Graph Neural Network

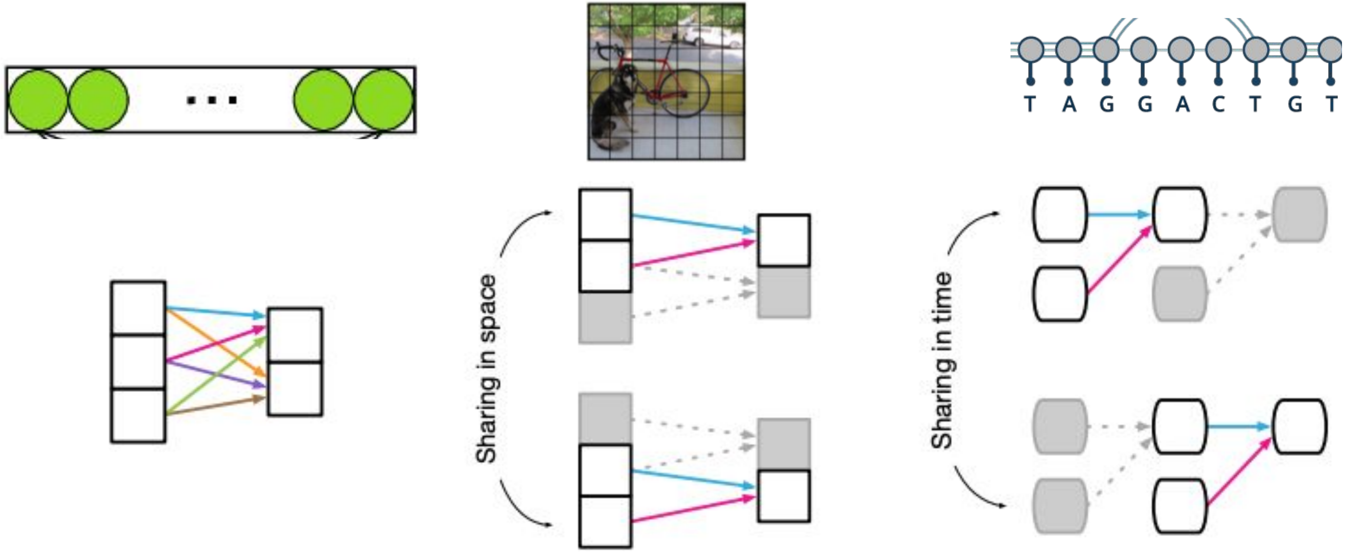
PHYS 591000 Spring 2021

J. Shlomi, P. Battaglia, J.-R. Vlimant, “Graph Neural Network in Particle Physics”

https://arxiv.org/abs/2007.13681

Common Deep Learning blocks

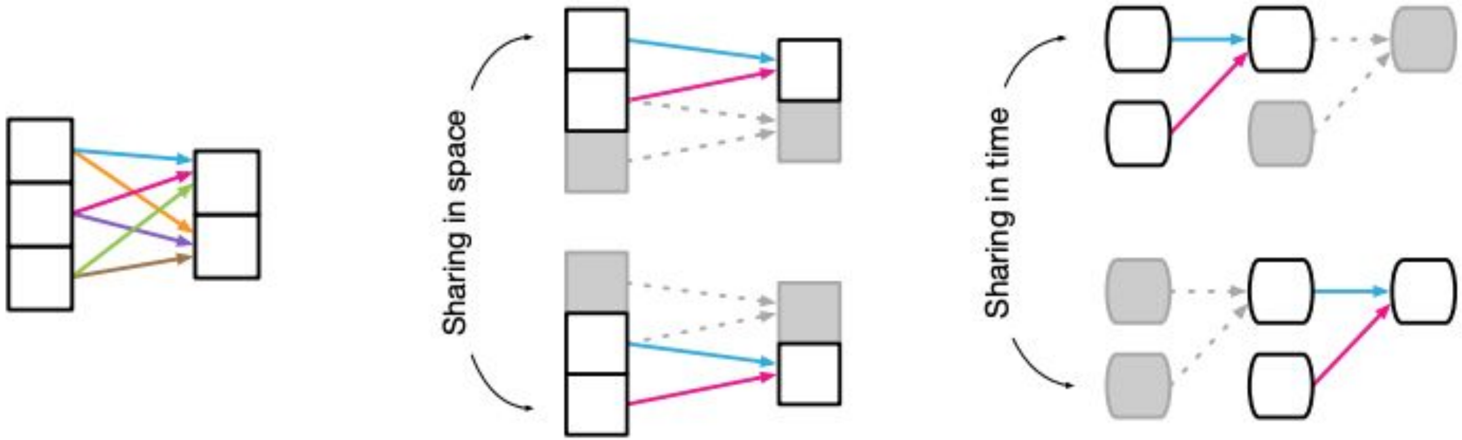
https://arxiv.org/pdf/1806.01261



(a) Fully connected (b) Convolutional (c) Recurrent

Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation

