WIFAI 2024

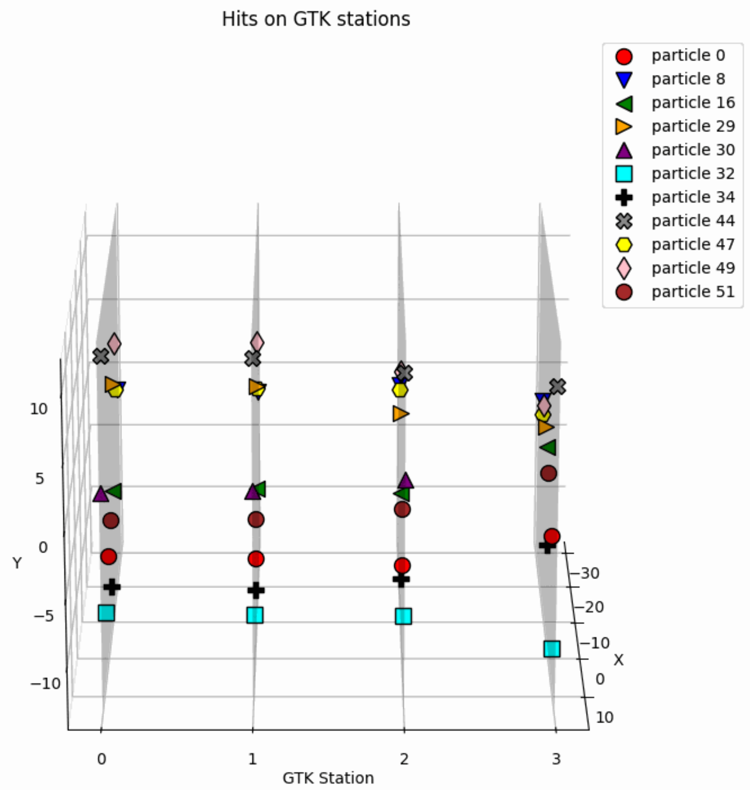
Leonardo Plini
National PhD student in Artificial Intelligence
INFN-LNF and Sapienza University of Rome
leonardo.plini@lnf.infn.it

INFN
Istituto Nazionale di Fisica Nucleare
Laboratori Nazionali di Frascati

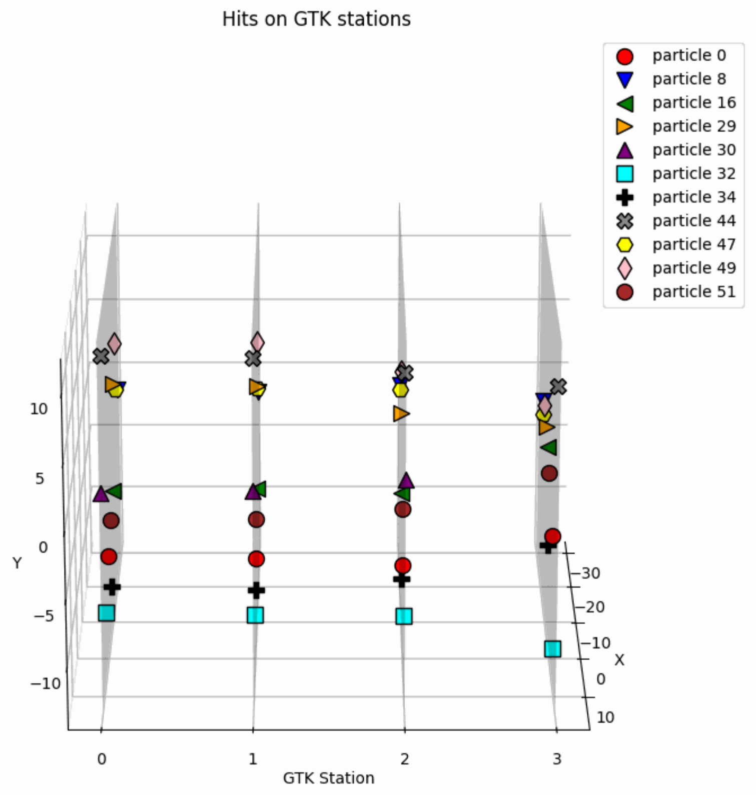
PINlab
PERCEPTION AND INTELLIGENCE

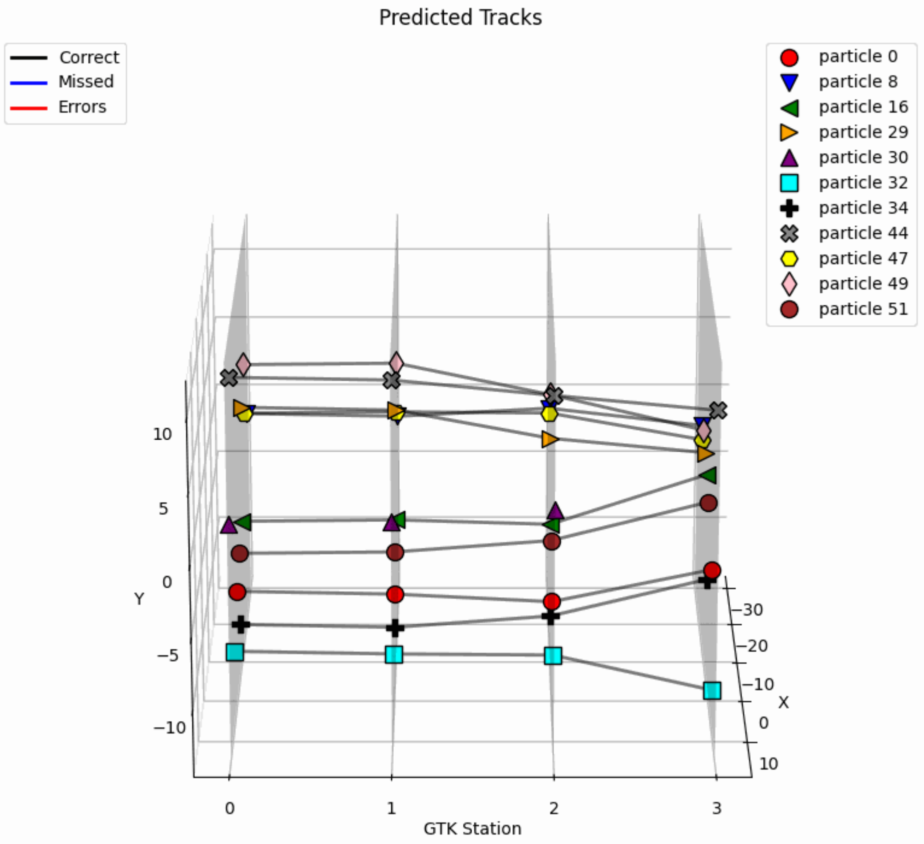
SAPIENZA
UNIVERSITÀ DI ROMA

Particle Tracking

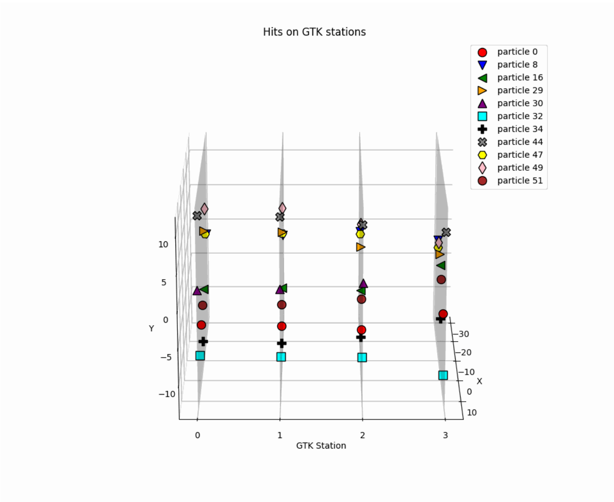


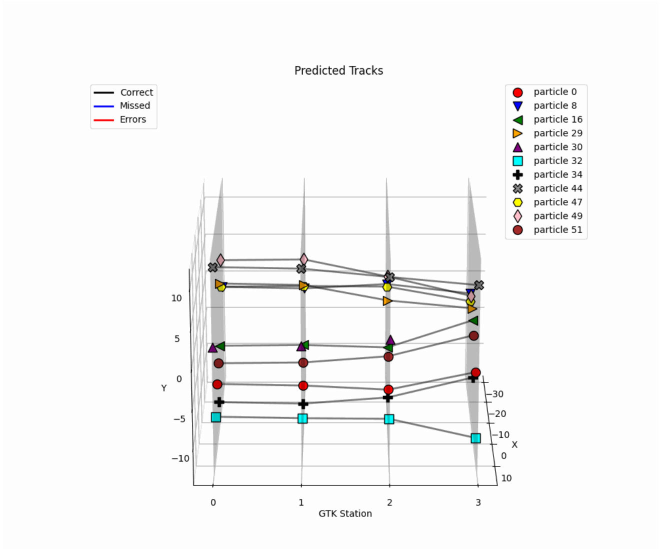
Particle Tracking



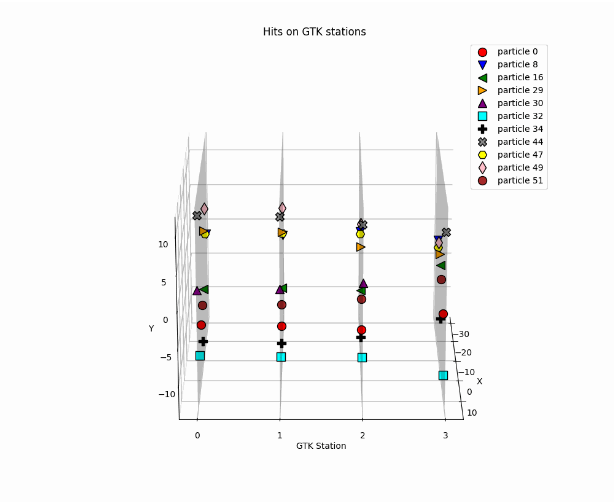


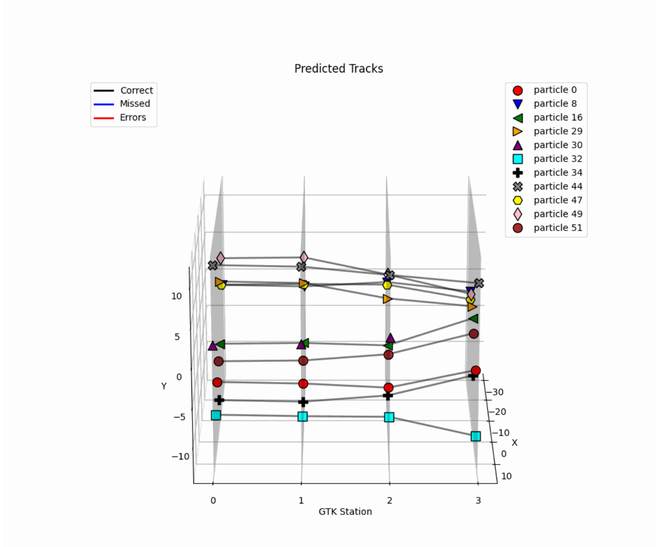
Particle Tracking





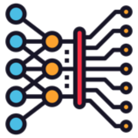
Particle Tracking



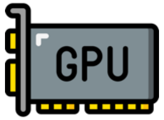


Motivations

Despite the variety of approaches and theoretical models tested in physical experiments, what they all have in common is the very **large volume of complex data** they produce

