NEURAL NETWORK APPROACH TO TRACKING PROBLEMS OF HIGH ENERGY PHYSICS

G.A. Ososkov[1a], P.V. Goncharov [1,2], A.V. Nechaevskiy[1], E.M. Shchavelev[3], A.N. Nikolskaya[3]

[1] *Joint Institute for Nuclear Research,6 Joliot-Curie St, Dubna, Moscow Region, 141980, Russia*

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

3 Saint Petersburg State University, 7-9 Universitetskaya emb., Saint Petersburg, 199034, Russia

E-mail: [a] ososkov@jinr.ru

The reconstruction of charged particle trajectories in tracking detectors is a key problem in the analysis of experimental data for high energy and nuclear physics. The amount of data in modern experiments is so large that classical tracking methods such as Kalman Filter cannot process them fast enough. Taking into account advantages of deep neural networks we have developed two algorithms of track recognition, based on deep learning architectures, for local (track by track) and global (all tracks in an event) tracking in the GEM tracker of the BM@N experiment at JINR (Dubna). Our neural network tracking algorithms are especially focused on overcoming the well-known difficulty inherent in GEM detectors, which, in addition to real measurements, generates many false measurements, which significantly complicate the tracking procedure. Particular attention is also paid to the problems of accelerating the learning procedures of deep neural networks. The results of test runs on the GOVORUN supercomputer and the HybriLIT cluster are presented, which shows the gain in computing power due to optimizing the sharing of resources between the training and validation parts of the tracking program.

Keywords: Track reconstruction, GEM detectors, Deep learning, Convolutional neural networks, Graph neural networks, supercomputer GOVORUN

Gennady Ososkov, Pavel Goncharov, Andrey Nechaevskiy,
Egor Shchavelev, Anastasia Nikolskaya

1. Introduction

In any analysis of experimental data in high energy and nuclear physics, it is extremely important to estimate as accurately as possible the kinetic parameters of particles formed during their interactions in experimental facilities. For these purposes, track detectors are installed, designed to record the trajectories of particles, called tracks. Those detectors are located near the zone of collision of a beam of charged particles with a target in the case of experiments with a fixed target or with each other in collider experiments. To follow the particle trajectory, track detectors are equipped with a set of coordinate planes, or so-called stations. A charged particle flies through a certain number of detector stations, releasing at each of them a portion of energy, which is registered in a way specific to the given detector, and from the results of this registration, the hit is calculated as a spatial coordinate. Besides depending on the design of the detector, in addition to the detector responses caused by the passage of the desired particle, the event may also contain hits caused by noise or false reconstruction of corresponding hits. For example, the problem of combining hits in track candidates on successive coordinate planes of a detector is seriously aggravated by the well-known drawback of popular multiwire and strip detectors, such as GEM detectors. To get hits in such detectors, two sets of strips intersecting at some angle are required. The charge-weighted sums of activated strips give us two coordinates, so the hit position is determined by their intersection. Therefore, in case of many tracks false intersections of fired strips occur inevitably, leading to the appearance of false hits, so-called fakes, and the number of these fakes grows quadratically with the growth of genuine hits. To determine the parameters of the particles, you need to restore the whole collision event and each of its tracks. The task of reconstruction of particle trajectories is called tracking and consists in the procedure for collecting separate groups of hits according to some criterion of belonging to a certain track. One can see that in cases of such detectors like GEM producing data contaminated by fakes and noise hits the tracking procedure becomes rather complicated from the both points of view as the algorithms used for fake removing and as the cost of computing resources.

Evaluation of the effectiveness of tracking methods is carried out using Monte Carlo data for modeling experimental interactions by comparing reconstructed trajectories with already known simulated tracks. The technological development of collider experiments aimed to increase the energies of particle beams and their luminosity (particle density in the beam) inevitably leads to a growth of the rate of experimental data stream at each stage of this evolution. Tracking methods have evolved along with the development of experimental installations: from intuitively understandable to any person ways of tracking along a path (along the trajectory of a particle), to complex models of deep artificial neural networks.

