BM@N Tracking with Novel Deep Learning Methods

Pavel Goncharov[1,⋆], *Egor* Shchavelev[2], *Gennady* Ososkov[3], and *Dmitriy* Baranov[3]

[1]*Sukhoi State Technical University of Gomel, October Ave. 48, 246746 Gomel, Republic of Belarus*
[2]*St. Petersburg State University, Universitetskaya Emb. 7/9, 199034 Saint Petersburg, Russia*
[3]*Joint Institute for Nuclear Research, Joliot-Curie 6, 141980 Dubna, Moscow region, Russia*

Abstract. Three deep tracking methods are presented for the BM@N experiment GEM track detector, which differ in their concepts. The first is a two-stage method with data preprocessing by a directional search in the k-d tree to find all possible candidates for tracks, and then use a deep recurrent neural network to classify them by true and ghost tracks. The second end-to-end method used a deep recurrent neural network to extrapolate the initial tracks, similar to the Kalman filter, which learns necessary parameters from the data. The third method implements our new attempt to adapt the neural graph network approach developed in the HEP.TrkX project at CERN to GEM-specific data. The results of applying these three methods to simulated events are presented.

1 Introduction

In fixed target experiments, such as the Baryonic Matter at Nuclotron (BM@N) [1], products of heavy ion interactions are recorded by the GEM track detector [2]. In the considered case it consists of six coordinate stations formed by sensitive electronic elements, in which a signal appears, induced by a passing particle. Those elements are linear parallel thin strips on a silicon substrate. The overflying particle activates several strips in the coordinate plane around the point of its passing. Another close plane with a different strip direction is needed to get the second coordinate. The intersection of the "fired" lines will give us a hit with the coordinates of a space point close to that at which the particle has flown. It is worth noting a serious drawback of the strip detectors, due to the fact that, for a large number of particles, in addition to the real intersections corresponding to the point of the passage of a particle, there appear, by an order of magnitude more numerous, false intersections, called fakes, considerably contaminating the results of the measurement.

The main task of the physicists analyzing the experimental data is to reconstruct all the tracks of each event in order to obtain the physical parameters of the charged particles producing these tracks. The track reconstructing procedure named tracking was historically based on the Kalman Filter (KF) [3]. Up to now KF was the most used and effective tracking algorithm. However, the initialization procedure needed to start Kalman filtering requires a really vast search of hits needed to obtain so-called "seeds", i.e. initial approximations of track parameters of the charged particles. Moreover, in the case of tracking GEM data with its very complex contamination by fakes, the initialization problem is especially complicated. There are several approaches to solve this case [4], but the KF methods are, generally, sequential by their nature and scale poorly with the expected increase in detector occupancy under new conditions, as for the planned NICA experiments. At the same time, a

⋆e-mail: kaliostrogoblin3@gmail.com

great contribution to tracking problems can be done by deep learning algorithms due to their capability to model complex non-linear data dependencies, to learn effective representations of high dimensional data through training, and to perform the algorithm parallelization on high-performance computing means such as GPUs. It is worth noting that there not exist whatever deep learning solutions suitable for tracking in GEM detectors.

Further, the results of applying a deep neural network will be described in generally accepted statistical metrics, such as accuracy, precision and recall [5]. Accuracy (also known as efficiency) is the fraction of predictions our model get right, but it becomes useless while dealing with an imbalanced dataset. Therefore, precision and recall metrics are used in these cases as more informative. Precision tells us how many of the objects classified as true tracks were correct. Thus, recall expresses the ability to find all true tracks in a dataset, while precision expresses the proportion of data, our model says was true, actually were true tracks.

2 Initial attempts and new solution of deep tracking approach

We have already made two former attempts with "deep" tracking, i.e. track reconstruction with the help of deep learning. At the first attempt, a preprocessing by directed k-d tree search [6] was performed in the first step to find all possible track-candidates as clusters joining all hits from adjacent GEM stations lying on a smooth curve. Then in the second step, a deep recurrent network (RNN) trained on the dataset from the obtained track-candidates was used to classify them in two groups: true tracks and, so named, ghost tracks formed by fakes and, possibly, by parts of different tracks citeRef7.

