PROCEEDINGS OF SCIENCE

Graph Neural Networks and Application for Cosmic-Ray Analysis

Paras Koundal[a,*]

[a] *Institute for Astroparticle Physics, Karlsruhe Institute of Technology, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Karlsruhe, Germany*

E-mail: paras.koundal@kit.edu

Deep Learning has emerged as one of the most promising areas of computational research for pattern learning, inference drawing, and decision-making, with wide-ranging applications across various scientific disciplines. This has also made it possible for faster and more precise analysis in astroparticle physics, enabling new insights from massive volumes of input data. Graph Neural Networks have materialized as a salient implementation method among the numerous deep-learning architectures over the last few years because of the unique ability to represent complex input data from a wide range of problems in its most natural form. Described using nodes and edges, graphs allow us to efficiently represent relational data and learn hidden representations of input data to obtain better model accuracy. At IceCube Neutrino Observatory, a complex multi-component detector, traditional likelihood-based analysis on a per-event basis, to reconstruct cosmic-ray air shower parameters is time-consuming and computationally costly. Using advanced and flexible models based on Graph Neural Networks naturally emerges as a possible solution, reducing the time and computing cost for performing such analysis while boosting sensitivity. This paper will outline Graph Neural Networks and discuss a possible application of using such methods at the IceCube Neutrino Observatory.

The 5th International Workshop on Deep Learning in Computational Physics
28-29 June, 2021
Moscow, Russia

*Speaker

PoS(DLCP2021)004

1. Introduction

Deep Learning with graphs is a research area with foundational ideas motivated from multiple research areas and different neural-network architecture categories. Consequently, it tries to bridge the gap between them. This affords us the flexibility to work with data structures and types, which were challenging to work with other Deep Learning (DL) architectures before.

Over the last few years, multiple interesting literature surveys and reviews [1–9] have been done which shown the length and breadth of this emerging area. This is accompanied by their rich and growing research application in natural sciences, recommender systems, or day-to-day use systems (traffic networks, social networks, computer networking).

This text is one of such reviews, albeit shorter; however, with a focus on the application of DL with graphs to understand Cosmic-Ray composition at IceCube Observatory. Section 2 introduces the formal definition of a graph and the ways to mathematically represent it. Section 3 delves into deep-learning with graphs and discusses the operative mechanism of Graph Neural Networks (GNNs). Section 4 discusses the possible application of GNNs for cosmic-ray analysis at IceCube Observatory. Finally, these are accompanied by a conclusion and acknowledgments.

2. Graphs - A Primer

In its simplest form, a graph is defined by nodes and the edges connecting these nodes. Depending on the graph, associated with nodes and/or edges we can also have attributes. The flight-route network [10] is a simple real-world example of a graph. The airports can be thought of as nodes, and a connection between two airports is equivalent to an edge between the two corresponding nodes in the graph (representing a relationship between the two airports/nodes). We can have associated attributes like airport capacity and flight time associated with the nodes/airports and edges/connections. Few general observations about graphs:

- If there is no associated spatial information associated with the nodes in a graph, the placement of a node w.r.t other nodes is irrelevant. The set describing the connectivity is sufficient to describe the graph structure. This can be seen in Fig. 1, where two graphs having the same underlying graph connectivity can be visually represented in two different ways.

- The structure of a graph is invariant to node-labeling. In general, there is no preferred node and we can number (or label) nodes starting from any one of them.

Most of the Graphs can be put into one of the following three categories:

1. **Undirected**: Have no specified direction associated with the edges.

2. **Directed**: The edge between two nodes has a single preferential direction.

3. **Bidirectional**: The edge between two nodes has a connection in both directions.

The direction of the connection is vital to describe the information exchange between two neighboring (i.e., connected) nodes.