Common Deep Learning blocks



(a) Fully connected

(b) Convolutional

(c) Recurrent

all weights are independent, and there is no sharing.

a local kernel function is reused multiple times across the input. Shared weights are indicated by arrows with the same color.

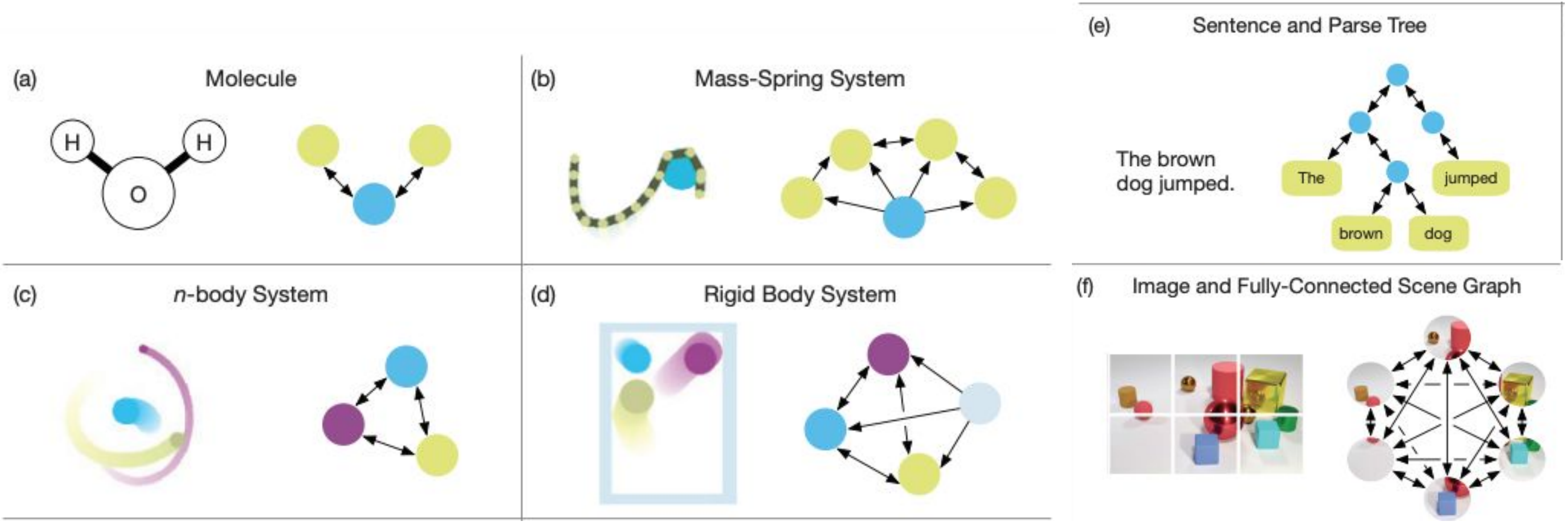
the same function is reused across different processing steps.

