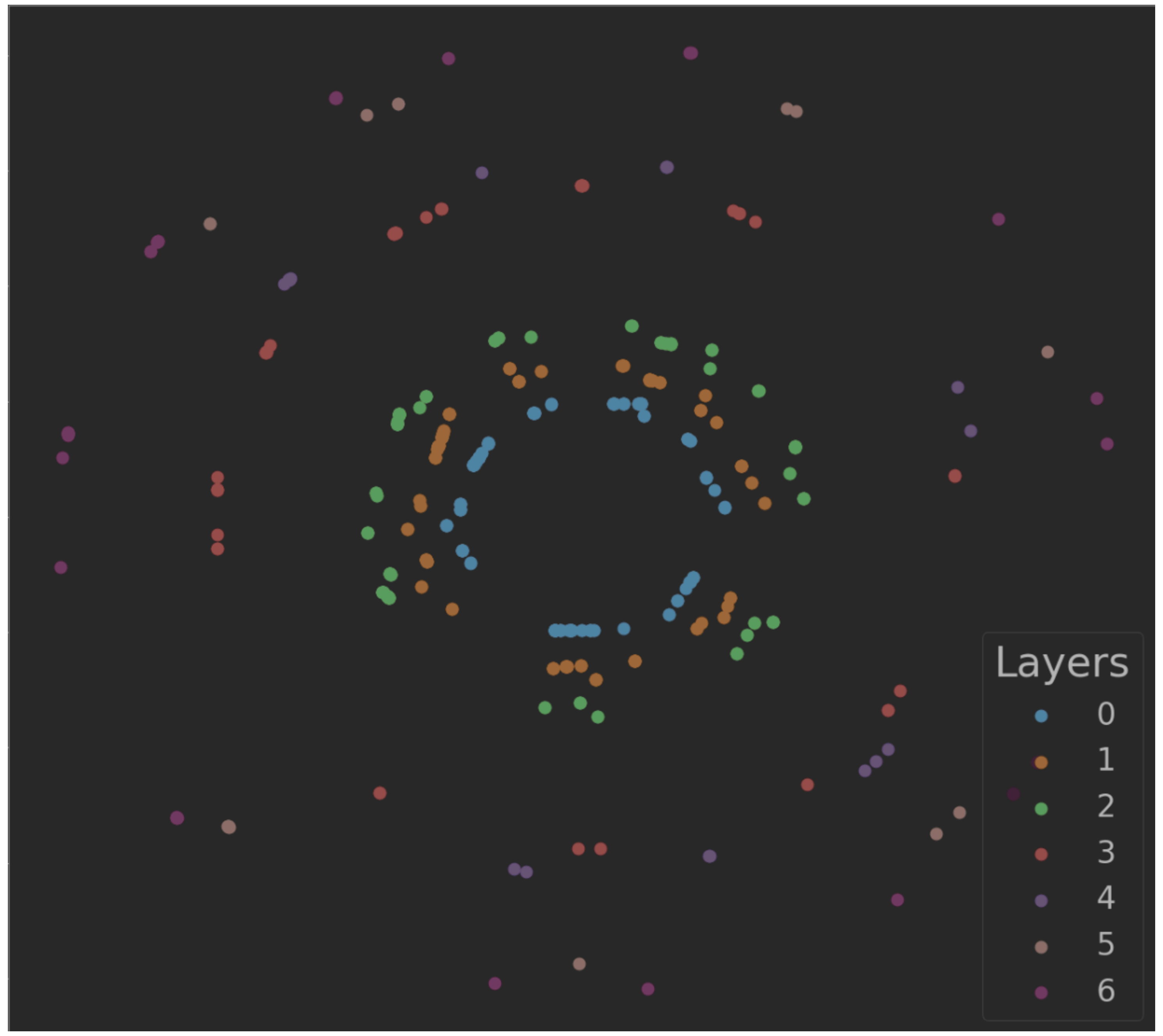
- After beams' collision
- Detection
- Analysis