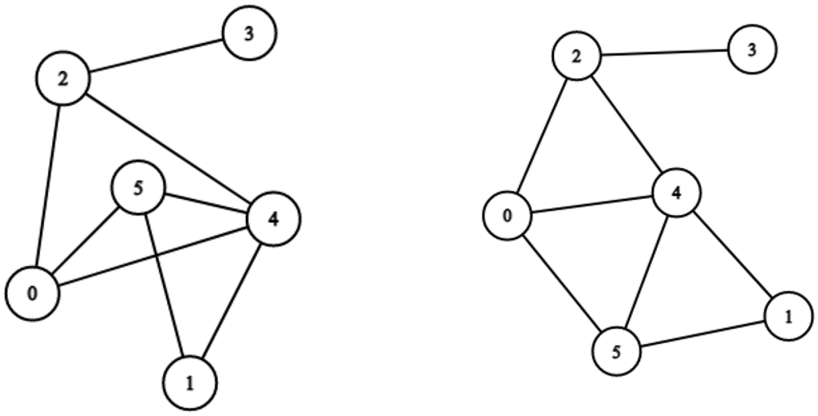

Figure 1: Two visually different graphs with the same underlying graph connectivity, and hence the same graphs.

2.1 Representing Graphs

Graph connectivity is one of the most important properties to define a graph. The connectivity can be expressed mathematically as matrices in multiple ways, namely:

- **Adjacency Matrix (A):**
  It is a square matrix representing whether an edge exists between two nodes. By convention, if an edge exists between node i and j, we fill the A_{ij} element of the matrix with 1, and 0 otherwise. Hence, for a simple undirected graph, the adjacency matrix will be symmetric. In general, the elements of the adjacency matrix can be different from 0 and 1. The adjacency matrix for the graphs in Fig. 1 is

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

- **Degree Matrix (D):**
  It is a square matrix representing the number of connected nodes to each node in a graph. Element D_{ii} of the matrix corresponds to the degree of node i. Hence, for an undirected graph the degree-matrix is a diagonal matrix. However, for a directed and/or weighted graph the non-diagonal elements can be non-zero. The degree matrix for the graphs in Fig. 1 is

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}.$$

Adjacency and Degree matrix can be used to understand importance of nodes in a graph, since a node with higher number of connections to other nodes can be considered as a central and important node. Other properties like **node-centrality**, **clustering coefficient** and so on (for details, read [11]) can also be used to describe graph connectivity and understand node importance.

3. Deep Learning with Graphs

MLP (Multilayer Perceptron), one of the simplest DL-architectures, involves mapping a set of input features onto the target variable(s) and optimizing the weights of connections between the layers to minimize a pre-defined loss-function in order to make good predictions on unseen data. As the dimension and size of input features grow, working with MLPs is mostly detrimental to model accuracy and reduces training time since the number of total trainable parameters can grow fast. Furthermore, we lose on benefiting from structure or patterns (spatial, temporal, etc.) inherent in the input training data. Convolutional Neural Networks (CNNs) alleviate this problem by introducing local connectivity. Moreover, the shared weights for the input data also help us train the network faster, than if we were using MLPs for the same dataset. Over the years, CNNs have proved to be very successful for pattern recognition tasks. However, their success is also generally limited to the domain where the input data structure is orthogonal. Graph Neural Networks (GNNs) extend this success in prediction and pattern recognition to data that can not be easily represented in Euclidean space. A large number of natural and research systems (or data) [12–15] consist of rich relational data where information is shared among individual constituents, where the data mostly has a non-euclidean form. Hence, GNNs are best suited for such tasks.

To motivate the necessity for understanding the operative mechanism of GNNs, we think it is crucial to first dissertate the variety of problem domains where GNNs have already been implemented and shown improvement over other DL methods:

- **Node Classification**

  It is the task of predicting label(s) associated with unlabeled nodes in a graph when given information about another set of connected and/or disjoint nodes. For more details see [16, 17].

- **Link Prediction**

  Contrary to node classification, link-prediction is the task of predicting label(s) or strength associated to unknown or non-existent connections, given the information on constituent nodes. For more details see [18, 19].

- **Cluster Detection**

  It is the task of predicting clusters in a graph, given information about nodes and edges. For more details see [20, 21].

- **Classification and Regression**

  It is the task of predicting global parameter(s) associated to a graph (for graph classification or regression), instead of learning features associated to the nodes. For more details see [22, 23].