Graph blocks

https://arxiv.org/pdf/1901.00596

Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

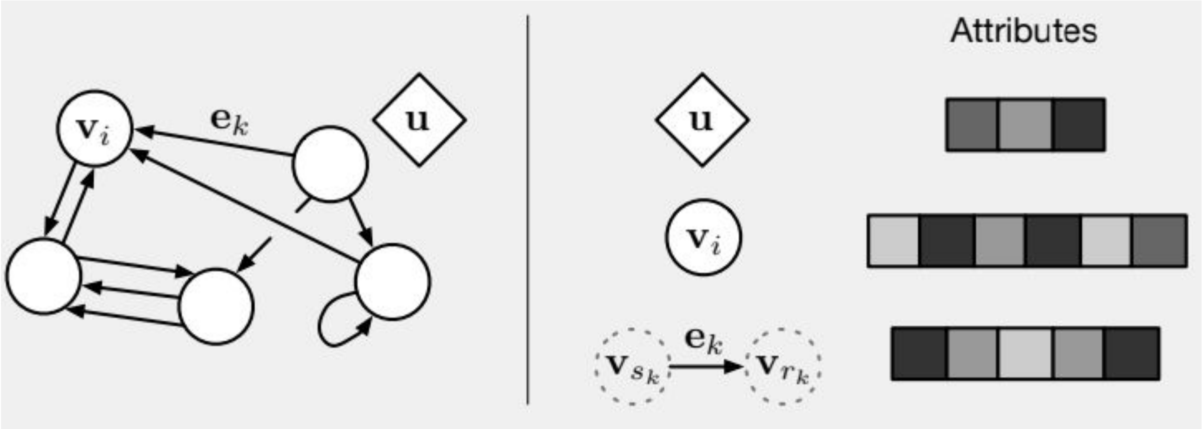
Different Graph Representation



Graph Network Formalism

https://arxiv.org/pdf/1806.01261

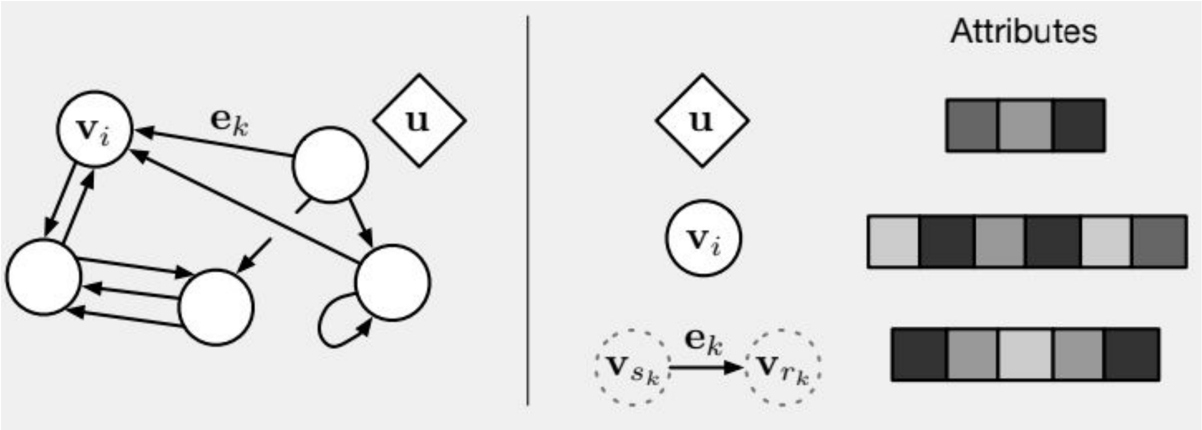
- A Graph can be represented by G = (u, V, E)



- u: graph-level attributes
- $V = \{v_i\}_{i=1:Nv}$: the set of nodes (or vertices)
    - Nv: number of vertice
- $E = \{e_k, r_k, s_k\}_{k=1:Ne}$: the set of edges
    - Ne: number of edges
    - e_k: k-th edge attributes
    - r_k: receiver indices of the two nodes connected by the k-th edge
    - s_k:sender index of the two nodes connected by the k-th edge

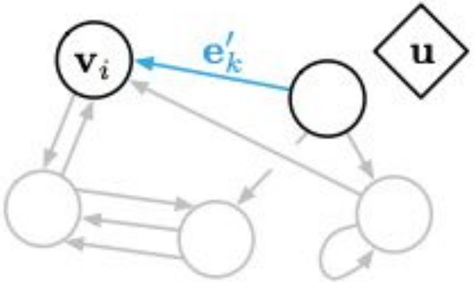
Graph Network Formalism

https://arxiv.org/pdf/1806.01261

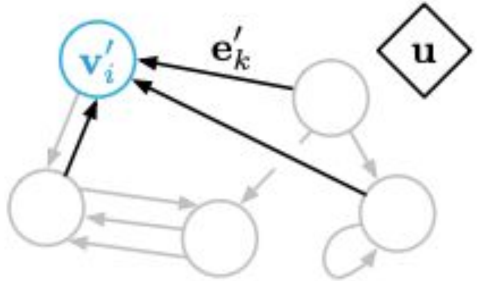


- Directed : one-way edges, from a “sender” node to a “receiver” node
- Attribute : properties that can be encoded as a vector, set, or even another graph.
- Multi-graph : there can be more than one edge between vertices, including self-edges.