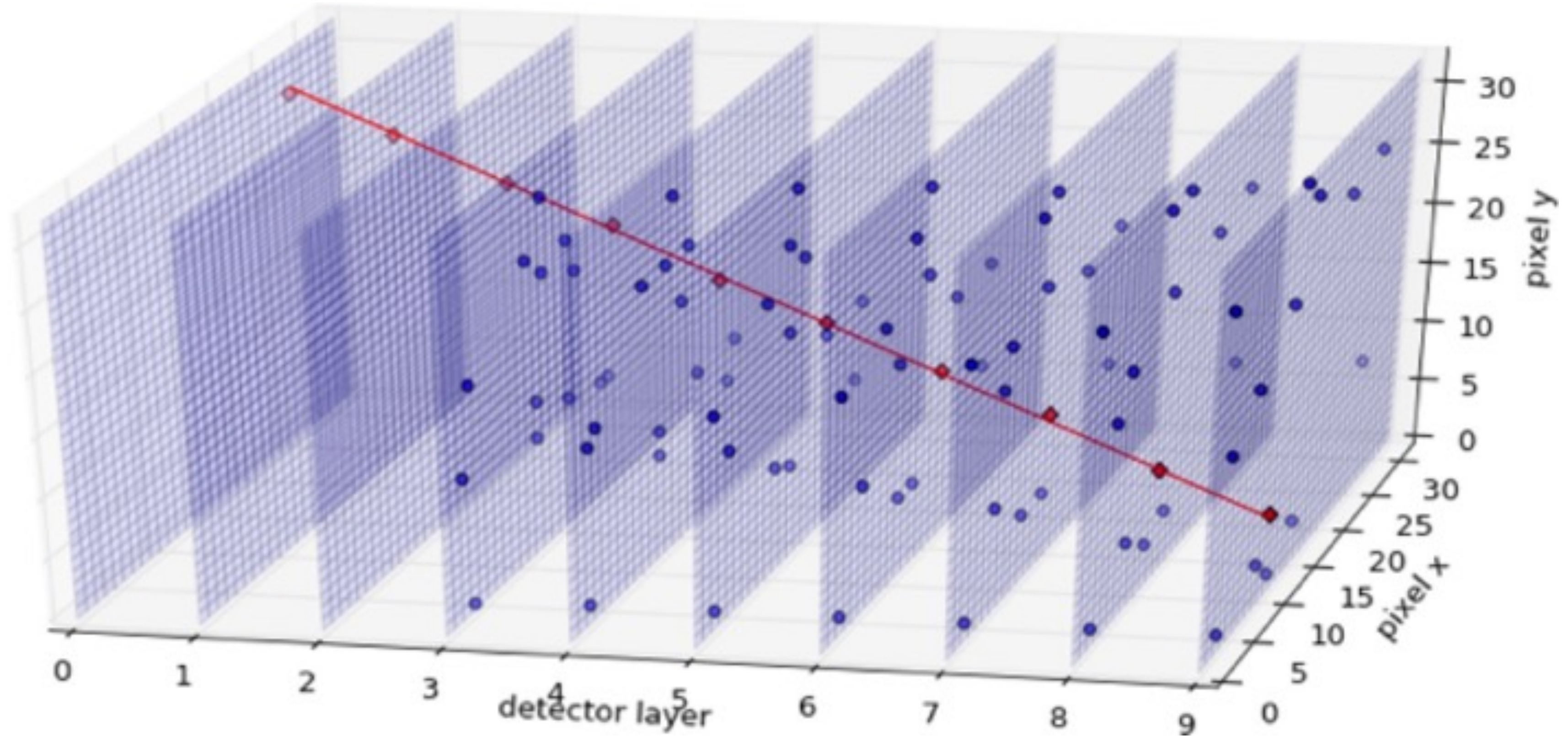


Detection

MVTX & INTT detectors

- Complex detector geometry
- High dimensionality (9k x 9k x 3)
- Variational input data
- Solutions: models with flexible data dimensionality + reduction of the useless data

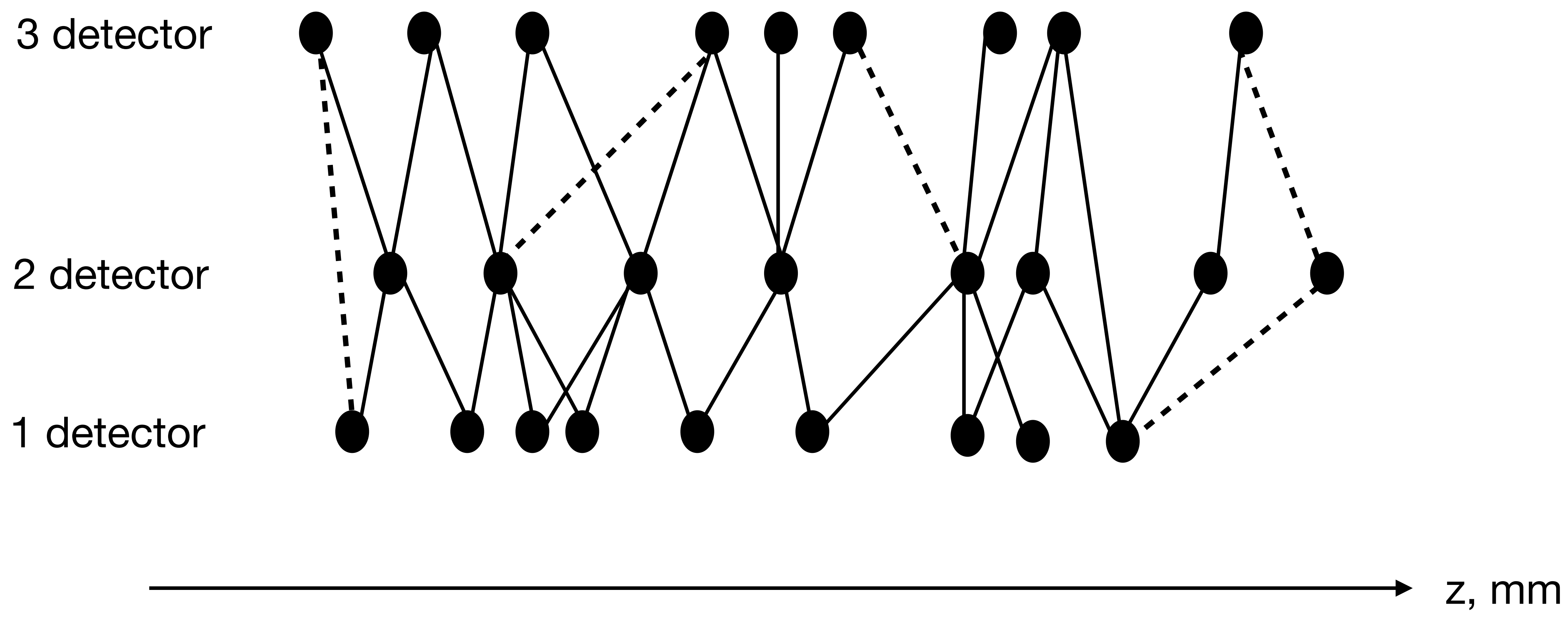




0	1	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	2	0	0
0	0	0	0	0	0	0	2	2	0	0
0	0	0	0	0	0	0	2	2	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	3	3	0	0	0	0	0
0	0	0	3	3	3	3	0	0	0	0

