LOOT: NOVEL END-TO-END TRAINABLE CONVOLUTIONAL NEURAL NETWORK FOR PARTICLE TRACK RECONSTRUCTION

P. Goncharov[1], G. Ososkov[2], D. Baranov[2], S. Shengsen[3], Z. Yao[3]

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

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

[3] *Institute of High Energy Physics of the Chinese Academy of Sciences, China*

E-mail: kaliostrogoblin3@gmail.com

We introduce a radically new approach to the particle track reconstruction problem for tracking detectors of HEP experiments. We developed the end-to-end trainable YOLO-like convolutional neural network named Look Once On Tracks (LOOT) which can process the whole event representing it as an image, but instead of three RGB channels, we use, as channels in-depth, discretized contents of sequential detector coordinate stations. The LOOT neural net avoids all problems of the existing sequential tracking algorithms because it does computations in one shot. The first promising results of the algorithm's application to the data from the Monte-Carlo simulations are presented and discussed.

The reported study was funded by RFBR, project number 19-57-53002.

Keywords: tracking, GEM detector, YOLO, convolutional neural network, particle track reconstruction

Pavel Goncharov, Gennady Ososkov, Dmitriy Baranov, Sun Shengsen, Zhang Yao

1. Introduction

Track reconstruction, so-called tracking, plays a significant role in the modern high energy and nuclear physics (HENP) analysis. During HENP experiments, the ion beam collides with another beam or with a fixed target, generating a huge number of secondary particles registered then by track detectors in accordance with the energy release of each particle as it passes through the detector element. Each energy trace allows to calculate the spatial point, the so-called hit, where some particle probably pierced the sensitive area of the detector. The general idea of tracking consists in joining these hits into groups which includes all hits belonging to the same track one of many others, discarding noise and fake hits. The knowledge of track curvatures allows physicists to recover the initial momentum of particles and, eventually, to recover the whole event. Tracking is especially difficult for modern HENP experiments with heavy ions where detectors register events with very high track multiplicity.

As we are working on the BM@N experiment of the NICA megaproject [1], we face the famous shortcoming of GEM strip detectors when a great amount of fake hits appears along with real hits because of extra spurious crossings of strips activated by other tracks. The number of those fakes is greater for some order of magnitude than for true hits.

Classical tracking methods as Kalman filter [2] used successfully for years, including improvements to solve tracking problems in the GEM environment [3]. However, in the recent BM@N experiments with xenon beam producing thousands of tracks the Kalman filter approach could not meet the desired tracking speed, largely due to its sequential nature.

In opposite to classical tracking methods, deep learning (DL) approaches have the explicit advantage due to their capability to model complex non-linear data dependencies. Besides, they imply linear algebra operations performing which are highly parallelizable. So, DL methods should make a great contribution to the tracking problem.

2. Previous study

Although DL methods are very popular nowadays, as far as we know, there is not any end-to-end trainable solution for tracking in GEM detectors. Therefore, we had to overcome the shortcomings of our first attempt in using deep neural networks for tracking which includes two stages: track-candidates 3D search and true track selecting among all candidates with the help of the deep recurrent neural classifier [4] by inventing a new end-to-end trainable neural network for tracking. It can search for the track-candidate continuation and simultaneously define the probability of whether it is a true track. It works sequentially from the first station of the detector to the last excluding fake track-candidates and prolongs the true ones. We called this model TrackNETv1, one can find a detailed description of it in [5].

Though TrackNETv1 performed pretty well, especially in its track-following part, it has several drawbacks. In particular, it requires a labeled dataset with true and fake tracks, also, we cannot use a single TrackNETv1 model to solve the tracking problems up to the hilt, because of the cohabitation of the classification part and regression part. To address all these problems, we simplified the TrackNETv1 structure dropping out the classification part. The new model TrackNETv2 performs like a trainable version of Kalman filter, but instead of recalculation of the state of the track, it predicts the area where the next hit of a track can be located [6].

Nevertheless, all proposed algorithms for tracking are sequential. They cannot see the whole picture, being limited only to individual trajectories pass while trying to ignore the neighbors. Also, as the number of tracks grows, we run into a memory limit since those algorithms are sensitive to the number of tracks and their length.

In [7] we proposed the novel DL model named LOOT – «Look Once On Tracks» to solve the abovementioned tracking problems. The model was tested there only on simple track collection with straight tracks only. The remaining part of the paper describes the main concept of LOOT and how it was developed to process the Monte-Carlo simulation data for carbon-carbon (C+C) interactions in the BM@N experiment.

3. Look Once On Tracks concept

When we were looking for the solution of the tracking problem in one single end-to-end trainable pipeline, we tried to find any similar applications in related areas of DL. There is a common model in object recognition named YOLO [8] whose name is a fun acronym inherited from the famous American slogan "You Only Live Once".

