Separation of particle trajectories by events accumulated over a single timeslice in the SPD NICA detector using graph neural networks

Omelianchuk S.S. [1,2] Talochka E.N. [2] Ososkov G.A. [2]

[1]Branch of MSU, Dubna [2]Joint Institute for Nuclear Research, LIT

Introduction

This work is devoted to untangling an array of previously reconstructed tracks into individual events recorded in a single timeslice. Classical methods for solving this problem are not applicable in conditions of experiments with such high event multiplicity and their pileup.

A Siamese neural network was previously developed for track sorting; however, it did not meet the required performance criteria. An effective neural network for this task must satisfy several key requirements:

- Compatibility with tracks composed of varying numbers of hits
- Ability to process a variable number of events per time slice
- Fully parallelizable model computation

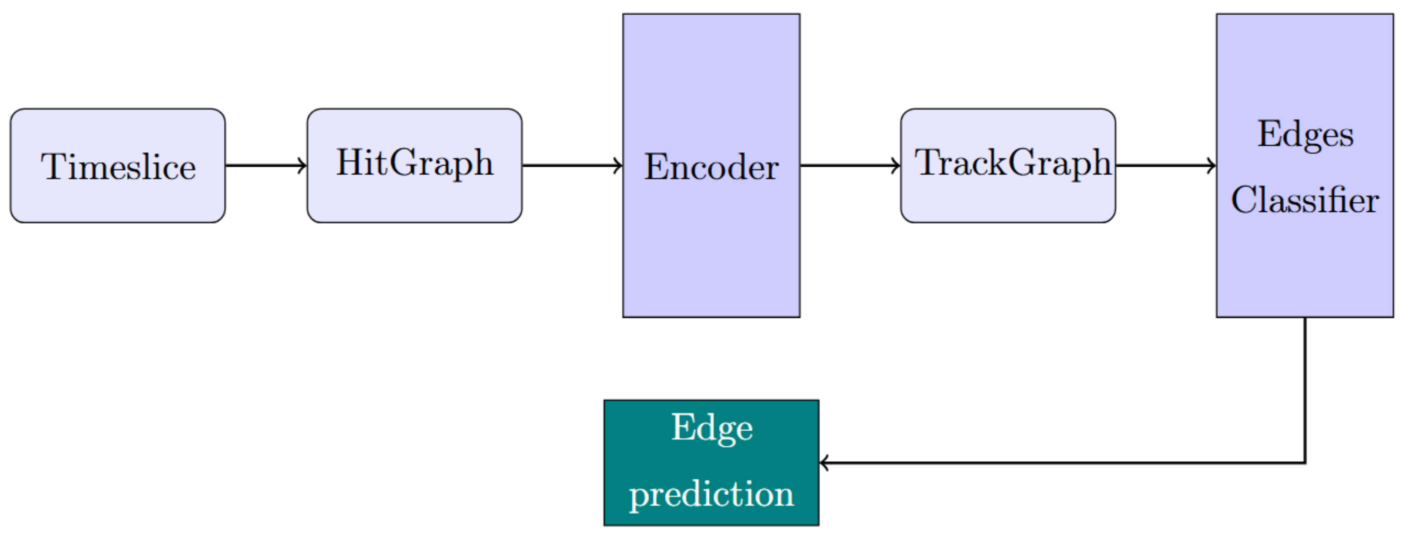

Figure 1. The general architecture of the model

We propose an **edge-classifying Graph Attention Neural Network (GANN)** with a preliminary track encoder. The model architecture is shown in (Fig. 1). The GANN classifies edges between encoded tracks, identifying pairs that originate from the same event. Clusters of such connected pairs are then interpreted as individual collision events.

Model

The first component of the model is the **track encoder** (Fig. 2). Hit coordinate data is transformed into a graph where each node represents a hit with its spatial coordinates, and edges connect adjacent hits within the same track, encoding each track as a separate graph.