The remainder of this article briefly highlights the evolution of tracking methods against the backdrop of advances in particle accelerators and computation, describes the authors' work on tracking by deep learning (DL), and illustrates the important role of high-performance computers on example of using DL to analyze experimental data of the BM@N experiment [1].

2. Evolution of tracking algorithms

Tracking algorithms have evolved along with the development of experimental facilities and technologies for registering particles in track detectors, moving to a new level with the advent of new methods of particle acceleration and more complicated types of detectors. In experiments of the 60s with bubble chambers, events in them were recorded on stereo photographs and points on the track were entered into a computer by an operator guiding coursor manually along the trajectory of a particle observed under a microscope. With the growth of the data stream, scanning devices of the Spiral Reader type [1] appeared, in which the operator put a point at the event vertex, from where the photograph was scanned along a spiral with a radial slit, while the coordinates of all registered points transmitted to the computer. Tracking this information required the use of coordinate transformations to "straighten" the tracks [2] and use the modification of the Hough transform to detect them [3]. Despite the advantages associated with the simplicity of the implementation of algorithms based on coordinate transformations, both they and the Hough transform do not allow effective work in three dimensions, moreover, the assumption of the homogeneity of the magnetic field used in these methods is not always fulfilled. With

the advent of electronic experiments, measurement data began to be digitized and immediately fed directly into a computer. After multi-stage filtering and alignment procedures, the time came for tracking, among the methods of which it is worth noting the first applications of artificial neural networks, both multilayer perceptrons and Hopfield's fully connected neural networks proposed by B. Denby [4]. The imperfection of these neural network algorithms, a sharp drop in their efficiency with an increase in the noise of track data and the multiplicity of events in collider experiments led to the emergence of a modification of Hopfield's neural network algorithms, called elastic neural networks [5]. Unfortunately, the effective use of elastic tracking methods is greatly complicated by the need to choose an initial approximation. However, with the further development of colliders and computer technologies, the most effective tracking method turned out to be the classical method using the Kalman Filter (KF) [7], since it allows one to take into account the inhomogeneity of the magnetic field, multiple scattering and energy losses when a particle passes through the detector medium. KF extrapolates the initial state of the track (as a rule, the first three points) to a small area on the next coordinate surface. The state vector $\vec{x} = (x, y, t_x, t_y, q/p)^T$ is iteratively evaluated to predict the position of the track on the next coordinate plane taking into account changes in the covariance matrix and error corridors. To test the hypothesis that the next hit near the extrapolated track belongs to this track, the criterion $\chi 2$ is used. Further, the procedure is repeated taking into account the new found hit. If the event has many tracks lying close to each other in space, a combinatorial version of the Kalman filter (CKF) [8] is used. For all tracks for which there are several possible continuations, CKF forms branches, trying to extrapolate original tracks in several directions. At the end of the procedure, false tracks are rejected according to the criterion $\chi 2$. The main KF disadvantage is the need in a cumbersome preliminary search of the first track hits (so-called seeding) to calculate the initial value of the state vector $\vec{X}$.

Despite the success of KF and many optimizations that can significantly reduce the time to find the initial state vector [9], this method has more disadvantages associated with its computational complexity and significant difficulties in the implementation of parallel computational schemes for forming tracks. The disadvantages of KF are especially pronounced in experiments with a huge multiplicity of tracks. This is, for example, the BM@N NICA [1] experiment with heavy ion beam, or experiments with high luminosity, such as at the Large Hadron Collider (LHC) [10]. By 2024, the fourth run of the LHC is planned, physicists expect about 10 thousand tracks by event, which will certainly mean a crisis of classical tracking [11]. In an attempt to reduce the computational complexity of KF, a method was proposed using a cellular automaton to set the initial state of Kalman filtration [9]. However, the program that implements the use of the cellular automaton is not documented, which makes it very difficult to adapt. With the development of parallel computing infrastructures, methods of deep learning are gaining more and more popularity due to their ability to detect hidden nonlinear dependencies in data and the ability to parallelize linear algebra operations that underlie these methods. The prospect of using deep learning methods for neural networks in the task of reconstructing tracks gives rise to the problem of creating effective algorithms for "deep tracking" that surpass classical algorithms like KF in speed.