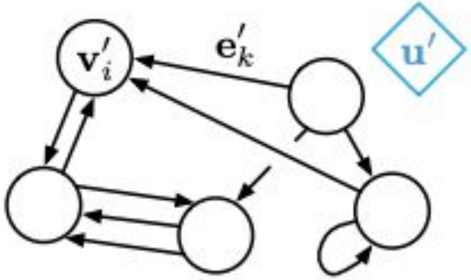
Graph Networks Formalism



Edge block computes one output for each edge, and aggregates them by their corresponding receiving node



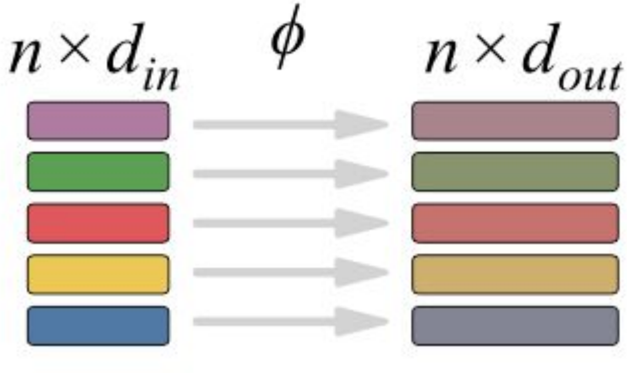
Vertex block computes one output for each node

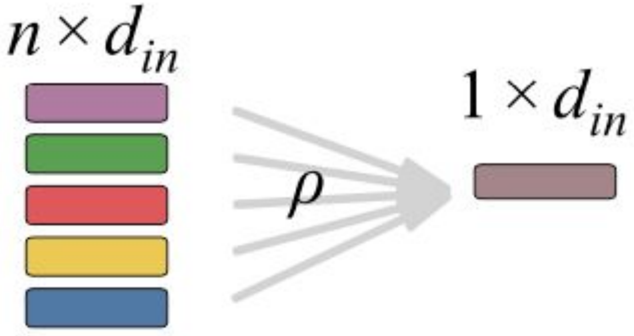


Global block aggregates the edge- and node-level outputs

- Blue indicates the element that is being updated
- Black indicates other elements which are involved in the update

Graph Network Blocks





Update Function

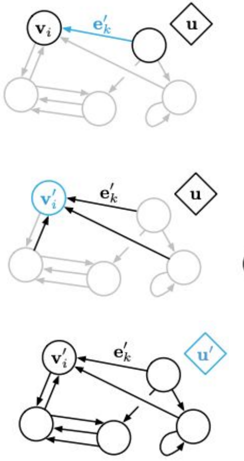
takes a set of objects with a fixed size representation, and apply the same function to each of the elements in the set, resulting in an updated representation (also with a fixed size).

Aggregation function

takes a set of objects and create one fixed size representation for the entire set, by using some order invariant function to group together the representations of the objects (such as an element-wise sum).