Tracks

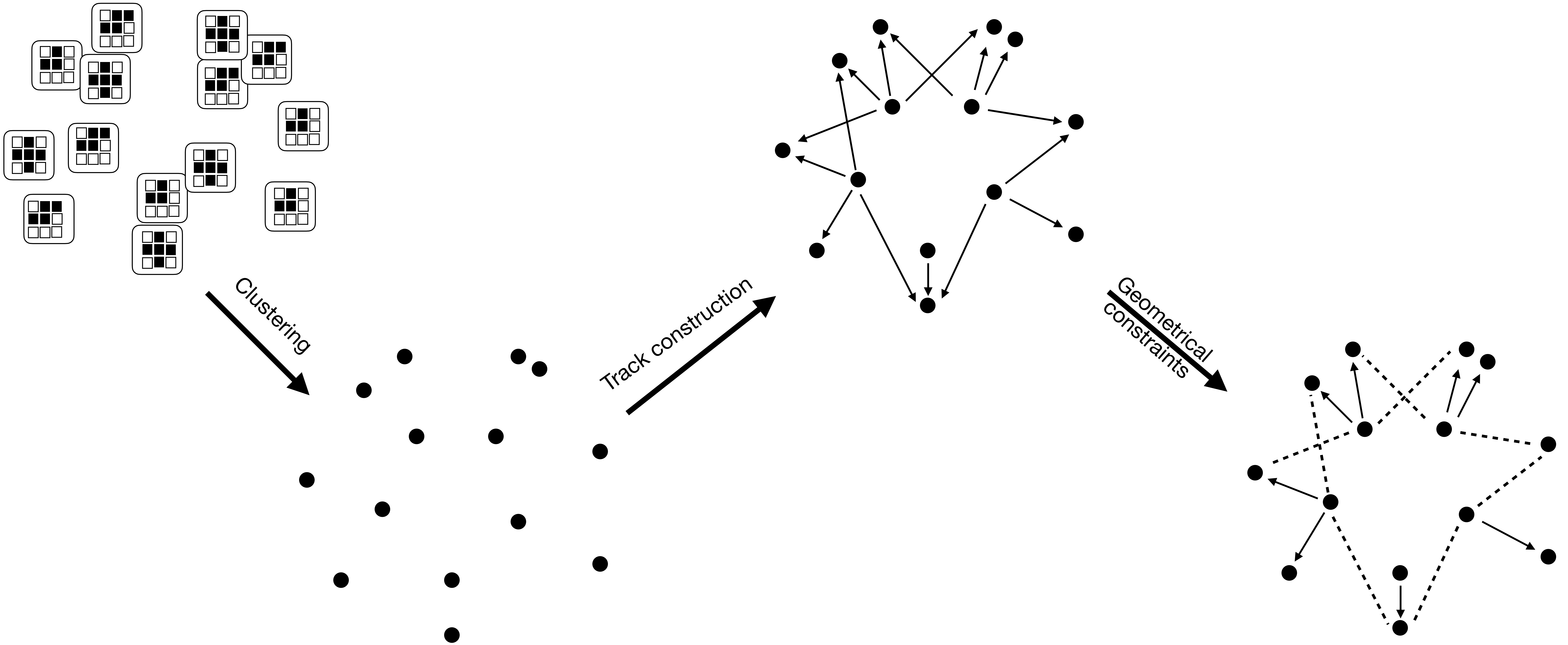
& their representations



- Tracks ~ graphs
- Geometrical constraints: $\delta(\phi) \leq \frac{\pi}{4}, \quad \delta(z) \leq 300$ mm
- With each iteration we find more relevant tracks and automise the process of matching the hits into graphs