3. Local tracking for the BM@N experiment

A two-step approach has been proposed for reconstruction of particle tracks as our first attempt to apply deep neural networks to this task [12]. We started with the first stage of spatial search for candidate tracks. The search was accelerated using the KD-tree technique and taking into account the magnetic field of the detector. A labelled sample of 82,677 tracks was obtained with labels "true" and other 695,887 tracks were classified as "false". The software implementation of this stage turned out to be very slow, and the sample was highly imbalanced. At the second stage, for the classification of track candidates, a deep recurrent network TrackNETv1 with convolution at the input and two BiGRU layers [13], as well as a special loss function that takes into account the sample imbalance, was used. After network training the result of the verification test was rather good: correct classification of candidate tracks for real tracks and false (ghosts) with 97% of efficiency. The trained RNN processed 10,666 track candidates per sec on one Nvidia Tesla M60 from the HybriLIT cloud service [14] and 34,602 candidate tracks per second using the Tesla V100 on the GOVORUN supercomputer [15].

The rich capabilities of the TrackNETv1 recurrent neural network allowed us to develop it in such a way as to overcome the difficulties of the 1st stage, by combining two stages in one - end-to-end - with a regression part of four neurons, two of which predict the center point of the ellipse on the next coordinate plane in which look for the continuation of the candidate track, and two more - determine the sizes of the semiaxes of this ellipse. In the resulting TrackNETv2 neural network, the prediction of the ellipse itself includes a track smoothness criterion, therefore, the classification part is no longer needed and for training it requires a labeled sample only from genuine tracks [15]. Thus, we got a neural network that performs tracking like a Kalman filter, albeit without final fitting to determine the physical parameters of the tracks.

The comparative performance of TrackNETv2 local tracking program on CPU and GPU of the GOVORUN supercomputer normalized to CPU productivity is shown in Fig 1. for different computing units and batch sizes.

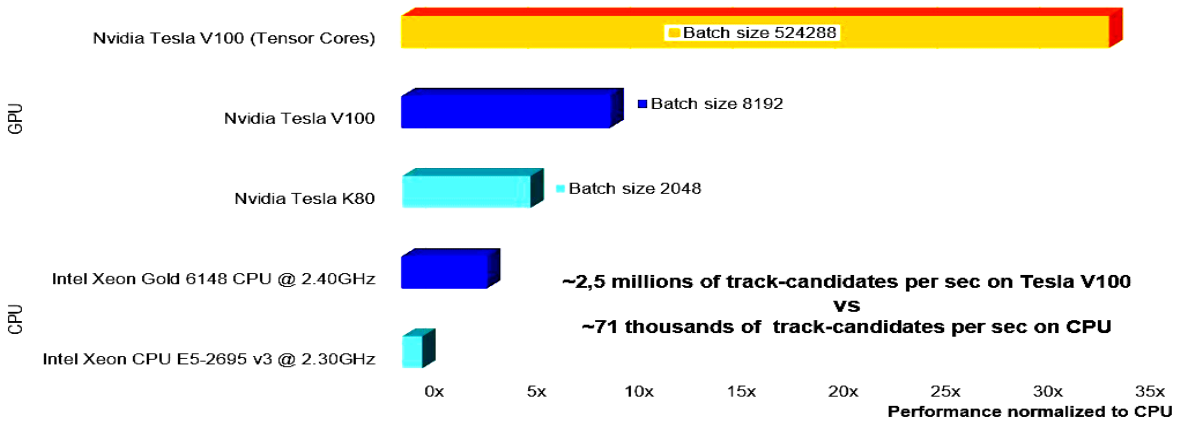

Fig.1. TrackNETv2 inference acceleration comparing CPU and GPU performances

It should be noted here that the performance of TrackNETv2 is measured here in track-candidates per sec because tracking method proposed above is local, i.e. tracks are reconstructed one after the other. At the same time, there is another global approach, in which the recognition of tracks among noises is performed immediately across the entire picture of the event, as it happened in the above-mentioned method of B. Denby using the Hopfield neural network. Local approaches to tracking have an obvious drawback: local methods do not allow us to assess the global picture of an event, to see the dependence between individual tracks or groups of tracks. Also, there is no direct possibility to track such phenomena as secondary vertices, as it is done in global tracking methods.

