PROCEEDINGS OF SCIENCE

Deep neural network applications for particle tracking at the BM@N and SPD experiments

D. I. Rusov,[a] **A. N. Nikolskaia,**[b] **P. V. Goncharov,**[a,c,*] **E. M. Shchavelev**[b] **and G. A. Ososkov**[c]

[a] *Dubna State University,*
19 Universitetskaya st., Dubna, Moscow Region, 141982, Russia
[b] *St. Petersburg State University,*
7-9 Universitetskaya Emb., St. Petersburg, 199034, Russia
[c] *Joint Institute for Nuclear Research,*
6 Joliot-Curie st., Dubna, Moscow Region, 141980, Russia
E-mail: dan4r@yandex.ru, nikolskaya2000@yandex.ru, pgoncharov13@gmail.com, egor.schavelev@gmail.com, ososkov@jinr.ru

Particle tracking is an essential part of any high-energy physics experiment. Well-known tracking algorithms based on the Kalman filter are not scaling well with the amounts of data being produced in modern experiments. In our work we present a particle tracking approach based on deep neural networks for the BM@N experiment and future SPD experiment. We have already applied similar approaches for BM@N RUN 6 and BES-III Monte-Carlo simulation data. This work is the next step in our ongoing study of tracking with the help of machine learning. Revised algorithms - combination of Recurrent Neural Network (RNN) and Graph Neural Network (GNN) for the BM@N RUN 7 Monte-Carlo simulation data, and GNN for the preliminary SPD Monte-Carlo simulation data are presented. Results of the track efficiency and processing speed for both experiments are demonstrated.

The 6th International Workshop on Deep Learning in Computational Physics (DLCP2022)
6-8 July 2022
JINR, Dubna, Russia

*Speaker

PoS(DLCP2022)005

1. Introduction

Particle track reconstruction is an important part of data analysis in high energy physics experiments. The purpose of the task is to group points of previously detected particle flights (hits) according to the criterion of belonging to some track. Efficient reconstruction of all particle trajectories allows to reconstruct the whole event and get physical insights from the data.

Solving the problem raises many difficulties. For example, the location of the primary vertex - the point of interaction of particles - is unknown in many experiments. Tracks may have different lengths, and it may be hard to understand which hit is the last for a given track. Also, many problems are associated with the design of the experimental facilities, for example, a microstrip gas electron multiplier (GEM) detector generates a large number of false hits that do not belong to any particle.

Classical algorithms based on the Kalman filter [1] have an ability to solve the above difficulties with sufficient accuracy, but they have several well-known drawbacks. Modern experiments are based on high luminosity events and produce a large amount of hits. Hence, classical algorithms cannot afford an acceptable data processing speed due to the high computational complexity of the Kalman filter and poor abilities for scaling and parallelization.

Reasons described above led to the necessity of developing new methods of tracking based on deep neural networks. These methods are successfully used to find complex implicit dependencies in the input data across various fields. Also, they are capable of modeling high-dimensional internal structure of data and, since all operations are reduced to sequential matrix multiplication, they are easier to parallelize.

The neural network approach has been successfully implemented in some works devoted to particle tracking in modern experiments. For example, graphs (GNNs) and recurrent neural networks (RNNs) have shown high efficiency for Monte Carlo simulated data of the BESIII and BM@N experiments [2, 3].

2. SPD

2.1 Experiment description

SPD (Spin Physics Detector) is a planned future experiment on the NICA megascience project developed in Dubna [4]. This experiment aims to verify the basis of quantum chromodynamics by colliding polarized nucleons and deithrons. SPD is constructed as a universal 4-pi detector with resolution of 50000 channels and dataflow of 20GB/sec because of a maximum collision frequency of 3 MHz. Collecting, processing, and storing such amounts of data leads to different engineering problems.