Pipeline overview

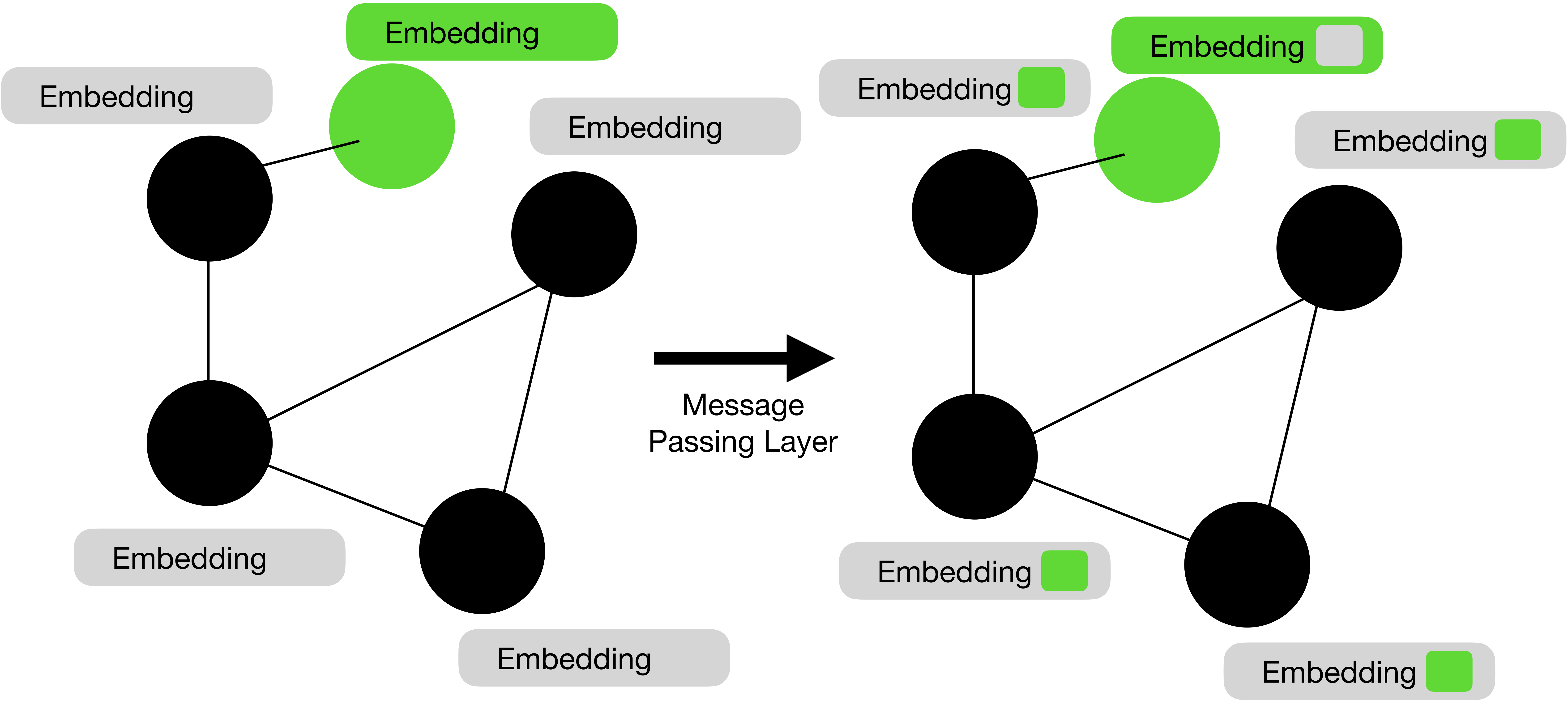
Track reconstruction



Graph Neural Networks

GNN idea

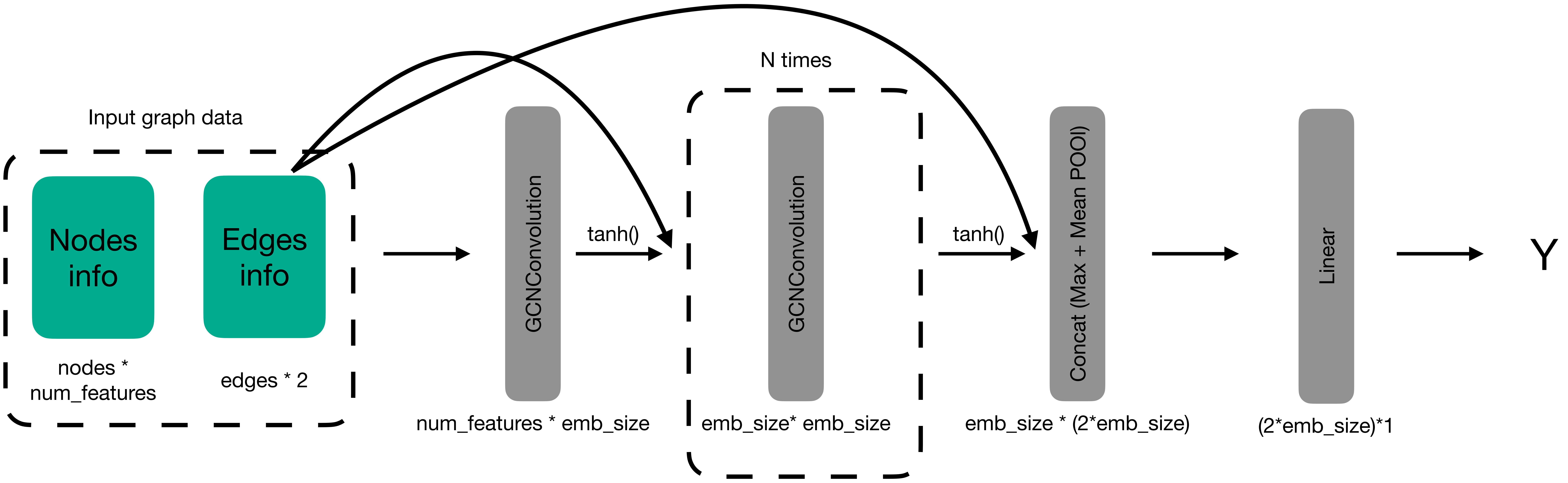
G = F(N, E)



- Predictions: node level, edge level, graph-level
- Hyper parameters: Embedding size, MPL amount.
- Python lib for working with GNN - PyG (PyTorch geometrical)

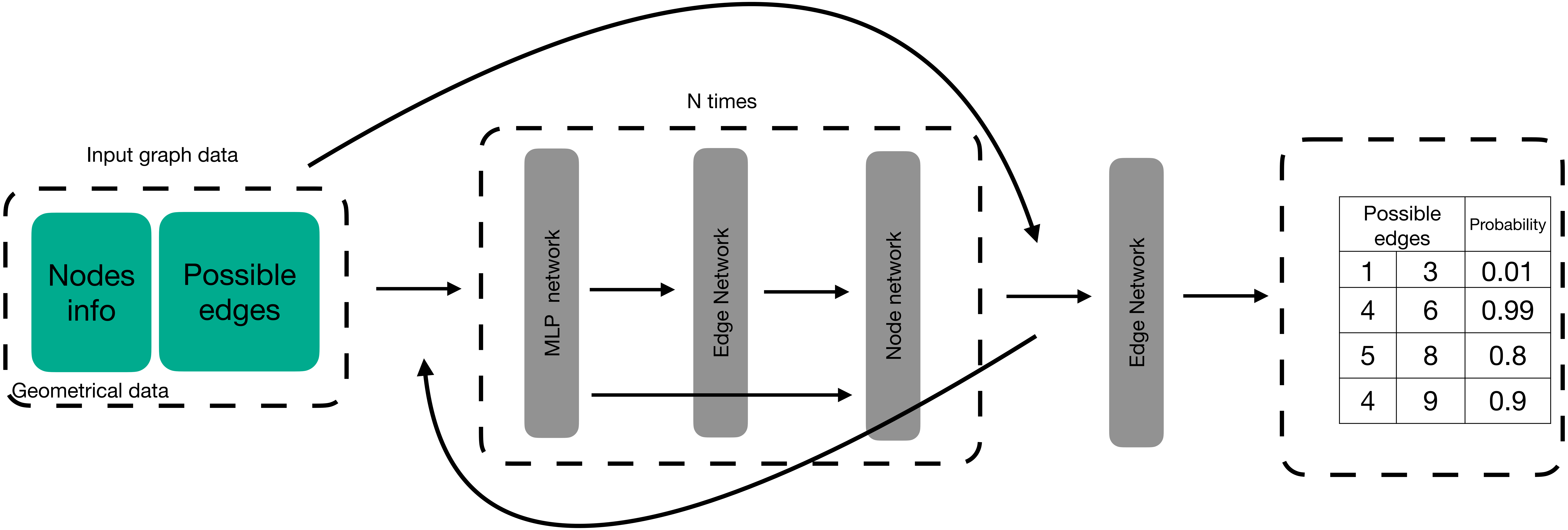
Graph Convolutional Network

graph level prediction



Graph Convolutional Network

node-level prediction - project task



Data analysis + preprocessing

Raw data

Simulated by sPHENIX project

- Generated json samples containing events
- Each event consist of: metadata, detectors data/positions/ids, points ids/pixel data/position/chip info, particles energy/momentum info, ground truth about vectors (containing particle ids etc.)


```
{
"Events" : [
{
    "MetaData": {
        "Description": "These are meta data for this event. Not intended to use in ML algorithm",
        "EventID": 0,
        "Unit": "cm",
        "CollisionVertex": [
            0.0026662557331342347,
            0.0025958270878951186,
            -10.565255027683989
        ],
        "Layer_Count": 3,
        "PixelHalfLayerIndex_Count": 6,
