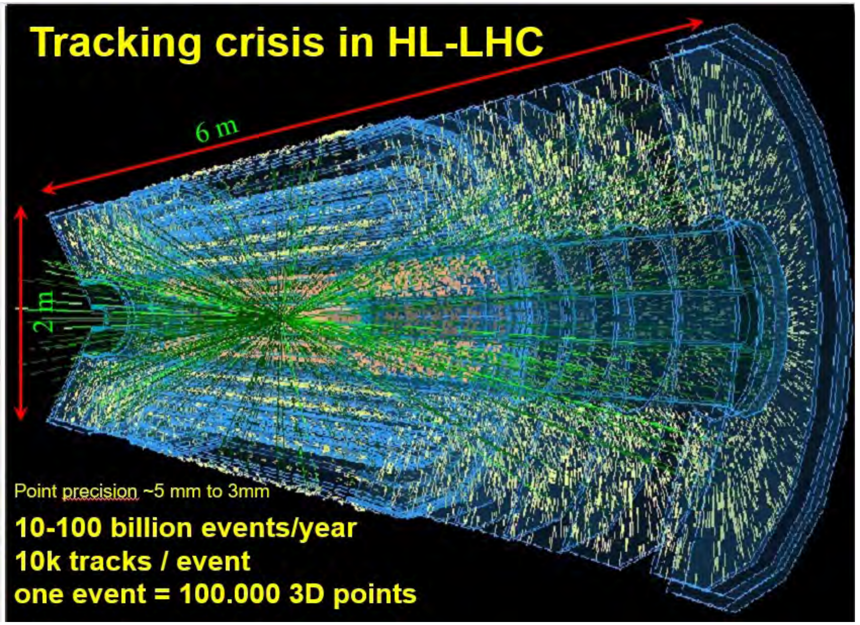
Deep Learning Methods and Software for Reconstruction of The Elementary Particles’ Trajectories

Goncharov Pavel, Shchavelev Egor, Nikolskaia Anastasia, Rezvaya Ekaterina, Rusov Daniil, Ososkov Gennady

Deep neural networks to face the tracking crisis challenge

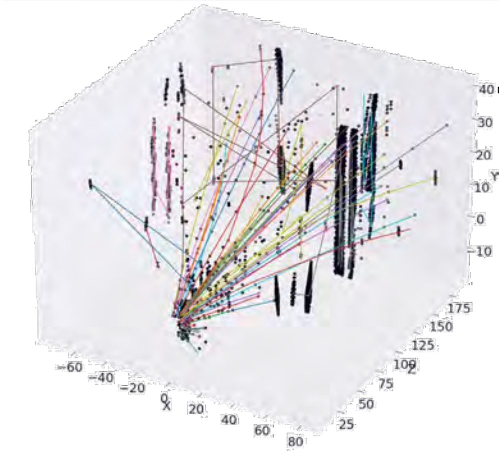
Motivation

- **Particle track reconstruction in dense environments such** as the Run-4 detectors of the High Luminosity Large Hadron Collider (HL-LHC) and of MPD NICA is a **challenging pattern recognition problem**.
- Current algorithms for tracking are highly performant physics-wise and scale badly computation-wise



Machine learning algorithms bring a lot of potential to the tracking problem, thanks to

- their capability to model complex non-linear data dependencies,
- learn effective representations of high-dimensional data through training
- parallelize easily on high-throughput architectures such as GPUs

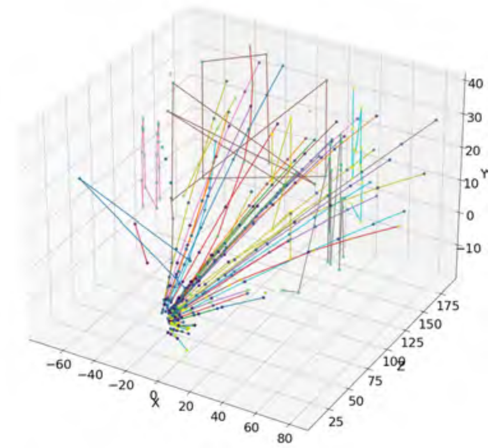


Neural network (online)