For example, high speed of data retrieval forces researchers to retrieve the data in the form of intersecting time slices of 10 ms instead of the common format of single event at time. Hence, it is required to differentiate between groups of hits from different events within the same time slice, filter out background events, find primary vertices of the events, etc. On the other hand, only a small part of all retrieved data is of scientific interest. Therefore, it is needed to reduce the amount of redundant time slices and not waste disk space. In classical systems various filters may be constructed based on the data from different subsystems of the detector like hardware triggers, but in SPD it is unacceptable due to the too high restrictions on the data delivery speed.

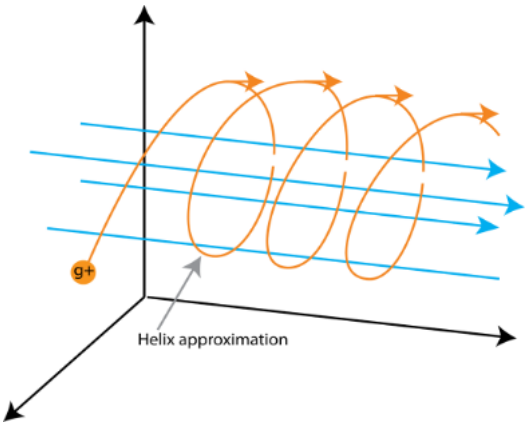

Figure 1: Helix representation of the particle path

All these together require a fast enough software to create a reconstruction system of the events and store only valuable events (2-5%). Therefore, requirements for tracking systems in the speed, memory and parallelization abilities are even more important than before.

2.2 Data generation

The SPD experiment is still in the design stage, so it is hard to use the existing software of Monte-Carlo simulation for training of the neural networks. The problem is that it is unclear how to find the right particle which produced the particular hit and label this point with appropriate track identifier. However, we created a simple Python program based on the particle trajectory approximation (fig.1) with a helix. Heterogeneity of the magnet field and energy losses are not considered in this very basic model. High performance and simplicity of adding new features compensate for these losses during early phases of experiment development.

The multiplicity of each event is a random number between 1 and 10. The vertex coordinates are random too. The transverse momentum of a particle is a random number with a uniform distribution in the range of values from 100 to 1000 MeV/c. The particle trajectory is represented by a selection of points on a segment of a helix with a helix pitch h=$\frac{2\pi}{B}|\frac{m}{q}|v\cos\alpha$ and radius $R = \frac{1}{B}|\frac{m}{q}|v\sin\alpha$. The detector insufficiency is considered too which means that with some probability the hit may be omitted. Fake hits are sampled from the uniform distribution according to the predefined fake hits ratio parameter of the script. The result of modeling is shown on the figure 2.

3. BM@N

3.1 Experiment description

BM@N (Baryonic Matter at Nuclotron) is the first experiment of NICA [5]. The main aim of the BM@N experiment is to study interactions of relativistic heavy ion beams with fixed targets. This experiment uses GEM detectors to register the hits. A well-known drawback of GEM detectors is that such detectors produce many fake hits due to the appearance of the spurious strip crossings in case of events with more than one particle. Therefore, it is necessary to classify whether the particular hit is real and belongs to some track or not. In comparison with SPD, the number of