YOLO is a deep convolutional network (DCN) for real-time object detection. It splits input image into parts using a regular grid; for each block predicts the probability of the object existence inside; the center of the bounding box area; the size of the bounding box; class label for the object inside the grid cell. The YOLO paper [8] inspired us and we asked ourselves: how we can represent the event as an image? Mostly, images have a 3d format: height, width, and RGB channels. Event data from each detector station can be considered as a sparse matrix of zeros and ones, where ones indicate the appearance of hits. Considering then each station, as a video frame, one can use DCN image format, where RGB channels are replaced by discretized contents of detector stations considering each of them as a channel in depth. It brings us to a radically new approach. In this way, we split all event data into stations, then we put all hits on a pixel grid with height and width equal to the height and width of the largest station depending on the selected resolution. For our BM@N configuration, we define the resolution of 512 and 256 pixels for width and height, respectively. Then we stack all resulting matrices on channels dimension, therefore, we obtained an object which is similar to image but instead of RGB we have stations.

In order to start with track reconstruction, we have to predict a LOOT mask for the first station as a simple binary matrix with the number of rows equal to height resolution and the number of columns equal to width resolution. It is a very sparse matrix in which rare ones locate in cells corresponding to the true hits on the first station. Nonzero cells in this mask represent the starting hits of true tracks.

Together with mask LOOT predicts coordinate shifts on OX and OY axes for every station except the first one. These shifts are also matrices with the same size as the mask. The value in each cell represents an axis shift from the previous hit of the track to obtain the coordinate position of the current hit. Thus, to predict the second hit of the track we have to know the position of the starting point – row and column indices (*ij*) of corresponding nonzero value in the predicted mask matrix. Then *ij* value of the first OX shift matrix indicates *j* shift, so to obtain the OX coordinate of the second hit of the track – $j + j\_shift$. To obtain the OY coordinate of the second hit of the track, one needs to sum i value (row number) with *ij* value from the first OY shift matrix. To obtain the third hit of the track, values from *ij* cell of the second OX and second OY shift matrices are required and then they are summed with the coordinates of the previous (second) hit of the track. The process repeats until the last station. In total: $1\ mask\ +\ (n\_stations - 1)\ *\ 2\ shifts$, e.g. for 6 stations – 11 matrices, so 11 channels on the output.

It should be noted that such an approach has some obvious limitations – it can only find tracks that start from the first station; short tracks and hits omissions not taken into consideration.

4. Model architecture

The first version of LOOT was applied to the toy dataset of 160K events with no magnetic field (straight tracks) registered in 5 stations, the number of tracks varied from 10 to 100 while fake hit numbers were taken twice greater than true hits and sampled uniformly. Mean tracks accuracy (fraction of tracks reconstructed accurately) was 96.5%. The model is a simple deep convolutional network with several blocks of 1x1 convolutions. A detailed description of it one can find in [7]. The main part of it is a special coordinate convolution layer [9].

There is a well-known coordinate transform problem, namely, the prediction of the coordinate grid that is extremely difficult for the common convolutional neural networks because they cannot learn coordinates from data samples without any additional knowledge about spatial positions of pixels. The idea is to add coordinate channels to the input of a deep neural network [9]. Without this layer, LOOT won't predict axes shifts accurately.

To train the LOOT model we proposed a special loss function with two terms – binary cross-entropy loss for the mask prediction and normalized squared error loss for the axes shifts. But while

adapting LOOT to the Monte-Carlo simulation data for the BM@N experiment we faced with the problem, when all output confidences (the mask for the starting hits of tracks) become equal to zero. It happened because the number of empty cells is much greater (n^2) than for the true tracks. This was overcome by replacing cross-entropy loss to the Dice loss function [10] which is a very common loss in the segmentation problems.

Also, we replace the original model architecture to the U-Net [11] like network. The network consists of a contracting path and an expansive path, which gives it the u-shaped architecture. During the contraction, the spatial information is reduced while feature information is increased. The expansive pathway combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features from the contracting path which is very useful to weed out fakes. We used U-Net with some small modifications, as the LOOT base network, preserving the output layer from the original LOOT.

5. Results and discussion

To train the LOOT model we generate 166K train and 62K test events of 4 GeV C+C interactions specific the BM@N experiment using LAQGSM generator. All tracks in these events have the number of hits equals to the number of stations (6 hits). We train the model for 50 epochs using SGD with momentum. We set the learning rate to 0.1 and momentum to 0.9. We set the batch size to 32. Also, we applied a cyclic learning rate update [12]. We measure several metrics: recall – expresses the ability to find all true tracks in a dataset, precision – expresses the proportion of data, our model says was true, actually were true tracks, shifts mean squared error and processing speed. We measure processing speed on a single Nvidia Tesla V100 GPU without float16 optimization and tensor cores utilization, so the processing speed may be optimized further.

Results from the tab. 1 look quite promising, although we found that LOOT suffers from the high error on the OY axis, which can lead to loss of tracks, as it shown in fig.1. It occurred because of using mean-squared error as shift loss. X shifts orders of magnitude greater than Y shifts, so the model learns to reduce OX error while preserving the OY error constant. To avoid this problem a special weighting of the shift loss is required.