        "Layer0": {
            "PixelPhiIndexInLayer_Count": 6144,
            "PixelPhiIndexInHalfLayer_Count": 3072,
            "PixelZIndex_Count": 9216,
            "HalfLayer_Count": 2,
            "Stave_Count": 12,
            "Chip_Count": 9,
            "Pixel_Count": 524288
        },
        "Layer1": {
            "PixelPhiIndexInLayer_Count": 8192,
            "PixelPhiIndexInHalfLayer_Count": 4096,
            "PixelZIndex_Count": 9216,
            "HalfLayer_Count": 2,
            "Stave_Count": 16,
            "Chip_Count": 9,
            "Pixel_Count": 524288
        },
        "Layer2": {
            "PixelPhiIndexInLayer_Count": 10240,
            "PixelPhiIndexInHalfLayer_Count": 5120,
            "PixelZIndex_Count": 9216,
            "HalfLayer_Count": 2,
            "Stave_Count": 20,
```

Preprocessing part

Goals and steps

Main steps should be:

1) unpack json raw data2

2) read carefully points data by ids

3)concatenate points from INTT and MVTX

4) reconstruct tracks from ground truth

5) cluster points on 1 detector

6)calculate cylindrical coordinates

7)segment possible edges by geometrical constraints

8) scale features

9) save in appropriate format.