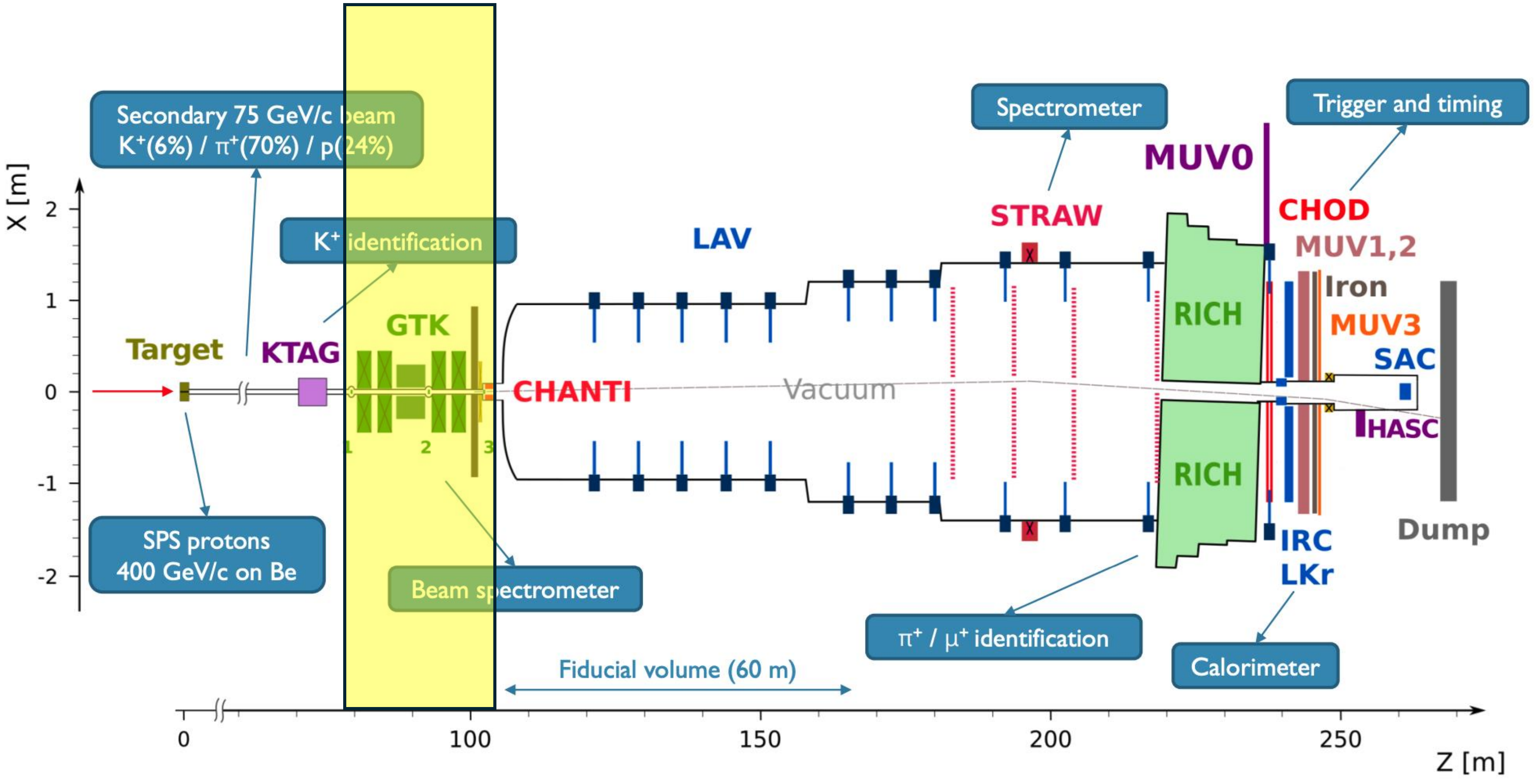


This data challenge calls for powerful computing methods like **Machine Learning** and **Deep Learning**

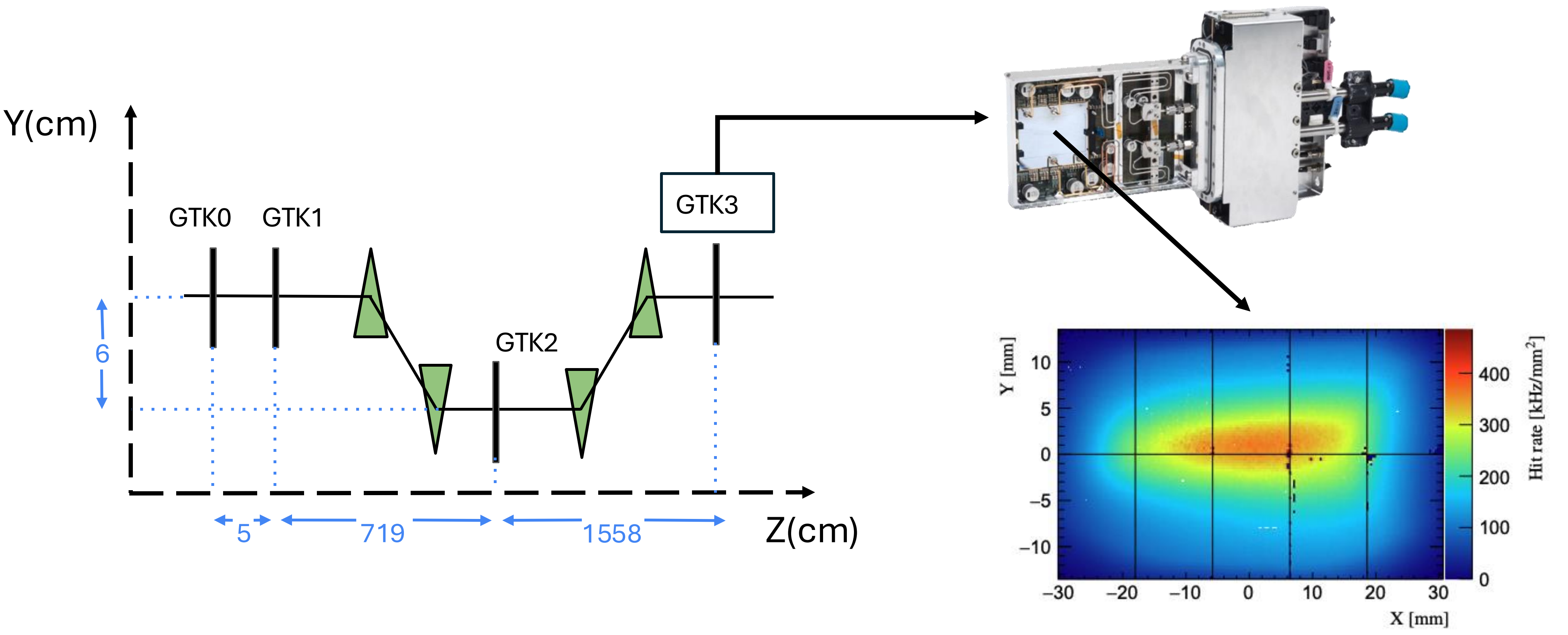


GPU parallelization and memory capacity allow to drastically reduce computational time

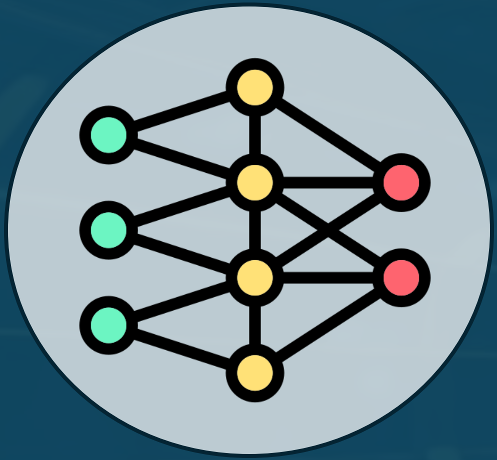
Introduction: NA62 experiment



Introduction: Giga Tracker Stations

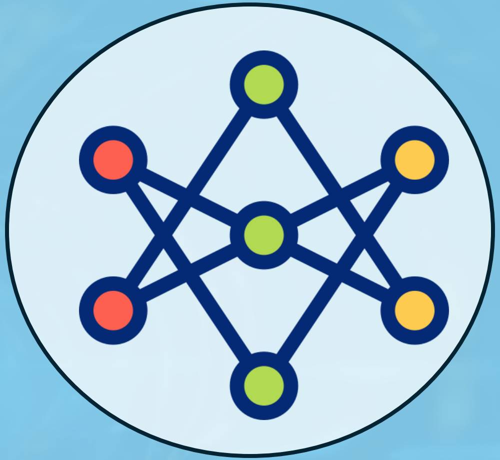


Proposal



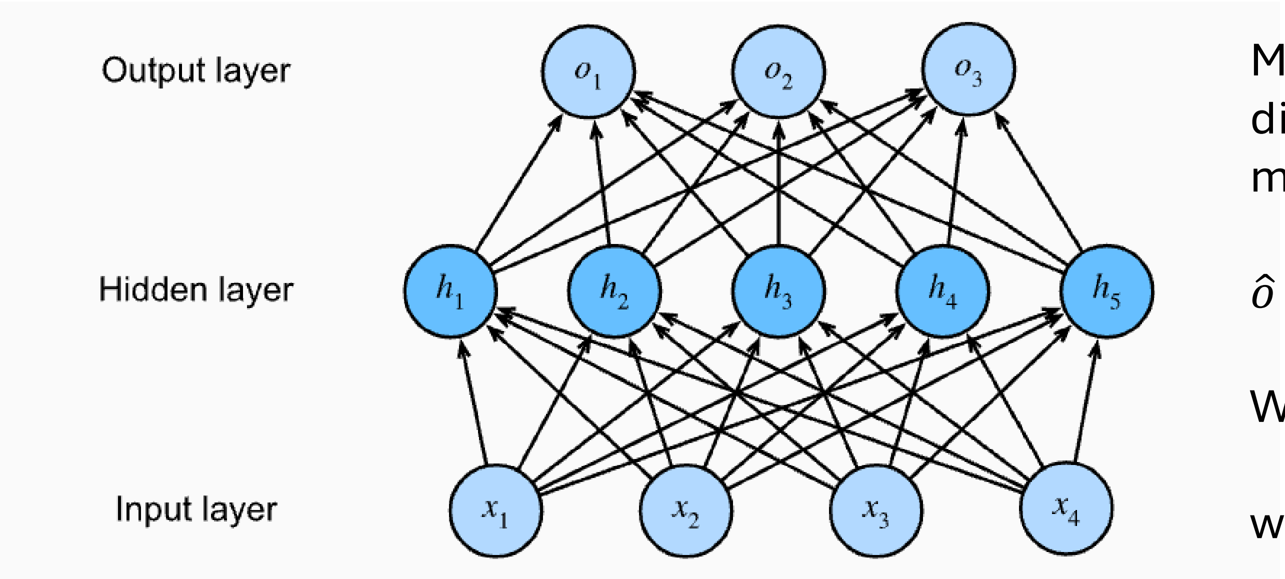
Multi-Layer Perceptron

Transformer



Graph Neural Network

Multi-Layer Perceptron



MLP is an architecture made up of composable and differentiable layers that optimizes its weights (W, b) by means of **Back-Propagation** to minimize a loss function L.

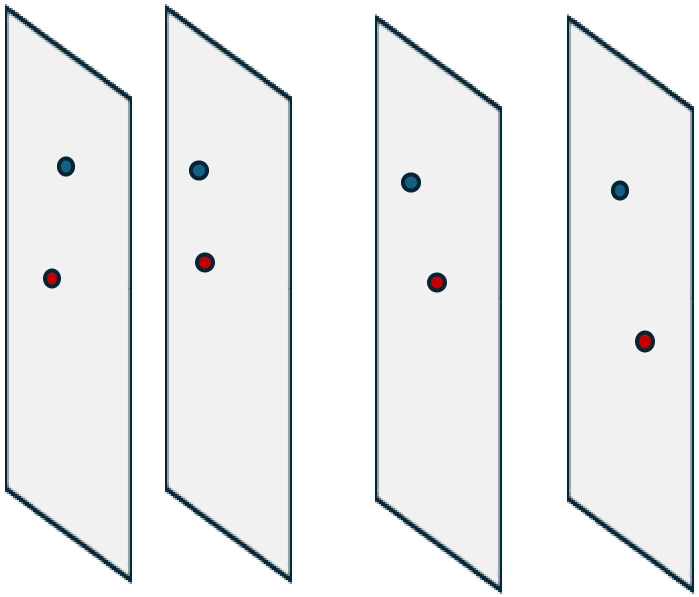
$\hat{o}$ = f(x) $= \sigma_2(W_2 \sigma_1(W_1 x + b_1) + b_2)$

W*, b* = argmin L(o, $\hat{o}$)