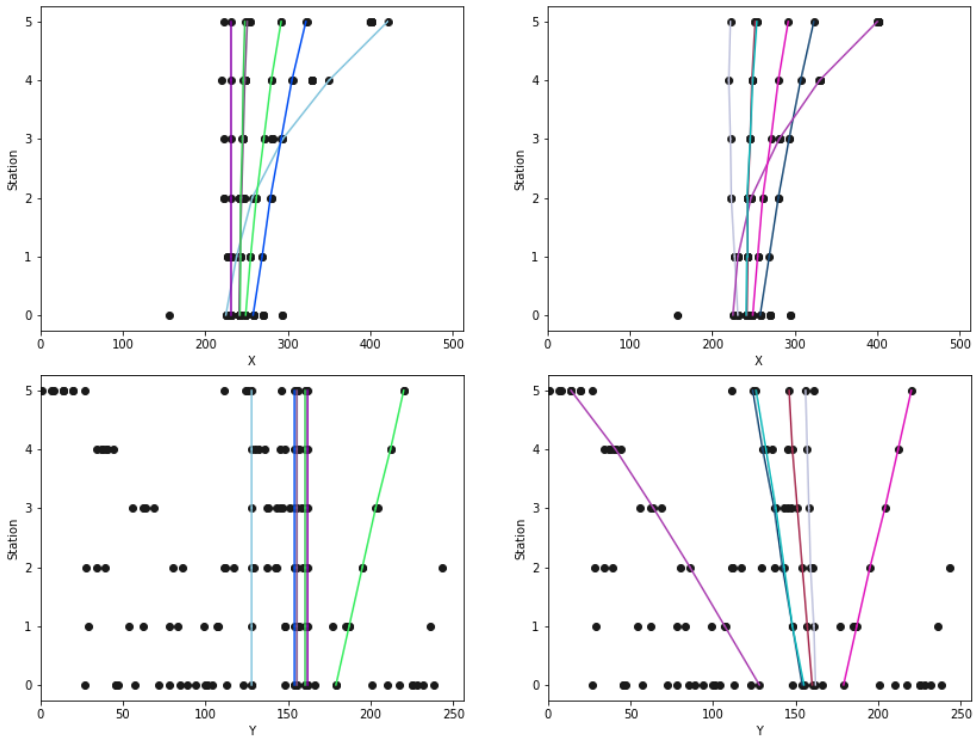

Figure 1. Illustrates LOOT problems in the reconstruction of OY shifts. Top row – OX projection, bottom – OY projection. Left column – reconstruction, right column – true event

Table 1. Result of the LOOT evaluation

Precision	0.9719
Recall	0.9690
Shifts MSE	0.0152
Processing speed (event/sec), batch size – 16	150

We have introduced the radically new approach to the problem of tracking and have presented the LOOT, which is fully end-to-end trainable; consumes the whole event at a time; doesn't depend on the number of fakes and tracks; memory cheaper than the other approaches; greatly drops out fake hits.

Now we are going to reduce the OY error by utilizing weighted shifts loss; improve the model to work on tracks with different lengths; try to predict track momentum instead of shifts; vertex prediction; expand the model's powers to solve tracking in a collider environment.

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] Baranov D., Merts S., Ososkov G., Rogachevsky O., New Algorithm of Seed Finding for Track Reconstruction, EPJ Web of Conferences, V. 108, 02012 (2016)

[4] D. Baranov, S. Mitsyn, G. Ososkov, P. Goncharov, A. Tsytrinov, Novel approach to the particle track reconstruction based on deep learning methods // Selected Papers of the 26th International Symposium on Nuclear Electronics and Computing (NEC 2017), Budva, Montenegro, September 25–29, 2017. – CEUR Proceedings. – Vol. 2023. pp 37–45.

[5] D. Baranov, Catch and Prolong: recurrent neural network for seeking track-candidates / D. Baranov, G. Ososkov, P. Goncharov, A. Tsytrinov // The XXII International Scientific Conference of Young Scientists and Specialists (AYSS-2018). – EPJ Web of Conferences. – EDP Sciences, 2019. – Vol. 201. – P. 05001

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

[7] Baranov D. et al. The Particle Track Reconstruction based on deep Neural networks //EPJ Web of Conferences. – EDP Sciences, 2019. – Т. 214. – С. 06018.

[8] Redmon J. et al. You only look once: Unified, real-time object detection //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – С. 779-788.

[9] Liu R. et al. An intriguing failing of convolutional neural networks and the coordconv solution //arXiv preprint arXiv:1807.03247. – 2018.

[10] Sudre C. H. et al. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations //Deep learning in medical image analysis and multimodal learning for clinical decision support. – Springer, Cham, 2017. – С. 240-248.

[11] Ronneberger O., Fischer P., Brox T. U-net: Convolutional networks for biomedical image segmentation //International Conference on Medical image computing and computer-assisted intervention. – Springer, Cham, 2015. – С. 234-241.

[12] Smith L. N. Cyclical learning rates for training neural networks //2017 IEEE Winter Conference on Applications of Computer Vision (WACV). – IEEE, 2017. – С. 464-472.