Clustering

Collect all nearby points -> calculate mean euclidian (center) -> save center coordinates and number of pixels

0	1	0	0	0
1	1	1	1	0
0	1	1	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Preprocessed data

Before training

Preprocessed data (per event), which can be considered to serve as input for the model, consist of next characteristics:

1) Scaled geometrical data: r vector value, phi angle value, z coordinate across cylindrical axe, amount of pixels on chip used in clustering

2) Possible edge combinations between point ids

3) Ground truth - each track consist of some points

Train process overview

Technology stack for training

Tools and technologies

- Remote developing, ssh, linux, conda3
- Python software engineering (OOP) for constructing training class, data loaders and models
- Torch, PyTorch geometrical, numpy, distributed programming
- Utils: wandb - for experiment tracking.

Training overview

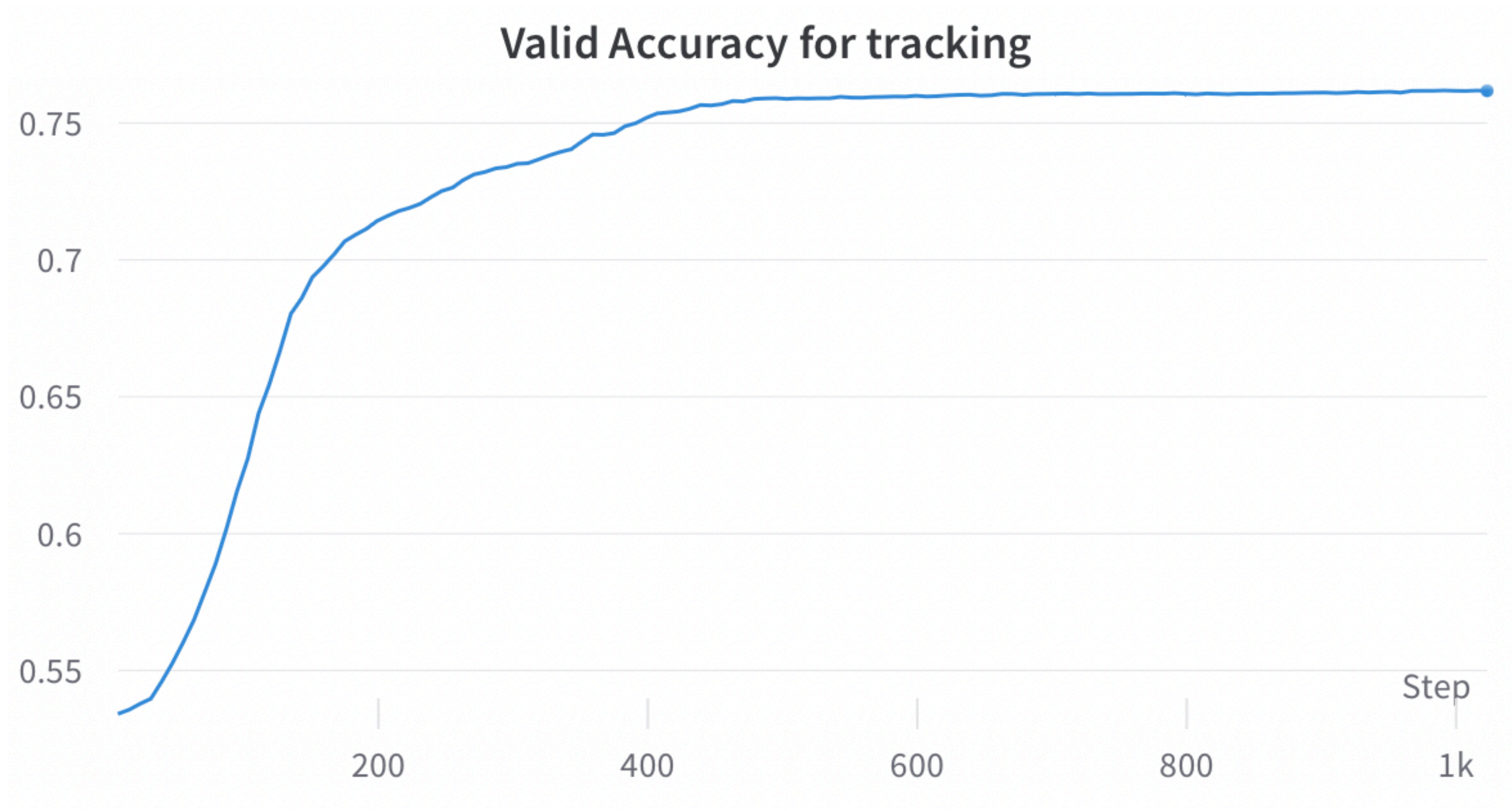
Optimizer and model

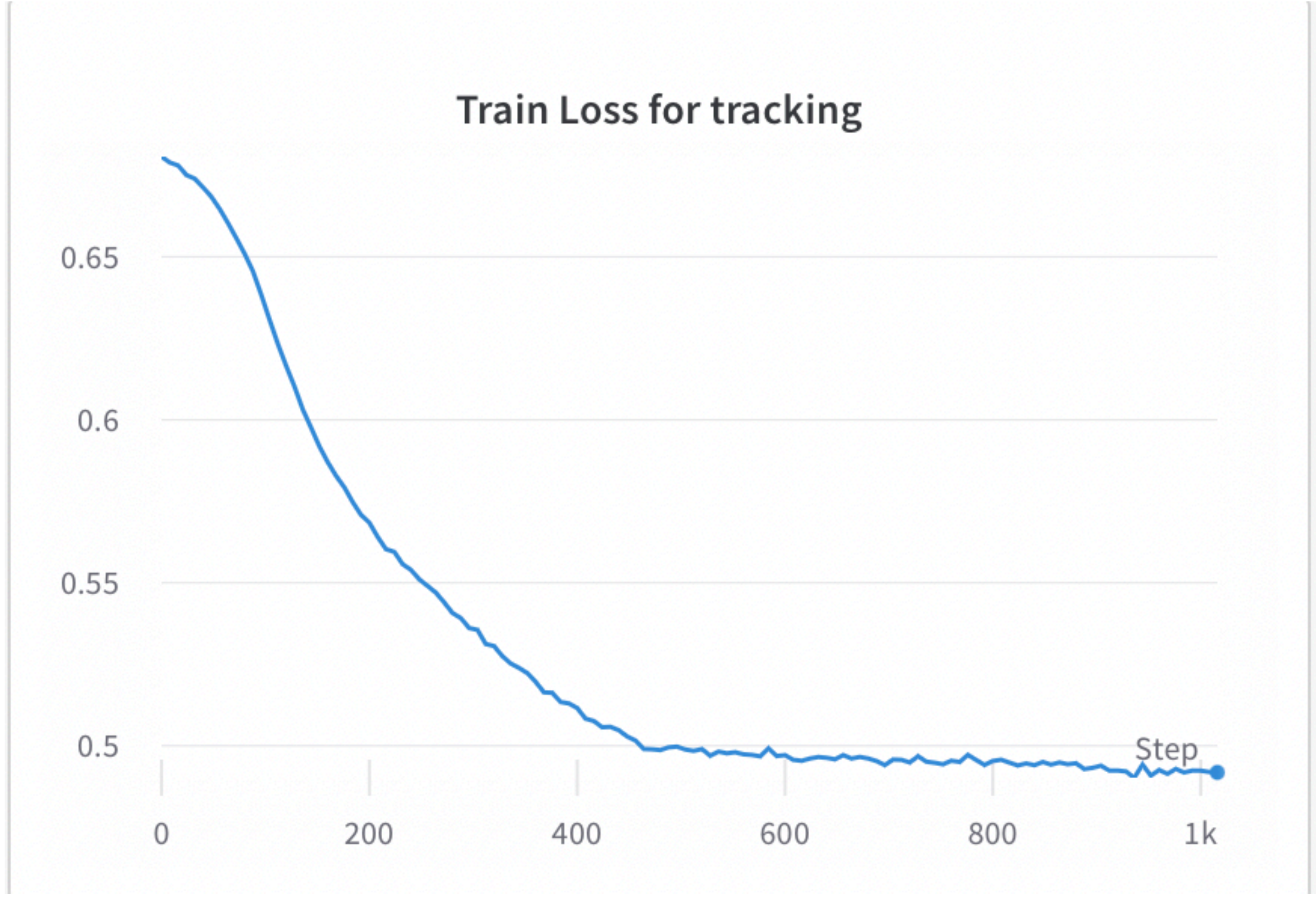
- Optimizer - Adam, with 1e-4 learning rate, weight_decay 1e-4, and lr decay schedule, starting from 60th epoch with 0.1 factor + l1+l2 regularisation
- Model: Graph Neural Network, consisting of MLP + Edge + Node networks (num of parameters = 753 and 2,5k)
- Loss: Binary cross-entropy
- Accuracy: precision (correctly predicted edges/all edges)
- Data: 800 training events + 200 validation (2000, 400)