High recall, **low precision**
Very fast

Kalman filter (offline)

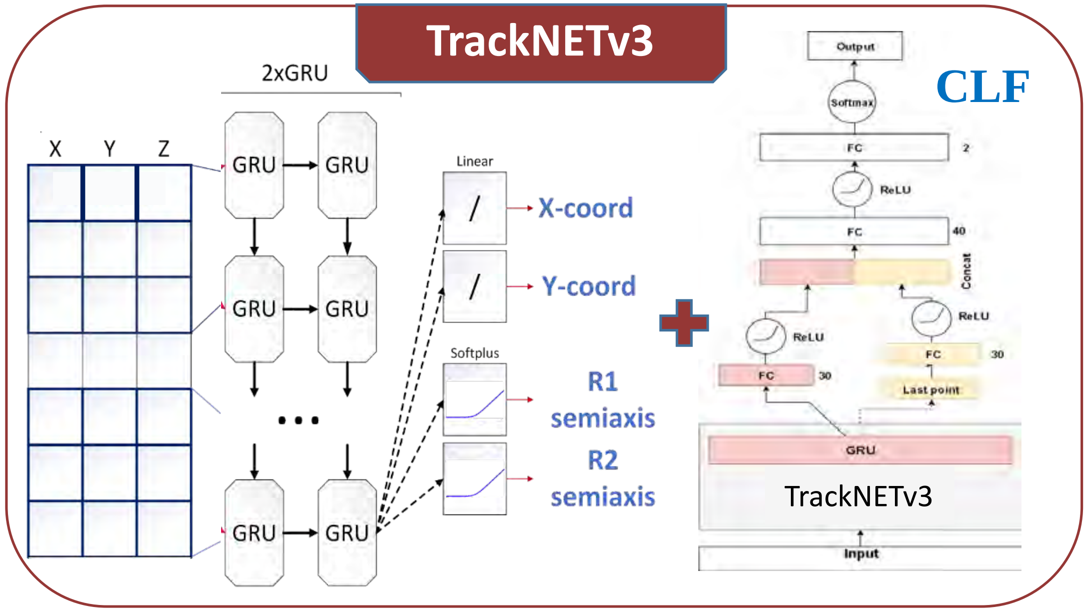
High recall,
high precision
Too slow



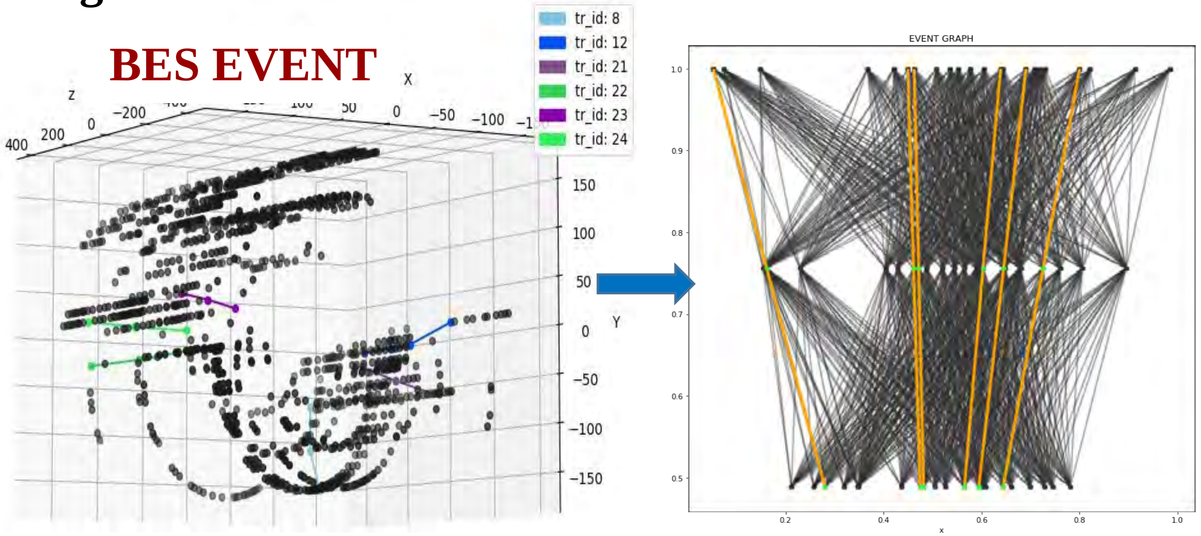
Reconstructed event

TrackNETv3 and RDGraphNet models for track reconstruction

TrackNETv3: a local approach that can be treated as **Kalman Filter “analogue”** powered by neural networks



RDGraphNet: a global approach, works with **event as a graph**. Hits are nodes and they are fully-connected between adjacent stations. The goal of the network is to **predict which edges are belong to true tracks.**