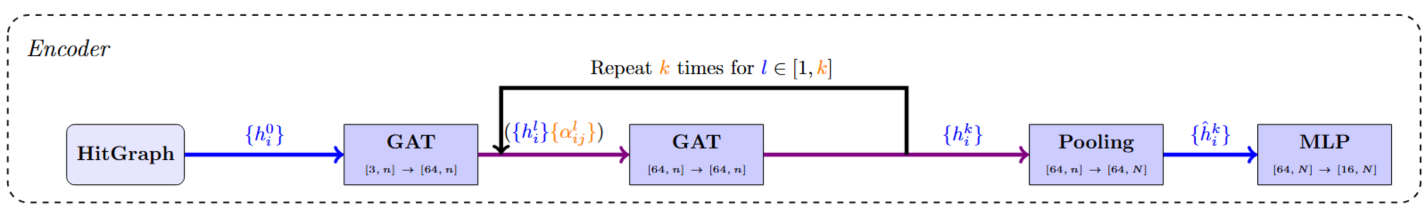

Figure 2. Encoder architecture

The second stage builds a supergraph where nodes represent encoded tracks. In this graph, edges connect every node to all other nodes, representing true ($\hat{y}_t = 1$) and false ($\hat{y}_t = 0$) edges. The third stage involves classifying these edges into false and true connections using a **classifier**, whose architecture is presented in (Fig. 3).

$$\alpha_{ij}^l = \sigma\left(\text{MLP}(h_i^l || h_j^l)\right) \qquad m_i^l = \sum_{k \in \mathcal{N}(i)} \alpha_{ik} h_k^l \qquad h_i^{l+1} = \text{MLP}(m_i^l || h_i^l)$$