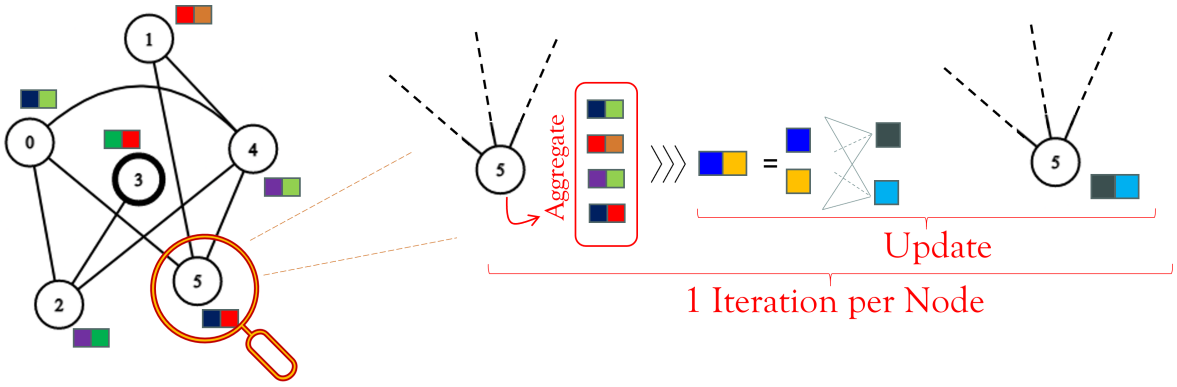

Figure 2: **Aggregate** and **Update** steps in a layer of a Graph Neural Network, using graph connectivity as well as local information. The step is repeated at each node in a layer.

3.1 Graph Neural Networks - A Whirlwind Tour

In order to bridge the gap between GNNs and other architectures, we need to define a schematic approach of learning graph structure and node representations. As also discussed in [11], flattening adjacency matrix (definition in Sub-Sec. 2.1) and using it as input for an MLP-like network architecture seems like a possible first solution. However, it suffers from a significant issue of not satisfying permutational invariance (mentioned in Sec. 2) of node-labeling in a graph. To elaborate, the same graph with different order of node labeling will be considered a different training/test example by the MLP. This treatment of graph data is undesirable and will also make our predictions on the data non-replicable. Moreover, the MLP-like treatment of graph data only uses graph connectivity and ignores the information associated with the neighboring graph nodes. CNNs excel at exploiting local-connectivity.

To benefit from and combine the simplicity of MLPs and expressivity of CNNs (using local-connectivity) into GNNs, the most straightforward framework is using a two-step process of **Aggregate** and **Update** (as shown in Fig. 2):

- **Aggregate**

  This step is vital to establish a message-passing framework in a graph. It involves accumulating embeddings (information) from the neighborhood of each node. The initial embeddings are hence the input features associated to the nodes. To keep the permutational invariance of node labelling in the graph, the aggregation function must be permutationally invariant. Few simple examples of such a function can be normalized mean, symmetric normalization [24], etc..

- **Update**

  This step takes the aggregation step's input, updates the embeddings using a MLP, and introduces non-linearity. The output of the update step is defined depending on the task at hand.

The aggregation step is similar to the local connectivity in CNNs, and the update step is similar to MLPs. Hence, GNNs benefit from the best of both worlds. After one layer (or iteration), each