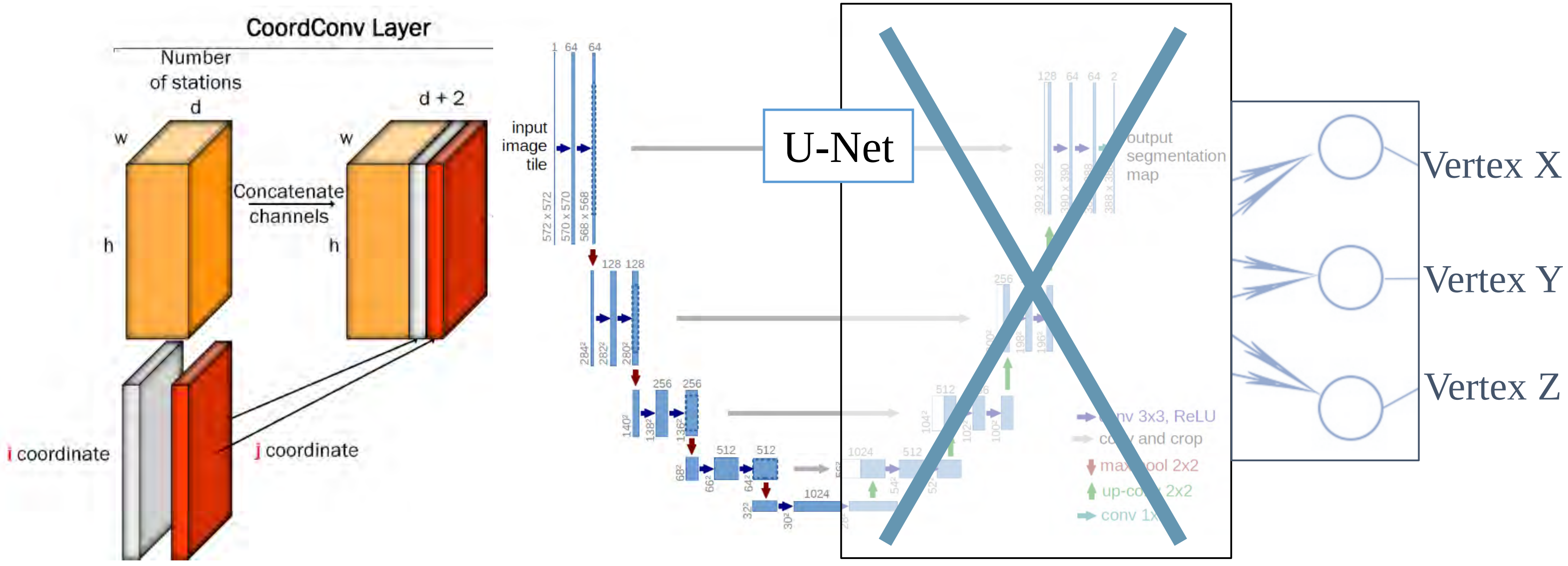


	BESIII Inner Tracker			BM@N RUN6		BM@N RUN7
	TrackNETv3	RDGraphNet	Kalman filter	TrackNETv2	RDGraphNet	TrackNETv3 (no clf)
Efficiency	0.9475	0.9548	0.9223	0.9593	0.87	0.9830
Ghost rate	0.2406	0.2596	0.0477	-	-	0.9791
Speed (events/sec)	74.17 (GPU)	283.70 (GPU)	0.2382 (CPU)			0.4545 (CPU)

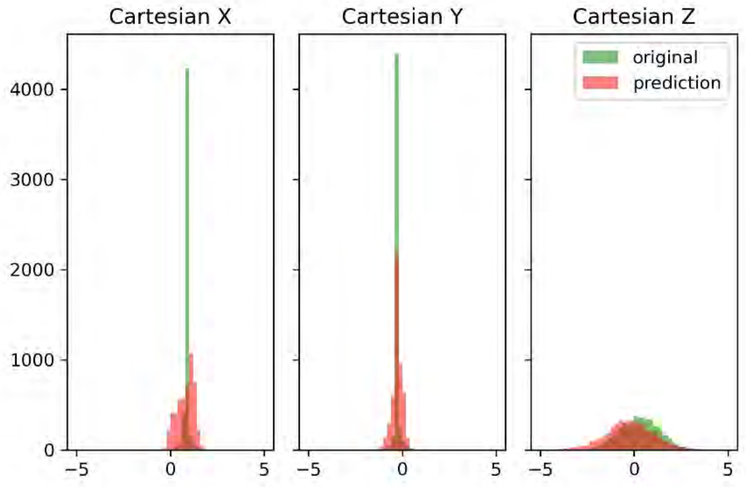
LOOT model for primary vertex prediction