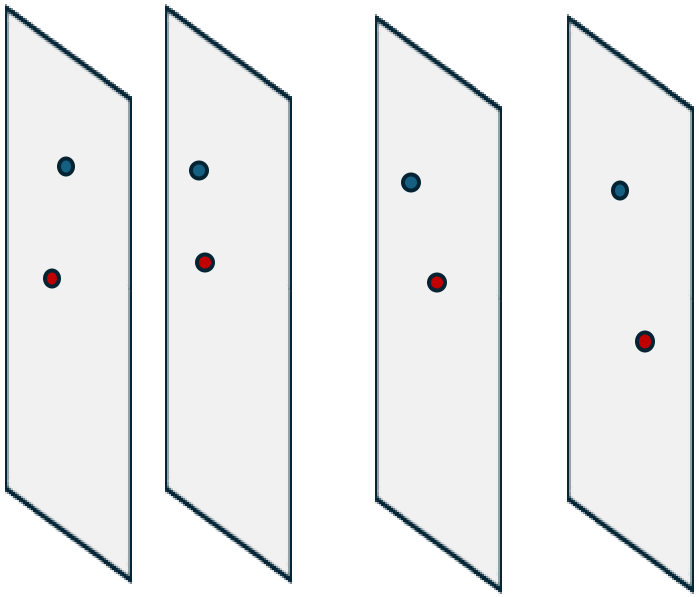
where σ *is an activation function*

* Image from Dive into Deep Learning, chapter 5

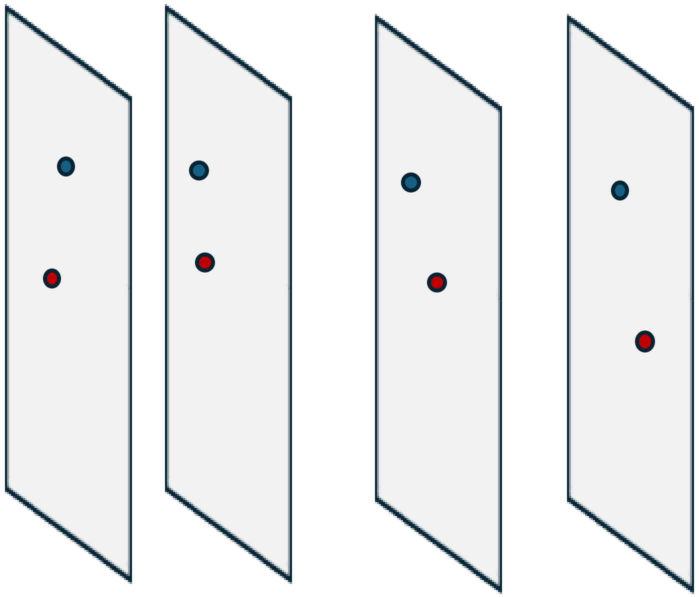
Multi-Layer Perceptron



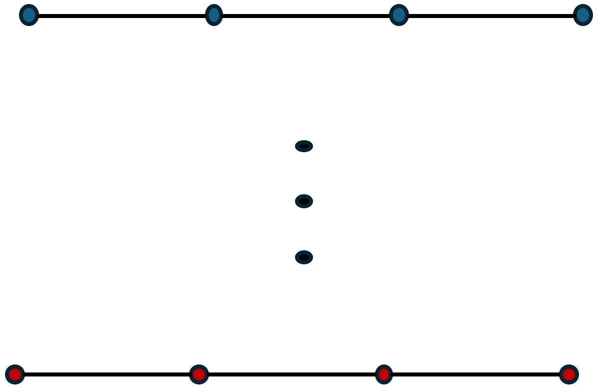
Multi-Layer Perceptron



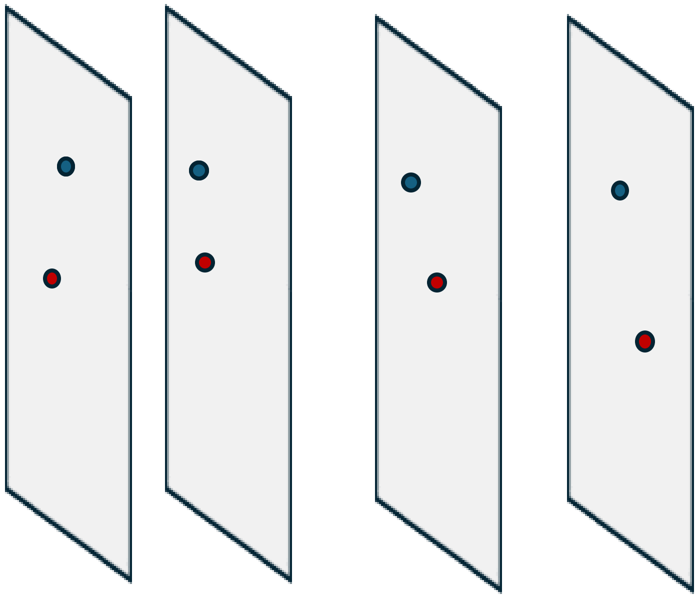
Multi-Layer Perceptron



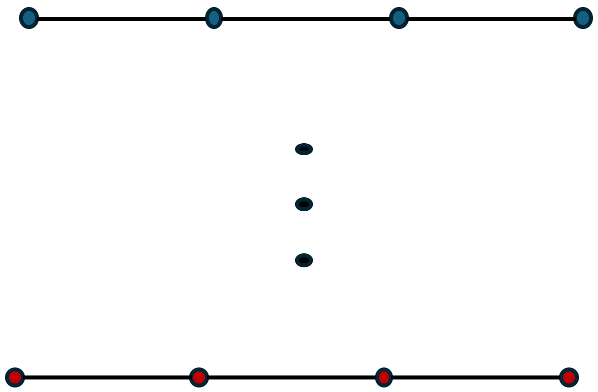
Admissible track combinations



Multi-Layer Perceptron

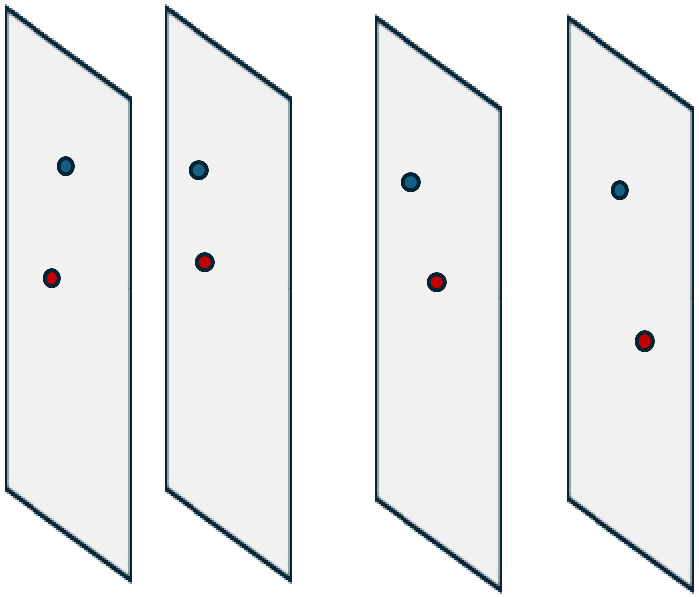


Admissible track combinations

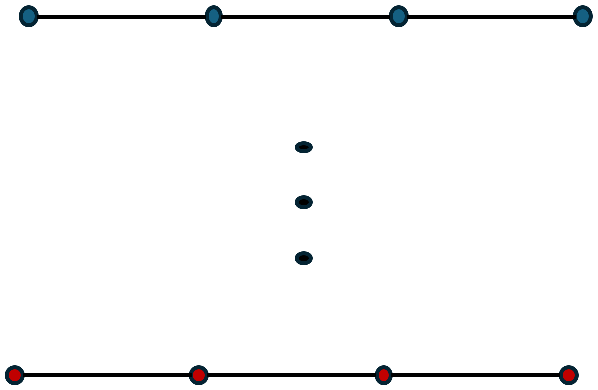


Not admissible track combination

Multi-Layer Perceptron

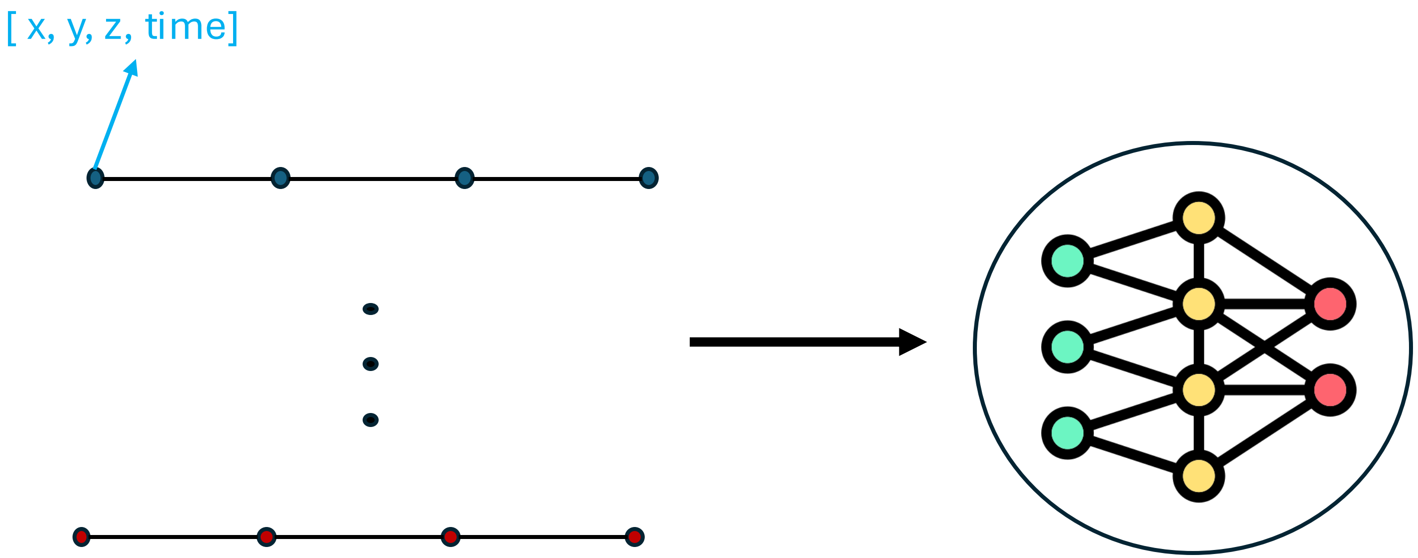


Admissible track combinations

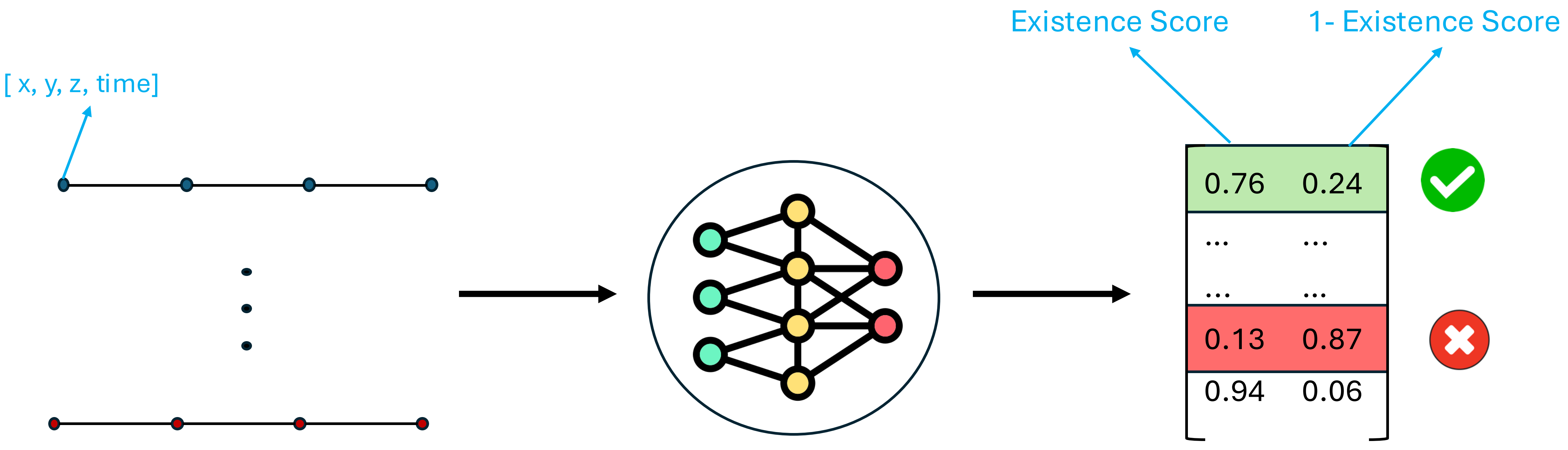


Not admissible track combination

Multi-Layer Perceptron



Multi-Layer Perceptron



Multi-Layer Perceptron: results

Efficiency $= \dfrac{\#\, correctly\ predicted\ tracks}{\#\, total\ true\ tracks}$

Purity $= \dfrac{\#\, correctly\ predicted\ tracks}{\#\, total\ predicted\ tracks}$

FAKE

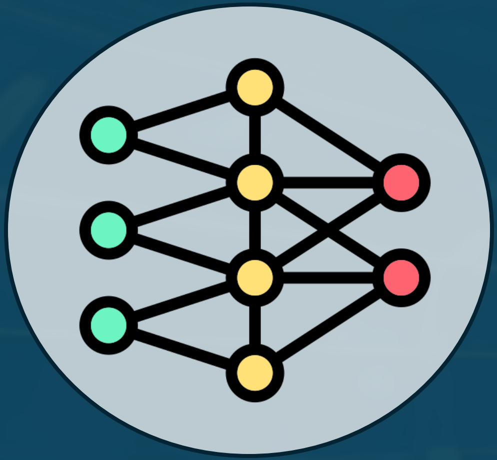
Fake Tracks $= \dfrac{\#\, wrongly\ predicted\ tracks}{\#\, total\ true\ tracks}$

- NA62 MC reproducing the datataking condition in 2022
- 200156 events {-10 ns, +10 ns} wrt KTAG reference time
- The models were implemented on an RTX 3060Ti Trio with 8 GB equipped with a Ryzen 7 3700X CPU
- 60% Train, 20% Validation, 20% Test
- Training on Train Set
- Hyper-parameters using Validation Set
- Results on Test Set

Efficiency	Purity	Fake Tracks
70.06 %	92.3 %	39.45 %

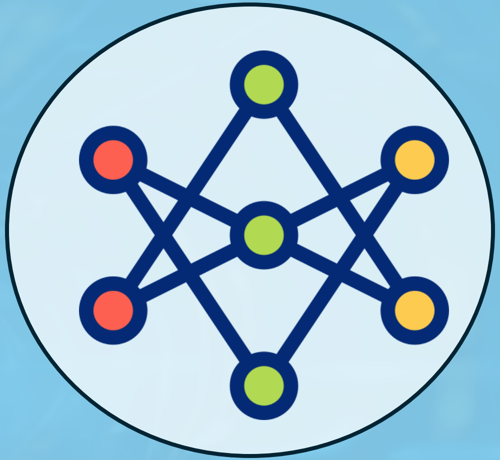
Multi-Layer Perceptron: conclusion

- Local awareness
- Poor Performances
- Big GPU memory needed to store all admissible combinations
- Slow computations
- **Class Imbalance**