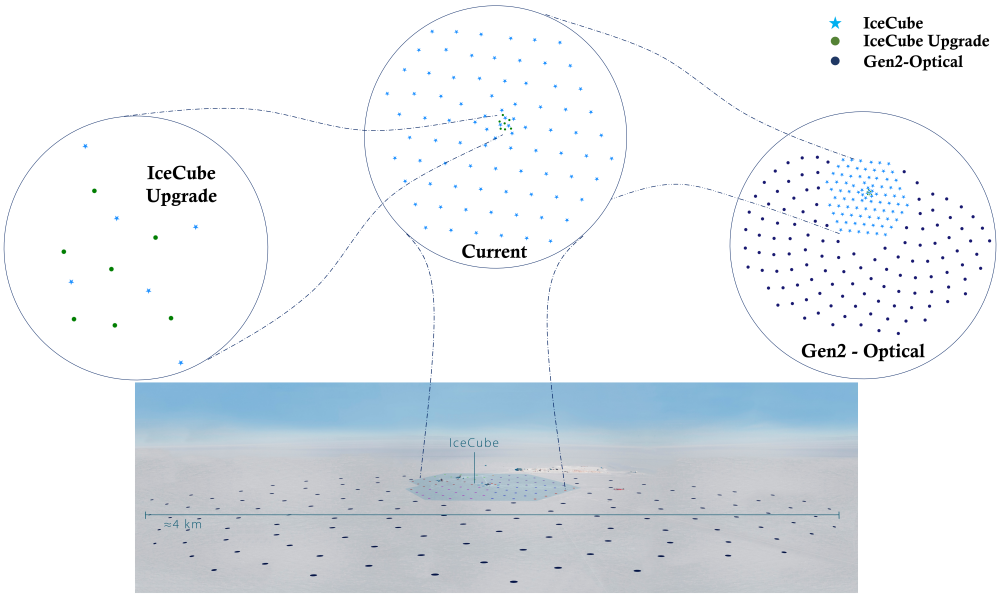

Figure 3: Top view of the planned IceCube-Gen2 (**right** : Dark-Blue Dots) Neutrino Observatory facility, with current (**center** : Light-Blue Stars) IceCube configuration and future seven IceCube Upgrade strings (**left** : Green Dots). **Adapted From:** Photo by National Science Foundation

node has its neighbors' aggregated (and updated) information which are 1-step away. Since the same process is repeated at each node in a layer, after k-iterations, each node has the information from a neighbour separated by k-steps. Hence, after k-steps, the embeddings at each node encode the information of graph-connectivity as well as feature information. This can be easily used to either do node classification or cluster detection. The concept can also be extended for updating embeddings associated with edges and can be used for link prediction. To make a graph-level prediction, we can add an additional pooling step over all nodes after k-iterations. An example of such an architecture will be presented in the next section.

4. Cosmic-Ray Analysis at IceCube Observatory

Cosmic-Rays (CRs) [25] are charged particles from astrophysical accelerators, with energies that can reach nearly six to seven orders higher than the energy achievable by the most powerful earthbound accelerators. Understanding the dynamics of the astrophysical sources that accelerate particles to these high energies is a pursuit that is very dear to many astrophysicists. CRs and CR-induced air-showers [26] furnish us with the possibility to discern the fundamental properties and behavior of such sources. IceCube Observatory [27], located at the South Pole, detects the particles from such astrophysical sources. The integrated operation of the in-ice array of IceCube (primarily a neutrino detector), with its surface array, IceTop (IT), affords us unique three-dimensional detection capabilities for CR-induced air showers.

IceCube Collaboration did an analysis [28] to understand the energy spectrum and composition of CR events detected at IceCube. An MLP was trained using Monte-Carlo (MC) simulations of the CR-induced air showers to estimate primary energy and corresponding mass from the measured

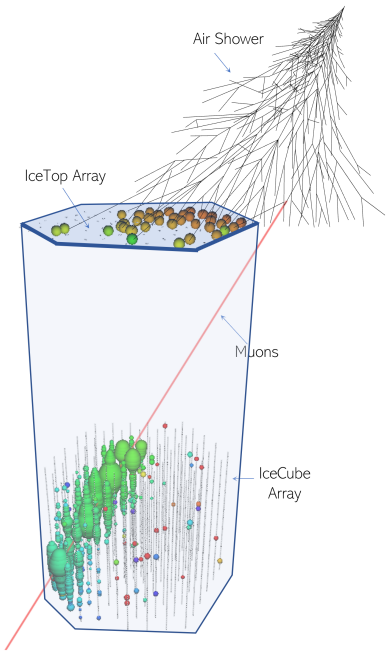

Figure 4: Example Air-Shower deposit in IT and IC. Size of the circles corresponds to signal strength and color corresponds to hit-timing (red = early; blue = late). **Adapted From:** IceCube Collaboration

shower footprint. Reconstructed air-shower observables were used as input features with Energy and Mass as the target variables. After sufficient network optimization on MC simulations, it was implemented on three years of real data. However, the analysis could not provide us good discrimination for mass estimation on a per-event basis. A future analysis, which also uses the full shower footprint and the reconstructed parameters, can improve the analysis to predict the target variables more precisely.