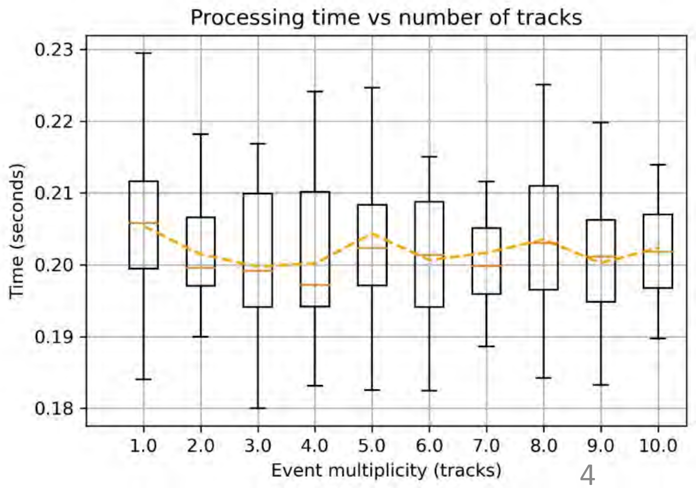
Due to the too-high expected rate of data receipt in the SPD experiment, there is a **pileup effect** when more than one event sticks together, so it is necessary to find all primary vertices for all these events to disentangle them.

To address this problem we introduce **LOOT** – a convolutional neural network that processes all event hits at once, like a three-dimensional image.