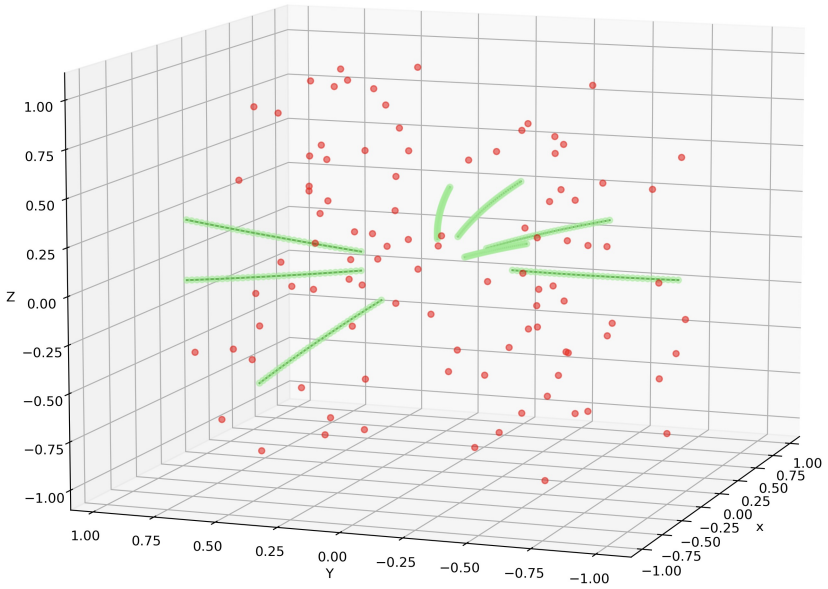

Figure 2: Model event of SPD experiment. Real hits - green dots, fake - red

fake hits is much higher. Multiplicity in the event may reach 100 tracks or more depending on the selected substance for the beam and for the target. In comparison, typical multiplicity in SPD is only about 1-10 tracks per event. Finding the method stable to a great noise background is vital to perform the tracking efficiently in such a dense environment.

Several deep neural networks were used to perform tracking at BM@N [2, 3]. Those models may be classified into local and global approaches depending on what part of data is used during the inference phase. TrackNETv3 [2] sees only one particular track at time, so it can be considered as a local model. On the contrary, the GraphNet model [3] operates globally, using all hits in the event at once. Both approaches achieved promising results, but only tracks without gaps were considered in the previous studies, which means that inefficiency of the detector hasn't been taken into account.

3.2 Data generation

Monte-Carlo simulation with the help of LAQGSM [6] was used to generate the dataset for training, validation, and evaluation of the models. We considered the RUN 7 geometry of the BM@N experiment which includes 9 stations in total - 3 silicon stations followed by 6 GEM detector layers. As the source of data, we decided to process interactions between the argon beam and plumbum target. We set up the energy of the beam to 3.2 GeV, and the magnet amperage to 1250 A. We configured the following target distribution: X (mean: 0,7 cm, std: 0,33 cm), Y (mean: -3,7 cm, std: 0,33 cm), Z (center: -1,1 cm, thickness: 0,25 cm). Finally, we generated 1 million of events.

Some events in the data may include up to 100 tracks, but on average there are 37 tracks per event. In extreme cases the number of hits on one station may exceed 500 hits. Worth to note, that about 68% of all generated hits are fakes. Hits were marked either with the identifier of the track (from 0 to N) or with the fakes label (-1). About 51% of tracks don't have missing hits, 14% of

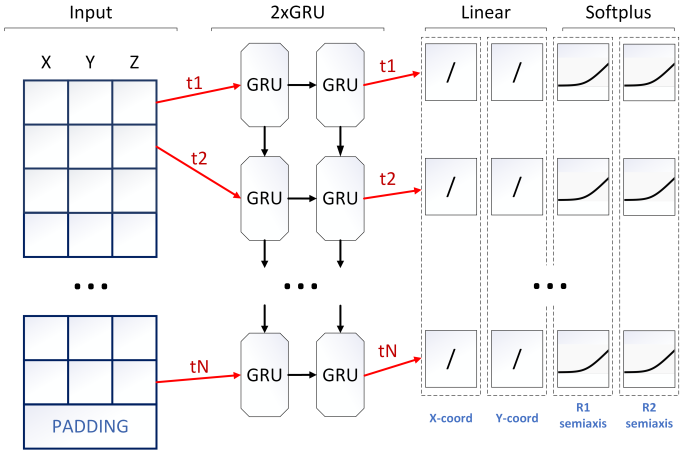

Figure 3: Main scheme of the TrackNETv3 model

all tracks don't have a hit on the first station, 23%, 11% and 10% of all tracks contain 1, 2 and 3 missed hits respectively. Tracks containing less than 4 hits, and spinning tracks were removed from the dataset by marking all their hits as fakes.