The simplest possible solution to use a full shower footprint is by using CNN, as shown in [29] for IceCube. The input pixels of the CNN is the individual photo-detector with electronics, i.e., Digital Optical Modules (DOMs) of the detector. With the enhancement of the current detector configuration to IceCube-Gen2, with larger-area [30] and multiple sub-components [31–33] as shown in Fig. 3, the detector will shift to a more irregular geometry. Data transformation of such future detector configuration to an orthogonal grid and specialized kernels (as shown in [29]) will possibly be complex and inefficient. As described in earlier sections, GNNs surely hold promise in their capability to allay the limits of MLPs and CNNs. IceCube already has few ongoing analysis using GNNs in cosmic-ray physics [34, 35] and neutrino physics [36, 37]. Results of one such preliminary test using GNNs for CR primary-mass prediction at IceCube will be presented here.

4.1 Primary Type Classification using GNNs

As shown in Fig. 4, particles in a CR-initiated air-shower deposit signal in the DOMs of the ice-cherenkov IT tanks. Of these, the high-energetic particles can propagate deep further in the ice to deposit signals in the IceCube (IC) array (starting at a depth of about 1500m). As discussed

earlier, we want to use the signal-footprint of such air-showers to predict the CR primary-type, and for this purpose, we will be using GNNs.

Looking at the signal footprint, using individual DOMs as nodes of a graph seems a natural option. Associated to each DOM (or node) in an air-shower, we have timing and charge-deposit information. Moreover, we also have in-ice spatial coordinates of the DOMs for every event. Hence, the charge and spatio-temporal information can be used as associated features to every DOM (hereon called as a node). The edges between nodes can be built in multiple ways (e.g., k-Nearest Neighbors, directed edges using temporal information, and so on). In this text, the connectivity between nodes is weighted by their spatial separation. This means that each node is connected to every other node; however, the features from neighboring nodes are weighted using a normalized Gaussian kernel using their spatial information for aggregation at each node. So, the nearest neighbor to any node will have the highest weight, whereas the farthest will have the least. The width of the Gaussian kernel is a learnable parameter for our case. Also, only the in-ice DOMs are mapped as a graph. To capture information from IT as well as from global features associated to each graph, another MLP based sub-component is used (details in [34]).

For the purpose of training, PyTorch [38] framework is used. The IC-based GNN and IT + Global Information incorporating MLP are concatenated into a fully connected network to train a primary-mass estimator. The data division between the training-validation-test set was 80%-10%-10%, and the network was trained on an equal number of proton and iron generated events. The preliminary results of such a test on a model trained for 100 epochs is shown in Fig. 5. Preliminary tests show a an improvement in mass resolution and preciseness in prediction of mass. More details of the work are presented in [34].

5. Conclusions

The article presents an overview of Graph Neural Networks (GNNs) and preliminary results for Cosmic-Ray Composition analysis at the IceCube Observatory. It was discussed how GNNs bridge the gap between MLPs (Multilayer Perceptrons) and Convolutional Neural Networks (CNNs), benefiting from the best principles and methods in both domains. Preliminary results for Cosmic-Ray Composition analysis at IceCube Observatory show promising results and more detailed results of this ongoing work are discussed in [34]. In the future, a full cosmic-ray energy spectrum and composition analysis at IceCube, in the transition region from galactic to extragalactic sources, using GNNs is envisioned.

Acknowledgments

This work is done with the support and collaboration of Karlsruhe Institute of Technology (KIT), Germany, and IceCube Collaboration. The author would also like to thank the organizers of the Deep Learning in Computational Physics (DLCP) - 2021 conference for their invitation for a talk.