4. Global tracking for the BM@N experiment

Global recognition of tracks among the noises is carried out at once over the entire picture of the event. The GraphNet program is based on the use of graph neural networks for tracking [16]. An event is represented as a graph with hits as nodes, and then this graph is inverted into a linear digraph, when the edges are represented by nodes and the nodes of the original graph are represented by edges. In this case, information about the curvature of track segments is embedded in the edges of the graph, which simplifies the recognition of tracks in the sea of fakes and noises.

During training, the network receives as input a reverse digraph with labels of true edges - segments of real tracks. The already trained neural network RDGraphNet (Reversed Directed Graph Neural Network,), as a result, connects each edge with the value $x \in [0,1]$ at the output. True track edges are those edges for which x is greater than some given threshold (> 0.5).

The comparative performance of RDGraphNet global tracking program on CPU and GPU of the GOVORUN supercomputer normalized to CPU productivity is shown in Fig. 2. for different computing units and batch sizes.

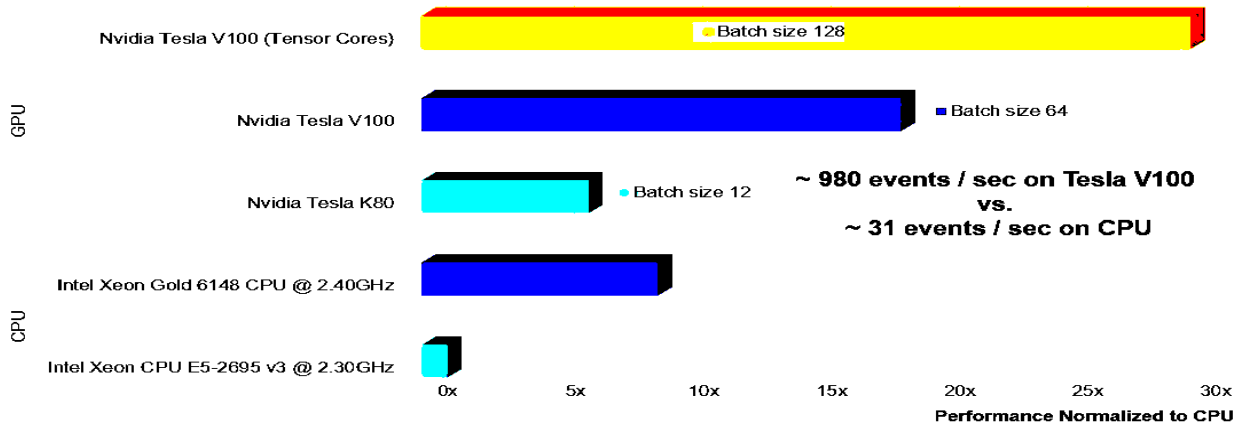

Fig.2. RDGraphNet inference acceleration comparing CPU and GPU performances

Note that, in contrast to the local neural network TrackNETv2, the productivity of the global neural network RDGraphNet is measured in the number of recognized events per second, which is very important for a large multiplicity of events.

5. Conclusion

Taking into account advantages of deep neural networks we have developed two algorithms of track recognition, based on deep learning architectures, for local and global tracking in the GEM tracker of the BM@N experiment at JINR (Dubna). Our neural network tracking algorithms are especially focused on overcoming the well-known difficulty inherent in GEM detectors, besides a particular attention is also paid to the problems of accelerating the learning procedures of deep neural networks. Test runs of deep tracking programs on the GOVORUN supercomputer and the HybriLIT cluster make it possible to evaluate the computing power gain of a special GPU with tensor cores compared to a conventional CPU and to optimize the sharing of resources between the training and validation parts of the tracking program. The training part of any deep neural network takes significantly longer than its testing.

The real inference productivity of TrackNETv2 after its training is over 2.5 millions of track-candidates per sec on GOVORUN GPU Tesla V100 what is 34 times faster than 71 thousands of track-candidates per sec on GOVORUN CPU Intel Xeon E5-2695.