However, our two-step solution spent too long time while building a k-d tree structure for every event always from scratch. Therefore, in our next attempt, the new model combining both stages in one end-to-end neural network TrackNETv1 was proposed [7]. The special trinomial cost function was invented to train the whole system where a focal loss multiplier [8] was used to deal with the very imbalanced training dataset. TrackNETv1 performes simultaneous search of the continuation of track-candidate and classification whether it belongs to true track or not. Despite achieving promising result, we observed that we can simply drop the classification part at all because the ellipse prediction comprises the track smoothness criterion by itself. By removing the classification part, we open the opportunity to train a single model end-to-end using only true tracks, which can be accurately extracted from Monte-Carlo simulation. Also, by removing the classification part we could simplify the loss function bringing more stability to the training process. So, we came to the new TrackNETv2 model. Its full description was reported in [9].

As data for training and evaluation of the proposed neural network, we used 550K 4 GeV simulated events of C+C interactions, specific for the BM@N run 2017. To prepare a clean train data we removed tracks containing less than 3 hits and spinning tracks registered more than once per station and, finally, we labeled all true hits by seeking for the corresponding Monte-Carlo points. Eventually, we have three data sets for the training (1 405 556 true tracks), for the validation (351 388 tracks) and for the testing (150 K events without any preprocessing and with presence of fakes). While evaluating TrackNETv2, we observed an obvious bias in the accuracy of the model to the tracks with the maximum length while for the shorter tracks the accuracy drastically felt. It happened because of the great imbalance in the distribution of the tracks with different lengths. To overcome such imbalance, a special batching strategy was applied. The mean accuracy for tracks of various lengths is about 96 % (where the accuracy is being defined as the fraction of tracks which can protracted from the very beginning to the last station without any mistake).

3 The graph neural network approach to the GEM data

The report about the success of the Graph Neural Network (GNN) approach for the HENP particle tracking by the HEP.TrkX project for the LHC detectors [10] inspired us to propose our GNN application for the simulated data from the GEM detector of the BM@N experiment. At the HEP.TrkX solution an event is represented as a graph where nodes are registered hits and edges are the segments fully connecting those nodes between adjacent detector planes. It should be stressed that the main difference between the LHC and GEM data is that the pixel detectors at the LHC do not produce fake hits, while in the GEM most of the hits are fakes, which makes it difficult the adaptation of the HEPTrkX GNN to the case of GEM. We observed that the average true-to-fake segment ratio of the whole GEM dataset is about 1:120, as it is shown in Fig. 1, where there are only about 40 true track segments (thick solid lines) while the others (>4500) are fakes (thin dashed lines).

Therefore, the straightforward adaptation of the GNN approach to the GEM data did not bring any decent results giving poor results as compared to the 99 % reported in [10]. We overcome such huge dataset imbalance with a data reduction (normalizing and transforming XYZ space into cylindrical one). Moreover, a new solution came from the old idea of using the Minimum Spanning Tree (MST)

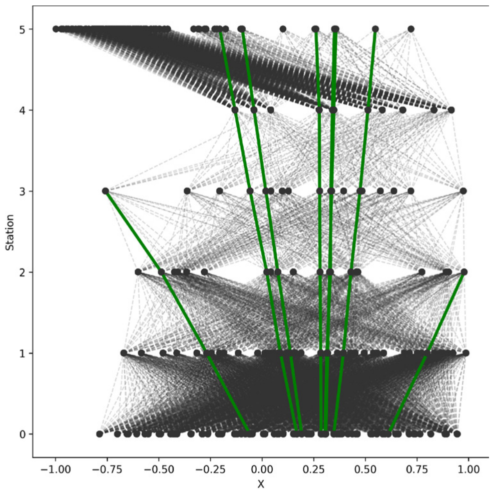

Figure 1. (Color online) The event-as-a-graph representation of the C + C 4GeV simulated BM@N experiment event. Dots are the hits, lines are the track-candidate segments.

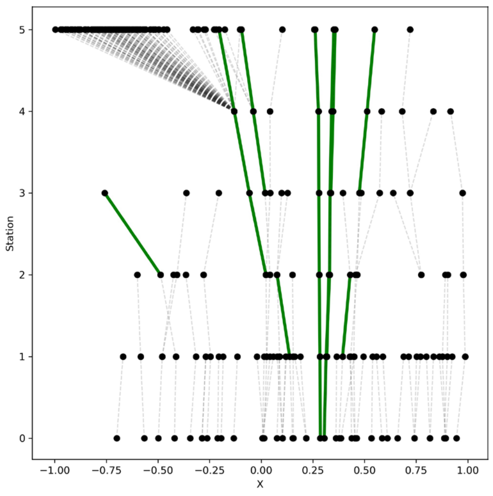

Figure 2. (Color online) Graph of the same event after applying the MBT algorithm. Green bold lines are true segments, dashed gray are fakes. Coordinate axes normalized by the detector dimensions

