TRACKING FOR BM@N GEM DETECTOR ON THE BASIS OF GRAPH NEURAL NETWORK

E. Shchavelev[1], P. Goncharov[2], G. Ososkov[3], D. Baranov[3]

[1] *Saint Petersburg State University, 7/9 Universitetskaya Emb., 199034, Saint Petersburg, Russia*

[2] *Dubna State University, Universitetskaya 19, Dubna, Moscow Region, 141982, Russia*

[3] *Joint Institute for Nuclear Research, 6 Joliot-Curie street, Dubna, Moscow region, Russia*

E-mail: egor.schavelev@gmail.com

Particle tracking is a very important part of modern high energy physics experiments. While the data stream from such experiments is increasing day by day, classic tracking methods lack the ability to fit these amounts of data. In order to solve this problem, new effective machine learning algorithms are actively developed in the HEP.TrkX project for Large Hadron Collider detector and for the GEM detector of the BM@N experiment. This work is a logical continuation of the research presented on the XXIII International Scientific Conference of Young Scientists and Specialists (AYSS-2019), where we have already introduced a new application for tracking with the Graph Neural Network based on the Minimum Branching Tree preprocessing. However, that approach has had some problems, including the overall inaccuracy with the segments purification of the preprocessed event graph. In this work, we overcome many of such problems and introduce a revised approach with improved tracking accuracy. Promising results of the improved GNN tracking are given.

Keywords: tracking, deep learning, graph neural networks, line graph, digraph

Egor Shchavelev, Pavel Goncharov, Gennady Ososkov, Dmitriy Baranov

1. Introduction

Modern high energy and nuclear physics (HENP) requires precise and effective reconstruction of events occurring during its experiments. One of the most important parts of event reconstruction is tracking of the particle trajectories (tracks). This process called particle tracking or just tracking, and it demands swiftest and most accurate algorithms to operate on, as in the modern experiments beams of heavy ions are collided on the very high speeds, and particle collisions produce enormous amounts of traces of the secondary particles in the sensitive elements of tracking detectors. These traces are to be converted then into so-called 'hits', which is usually points in some coordinate space, and one of the main parts of the event reconstruction process is to search for groups of those hits each connected by affiliation to the same track.

As we are working on the BM@N experiment of the NICA megaproject [1], we obtain hits information from the microstrip gaseous chamber GEM detector which has the one main drawback: with any N actual particles hits, we also get $N^2 - N$ spurious hits. Although this problem has successful classical solutions, as the Kalman filter method [2], it meets difficulties at achieving satisfying speed on the events with the heavy ions in the recent BM@N experiments due to its sequential nature and problems with parallelization.

At the same time, machine learning approaches have the capability of modeling such complex environments and can be effectively run in parallel. There are some existing machine learning approaches for tracking [8], but a majority of them are sequential, they operate over the chain of hits and do not process the whole event at a time and can miss important informational patterns. One of the possible options to represent an event is to consider it as a graph. This approach is not completely new, there is a successful approach based on the Hopfield Network [3], but it will not be able to handle properly current amounts of data.

Recently the idea of graph representation in machine learning found great success with the Geometric Deep Learning methods known as Graph Neural Networks (GNN) [4], and the recent HEP.TrkX approach [5] based on GNN showed the huge potential for the data from the LHC pixel-based detector. We were inspired of this approach, so in our previous study, we tried to adopt the same approach for the data from the GEM detector of the BM@N experiment [6]. However, our approach was not able to achieve similar success compared to the LHC one. In this work, we propose some improvements which lead us to more precise and robust results.

2. Previous study

The key idea of our approach is an event-as-a-graph representation when registered hits are nodes of the graph and edges of the graph fully connect the nodes between the adjacent layers of the detector. One can see the full graph of a particular event simulated for the GEM BM@N in figure 1.

GNN consumes the nodes' features, namely, 3-D coordinates (X, Y, Z) of the hit and their connectedness of each other and the adjacency matrix of nodes (hits). The forward pass of the model consists of iterating data through the 2 subnetworks, so-called 'Edge' and 'Node' networks. The Edge network selects nodes' features and the Node network aggregates neighbor node features. The full details and explanation of the model one can find at [5].

During the integration of this approach for the GEM data we recognized the biggest pitfall of the initial approach – while it works on the data without the fake hits perfectly, it is highly afflicted by the great fake count, which is a common problem of the microstrip-based detectors like GEM. This approach cannot handle huge noise rate and for that purpose, we had to introduce a special filtering process containing two main steps:

- Truncating the segments based on their statistical behavior (i.e. cutting all segments which angle exceeds some constant). This step brings substantial influence on the overall noise rate, reducing the noise factor by half.
- Utilizing the Minimum Branching Tree. It brings a huge impact on the overall fake segment rate. We discovered that MBT-based filtering reduces the fake-to-real ratio from 1:120 to 1:10 (for one true segment we observe 10 fake segments). But this algorithm has one big disadvantage – it discards about 18% of true segments.