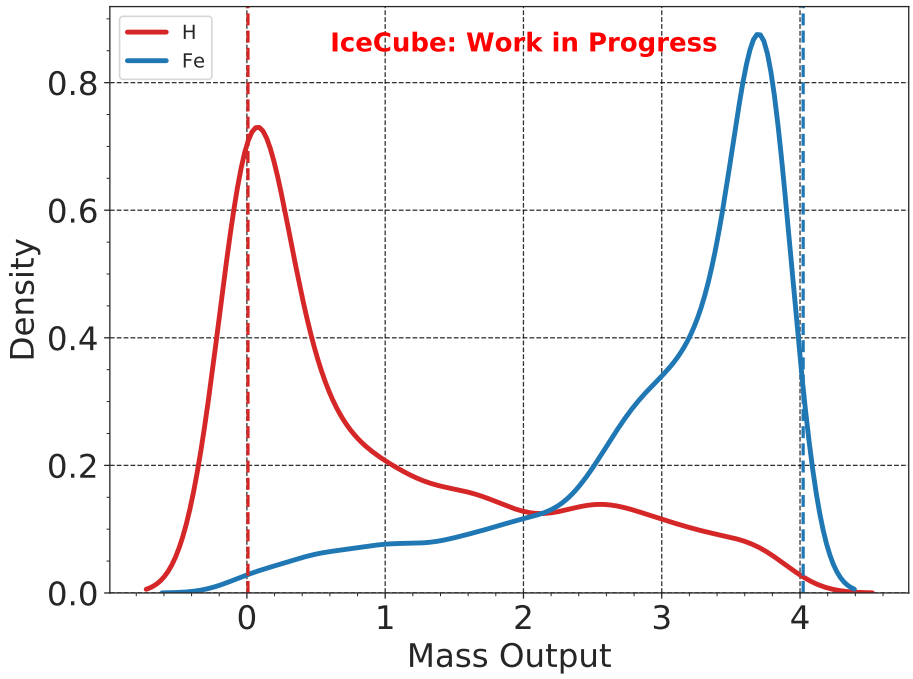

Figure 5: KDE of the natural logarithm of the cosmic-ray primary mass as predicted by the GNN-based method, trained on equal fraction of H and Fe simulations (vertical dashed lines: true primary mass). Plot taken from [34].

References

[1] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu et al., *Graph Neural Networks: A Review of Methods and Applications*, 2021.

[2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang and P.S. Yu, *A Comprehensive Survey on Graph Neural Networks*, *IEEE Transactions on Neural Networks and Learning Systems* **32** (2021) 4–24.

[3] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu et al., *Graph neural networks: A review of methods and applications*, *AI Open* **1** (2020) 57.

[4] Z. Zhang, P. Cui and W. Zhu, *Deep Learning on Graphs: A Survey*, 2020.

[5] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner and G. Monfardini, *Computational Capabilities of Graph Neural Networks*, *IEEE Transactions on Neural Networks* **20** (2009) 81.

[6] V.P. Dwivedi, C.K. Joshi, T. Laurent, Y. Bengio and X. Bresson, *Benchmarking Graph Neural Networks*, 2020.

[7] Z. Chen, F. Chen, L. Zhang, T. Ji, K. Fu, L. Zhao et al., *Bridging the Gap between Spatial and Spectral Domains: A Survey on Graph Neural Networks*, 2021.

[8] J. Skarding, B. Gabrys and K. Musial, *Foundations and Modeling of Dynamic Networks Using Dynamic Graph Neural Networks: A Survey*, *IEEE Access* **9** (2021) 79143–79168.

[9] M.M. Bronstein, J. Bruna, Y. LeCun, A. Szlam and P. Vandergheynst, *Geometric Deep Learning: Going beyond Euclidean data*, *IEEE Signal Processing Magazine* **34** (2017) 18–42.

[10] M. Grandjean, *Connected World: Untangling the Air Traffic Network*, 2016.

[11] W.L. Hamilton, *Graph Representation Learning*, *Synthesis Lectures on Artificial Intelligence and Machine Learning* (2019) .

[12] W.L. Hamilton, R. Ying and J. Leskovec, *Inductive Representation Learning on Large Graphs*, 2018.

[13] P.W. Battaglia, R. Pascanu, M. Lai, D.J. Rezende and K. Kavukcuoglu, *Interaction Networks for Learning about Objects, Relations and Physics*, *CoRR* **abs/1612.00222** (2016) [1612.00222].

[14] A. Fout, J. Byrd, B. Shariat and A. Ben-Hur, *Protein Interface Prediction Using Graph Convolutional Networks*, in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 6533–6542, Curran Associates Inc., 2017.

[15] H. Dai, E.B. Khalil, Y. Zhang, B. Dilkina and L. Song, *Learning Combinatorial Optimization Algorithms over Graphs*, *CoRR* **abs/1704.01665** (2017) [1704.01665].

[16] H. Park and J. Neville, *Exploiting Interaction Links for Node Classification with Deep Graph Neural Networks*, in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 3223–3230, International Joint Conferences on Artificial Intelligence Organization, 7, 2019, DOI.