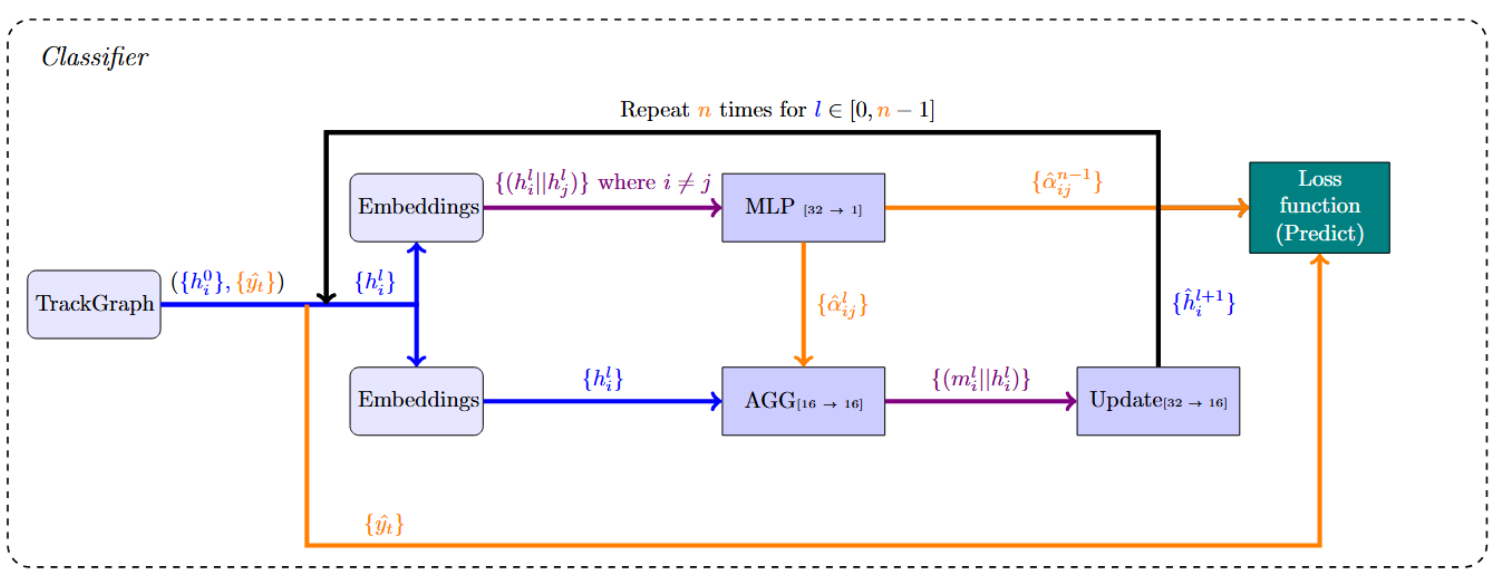

Figure 3. Classifier architecture

Datasets

- **SPD timeslice**. Simulated tracks. All of the hits in a track are equidistant in r_ϕ. Each track consists of a large number of hits in the range $(28, 35)$. The transverse dimensions of the detector $r_\phi = \sqrt{x^2 + y^2}$ are limited: $r_\phi^{min} = 150\,mm, \quad r_\phi^{max} = 850\,mm.$
- **TrackML timeslice**. Tracks simulation based on LHC detectors. All of the hits in a track are not equidistant in r_ϕ. Each track consists of a small number of hits in the range $(3, 20)$. The transverse dimensions of the detector $r_\phi = \sqrt{x^2 + y^2}$ are limited: $r_\phi^{min} = 50\,mm, \quad r_\phi^{max} = 1000\,mm.$

Training

We use hierarchical learning by parameter $S_\mathcal{N}$. By adjusting this parameter starting from $S_\mathcal{N} = 2$, we can simplify the model training process while maintaining connections between tracks from the same event and those from a specified number of adjacent events. We used BFL as a loss function.

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \omega_i (1 - p_t)^\gamma \ln(p_t) \qquad \omega_i = y_i \omega_p + (1 - y_i)\omega_n \qquad p_t = \begin{cases} p_i & \text{false} \\ 1 - p_i & \text{true} \end{cases}$$

Results

Testing of the hierarchically trained model with parameters: $S_\mathcal{N}^i = 2, S_\mathcal{N}^f = 4, \omega_p = 1, \omega_n = 0.5, \gamma = 1$