Multi-Layer Perceptron

Transformer



Graph Neural Network

Transformer

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

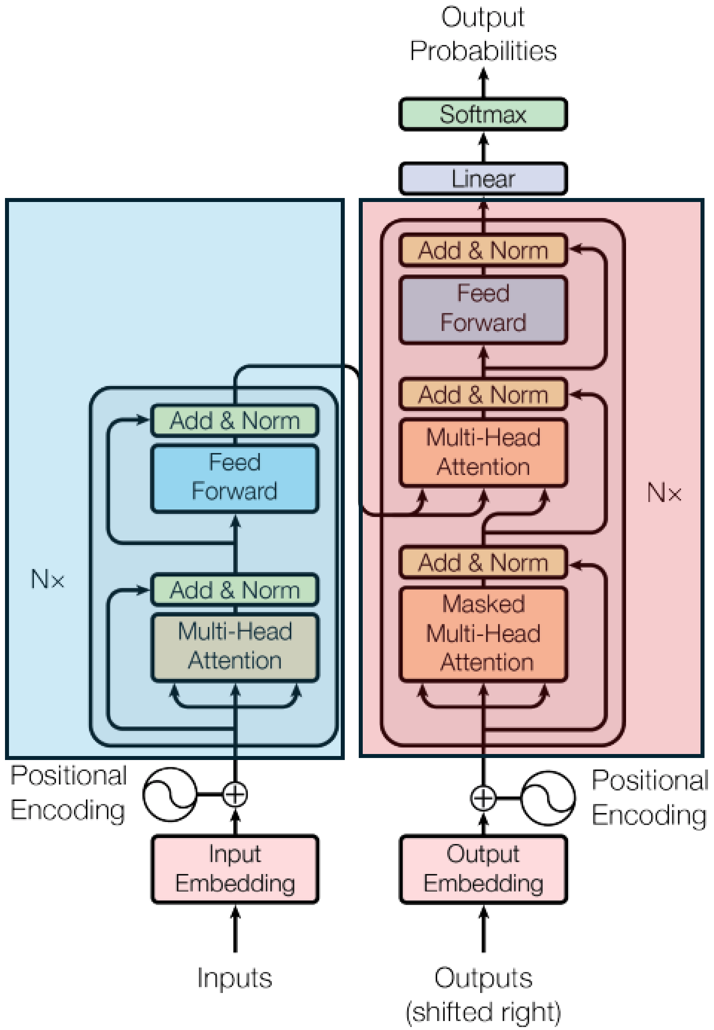
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Presented in 2017 and now widely adopted due this their incredible performances and parallelization.

They represented a revolution in Natural Language Processing, Computer Vision and Multimodal Learning

Transformer



Encoder: processes the input sequence by applying self-attention to capture relationships between all tokens in the sequence. It then passes the resulting representations through feed-forward layers to create a context-aware representation of the input data

Decoder: uses this representation to generate the output sequence, often performing tasks like translation or text generation

Note: we used only the Encoder

Transformer

Local awareness

If we have n hits per station, we will create:

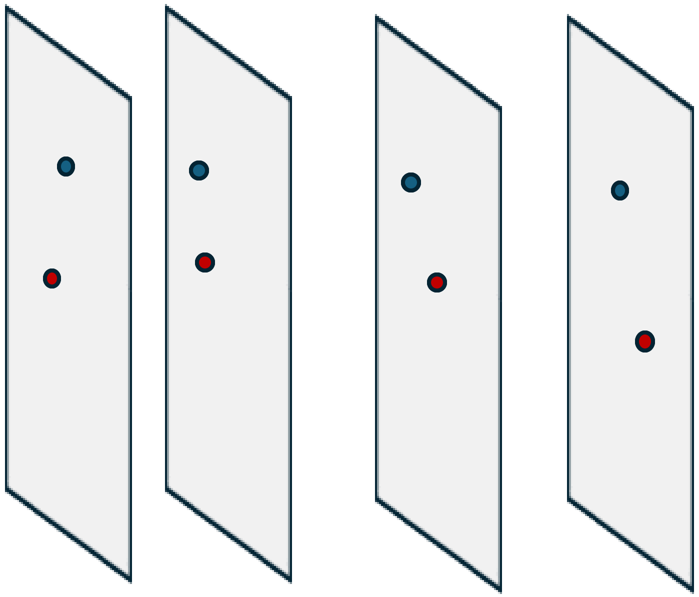
- n^4 (GTK0-GTK1-GTK2-GTK3)
- n^3 (GTK0-GTK2-GTK3)
- n^3 (GTK1-GTK2-GTK3)

Total: n^4+2n^3

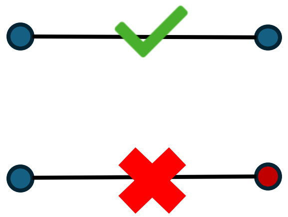
Global awareness with Transfomer

What if move the problem from tracks to edges?

Transformer

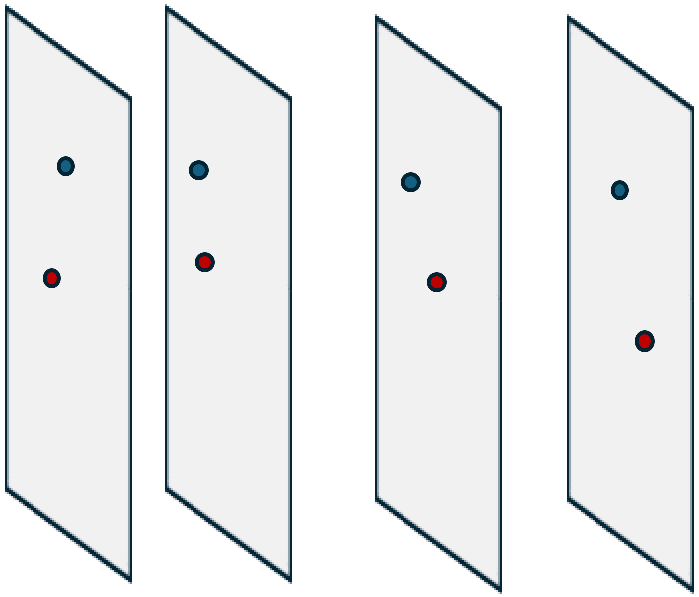


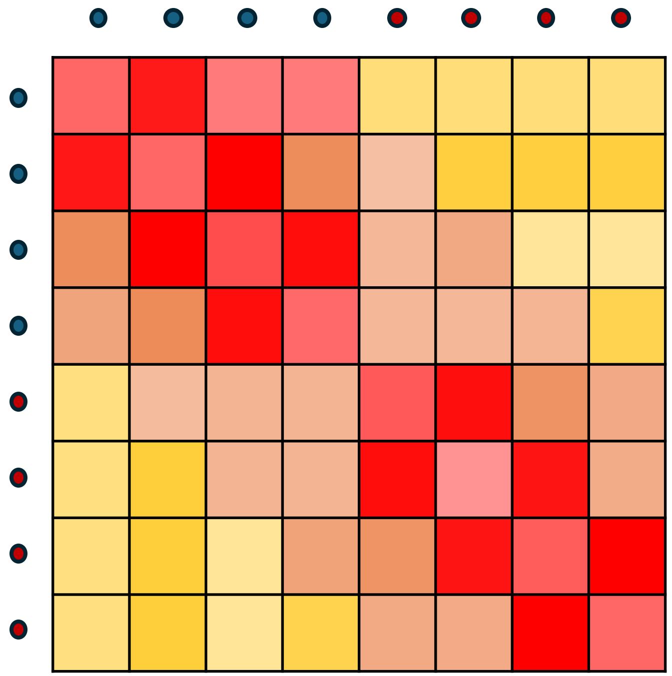
Binary Classification



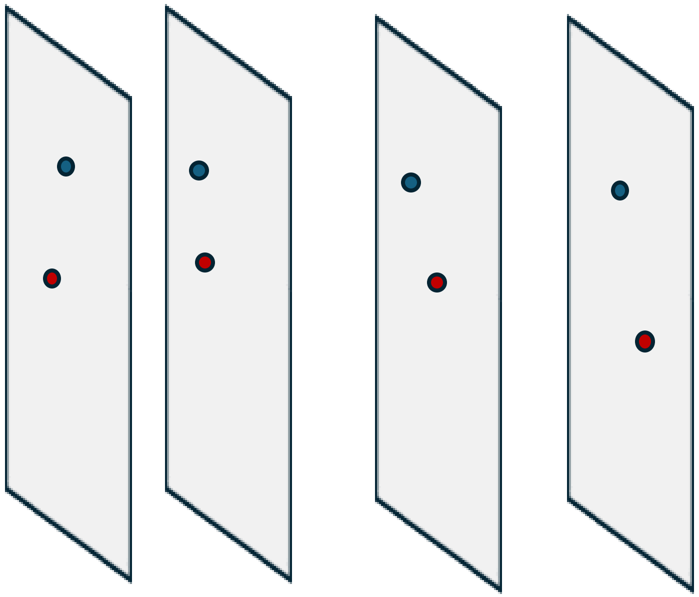
The number of candidates we need to evaluate is $(4n)^2 = 16n^2 << n^4 + 2n^3$