4. Neural Networks

4.1 Recurrent tracking

TrackNETv3 receives as input the coordinates of points of track-candidates used as seeds and predicts the center and semi-axes of the ellipse on the next coordinate plane, on which the candidate track continuation is searched. If any hits fall within the ellipse, several of them are selected with K-nearest neighbors search. Selected hits are appended to the track-candidate to prolong it and produce K new track-candidates. After all mentioned steps the model takes newly created seeds as inputs and repeats the prediction. The procedure continues until there are no more stations or the candidate track leaves the detector area.

If no hits fall within the ellipse, track is not prolonged and is either moved from seeds set to finished track-candidates set or dropped out as a too short candidate if the number of hits doesn't exceed 4. The main scheme of the TrackNETv3 model is presented in Fig. 3. It consists of 2 stacked recurrent layers with GRU units as base neural blocks. There are two fully connected layers on the top of the network to predict the parameters of the ellipse.

The base training procedure of TrackNETv3 is described in detail in [2]. Detector insufficiency was considered too. Tracks with missing hits have also been added to the training dataset. To ensure that the model handles such tracks correctly, changes in the model forward path are needed. We use a mask to specify holes in the track. Before returning the prediction, these holes will be replaced with coordinates of the virtual point as the center of the ellipse, calculated based on the previous hits of the track. Such points will not be taken into consideration during loss function measurement and backpropagation.

4.2 Graph Neural Network Classifier

Consider the event as a graph with hits represented as nodes. Nodes are fully-connected between the adjacent stations. Graph is represented as the following tensors - nodes feature matrix X, two edge matrices R_i and R_o (each edge matrix indicates how nodes are connected to each other), and the resulting vector Y (which masks the edges of graph).

The goal of Graph Neural Network (GNN) is to predict vector Y which has '1' for the edges which are part of the real track and '0' otherwise. Detailed construction of the GNN can be observed in [7]. Another approach on the base of GNN is to 'reverse' the graph described above (RDGraphNet approach). The idea of the reversed graph is to consider the edges from the original graph as nodes of the reversed graph, and pairs of nodes as edges, respectively. It allows us to take into account the connectivity between segments of track instead of simple hit-to-hit relations. Such approach significantly improves GNN tracking efficiency for the strip-based detectors with the cost of more memory consumption and decreased processing speed [8]. Extra care must be taken when applying RDGraphNet as it produces $O(N^3)$-size edge matrices where N is the number of hits of the event. Extra edge pruning step based on statistical information is performed to reduce the count of edges in the reversed graph, but it does not solve the combinatorial complexity of the algorithm. Looking at the amount of hits in BM@N RUN7 and SPD experiments, straightforward adaptation of the RDGraphNet approach can quickly run out of memory on a modern GPU and the approach needs to be revised and optimized further.

4.3 Combined tracking

TrackNETv3 aims to extract potential track-candidates from the raw event hits and has the ability to extract track-candidates of different lengths. The drawback of the method is its local nature. The locality means that the model sees only one particular track-candidate during the prediction phase and cannot rebuild it in time. Therefore, it may produce many false positive track-candidates. Fortunately, the combined approach greatly eliminates the vast number of fake edges if we consider TrackNETv3 track candidates as an input of the Graph Neural Network described above. GNN can now observe the whole event but looking only at hits from the potential track-candidates which are presented in the form of a plain (non-reversed) graph. Combination of the two models significantly improves the overall precision of tracking without high decrease of the recall, and at the same time, it greatly reduces the memory consumption of the GNN model.

