Application of graph neural networks in particle track reconstruction

Wojciech Gomułka[1[0009−0003−2873−425X]], **Piotr A. Kowalski**[2,3[0000−0002−4850−1758]], **Tomasz Szumlak**[2[0000−0002−2562−7163]], **Tomasz Bołd**[2[0000−0002−2432−411X]]

[1]*AGH University of Krakow*
Center of Excellence for Artifical Intelligence
al. Adama Mickiewicza 30, 30-059 Kraków
wgomulka@agh.edu.pl
[2]*Faculty of Physics and Applied Computer Science,*
AGH University of Krakow,
al. A. Mickiewicza 30, 30-059 Krakow, Poland
{pkowal,szumlak,tbold}@agh.edu.pl
[3]*Systems Research Institute, Polish Academy of Sciences*
ul. Newelska 6, 01-447 Warsaw, Poland

Abstract. *This study describes the usage of graph neural networks (GNN) as one of potentially promising directions to address the problem of reconstructing the tracks of charged particles in high-energy physics. We point to previous attempts of such applications and use those approaches as entry point of our research. We present a proof-of-concept example of the usage of GNN in the problem, through the application of the basic architecture for the edge labeling task as a template. We also train and compare similar architectures using either SAGEConv, GCNConv or GatedGraphConv layers.*

Keywords: *high energy physics, artificial intelligence, particle track reconstruction, graph neural networks*

1. Introduction

High-energy physics (HEP) remains one of the most prominent examples of interdisciplinary research among physicists, computer and electronics scientists. Every new generation of experiments pose a new challenges that still demand for novelty in software that helps in research. A particular use case is fast (near run-time) labeling of the data. It is already challenging and the data volumes are expected to grow even more in the future. Specifically, an incoming High-Lumi upgrade of Large Hadron Collider (LHC) which shall finish within a few years is hoped to increase the number of collisions by a factor of 5 - 7.5 [1].

In this paper, we focus on the topic of track reconstruction of charged particles, for which many different techniques have been used till now. They are taking into account physical reality of the process, like the helical movement of charged particles in a uniform magnetic field, with well-known algorithms, such as decision trees [1], Kalman filter [2], and the Hough transform [3]. Some of those approaches have a well-documented history of applications in tracking subjects in different domains and were also used in the field of particle physics. Hardware might be an essential limitation for application of these techniques, but recent interest in artificial intelligence combined with a growth of suitable computing powers might encourage more research in this direction. One of possibilities is the application of graph neural networks (GNN). For example, *X.Ju et al., 2020* [4] prepared GNN, which used differences between the coordinates of pairs of measurements left by particles in the detector to form GNN edge features and the measurement themselves as nodes. Track reconstruction was then treated as task of binary edge labeling on the pre-filtered set of hits. The concept of using GNNs for HEP tracking was also comprehensively described in *Duarte et al., 2022* [5]. We would like to leverage this idea to construct own architecture of GNN to address the problem of track reconstruction and present the initial research on the matter.

2. Idea of particle track reconstruction

High energy experiments, such as those at the LHC consist of many detector elements used to register signals of particle passage. After proper mapping, they give information about the coordinates of the signal; however, to restore tracks of the particles, there is a need to associate them with specific hits and then obtain properties of the particle. The task is trivial for single particle in the detector, all signals belong to that particle[2]. In the experimental reality, however, there is a significant number of particles that leave the signals in the detector at the same time.

[1]According to: `https://home.cern/resources/faqs/high-luminosity-lhc`.

[2]Assuming absence of the detector noise.

Several approaches are used to perform track reconstruction, and the tracking algorithm might consist of several steps using different computational techniques. For example, ACTS (*A Common Tracking Software*) toolset, divides the track reconstruction process into steps such as seeding, track finding, track fitting and ambiguity resolution [3]. Seeding means finding a few initial hits believed to belong to the specific particle. This step has a major impact on the further reconstruction process. The toolset also provides virtual detectors such as ODD (*Open Data Detector*) and simulations of collisions inside those detectors. Example hits generated in the simulation of ODD are shown in Figure 1 and 2. The scatter plot of hits allows to implicitly visualize the detector geometry.

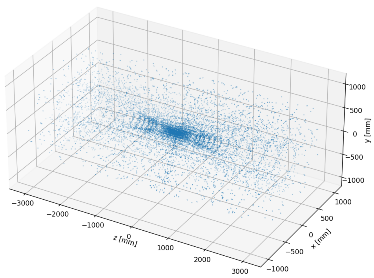

Figure 1. Example hits from simulation of the *OpenDataDetector* in the ACTS toolkit

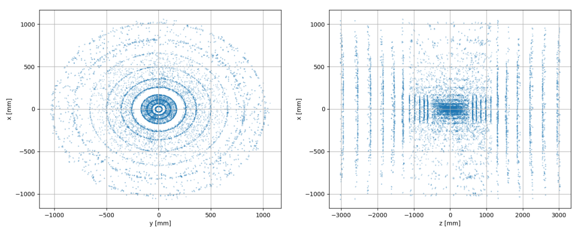

Figure 2. Example hits from simulation of the *OpenDataDetector* in the ACTS toolkit, projected on Y-X (left) and Z-X (right) planes

[3]Tracking explanation based on: `https://acts.readthedocs.io/en/latest/tracking.html`.