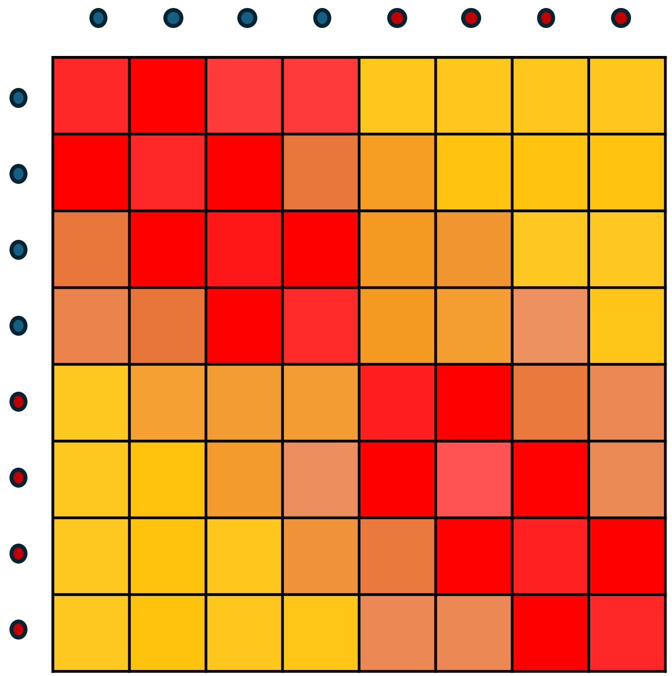
Transformer



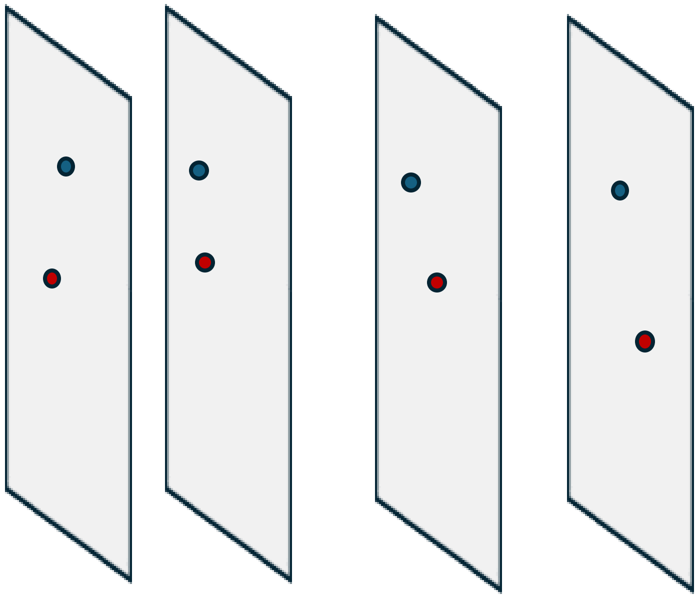


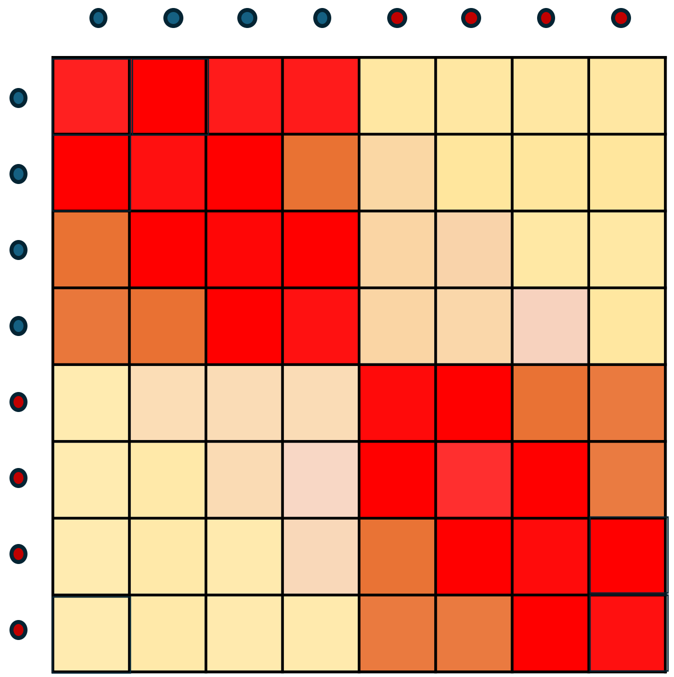
Transformer



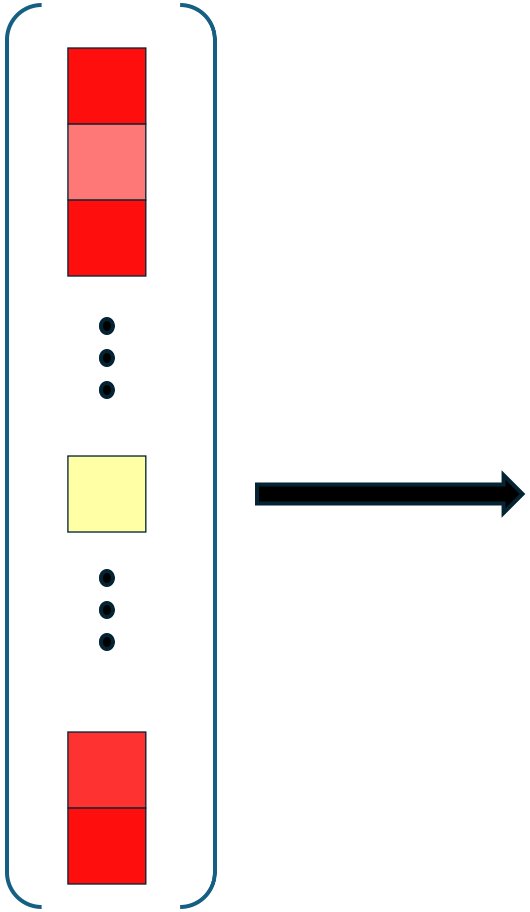


Transformer



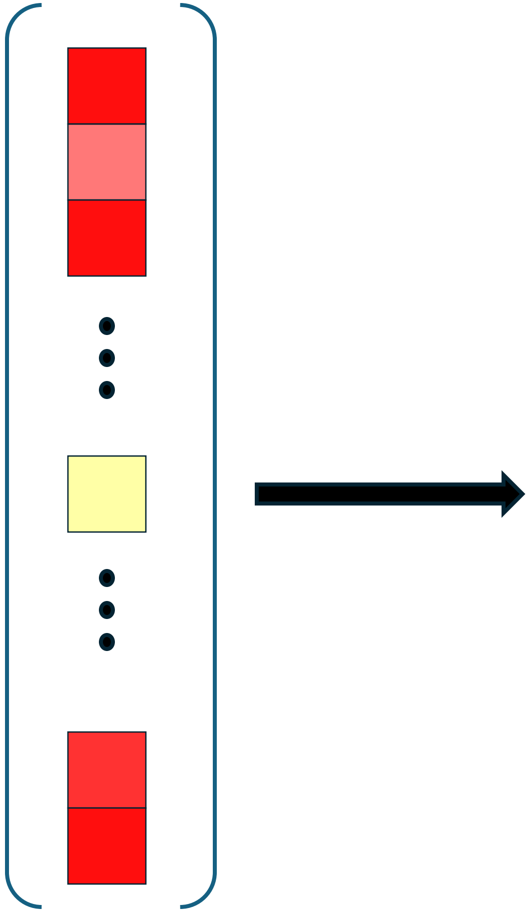


Transformer



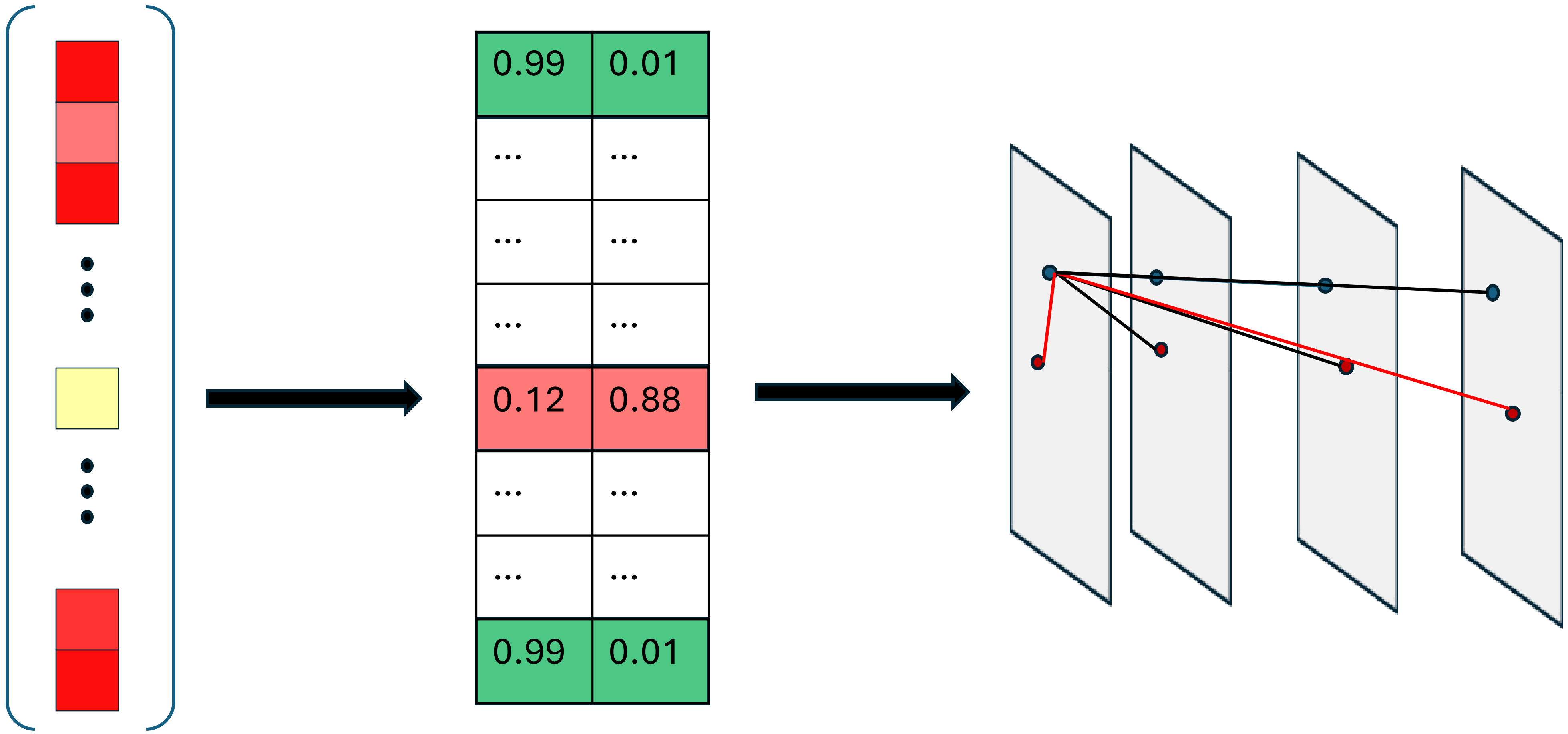
0.99	0.01
…	…
…	…
…	…
0.12	0.88
…	…
…	…
0.99	0.01

Transformer



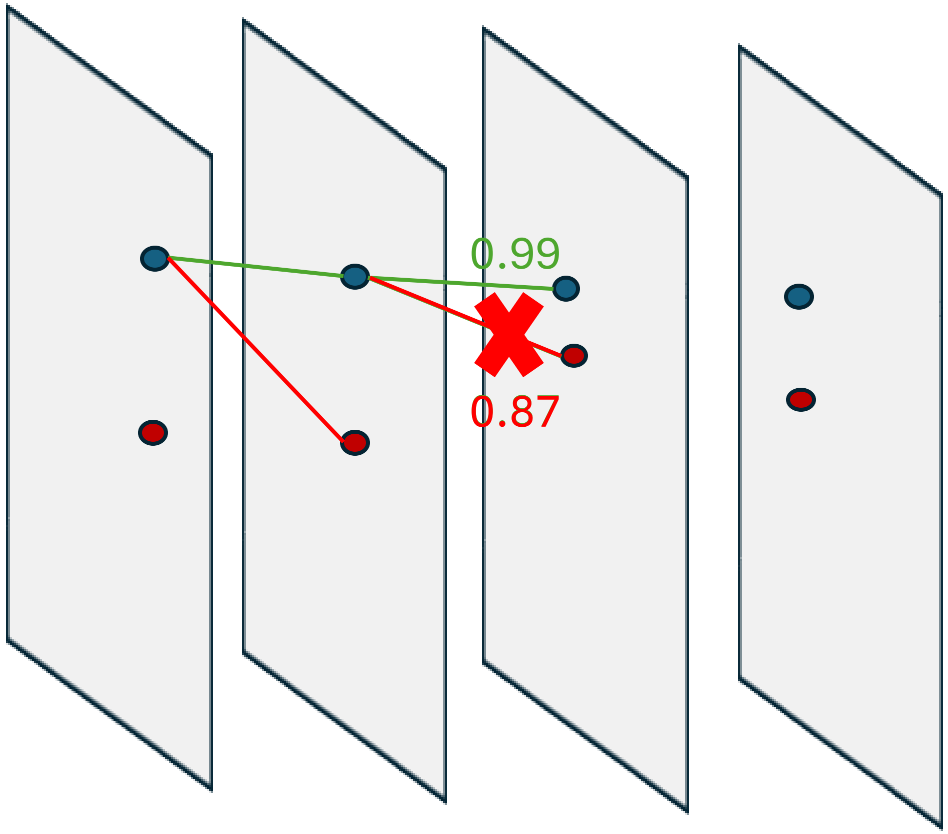
0.99	0.01
...	...
...	...
...	...
0.12	0.88
...	...
...	...
0.99	0.01

Transformer



Transformer

Max Edge selection



Transformer

Efficiency $= \dfrac{\#\ correctly\ predicted\ tracks}{\#\ total\ true\ tracks}$

Purity $= \dfrac{\#\ correctly\ predicted\ tracks}{\#\ total\ predicted\ tracks}$

Fake Tracks $= \dfrac{\#\ wrongly\ predicted\ tracks}{\#\ total\ true\ tracks}$

Multi-Layer Perceptron

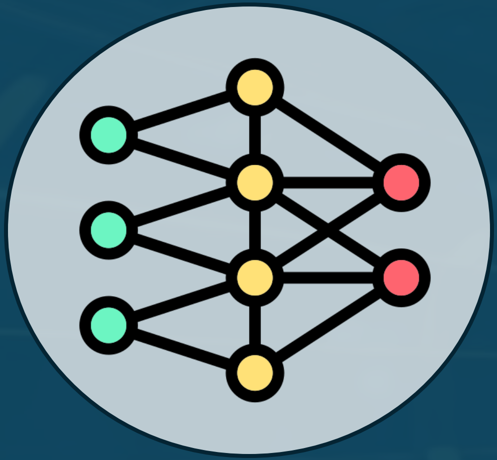
Efficiency	Purity	Fake Tracks
70.06 %	92.3 %	39.45 %

Transformer

Efficiency	Purity	Fake Tracks
95.95 %	98.62 %	1.22 %

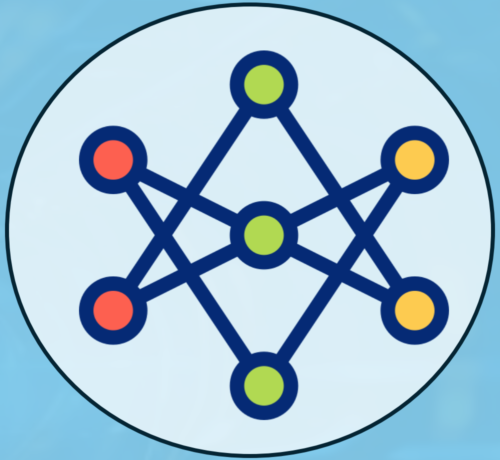
Transformer

- Solved class imbalance
- Global awareness
- Efficient computations
- Many computations to consider all the connections



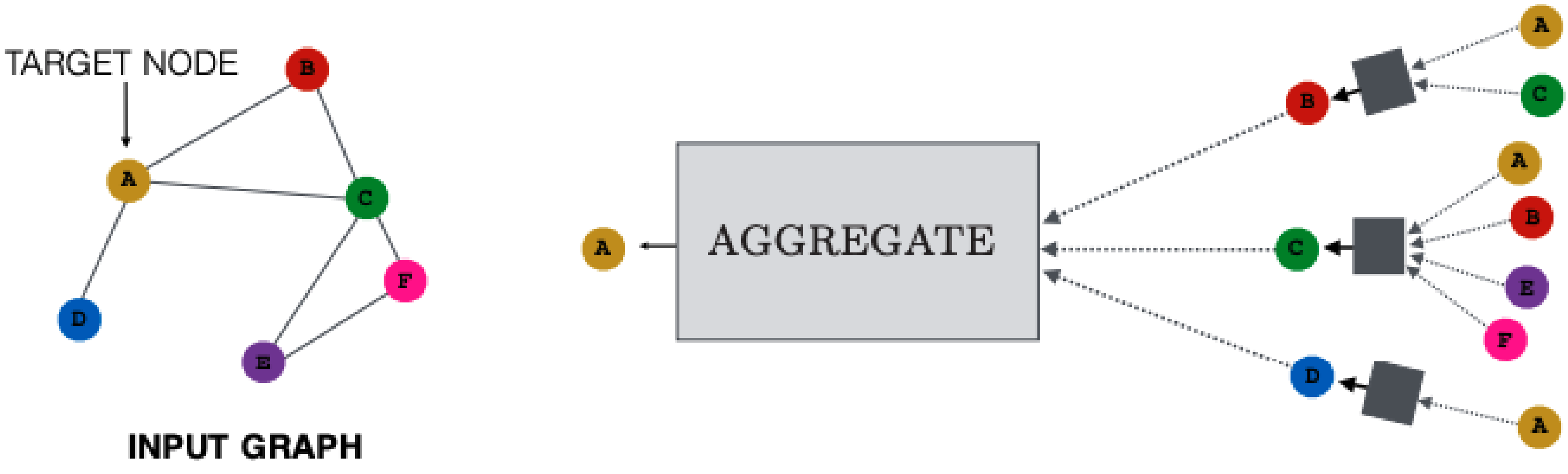
Multi-Layer Perceptron

Transformer



Graph Neural Network

Graph Neural Network



Graph Neural Networks are a type of neural network designed to work with graph-structured data.