Graph Networks Formalism

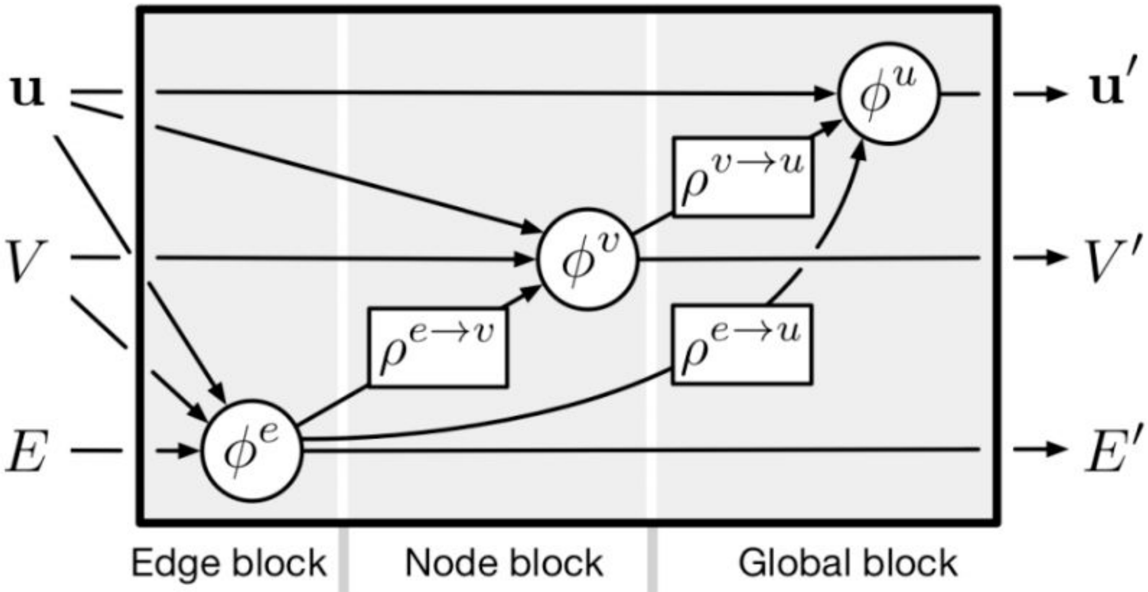
$$
\begin{aligned}
\mathbf{e}'_k &= \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right) & \bar{\mathbf{e}}'_i &= \rho^{e\to v}\left(E'_i\right) & &\triangleright \text{ Edge block} \\
\mathbf{v}'_i &= \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right) & \bar{\mathbf{e}}' &= \rho^{e\to u}\left(E'\right) & &\triangleright \text{ Vertex block} \\
\mathbf{u}' &= \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right) & \bar{\mathbf{v}}' &= \rho^{v\to u}\left(V'\right) & &\triangleright \text{ Global block}
\end{aligned}
$$



- Updated functions: $(\phi^e,\ \phi^v,\ \text{and}\ \phi^u)$
- Aggregation functions: $(\rho^{e\to v},\ \rho^{e\to u},\ \text{and}\ \rho^{v\to u})$

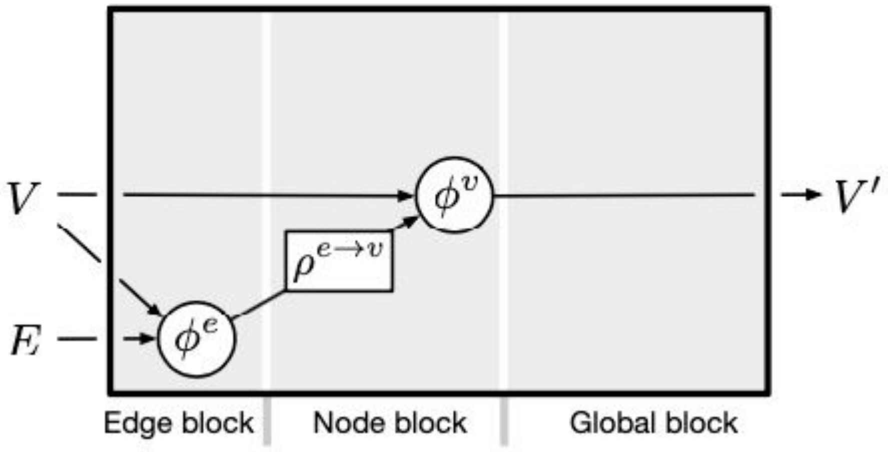
Fully Graph Networks

- In practice, the update functions are often implemented as simple trainable NNs;
- Aggregation functions are typically implemented as permutation invariant reduction operators, such as element-wise sums, means, or maximums;