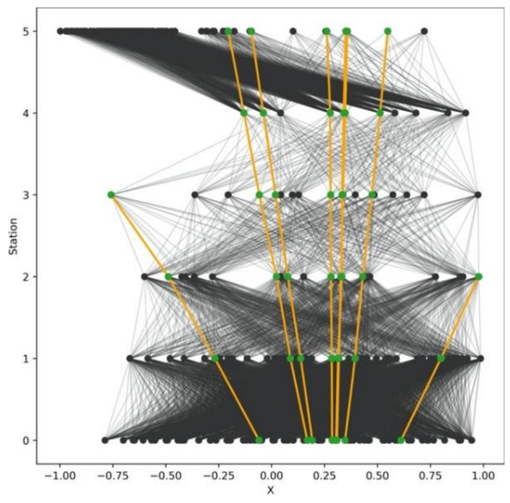

Figure 1. Event-as-a-graph representation of a 4 GeV C+C BM@N experiment event. Orange edges, also called 'segments', are parts of true tracks, black edges represent fake segments. Green nodes are true hits, black nodes are fakes produced by the GEM detector

After applying the filtering described above, GNN showed promising results achieving 77% recall and 96% precision on the preprocessed data. This result has inspired us towards the continuation of our studies, so we needed to improve the filtering process and increase the overall network accuracy.

3. New event representation

3.1 New network feature idea

The common straightforward method for improving neural network model accuracy is to feed the network with more features. As it was mentioned above, the GNN model uses a hits 3-D coordinate position as a feature. Looking at the physical interpretation of particle track one can notice that track in the GEM detector is usually a smooth curve that has a concavity, which is not varying much, being approximately constant. In other words, the track curve has about the same value for its first-order derivative (and always the same sign). The information of the curve monotonicity and concavity can be obtained as the gradient of particle track function and its Laplacian, correspondingly. We would like to use such characteristics as features and feed it to the network. But the network consumes only the hits positional information as nodes features. The information about the segment direction ('monotonicity') can be obtained only if we use it as an edge feature (because segment direction depends on the two nodes positions, which cannot be encoded into a single node). Unfortunately, current GNN architecture utilizes the graph adjacency matrix as a binary mask for the network message passing mechanism so it is considerably hard to encode such information to the current event-as-a-graph representation. Therefore, we found a new type of graph representation that is more in line with our intentions.

3.2 Line graph

Line graph [7] of a graph G is another graph $L(G)$ that represents the adjacencies between edges of G. Given a graph G, its line graph $L(G)$ is a graph such that:

- each node of $L(G)$ represents an edge of G; and
- two nodes of $L(G)$ are adjacent if and only if their corresponding edges share a common endpoint in G.

Roughly speaking, the line graph inverts the nodes and edges of the original graph. We also can use the directed version of line graph, called 'line digraph'. If G is a directed graph, its directed line graph or line digraph has one vertex for each edge of G. Two vertices representing directed edges from a to b and from c to d in G are connected by an edge from ab to cd in the line digraph when $b = c$. That is, each edge in the line digraph $L(G)$ represents a length-two directed path in G. The biggest advantage of such transformation is the fact that *edge* features from the initial graph turn into *node* features of the line graph which becomes really convenient to use within the GNN model.

3.3 Applying line digraph approach

Consider the initial event-as-a-graph conception we are transforming the initial graph G to the line digraph $L(G)$. As an initial graph fully-connects adjacent layers nodes we can apply the same approach described above: each (graph) path of two consecutive segments will be the edge of $L(G)$; each segment will be the node of the new line digraph $L(G)$. In this situation, each node of $L(G)$ gathers two 3-D coordinate positions from the two initial event hits. One can see the resulting line digraph representing the same event as in figure 1 is depicted in figure 2 where almost all tracks become more straighten in such representation because they are in fact first-order derivatives of initial tracks.