They **propagate** and **aggregate** information across nodes and edges in a graph, allowing them to learn representations that capture the relationships and structure within the data

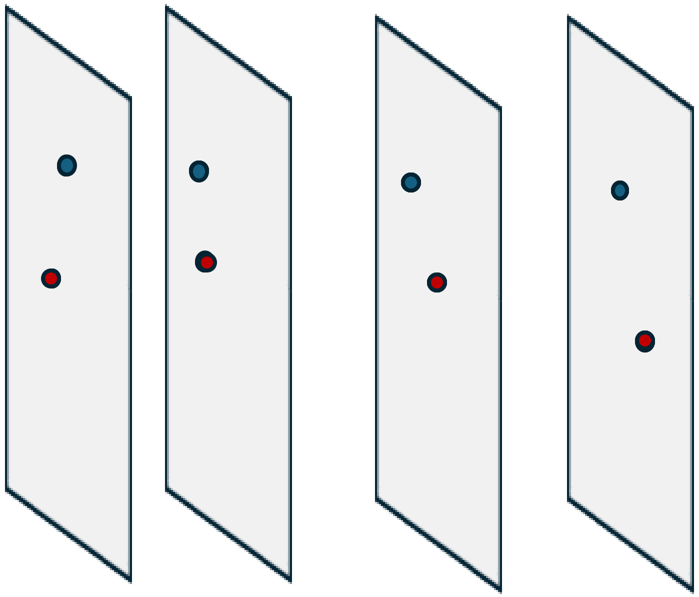
* Image from "Representation Learning on Graphs: Methods and Applications"

Graph Neural Network

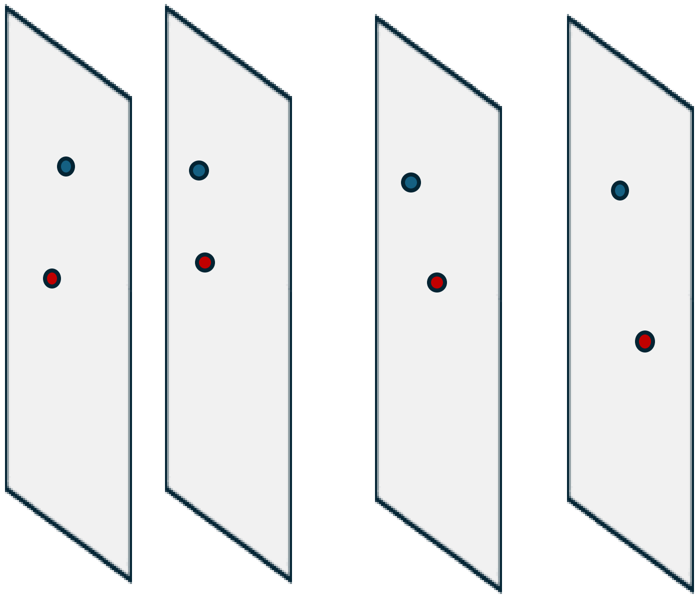
Many computations to consider all the connections

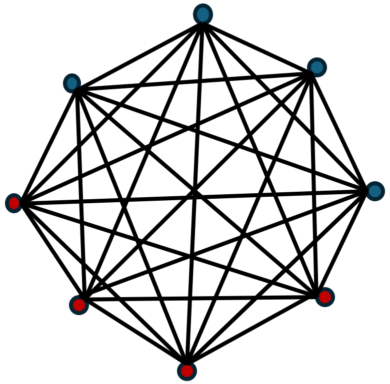
Graphs allow to decide the topology (e.g. the connections between nodes)

Graph Neural Network

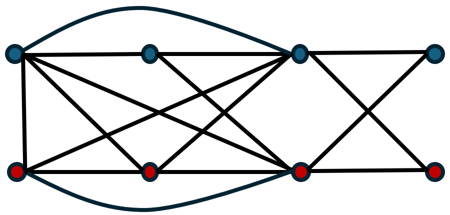


Graph Neural Network





Complete Graph

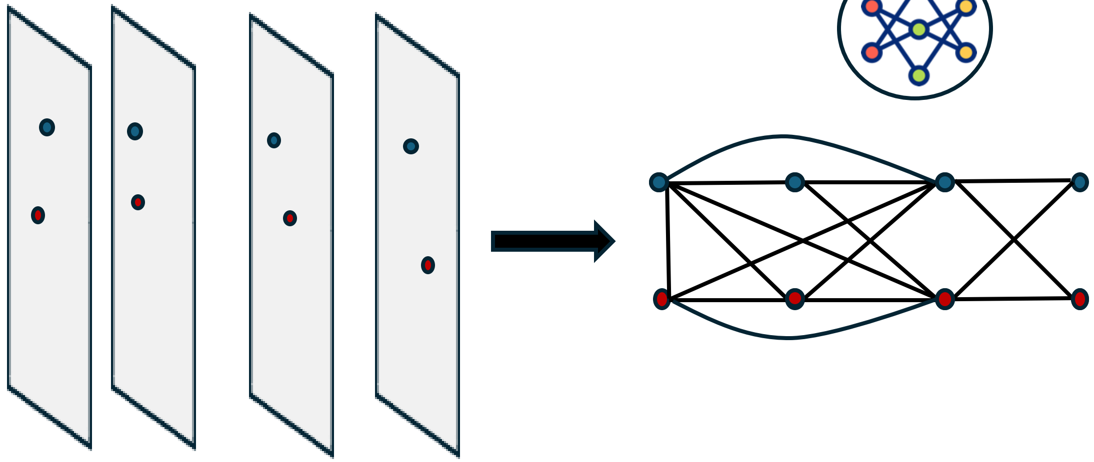


Sparse Graph

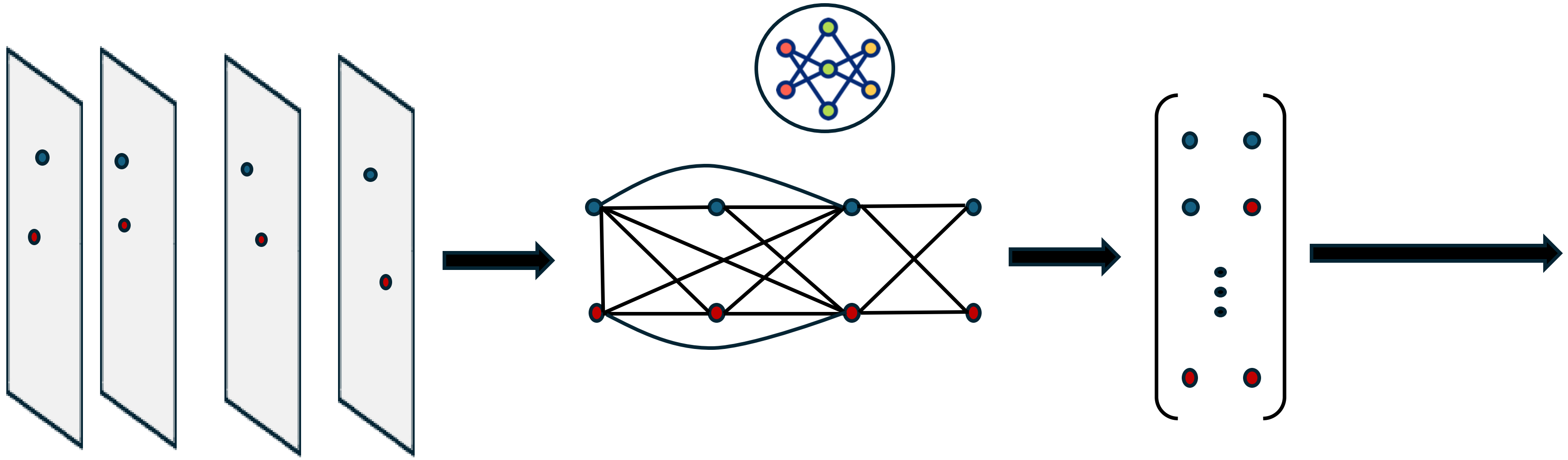
If we consider possible edges between hits:
- $2n^2$ (GTK0-GTK1 and GTK0-GTK2)
- n^2 (GTK1-GTK2)
- n^2 (GTK2-GTK3)