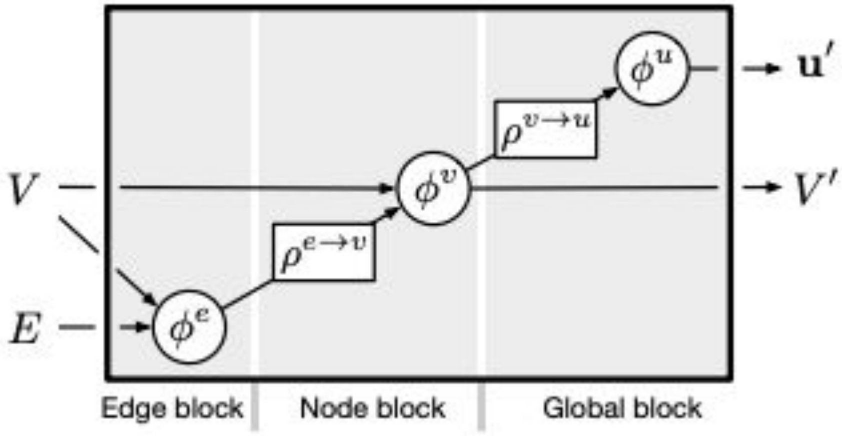


Graph Neural Network architecture

Graph Convolutional Network



Message Passing Graph Network

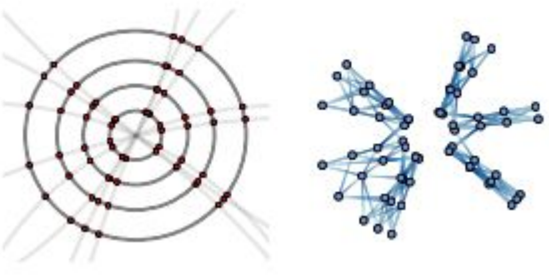


Applications of GNN

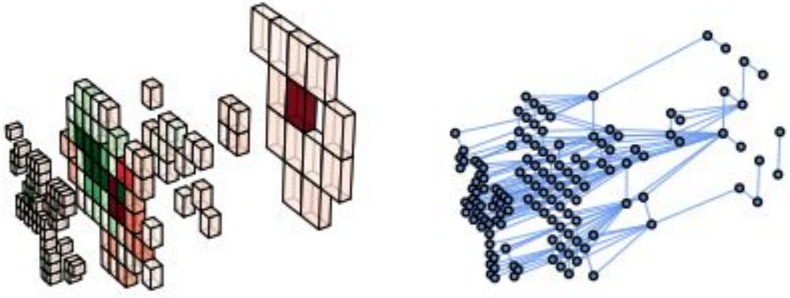
- Decide how the data could be expressed as a graph:
    - What are the entities and relations that will be represented as nodes and edges?
    - What is the required output, i.e., edge-, node-, or graph-level predictions?
- Choices about the specific GNN architecture:
    - Is a global output network required to produce graph-level outputs?
    - Should pairwise interactions among nodes be computed, or more GCN-like summation and non-linear transformation?
    - How many message-passing steps should be used to propagate information among distant nodes in the graph?

Graph Presentation in Particle Physics

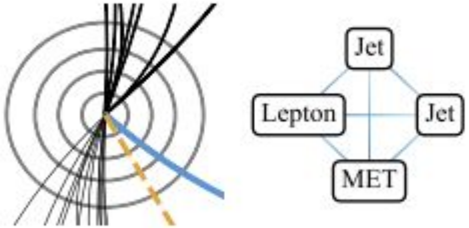
- Using Graph data structure to represent low level feature of data