Training results

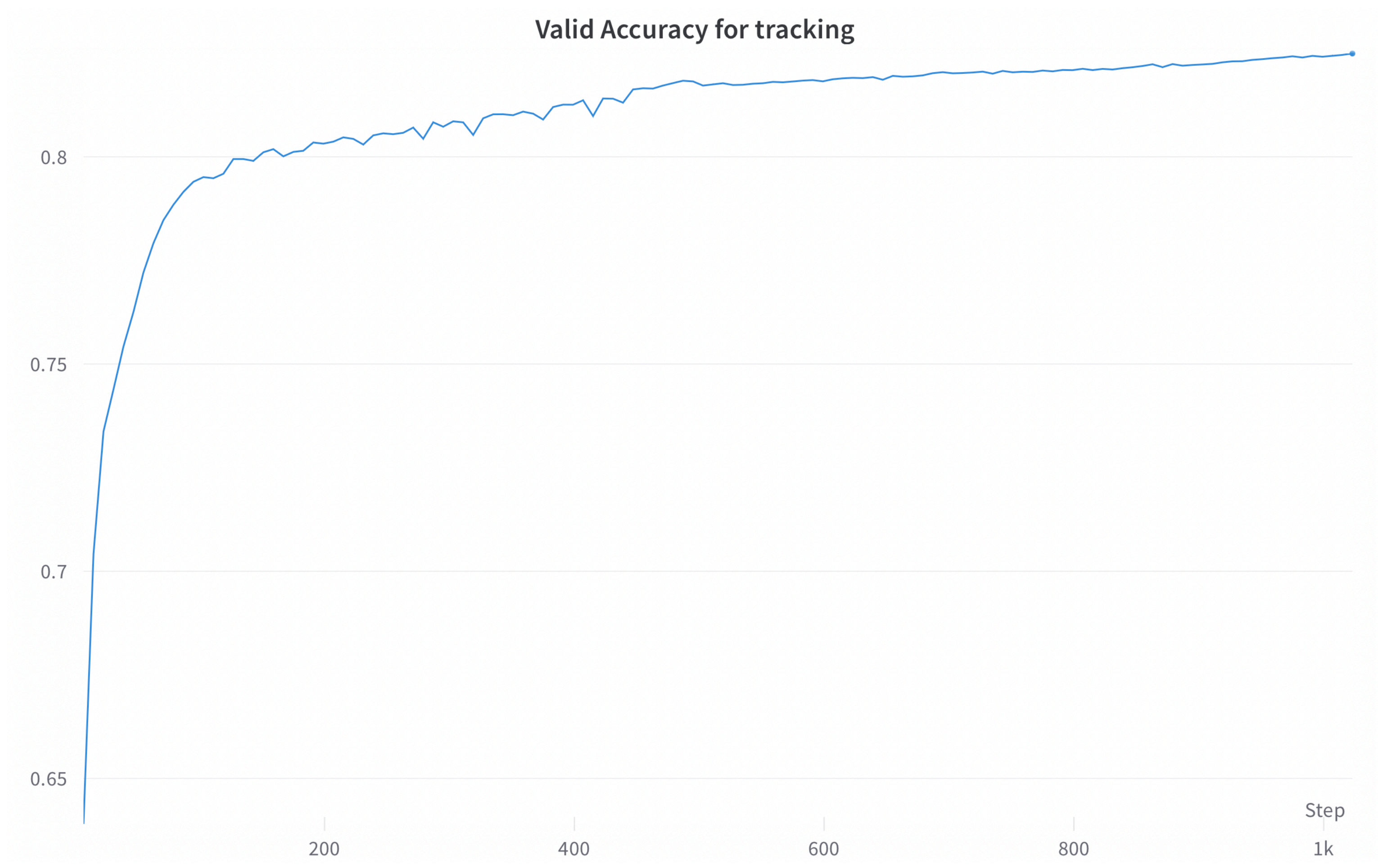
Process and performance

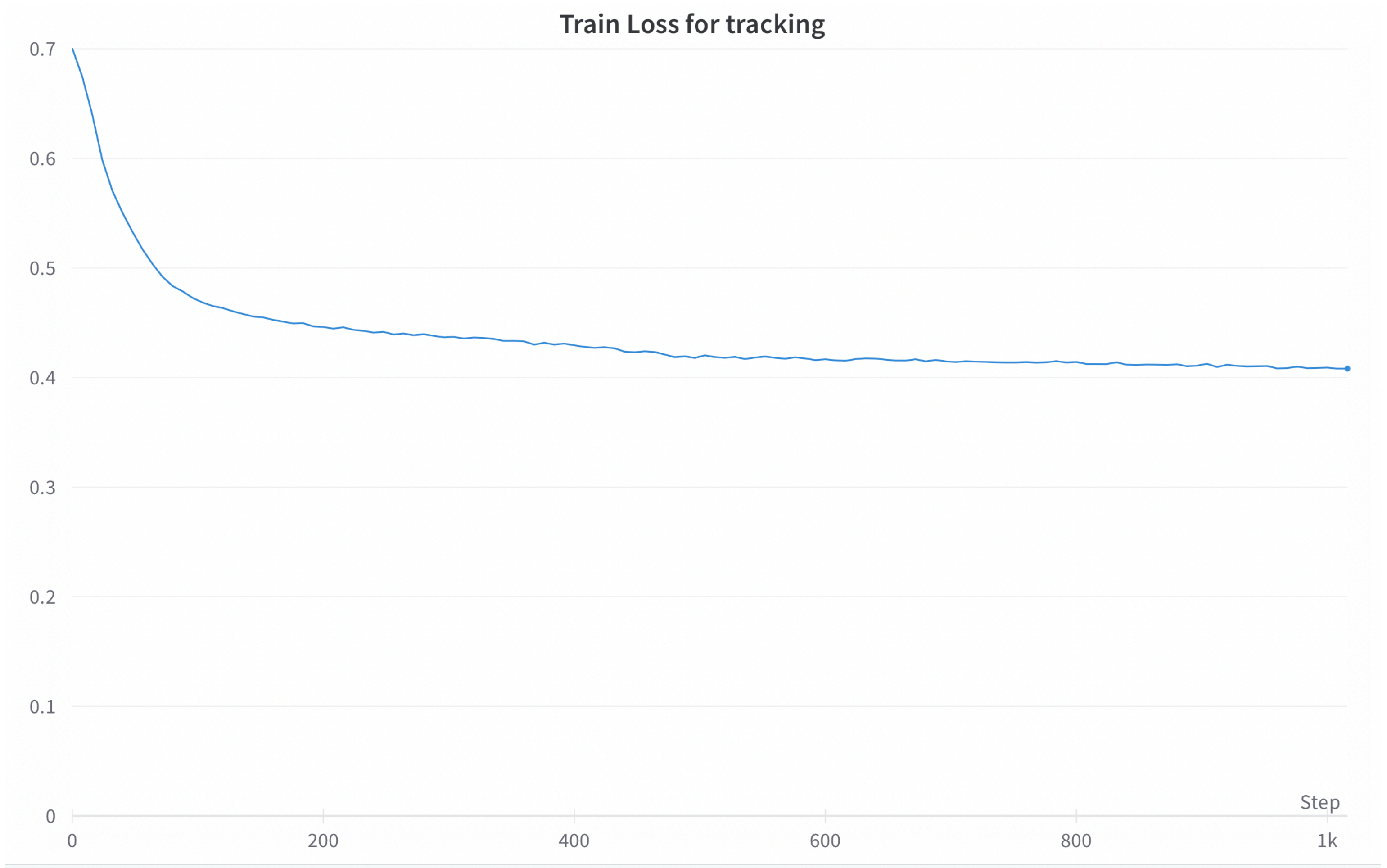
- 700 parameters, 800 training events ->