RDGraphNet performance is ~ 980 events per second on the Tesla V100 versus ~ 31 events per second on the processor, which means a 30x speedup.

An additional performance reserve is the launch of the tensor core system in half precision FP16 mode, which gives significantly increase performance (1.5 - 3.5 times). It should be noted that the most important reserve in terms of processing speed by a trained neural network is the transition from an interpreted Python programming language to the C ++ language widely accepted in physics. This can speed up tracking programs even more.

Acknowledgement

The reported study was funded by RFBR according to the research project № 18-02-40101.

References

[1] BM@N Conceptual Design Report (BM@N collaboration) [Electronic resource]. – Mode of access: http://nica.jinr.ru/files/BM@N/BMN_CDR.pdf

[2] Gouache J.-C. Status report of the CERNLSD system // European Spiral Reader Symposium, P.21-32 https://cds.cern.ch/record/870023/files/p21.pdf

[3] Hansroul M., Savard D., Jeremie H. Fast circle fit with the conformal mapping method // Nucl. Instrum. Meth. A, 1988, v. 270, pp 498-501.

[4] Никитин В.А., Ососков Г.А., Автоматизация измерений и обработки данных физического эксперимента (монография), Изд. МГУ, Москва, 1986, 185 с.

[5] Denby B., Neural networks and cellular automata in experimental high energy physics // Comput. Phys. Commun. 49, 1988, pp 429-448.

[6] Gyulassy M. and Harlander M. Elastic tracking and neural network algorithms for complex pattern recognition // Comput. Phys. Commun. 66, 1991, 31-46.

[7] Fruhwirth R. Application of Kalman filtering to track and vertex fitting // Nuclear Instruments and Methods in Physics Research Section A, 1987, Vol. 262, No. 2-3, 444-450.

[8] Mankel R. A., Concurrent track evolution algorithm for pattern recognition in the HERA-B main tracking system // Nuclear Instruments and Methods in Physics, Research Section A, 1997, Vol. 395, No. 2, pp 169-184.

[9] Abt I., Kisel I., Masciocchi S. and Emelyanov D., CATS: a cellular automaton for tracking in silicon for the HERA-B vertex detector // Nuclear Instruments and Methods in Physics, Research Section A, 2002, Vol. 489, No. 1-3, pp 389-405.

[10] The High-Luminosity Large Hadron Collider (HL-LHC) project [Electronic resource]. – Mode of access: https://home.cern/science/accelerators/high-luminosity-lhc

[11] D. Rousseau, Tracking Machine Learning challenge [Electronic resource]. – Mode of access: https://indico.in2p3.fr/event/17295/contributions/61362/attachments/47569/59834/tr180329_david Rousseau_IN2P3ML_trackML.pptx.pdf

[12] Baranov D., Mitsyn S., Ososkov G., Goncharov G., Tsytrinov A., Novel approach to the particle track reconstruction based on deep learning methods // CEUR Proceedings, Vol. 2023, 37-45.

[13] H. M. Lynn, S. B. Pan, P. Kim, A Deep Bidirectional GRU Network Model for Biometric Electrocardiogram Classification based on Recurrent Neural Networks// IEEE Access PP(99), DOI: 10.1109/ACCESS.2019.2939947.

[14] N. Balashov, M. Bashashin, P. Goncharov et al, Service for parallel applications based on JINR cloud and HybriLIT resources// EPJ Web Conf., 214 (2019) 07012, DOI: 10.1051/epjconf/201921407012

[15] Supercomputer «GOVORUN» [Electronic resource]. – Mode of access: http://hlit.jinr.ru/about_govorun/

[16] Shchavelev E., P. Goncharov, G. Ososkov, D. Baranov Tracking for BM@N GEM Detector on the Basis of Graph Neural Network // Proceedings of the 27th Symposium on Nuclear Electronics and Computing (NEC 2019), Budva, Montenegro, 2019, [Electronic resource]. – Mode of access: http://ceur-ws.org/Vol-2507/280-284-paper-50.pdf