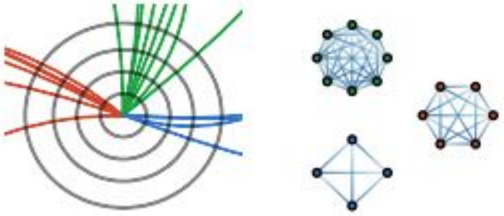
Tracking



Clustering



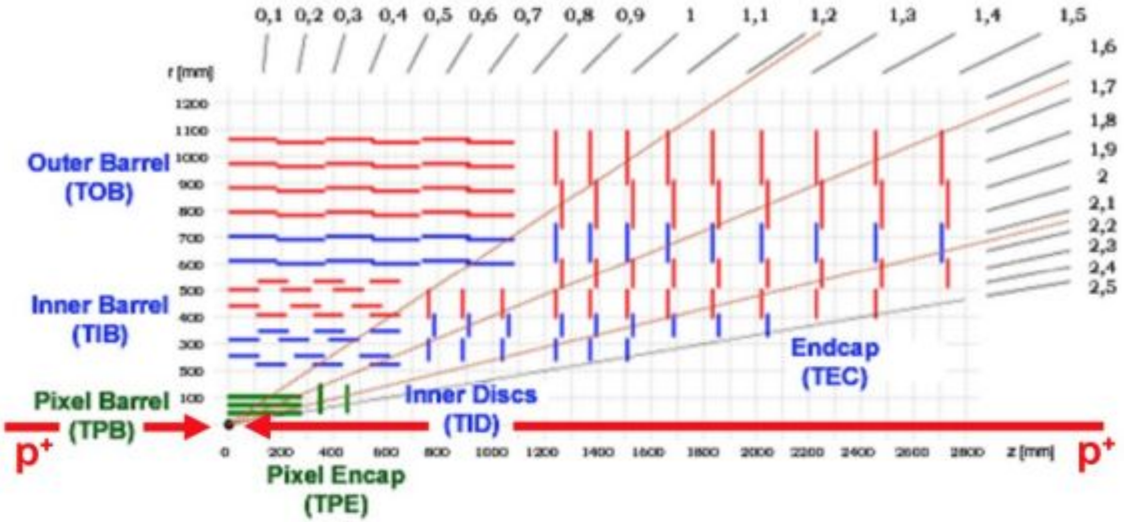
Physics Events

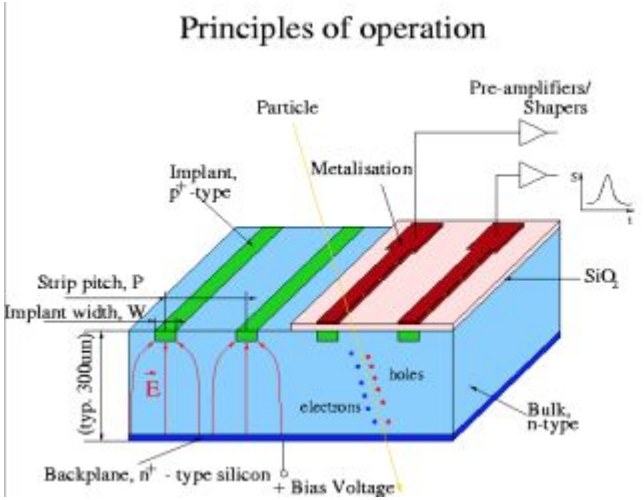


Jet

Jean-Roch Vlimant
https://indico.cern.ch/event/768915/contributions/3474674/

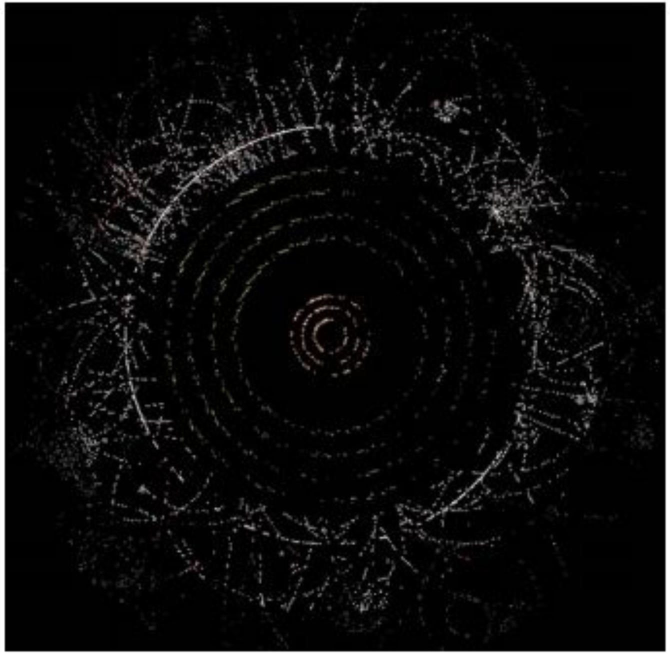