[17] N. Park, A. Kan, X.L. Dong, T. Zhao and C. Faloutsos, *Estimating Node Importance in Knowledge Graphs Using Graph Neural Networks*, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2019) .

[18] R. Wang, B. Li, S. Hu, W. Du and M. Zhang, *Knowledge Graph Embedding via Graph Attenuated Attention Networks*, *IEEE Access* **8** (2020) 5212.

[19] B. Wang, T. Shen, G. Long, T. Zhou, Y. Wang and Y. Chang, *Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion*, *Proceedings of the Web Conference 2021* (2021) .

[20] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang and C. Zhang, *Attributed Graph Clustering: A Deep Attentional Embedding Approach*, 2019.

[21] X. Zhang, H. Liu, Q. Li and X.-M. Wu, *Attributed Graph Clustering via Adaptive Graph Convolution*, 2019.

[22] J. Li, Y. Rong, H. Cheng, H. Meng, W. Huang and J. Huang, *Semi-Supervised Graph Classification: A Hierarchical Graph Perspective*, 2019.

[23] F.-Y. Sun, J. Hoffmann, V. Verma and J. Tang, *InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization*, 2020.

[24] T.N. Kipf and M. Welling, *Semi-Supervised Classification with Graph Convolutional Networks*, 2017.

[25] A.M. Hillas, *Cosmic Rays: Recent Progress and some Current Questions*, in *Conference on Cosmology, Galaxy Formation and Astro-Particle Physics on the Pathway to the SKA*, 7, 2006 [astro-ph/0607109].

[26] R. Engel, D. Heck and T. Pierog, *Extensive Air Showers and Hadronic Interactions at High Energy*, *Annual Review of Nuclear and Particle Science* **61** (2011) 467 [https://doi.org/10.1146/annurev.nucl.012809.104544].

[27] F. Halzen, *Neutrinos at the Ends of the Earth*, *Scientific American* **313** (2015) 58.

[28] M. Aartsen, M. Ackermann, J. Adams, J.A. Aguilar, M. Ahlers, M. Ahrens et al., *Cosmic ray spectrum and composition from PeV to EeV using 3 years of data from IceTop and IceCube*, *Physical Review D* **100** (2019) .

[29] R. Abbasi, M. Ackermann, J. Adams, J. Aguilar, M. Ahlers, M. Ahrens et al., *A convolutional neural network based cascade reconstruction for the IceCube Neutrino Observatory*, *Journal of Instrumentation* **16** (2021) P07041.

[30] M.G. Aartsen, R. Abbasi, M. Ackermann, J. Adams, J.A. Aguilar, M. Ahlers et al., *IceCube-Gen2: the window to the extreme Universe*, *Journal of Physics G: Nuclear and Particle Physics* **48** (2021) 060501.

[31] A. Omeliukh, *Optimization of the optical array geometry for IceCube-Gen2*, 2021.

[32] A. Leszczyńska, M. Weyrauch, A. Coleman, R. Abbasi, M. Ackermann, J. Adams et al., *Study of mass composition of cosmic rays with IceTop and IceCube*, *Proceedings of 37th International Cosmic Ray Conference — PoS(ICRC2021)* (2021) .

[33] S. Hallmann, B. Clark, C. Glaser and D. Smith, *Sensitivity studies for the IceCube-Gen2 radio array*, 2021.

[34] P. Koundal, M. Plum, J. Saffer, R. Abbasi, M. Ackermann, J. Adams et al., *Study of mass composition of cosmic rays with IceTop and IceCube*, *Proceedings of 37th International Cosmic Ray Conference — PoS(ICRC2021)* (2021) .

[35] P. Koundal for IceCube Collaboration, "Composition Study of Cosmic Rays with IceCube Observatory using Graph Neural Networks." Vortrag gehalten auf DPG Frühjahrstagung der Fachverbände Teilchenphysik, Strahlen- und Medizinphysik (2021), Online, 15.–19. März 2021, 2021.

[36] M.H. Minh, *Reconstruction of Neutrino Events in IceCube using Graph Neural Networks*, 2021.

[37] N. Choma, F. Monti, L. Gerhardt, T. Palczewski, Z. Ronaghi, Prabhat et al., *Graph Neural Networks for IceCube Signal Classification*, 2018.

[38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, 2019.