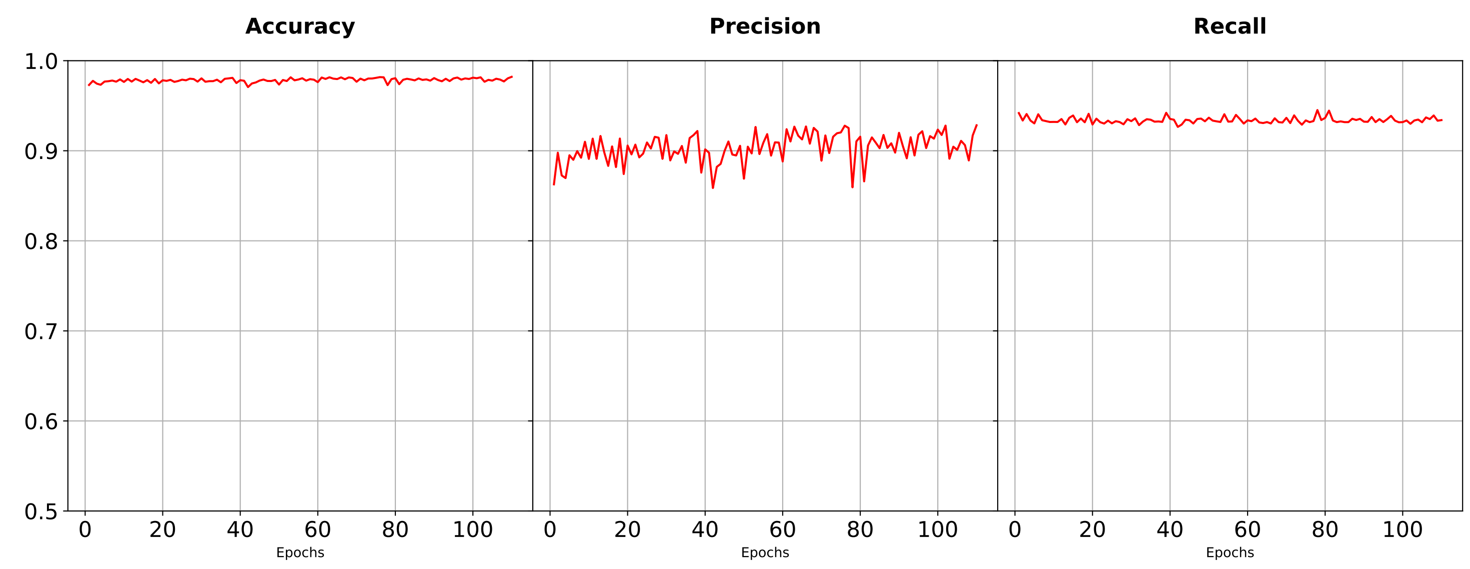

Figure 4. Testing of the trained model on **SPD simulation** data with $S_\mathcal{N} = 4$

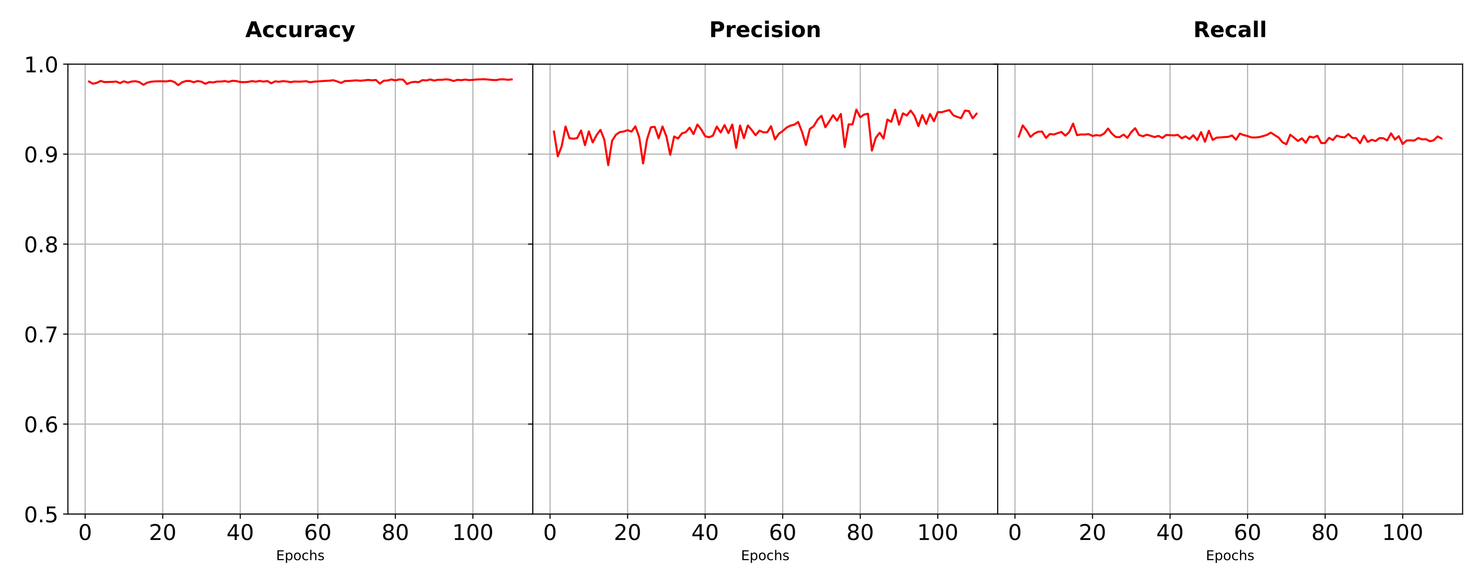

Figure 5. Testing of the trained model on **TrackML** data with $S_\mathcal{N} = 4$

Conclusions

The final processing speed of the model on an Nvidia V100 Tesla GPU is:

- SPD simulation:
  - Training: 4 timeslice/sec
  - Evaluation: 5.5 timeslice/sec
- TrackML dataset:
  - Training: 3.5 timeslice/sec
  - Evaluation: 4 timeslice/sec

- The model handles variable numbers of events per time slice and hits per track through adaptive graph structures.
- All computations are fully parallelized on GPU architectures.

savelii@jinr.ru yauheni.talochka@cern.ch ososkov@jinr.ru