Total: $4n^2 < 16n^2$

Graph Neural Network

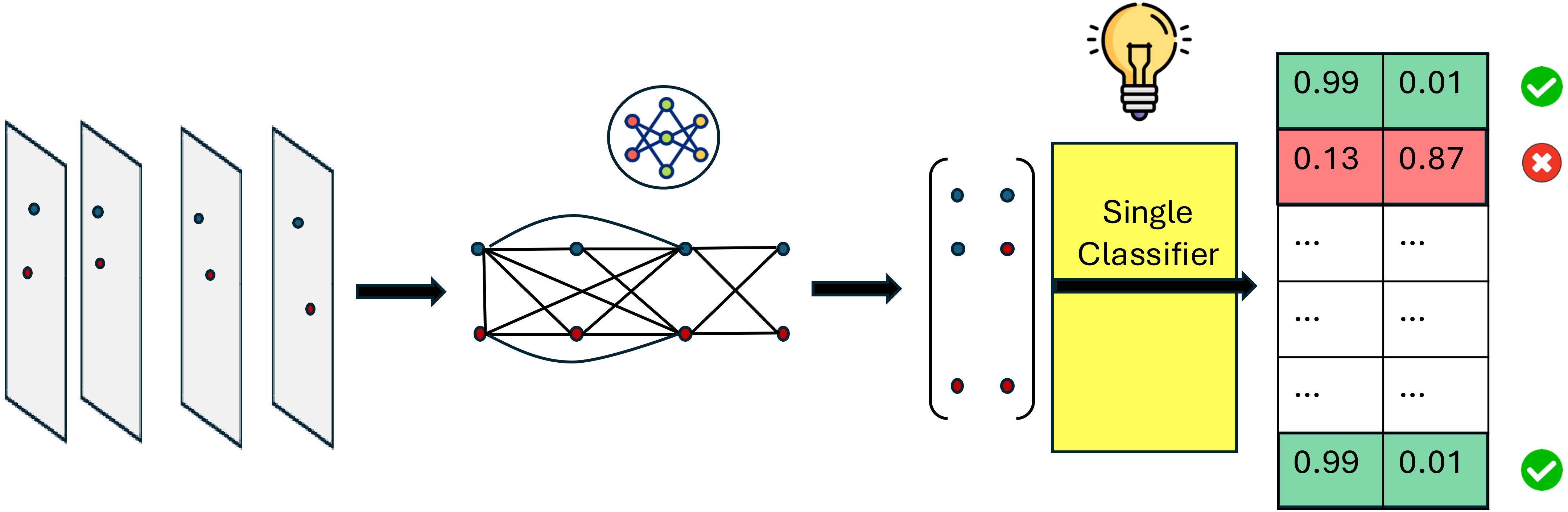


Graph Neural Network

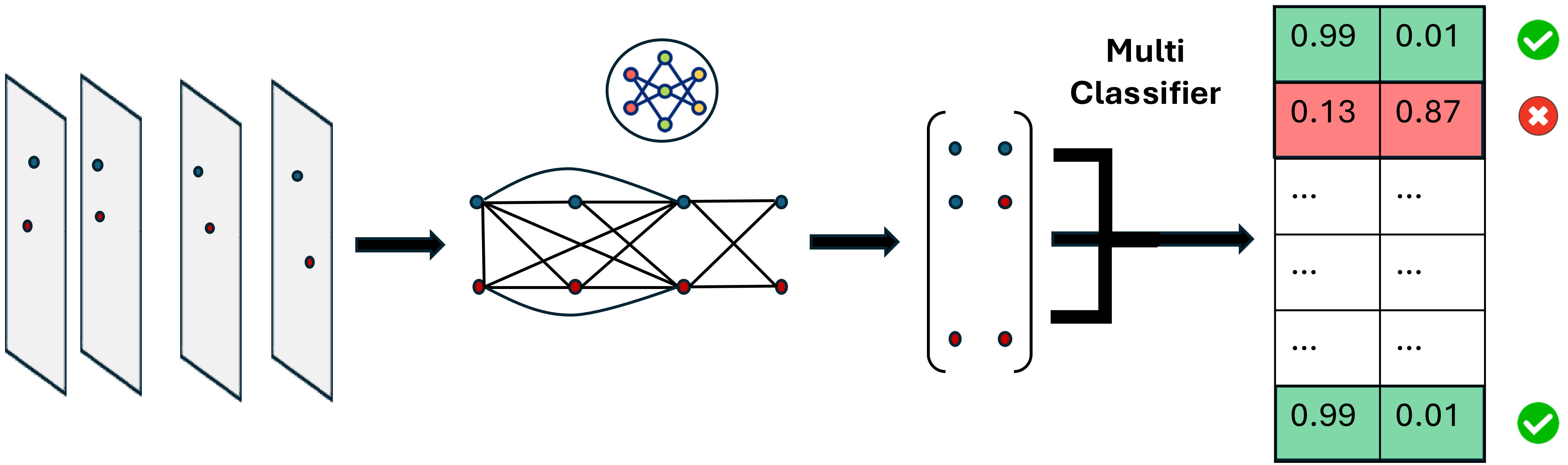


0.99	0.01
0.13	0.87
…	…
…	…
…	…
0.99	0.01

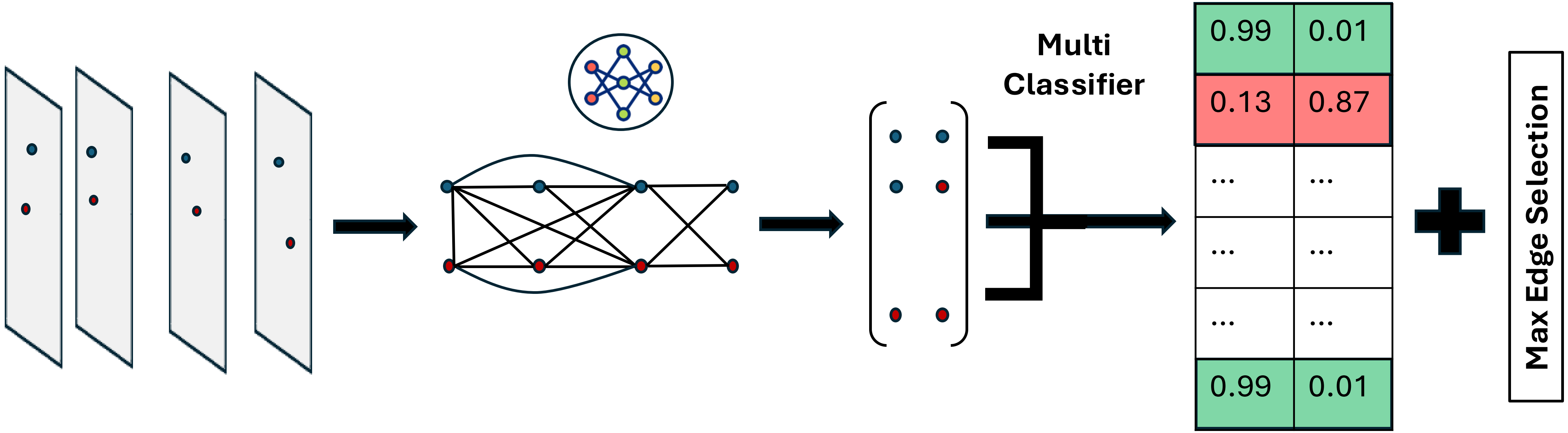
Graph Neural Network



Graph Neural Network



Graph Neural Network



Graph Neural Network

Efficiency $= \dfrac{\#\ correctly\ predicted\ tracks}{\#\ total\ true\ tracks}$

Purity $= \dfrac{\#\ correctly\ predicted\ tracks}{\#\ total\ predicted\ tracks}$

Fake Tracks $= \dfrac{\#\ wrongly\ predicted\ tracks}{\#\ total\ true\ tracks}$

Multi-Layer Perceptron

Efficiency	Purity	Fake Tracks
70.06 %	92.3 %	39.45 %

Transformer

Efficiency	Purity	Fake Tracks
95.95 %	98.62 %	1.22 %

Graph Neural Network*

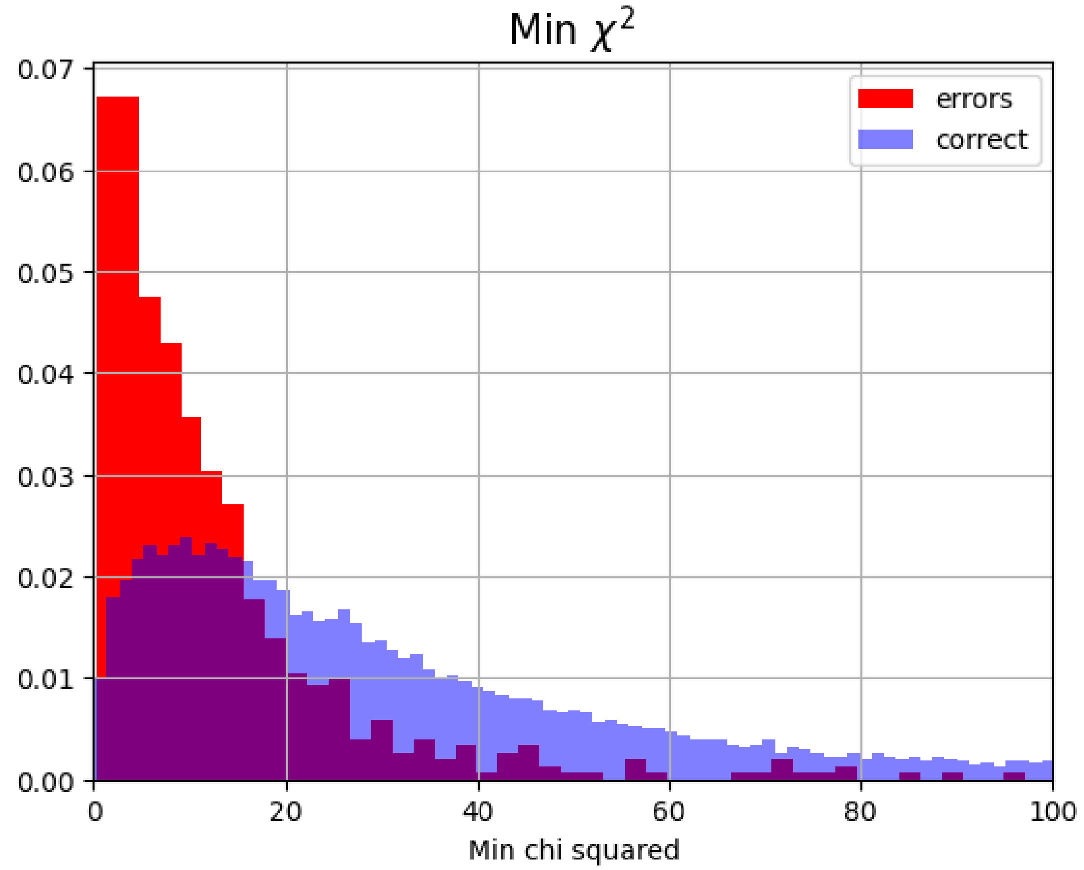
Efficiency	Purity	Fake Tracks
94.78 %	99.78 %	0.21 %

* Direct, Fully-connected graph with Graph Convolutional Network and Multi-Classifier

Graph Neural Network

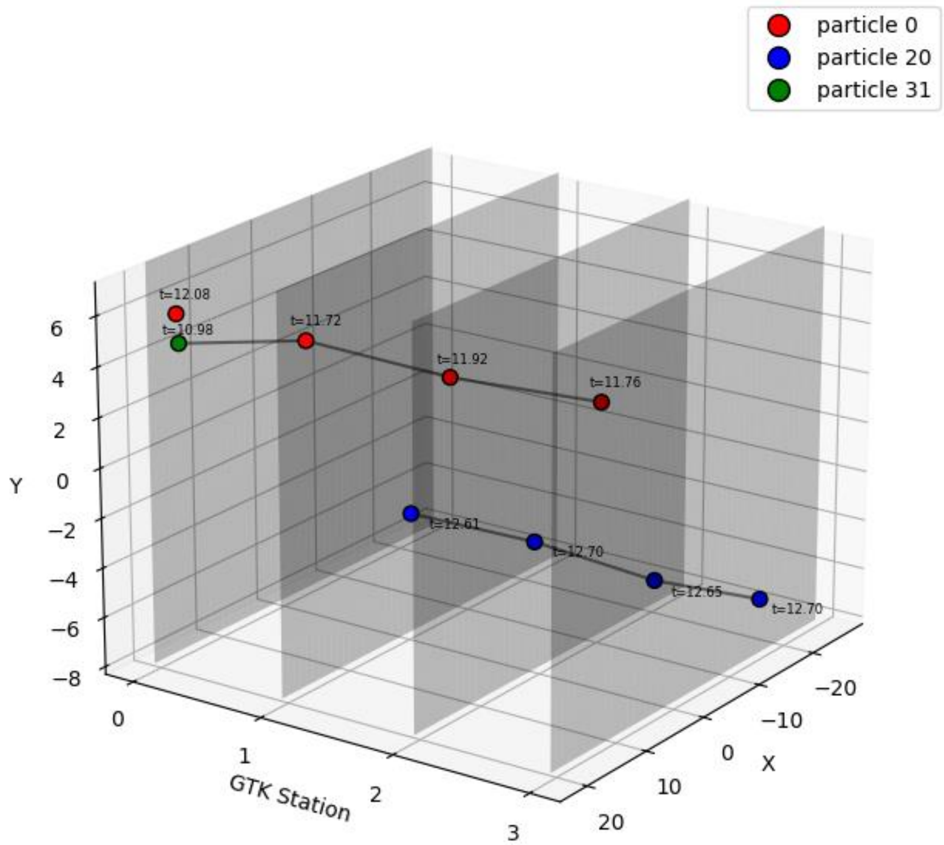
Difficulty Score

$$\chi^2 = \min_{k=0,1,2,3}\left(\min_{1\le i<j\le n} [(x_{i,k} - x_{j,k})^2 + (y_{i,k} - y_{j,k})^2 + (t_{i,k} - t_{j,k})^2] \right)$$