5. Experiments

5.1 SPD

To evaluate the models, the following metrics were used:

$$recall = \frac{N^{rec}_{true}}{N^{in}} \tag{1}$$

$$precision = \frac{N^{rec}_{true}}{N^{rec}}, where \tag{2}$$

- N^{rec}_{true} - no. real tracks that the model has found;

	No. generated fakes	Preprocessing time (event/sec.)	GNN inference time (event/sec.)	Track building procedure time (event/sec.)
Ordinary graph	100 1000 10000	427.59 46.26 10.5	312.99 299.52 250.69	319.33 309.83 184.71
Reversed graph	100 1000 10000	261.52 32.61 12.51	208.94 211.57 166.64	248.86 214.43 134.59

Table 1: Speed comparison for different event representations and different phases of tracking

- N_{in} - no. all real tracks known from the Monte-Carlo simulation;
- N^{rec} - no. all tracks that the model reconstructed as real ones.

Recall indicates how many real tracks have been successfully extracted using the model. Precision represents the fraction of real tracks among all tracks reconstructed by the model.

To model detector inefficiency, each hit was saved with a given probability. Dropped hits are considered as detector failures. To reconstruct such tracks, the following procedure is used:

1. GNN makes predictions for the edges.
2. Classification threshold is applied to drop out not interesting segments.
3. All continuous track segments are collected (full tracks and parts of tracks).
4. Track segments are saved as pairs of nodes – node without input edge, node without output edge
5. IF
    a. The station number of the 1st element of the j-th pair is 2 more than the station number of the 2nd element of the i-th pair (e.g., node without output edge is at 2nd station, node without input edge from the second pair is at 4th station).
    b. AND the corresponding coordinate differences do not exceed $2d\phi_{max}$, $2dz_{max}$ and exceed $2d\phi_{min}$, $2dz_{min}$ (criterion from the graph pruning).
6. Connect such nodes and traverse the new graph

To get better results, track-candidates may be filtered out using polynomial interpolation, but we left it for future study.

Tables 1 and 2 show performance and efficiency comparison of event representation with ordinary and reversed graphs. All calculations were carried by Intel Pentium G4560 @ 3.50GHz. Reversed graph approach shows better recall and precision metrics in comparison to the ordinary graph for both ideal events and events with missing hits. At the same time, the usage of reversed graph leads to significant drop of event processing speed.

	Standard algorithm (no handling of missed hits)		Improved algorithm (dealing with inefficiency)	
	Ordinary graph	Reversed graph	Ordinary graph	Reversed graph
Recall	0.6767	0.6887	0.9104	0.9113
Precision	0.8582	0.9543	0.8324	0.9364

Table 2: Efficiency comparison for different algorithms

5.2 BM@N

To test the model on the BM@N data, the same metrics were used. However, we assume that the track is reconstructed if 70 percent of hits of this particular track were reconstructed correctly. This assumption makes the metrics not so strict, but it is expected that the track parameters may be restored using the majority of correctly reconstructed hits.

Table 3 shows that both recurrent and combined methods achieve high recall without considering tracks with missing hits but using GNN to filter out fake track-candidates leads to significant raise of track purity without drastic speed loss. The result of the combined method is acceptable to use in a real scenario.

	TrackNETv3	TrackNETv3 + GraphNet
Track efficiency (recall) (%)	97,67	94,89
Track purity (precision) (%)	2,35	74,98
Event/sec	12,78	7,56

Table 3: Recurrent and combined approach for tracking in BM@N

We used two different ways of inference for the case of missing hits. The first one - is an ideal situation where in addition to K nearest hits to the center of the predicted ellipse, we create a virtual point with the coordinates of that center, but we do this only for missing hits. Such an approach may help us to figure out the maximum that we can reach in terms of efficiency of tracking. As the simplest behavior in reality, we can always add a virtual point in addition to found event hits to prolong track-candidates. It produces a lot of fake candidates consisting of virtual points, so the number of missing hits per track was set to a maximum of 2. For each ellipse 2 nearest hits and the virtual point were considered as a possible continuation. Table 4 shows results of both approaches for the TrackNETv3 model measured on 23000 events. Time measurements were taken using the Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz and single Nvidia V100 32Gb GPU.