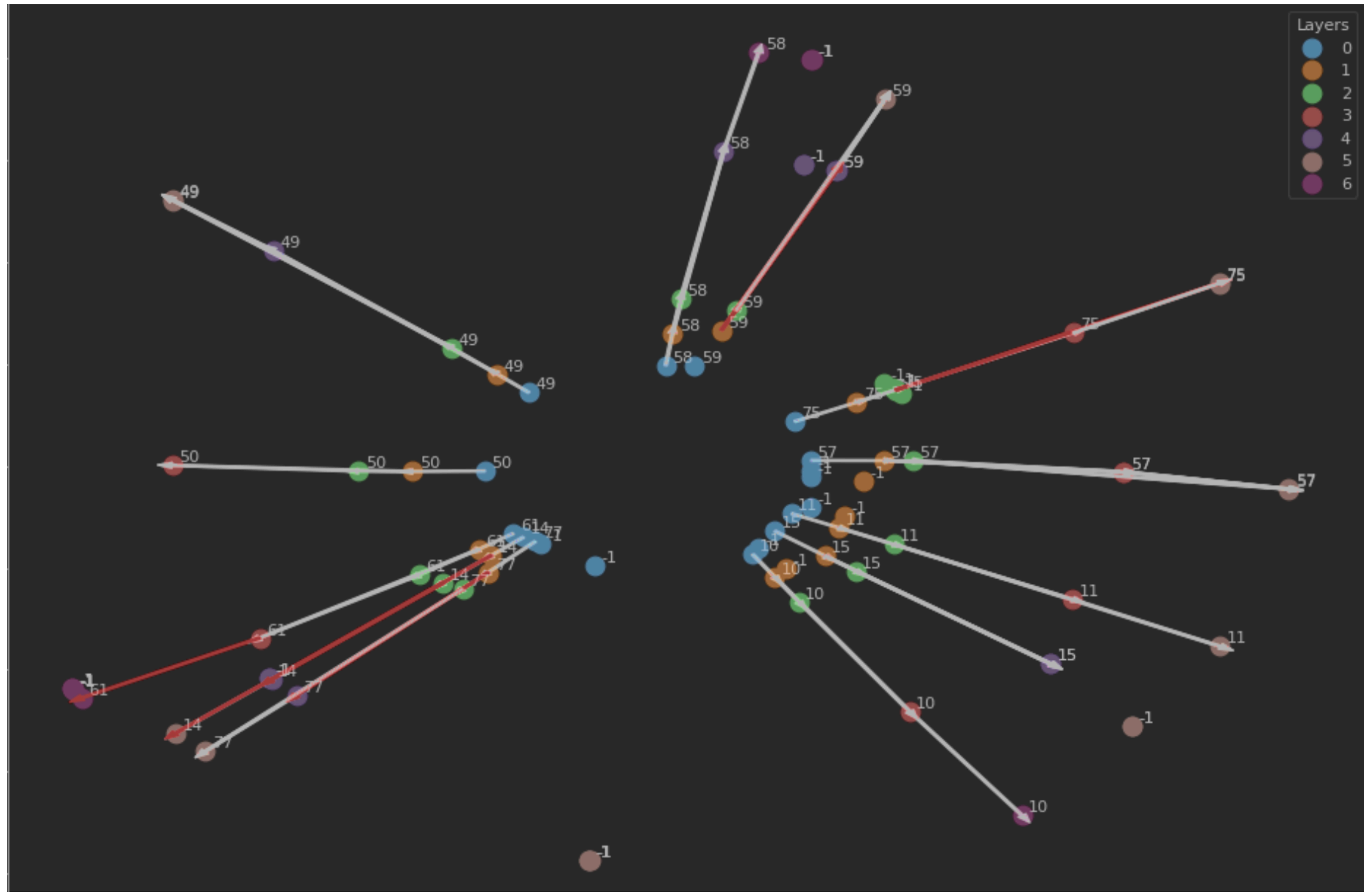
- 2,5k parameters, 2000 events ->

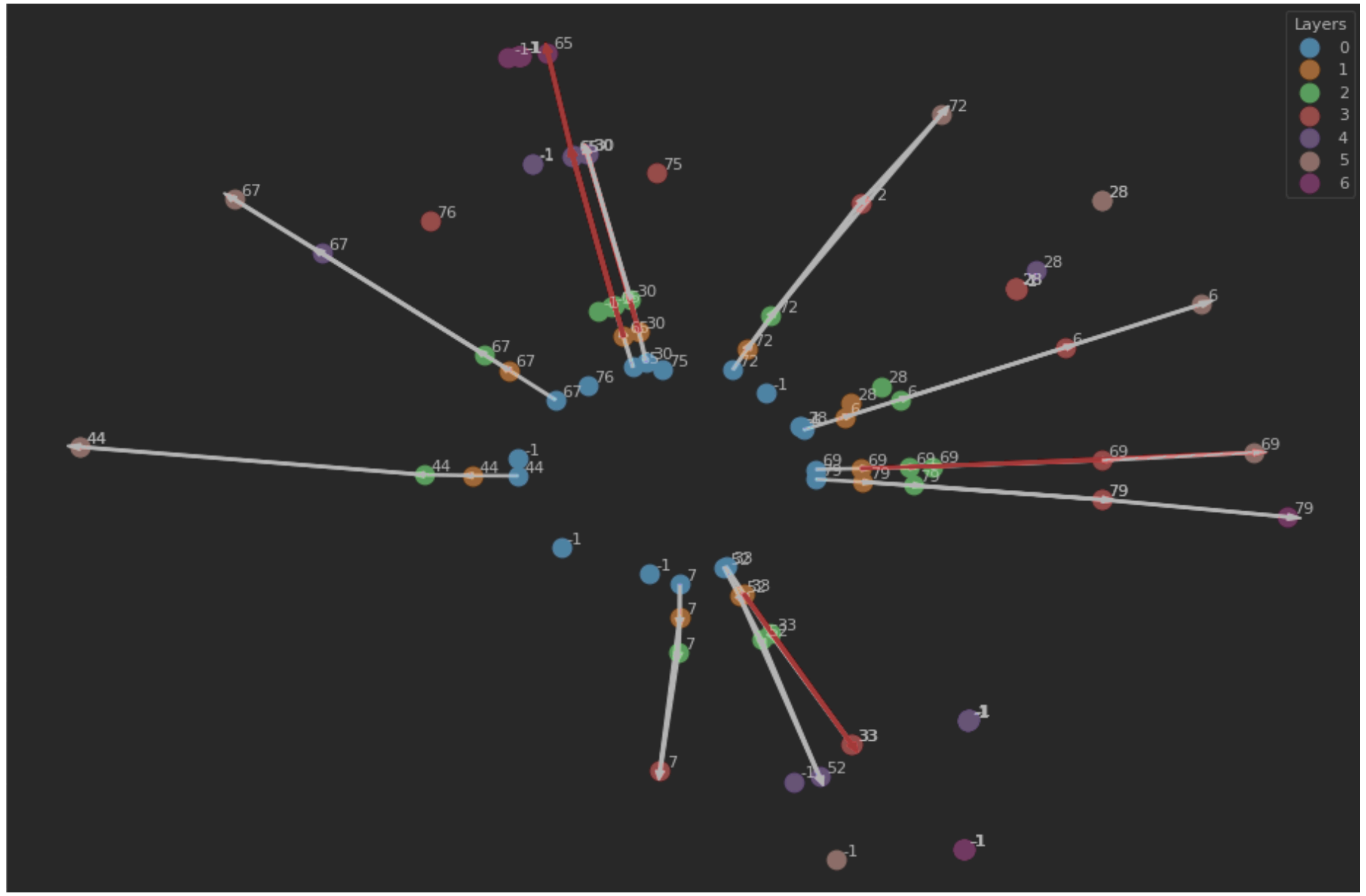




Example picture

Inference track reconstruction





Ideas for improving

Training and data preprocessing

- Any scaling of the raw parameters should be applied in the initial layers of the network, since scaling parameters can harm feature importance and should be fine-tuned by the model to maximise the performance

- Energy and momentum are not counted right now in the pipeline, which can be a significant improve in results, because of extra input information.

- Hyperparameters tuning

- Models increasing + dataset expanding

Conclusion

Conclusion

- I learned Graph Neural Networks models from zero and implemented some of their variations in the new library PyTorch geometrical
- I met new people, who taught me a lot on coding, data analysis, physics context overview and nuclear physics itself.
- I expanded my thoughts about international scientific cooperation, by directly participating in it.
- It is an honour to be a part of such interesting and cutting-edge project, which combines classical scientific subjects, such as physics, together with machine learning engineering.

Acknowledgment

- I would like to acknowledge help of Dantong Yu in providing working resources and inviting me for his project.
- Also I would like thanks Tingting Xuan for helping with pipeline and sharing ideas of the implementation.

Thanks for attention