for the reconstruction of bubble tracks, dating back in 1978 [11]. The spanning tree is a subset of the edges of a connected, undirected graph that connects all the vertices together, without any cycles. The Minimum Spanning Tree (MST) is a spanning tree for an edge-weighted graph with the minimum possible total edge weight. To prevent the MST algorithm possible tendency to pick back as the shortest path, we have to use directed graphs, i.e. the Minimum Branching Tree (MBT) algorithm which is the MST but on the directed graphs. As the graph edge weight, the Euclidian distance between two hits in cylinder coordinate space was used.

Thanks to the MBT algorithm preprocessing the real-to-fake ratio was radically reduced by 12 times, from 1:120 to 1:10. One can see that in Fig. 2. The model achieves 99 % accuracy, 92 % recall and 0.983 AUC score [5] for the segments obtained from the MBT algorithm. However, it should be admitted that these results were obtained on data after their preliminary filtration and application of the MBT algorithm, during which, on average, up to 18 % of the real graph edges were lost (as illustrated in the Fig 2, one can see that some segments of the real tracks, represented with the bold lines, are missing). The overall recall (computed as the count of correctly predicted track segments divided by the total track segments before the preprocessing) is 77 % and the overall precision is 96 %.

Given the fact that the long 6-layer tracks dominate the data set (which is the same case for the TrackNETv2), we observed that MBT behaved poorly on the short tracks. However, this result was obtained from scratch, following the almost instant leap to the 77 % recall and 96 % precision, so one should expect that this approach is really promising and could be significantly improved in the future.

4 Conclusions

The new fully end-to-end trainable tracking network TrackNETv2 [9] has the following advantages: it does not require the tremendous fake tracks preparing via directed search, has fewer parameters for tuning due to the lack of the classification part, achieves 0.962 accuracy and can process around 16 K tracks/sec on a single Intel Core i3-4005U @1.70 GHz.

For the time being, we are improving the TrackNETv2 behavior dependence on the volume of information about the vertex location.

The Graph Neural Network approach for the LHC detectors was adapted and successfully developed to handle data from GEM detectors. The idea of data preprocessing by the Minimum Branching Tree (MBT) algorithm allowed to improve GNN results achieving a promising 99 % accuracy on the preprocessed dataset. However the MBT use resulted in 18 % true segments loss on the whole dataset since MBT behaved poorly on short tracks. All results where obtained with simulated data.

In the next future we are going to improve the MBT algorithm to overcome the inaccuracy related to the segment purification; to port the models and the code to a C++ package in order to speed up computing time; to expand the ability of the model to solve tracking in a collider environment.

Acknowledgment

The reported study was funded by RFBR according to the research projects №19-57-53002 and №18-02-40101.

References

[1] M. Kapishin, The European Physical Journal A **52**, 8, 213 (2016)
[2] F. Sauli, Nuclear Instruments and Methods in Physics Research Section A **805** 2–24 (2016)
[3] R. Frühwirth, Nuclear Instruments and Methods in Physics Research Section A: **262**, No. 2–3 444–450 (1987)
[4] D. Baranov, S. Merts, G. Ososkov, O. Rogachevsky, EPJ Web of Conferences **108**, 02012 (2016)
[5] M. Sokolova, N. Japkowicz, S. Szpakowicz, AI 2006: Advances in Artificial Intelligence, Lecture Notes in Computer Science **4304** 1015–1021 (2006)
[6] B. Louis, Multidimensional binary search trees used for associative searching. Communications of the ACM **18.9** 509-517 (1975)
[7] D. Baranov, G. Ososkov, P. Goncharov, A. Tsytrinov, EPJ Web of Conferences **201**, 05001 (2019)
[8] T.Y. Lin, et al., `arXiv:1708.02002` (2017)
[9] D. Baranov, G. Ososkov, P. Goncharov, AIP Conference Proceedings **2163**, 040003 (2019)
[10] *HEP.TrkX project*, `https://arxiv.org/abs/1810.06111`
[11] G. Ososkov, V. Pahomov., Comm. JINR, B10 11-11264, Dubna, 288–293 (1978)