3. Graph neural networks

Graph neural networks [6] can address a wide range of problems, which can be translated to graphs. The rise in their popularity is relatively recent, yet already several different graph architectures have been designed and systematized in different surveys and review articles [6, 7]. The GNNs are significantly inspired by other architectures, among which graph convolutional networks (GCN), recurrent graph neural networks, or graph generative-adversarial networks (GGAN). Therefore, GNNs do not limit themselves to graph classification, but they can also be helpful at the node and edge level problems. The specific yet popular task is link prediction, which can be used in recommendation systems or social networking sites. Other popular applications of GNNs include, e.g. natural language processing, where GNNs can be helpful in modeling relations among text elements, by providing hierarchy [8], computer vision, where e.g. graph representation can support reasoning based on images or video frames [9], or chemistry, where graphs are well suited to represent the structure of the molecules.

In this paper, we opt for Graph Neural Networks, as every reconstructed particle track could be treated as a graph of the measurments forming it, as shown by [4]. The architecture of network proposed here and the training process used implementation of GNN from [10] as a template, upon which three similar architectures were build and compared.

4. Experiments

The three compared architectures differed by leveraged graph layers (SAGEConv, GCNConv or GatedGraphConv). GCNConv is an example of a spectral graph convolutional neural network [11]. SAGEConv (*sample and aggregate*) was introduced as a result of research related to the GCNConv. It is a graph neural network that addresses the problem of generating low-dimensional embeddings for node features in GNN [12]. Although convolutional GNNs seemed appropriate due to their ability to gather information from the neighborhood, recurrent neural networks (RNNs) were believed to be even more suitable, as consecutive hits of the specific track remain in the sequantial relation. Therefore, we use GatedGraphConv, which is a graph RNN inspired by the Gated Recurrent Unit (GRU) [13]. The general architecture of the models is presented in Figure 3.

The training was performed on the graph obtained from the ODD simulation of a single collision event with about 3000 particles/labels. Therefore, the input graph consisted of approximately 200,000 nodes and 15,000 edges, from which half of the edges were labeled with positive class - edges between two consecutive hits belonging to the same track. The second half served as negative samples - edges generated between random hits. Although such an example is simplified when compared to real task of searching for the connections in the whole graph, yet it can still serve as a proof-of-concept for usage of GNNs in the tracking problem, and can be helpful for initial evaluation of the models. The training was carried out for 400 epochs in each case with a learning rate of 10^{-3} and ADAM chosen as optimizer. No early stopping was added. As a loss function, binary cross entropy with logits was selected. All the experiments were conducted with use of PyTorch Geometric framework. The final models were then evaluated on the few graphs built with different events generated from ODD. In case of SAGEConv and GCNConv, there were no significant differences from the testing/training metrics reached for the data from the original event. The results of data from one of such events is presented in Figure 5 in the form of confusion matrices. These data were not used for training of the models.

From Table 1. and Figure 4 we can argue that the model based on the SAGEConv or GCNConv layers outperformed the recurrent GatedGraphConv network in most of the metrics calculated. Model leveraging GatedGraphConv was converging faster, but last over 200 epochs had no significant impact on the loss.

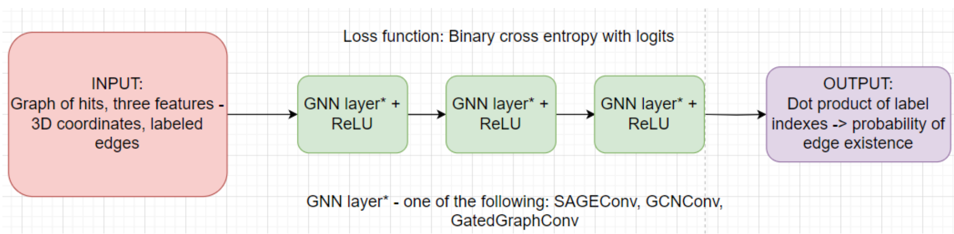

Figure 3. General architecture tested in the problem. For SAGEConv and GCNConv 256 channels were passed to the layers. For GatedGraphConv only one layer with 64 channels was used in the network to speed up the experiment