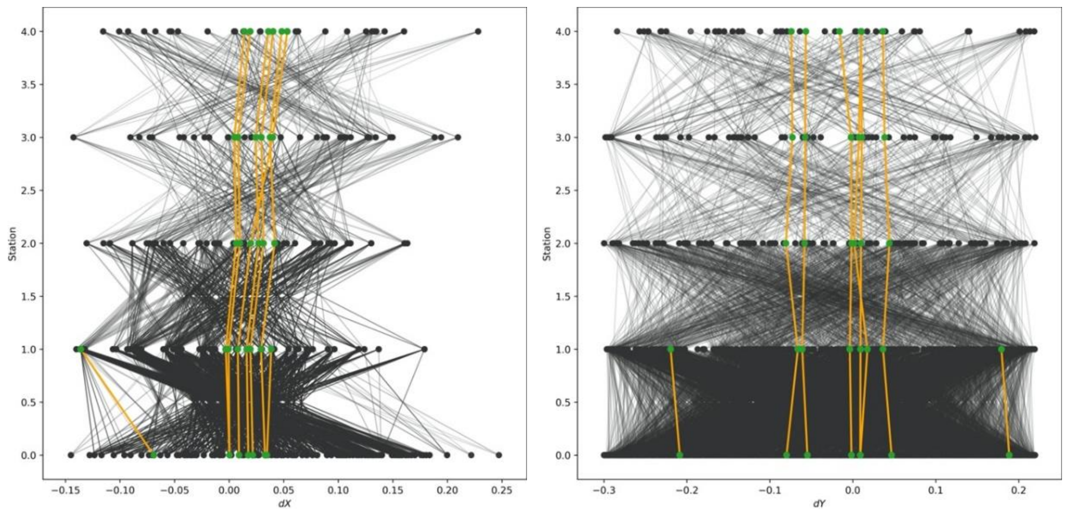

Figure 2. Line graph representation of the same event shown in figure 1. Two pictures show both XoZ and YoZ planes. Nodes of this graph (being the segments of the initial graph) here are being positioned by their $dX = x_i - x_{i-1}$ coordinate, where x_i is a segment's end X coordinate and x_{i-1} is a segment's start X coordinate (same for the YoZ plane)

Despite the fact that this representation brought new features (dX and dY values) to the network model, it still suffers from a big amount of fake segments. But now filtering fake segments is a much simpler procedure: the segment of a line digraph now, in fact, represents hits from three consecutive layers, so we can compute the second-order derivative of a curve which passes through these hits. So, the current filtering process is the following:

- construct line digraph;
- compute edges weight; edge weight computed as $w = (d^2X, d^2Y)$, where d^2X is computed as $d^2X = dX_1 - dX_0 = (x_2 - x_1) - (x_1 - x_0) = x_2 - 2 * x_1 - x_0$. x_2, x_1, and x_0 are three consecutive hits' X coordinates of a track curve.
- filter out all segments with weight value more than a constant.

This filtering process achieves 100% purity (purity is the count of true segments left after filtering divided by the count of initial true segments) and reduces overall fake segment rate up to 10 times (fake segment rate is the count of fake segments divided by the count of true segments). It is worth to note that constructing line digraph, finding and filtering segments can be computed with up to $O(N)$ operations with usage up to $O(N^2)$ memory, where N is a number of segments in the graph. We also vectorized this algorithm and achieved the resulting speed about ~7 events/sec; which is, in fact, satisfying for the current study but an important thing to improve in the future.

3.4 Graph Neural Network training and results

We used the same network architecture and structure described in [6] except feeding it with segments produced by the filtered line digraph. During the network training, we observed that optimal feature configuration is a vector of the following five line digraph node features: $(dX_0, dX_1, dY_0, dY_1, Z)$. With this configuration, our model achieves 85% precision and 96% recall on the validation dataset containing 4k events.

Moreover, we also have measured more strict metrics introduced in [8]: it achieves 94% hit and 87% track accuracy (hit accuracy is a percentage of true hits found by network out of all true hits in a single event; track accuracy is a percentage of full tracks without gaps found by network out of all tracks in a single event). Although overall results are totally satisfying, we discovered the main shortcoming of the line digraph approach – our algorithm runs out of memory when handling events with a huge number of hits (more than 3000 hits in a single event). On the other side, the total percentage of such events in the whole dataset is about 1-2% out of all events. We will investigate and resolve memory issues in future studies.

4. Conclusion

In this paper, we have introduced a novel approach for the effective graph representation of the event in GEM detector – line digraph. It facilitated both filtering and feature extraction processes. GNN being trained on line digraph achieves significant results even with the strict metrics reaching 94 % hit accuracy and 87% track accuracy and handles tracks of any length. The main drawback of the line digraphs is high memory consumption (because it is quadratic from the total hit count in the event). Now we are going to speedup the filtering process; embed line graph architecture into the network model itself; overcome memory issues; expand the approach to solve tracking in a collider environment.

5. Acknowledgement

The reported study was funded by RFBR, project number 18-02-40101.

References

[1] M. Kapishin, The fixed target experiment for studies of baryonic matter at the Nuclotron (BM@N) // The European Physical Journal A. – 2016. – V. 52. – No. 8. – pp. 213.

[2] R. Frühwirth, 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. – pp. 444-450.

[3] G.Ososkov et al, Neural Network Applications for Efficiency Improving of Geometric Reconstruction of Events Detected in the EXCHARM Experiment at the JINR, Proc. of VI Intern. Workshop on Software Engineering, Artificial Intelligence and Expert Systems, (Greece), 1999

[4] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," CoRR abs/1611.08097 (2016), arXiv:1611.08097.

[5] 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

[6] E. Shchavelev et al., Graph neural network application to the particle track reconstruction for data from the GEM detector // AIP Conference Proceedings, 2019. – T. 2163. – №. 1. – C. 040003.

[7] H. Frank, R. Norman, "Some properties of line digraphs" in Rendiconti del Circolo Matematico di Palermo, 1960, V. 9, pp 161-168.

[8] P. Goncharov, G. Ososkov, D. Baranov Particle track reconstruction with the TrackNETv2 //AIP Conference Proceedings. – AIP Publishing, 2019. – T. 2163. – №. 1. – C. 040003.