6. Conclusion

The current GNN model achieves promising results on generated data for SPD. RDGraphNet approach achieves even better results in terms of tracking efficiency with a slight decrease in processing speed, but still more sophisticated simulation is required.

To increase precision of the models for BM@N we tried to fit each track-candidate by helix to filter out tracks with Chi-square greater than a threshold. Attempts to use the helix fitting did not show any improvement.

	Recall	Precision	Mean processing time (events/sec)
Perfect case	0.9325	0.0911	13.72
With virtual point always added	0.8985	0.0168	1.82

Table 4: Different approaches to dealing with detector inefficiency

Handling tracks with missing hits is a specific process which requires extra tricks. For example, during the TrackNETv3 inference we can decide where to place a virtual point instead of prolonging the candidate with the hits from the predicted ellipse. When setting a point always, a large number of ghosts will be produced. We can use some simple classifier to reduce the number of ghosts consisting of virtual points. At the same time, such a method must not affect the performance of inference. We left it for future study. It is also required to train the GNN model for the case when we have tracks with missed hits.

Tracks fitting with parameters calculation is required for better analysis of the results.

Acknowledgments

The work was supported by the Russian Science Foundation under grant No. 22-12-00109. Calculations were carried out on the basis of the HybriLIT heterogeneous computing platform (LIT, JINR)[9].

References

[1] Frühwirth R. *Application of Kalman filtering to track and vertex fitting*, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. – 1987. – Vol. 262. – No. 2–3. – P. 444–450

[2] Nikolskaia A. et al. *TrackNETv3 with Optimized Inference for BM@N Tracking*, *Proceedings of the 9th International Conference "Distributed Computing and Grid Technologies in Science and Education" (GRID'2021), Dubna, Russia, July 5-9.* – 2021. – Vol. 3041. – P. 332-337.

[3] Shchavelev E. et al. *Global strategy of tracking on the basis of graph neural network for BES-III CGEM inner detector* /O. Bakina, I. Denisenko, P. Goncharov, Y. Nefedov, A. Nikolskaya, G. Ososkov, E. Shchavelev and A. Zhemchugov //*AIP Conference Proceedings*. – *AIP Publishing LLC*, 2021. – Vol. 2377. – No. 1. – pp. 060001.

[4] Savin, I., A. Efremov, D. Peshekhonov, A. Kovalenko, O. Teryaev, O. Shevchenko, A. Nagajcev, A. Guskov, V. Kukhtin, and N. Toplilin. *Spin Physics Experiments at NICA-SPD with polarized proton and deuteron beams.* In *EPJ Web of Conferences*, vol. 85, p. 02039. EDP Sciences, 2015.

[5] Kapishin M. et al. *Studies of baryonic matter at the BM@ N experiment (JINR)* //Nuclear Physics A. – 2019. – Vol. 982. – pp. 967-970.

[6] K. Gudima, S. Mashnik, and A. Sierk. *User Manual for the Code LAQGSM*. Tech. rep. LA-UR-01-6804. Los Alamos National Laboratory, 2001.

[7] S. Farrell et al., *Novel deep learning methods for track reconstruction*, in *4th International Workshop Connecting The Dots 2018 (CTD2018)*, Seattle, Washington, USA, March 20-22, 2018 (2018), arXiv:1810.06111 [hep-ex].

[8] Shchavelev E. et al. *Tracking for BM@N GEM Detector on the Basis of Graph Neural Network* / E. Shchavelev, P. Goncharov, G. Ososkov, D. Baranov //*Proceedings of the 27th Symposium on Nuclear Electronics and Computing (NEC 2019), Budva, Montenegro*. – 2019. – Vol. 2507. – pp. 280-284.

[9] Adam G. et al. *IT-ecosystem of the HybriLIT heterogeneous platform for high-performance computing and training of IT-specialists // Selected Papers of the 8th International Conference "Distributed Computing and Grid technologies in Science and Education" (GRID 2018), Dubna, Russia*. – 2018. – Vol. 2267. – pp. 638-644.