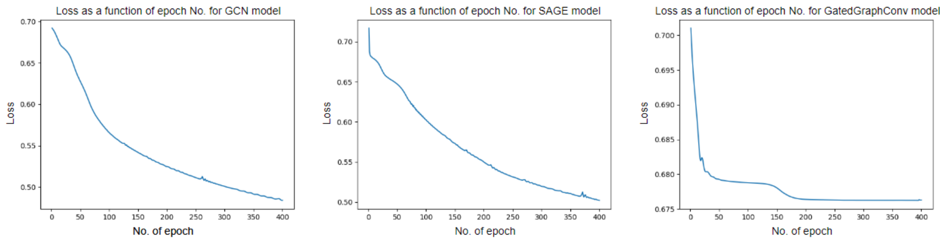

Figure 4. Training loss of the different models

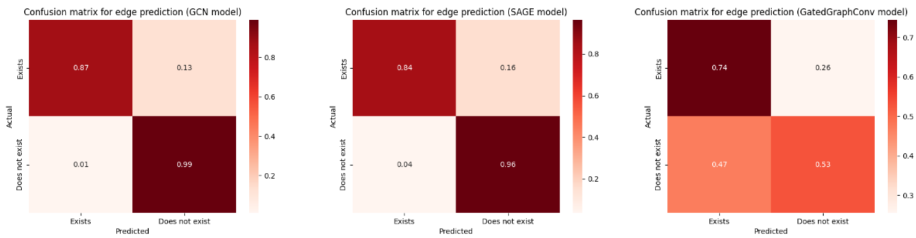

Figure 5. Confusion matrices for evaluation on example event

Table 1. Metrics for training of models using different layers

Metric	**SAGEConv**	**GCNConv**	**GatedGraphConv**
Sensitivity	0.976	0.993	0.402
Specificity	0.830	0.849	0.860
Precision	0.852	0.868	0.742
Negative predictive value	0.972	0.991	0.590
Accuracy	0.903	0.921	0.631

5. Conclusions

Usage of GatedGraphConv layer led to the least successful results among the compared approaches, despite the intuitive sequential nature of the hits in the track. Both GCNConv and SAGEConv allowed us to reach over 80% for all training metrics, with a slight edge of the first of the mentioned networks. It should be added, however, that the most important metric in measurements association is the efficiency (the ratio of the number good candidates/number of genuine tracks).

The results achieved in our experiments look promising in terms of GNN usage in the problem of particle tracking, despite the very initial phase of the research and simplified conditions of the task. Future studies will try to address the issue of track reconstruction in more realistic conditions, with taking advantage of filters inspired by physical and geometrical properties, which shall limit significantly the amount of hits considered for tracking. The training will be performed for bigger data volumes and different detectors with consideration of situations when there is no uniform magnetic field.

Acknowledgement

We gratefully acknowledge the funding support by program "Excellence initiative—research university" for the AGH University of Krakow as well as the ARTIQ project: UMO-2021/01/2/ST6/00004 and ARTIQ/0004/2021

References

[1] Davis, A., Dendek, A. M., De Cian, M., and Szumlak, T. PatLongLivedTracking: a tracking algorithm for the reconstruction of the daughters of long-lived particles in LHCb. Technical report, 2017.

[2] Astier, P., Cardini, A., Cousins, R. D., Letessier-Selvon, A., Popov, B. A., and Vinogradova, T. Kalman filter track fits and track breakpoint analysis. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 450(1):138–154, 2000.

[3] Pozzobon, N., Montecassiano, F., and Zotto, P. A novel approach to hough transform for implementation in fast triggers. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 834:81–97, 2016.

[4] Ju, X., Farrell, S., Calafiura, P., Murnane, D., Prabhat, Gray, L., Klijnsma, T., Pedro, K., Cerati, G., Kowalkowski, J., Perdue, G., Spentzouris, P., Tran, N., Vlimant, J.-R., Zlokapa, A., Pata, J., Spiropulu, M., An, S., Aurisano, A., Hewes, V., Tsaris, A., Terao, K., and Usher, T. Graph neural networks for particle reconstruction in high energy physics detectors, 2020. doi:arXiv: 2003.11603.

[5] Duarte, J. and Vlimant, J.-R. Graph neural networks for particle tracking and reconstruction. In *Artificial intelligence for high energy physics*, pages 387–436. World Scientific, 2022.

[6] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020. doi:10.1109/TNNLS.2020.2978386.

[7] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

[8] Wu, L., Chen, Y., Shen, K., Guo, X., Gao, H., Li, S., Pei, J., and Long, B. Graph neural networks for natural language processing: A survey, 2022.

[9] Chen, C., Wu, Y., Dai, Q., Zhou, H.-Y., Xu, M., Yang, S., Han, X., and Yu, Y. A survey on graph neural networks and graph transformers in computer vision: A task-oriented perspective, 2022.

[10] Link prediction on heterogeneous graphs with PyG. URL `https://medium.com/@pytorch_geometric/link-prediction-on-heterogeneous-graphs-with-pyg-6d5c29677c70`.

[11] Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks, 2017.

[12] Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs, 2018.

[13] Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. Gated graph sequence neural networks, 2017.