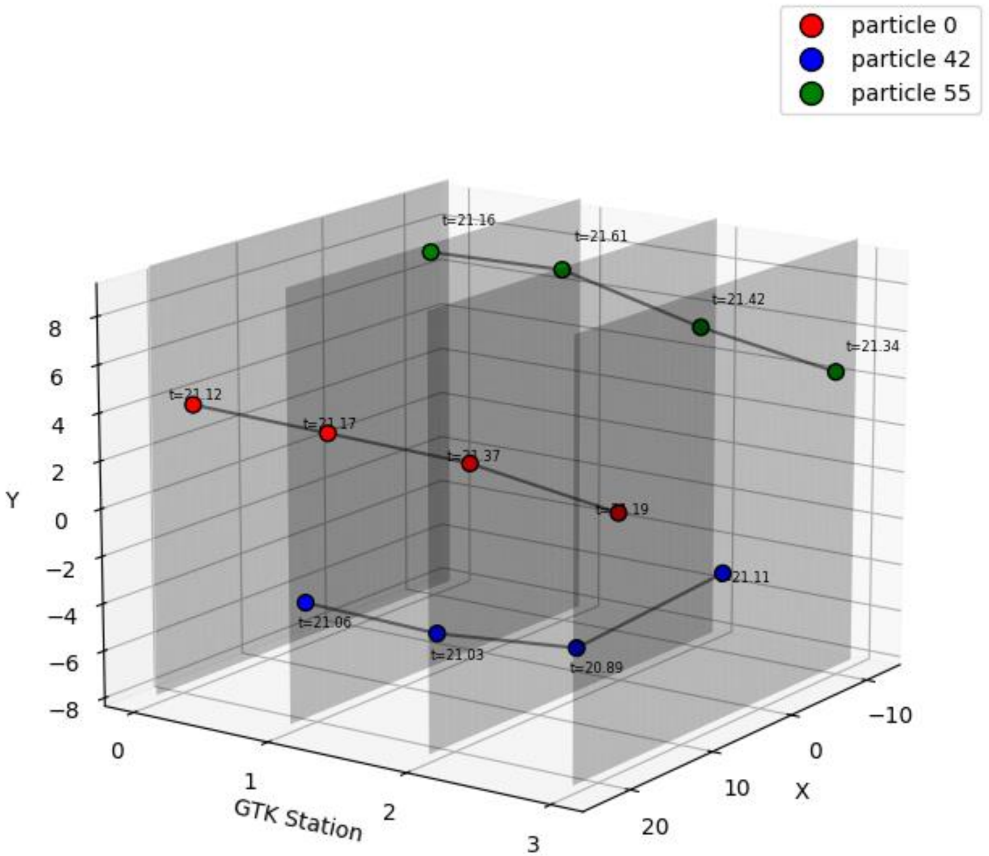


Graph Neural Network

Example of an error



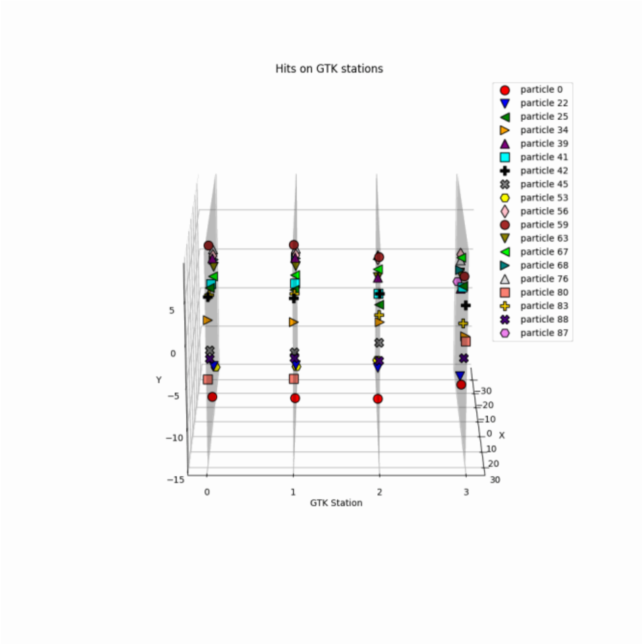
Example of a good prediction

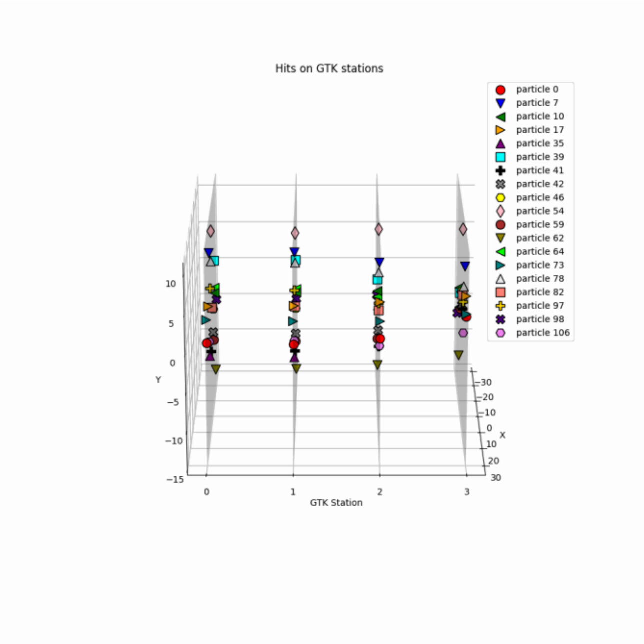


Graph Neural Network

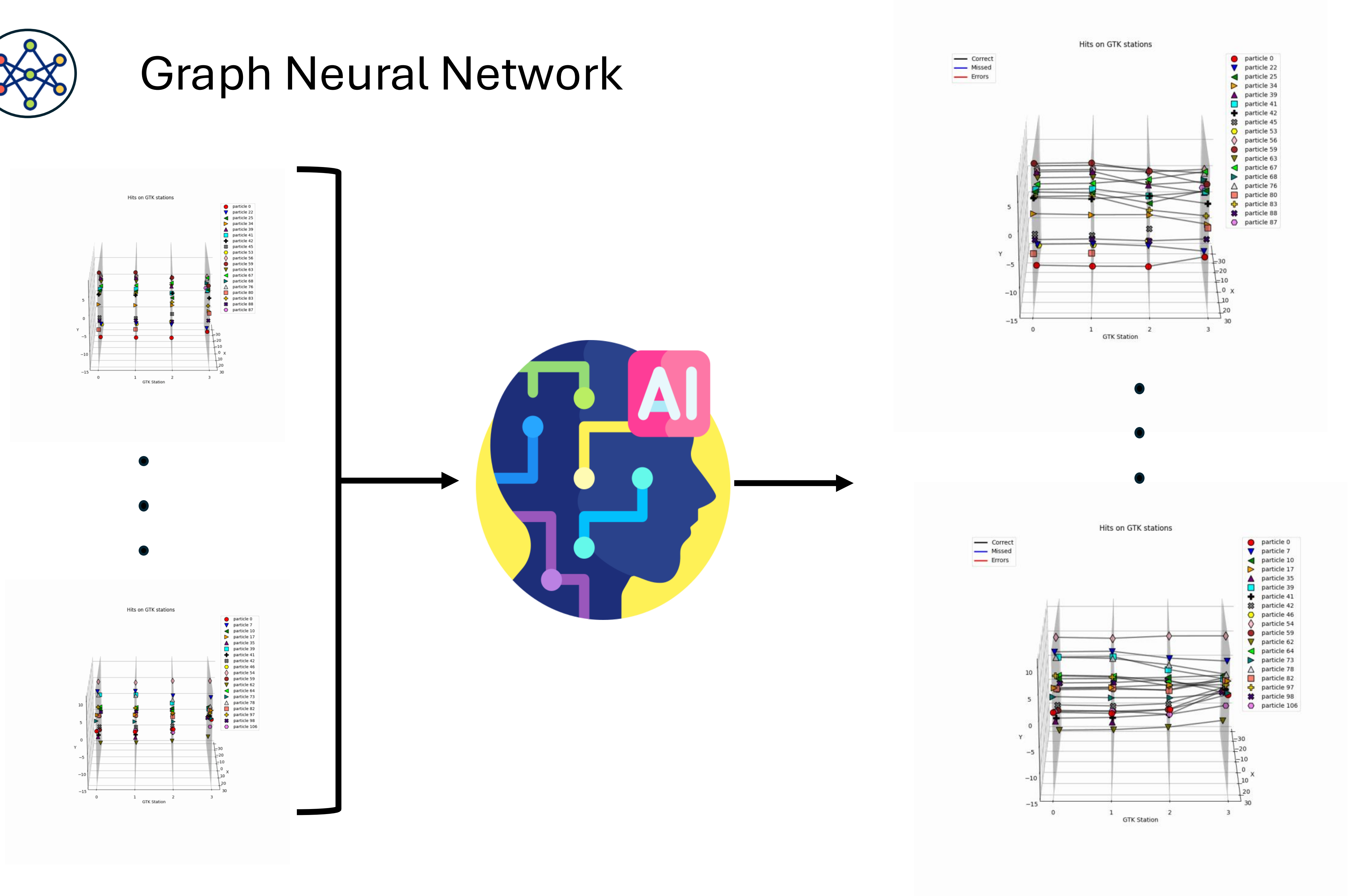
- Solved class imbalance
- Global awareness
- Computationally Efficient
- Flexible Topology

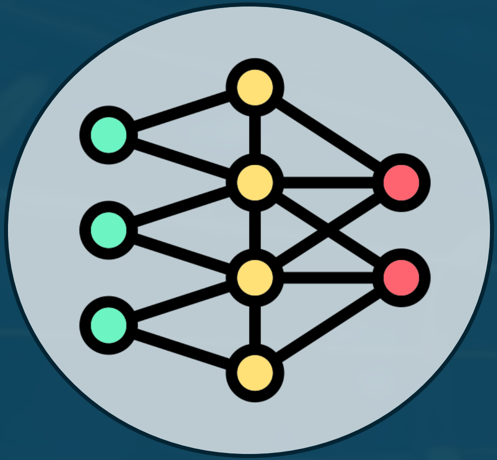
Graph Neural Network





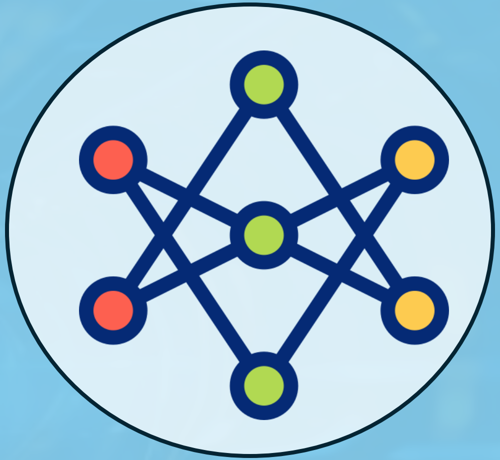
Graph Neural Network





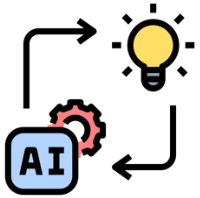
Multi-Layer Perceptron

Transformer

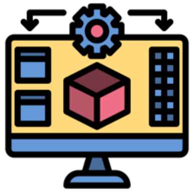


Graph Neural Network

Conclusions



- Machine Learning algorithms can be used to track particles



- 3 different algorithms were proposed (MLP, Transformer, Graph Neural Network)

Acknowledgments

This project was based on Monte Carlo simulations kindly provided by the NA62 Collaboration.

References

[1] B. Denby. Neural networks and cellular automata in experimental high energy physics. Computer Physics Communications, 49(3):429–448, 1988.
[2] Alexander Radovic, Mike Williams, David Rousseau, Michael Kagan, Daniele Bonacorsi, Alexander Himmel, Adam Aurisano, Kazuhiro Terao, and Taritree Wongjirad. Machine learning at the energy and intensity frontiers of particle physics. Nature, 560(7716):41—48, August 2018.
[3] Bourilkov D. Machine and deep learning applications in particle physics. International Journal of Modern Physics A, 34(35):1930019, December 2019.
[4] M. Feickert and B. Nachman. A living review of machine learning for particle physics, 2021.
[5] M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. IEEE Signal Processing Magazine, 34(4):18–42, July 2017.
[6] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. IEEE Transactions on Neural Networks, 20(1):61–80, Jan 2009.
[7] T. Gaudelet and et al. Utilizing graph machine learning within drug discovery and development. Briefings in Bioinformatics, 22(6):bbab159, 05 2021.
[8] F. Liu, S. Xue, J. Wu, C. Zhou, W. Hu, C. Paris, S. Nepal, J. Yang, and P.S. Yu. Deep learning for community detection: Progress, challenges and opportunities. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-PRICAI-2020. International Joint Conferences on Artificial Intelligence Organization, July 2020.
[9] Z. Zhang, P. Cui, and W. Zhu. Deep learning on graphs: A survey, 2020..
[10] 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(1):4–24, January 2021.
[11] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C.Li, and M. Sun. Graph neural networks: A review of methods and applications, 2021.
[12] S. Farrell et al. Novel deep learning methods for track reconstruction, 2018.
[13] X. Ju and et al. Graph neural networks for particle reconstruction in high energy physics detectors, 2020.
[14] I. Henrion, J. Brehmer, J. Bruna, K. Cho, K. Cranmer, G. Louppe, and G. Rochette. Neural message passing for jet physics. 2017.
[15] J. Shlomi, S. Ganguly, E. Gross, K. Cranmer, Y. Lipman, H. Serviansky, H. Maron, and N. Segol. Secondary vertex finding in jets with neural networks. The European Physical Journal C, 81(6), June 2021.
[16] J. A. Martinez, O. Cerri, M. Pierini, M. Spiropulu, and J. Vlimant. Pileup mitigation at the large hadron collider with graph neural networks, 2019.
[17] E. Cortina Gil. et al. The beam and detector of the na62 experiment at cern. Journal of Instrumentation, 12(05):P05025, may 2017.
[18] Rinella et al. The na62 gigatracker: a low mass high intensity beam 4d tracker with 65 ps time resolution on tracks. Journal of Instrumentation, 14(07):P07010–P07010, July 2019.
[19] A. Tsaris and et al. The hep.trkx project: Deep learning for particle tracking. Journal of Physics: Conference Series, 1085(4):042023, sep 2018.
[20] Exa.TrkX Collaboration. Exa.trkx. https://exatrkx.github.io/.
[21] S. Caillou, P. Calafiura, S. Farrell, X. Ju, D. Murnane, C. Rougier, J. Stark, and A. Vallier. ATLAS ITk Track Reconstruction with a GNN-based pipeline. Technical report, CERN, Geneva, 2022.
[22] X. et al. Ju. Performance of a geometric deep learning pipeline for hl-lhc particle tracking. The European Physical Journal C, 81(10), October 2021.
[23] M. Kiehn and et al. The trackml high-energy physics tracking challenge on kaggle. EPJ Web of Conferences, 214:06037, 01 2019.
[24] S. Thais and et al. Graph neural networks in particle physics: Implementations, innovations, and challenges, 2022.
[25] V. Mikuni and F. Canelli. ABCNet: an attention-based method for particle tagging. The European Physical Journal Plus, 135(6):463, 2020.
[26] R. Liu, P. Calafiura, S. Farrell, X. Ju, D. Murnane, and T. Pham. Hierarchical graph neural networks for particle track reconstruction, 2023.
[27] G. DeZoort, S. Thais, J. Duarte, V. Razavimaleki, M. Atkinson, I. Ojalvo, M. Neubauer, and P. Elmer. Charged particle tracking via edge-classifying interaction networks. Computing and Software for Big Science, 5(1), November 2021