Result distribution of the coordinates of the true and the predicted vertex (BESIII Monte-Carlo)

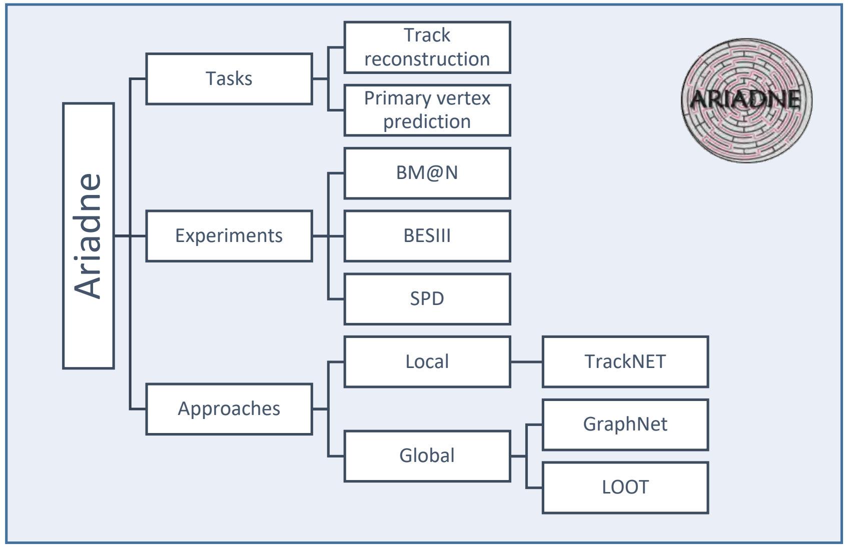


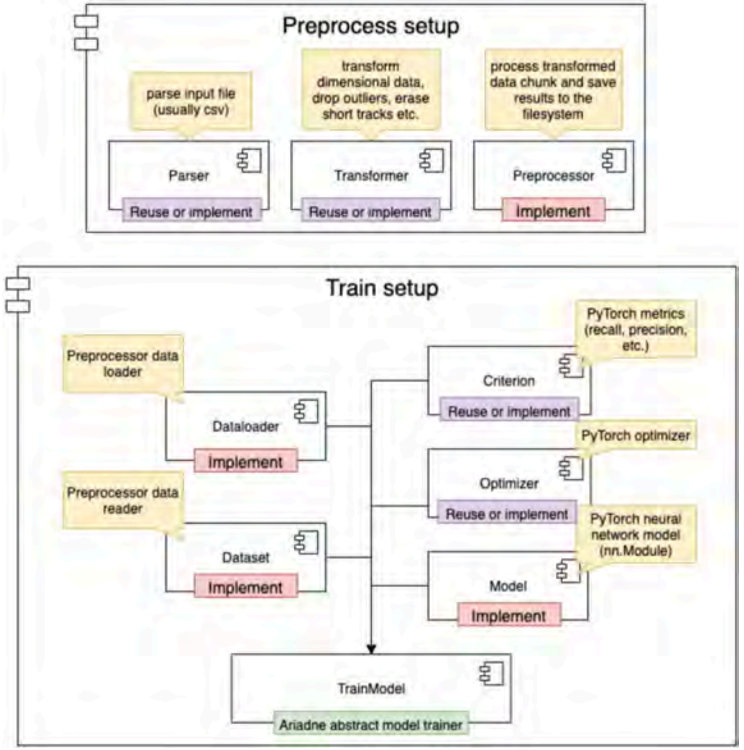


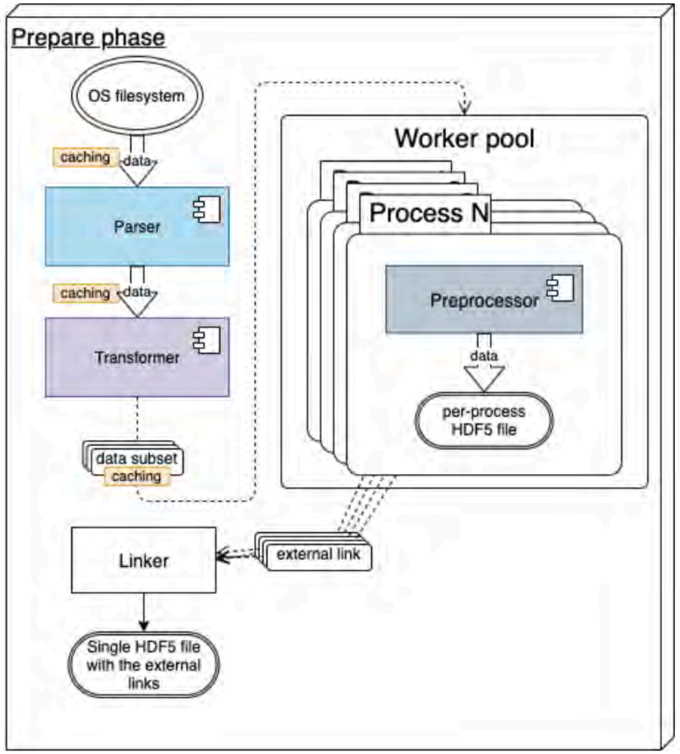
Ariadne: PyTorch Library for Particle Track Reconstruction Using Deep Learning

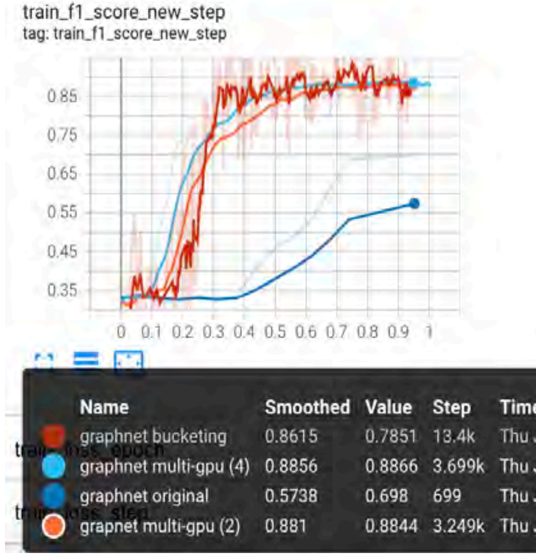
Ariadne – the first library for deep learning tracking on Python:

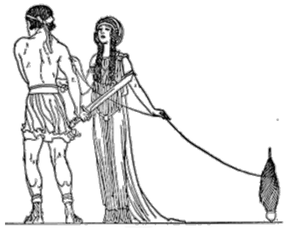
- tracking and vertex reconstruction tasks
- rapid prototyping a new NN model
- metrics logging, multiprocessing for data preparation, multi-GPU training
- open source and fully deterministic (https://github.com/t3hseus/ariadne)











Conclusion and Outlook

Conclusion

- The TrackNETv3 model which operates like a trainable Kalman filter was developed.
- RDGraphNet model that is able to consume the whole event data as a graph was introduced.
- A special convolutional neural network named LOOT was proposed to solve the problem of primary vertex recognition.
- We developed Ariadne – the first library for deep learning tracking on Python.
- With the help of Ariadne, we've trained RDGraphNet, TrackNETv3, and LOOT models on Monte-Carlo data from BESIII and BM@N experiments.
- For the BESIII experiment, we achieved more than 94% of track reconstruction efficiency that is superior to the Kalman filter!
- Each of the proposed deep learning models is more than 100 times faster than the Kalman filter.
- Besides, we've trained the regression part of the TrackNETv3 model on BM@N RUN7 data (Ar+Pb interaction) and achieved the descent tracking efficiency equal to 98.3%.
- The LOOT model was successfully trained on the task of primary vertex recognition in the BESIII experiment and showed sufficient results.

Outlook

- We are going to create the classifier of track candidates for TrackNETv3 required for the BM@N RUN7 data.
- When the full track reconstruction pipeline for BM@N RUN7 is trained, we **will run inference on experimental data!**
- We have to do a lot of refactoring in Ariadne in order to build a production-ready Python package.
- Besides, now we are training our models – LOOT and TrackNETv3 on Monte-Carlo data from SPD.