Tracker 101



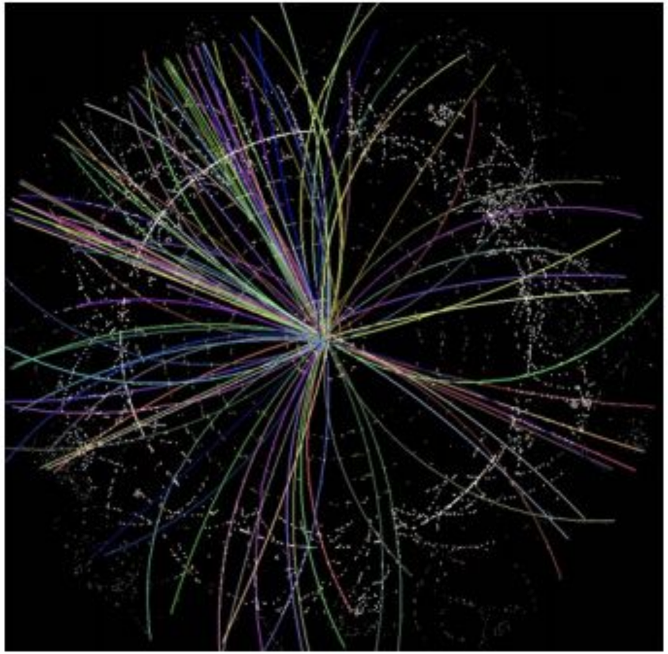


- Ionizing particle leaves a signal in silicon detector

Name of the game

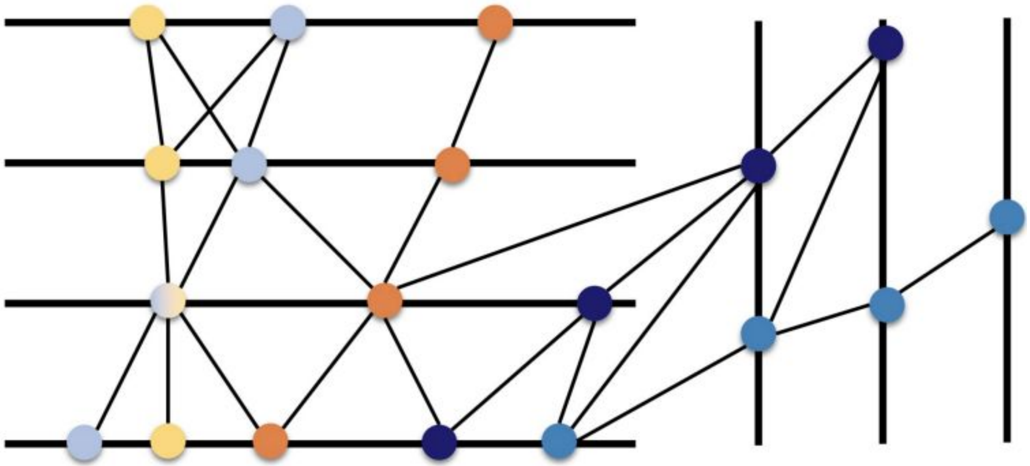


From hits ...



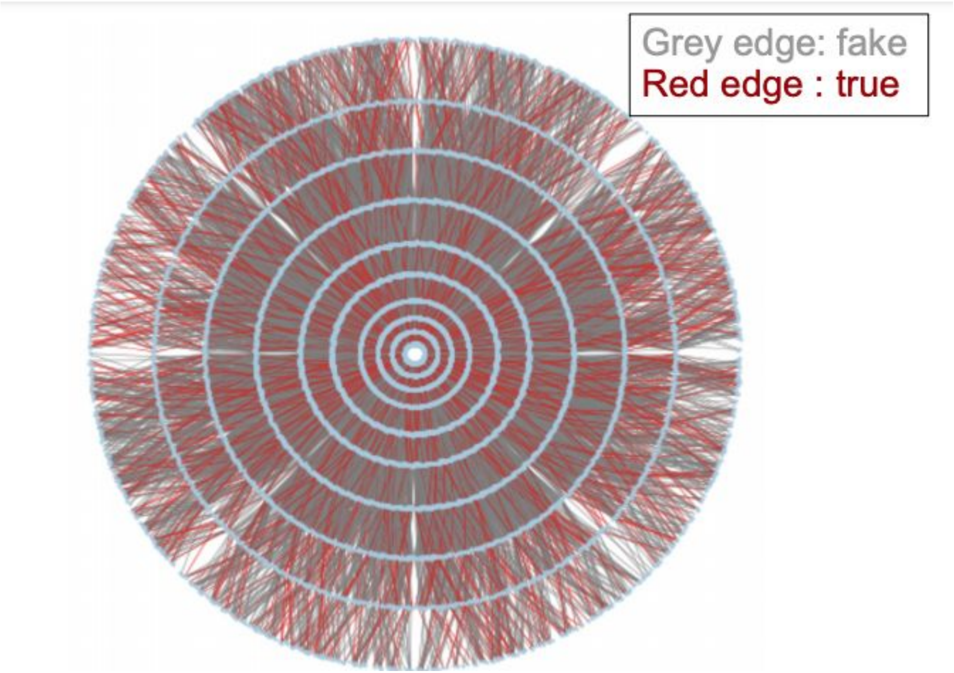
... to trajectory & parameters

Tracker Hit Graph

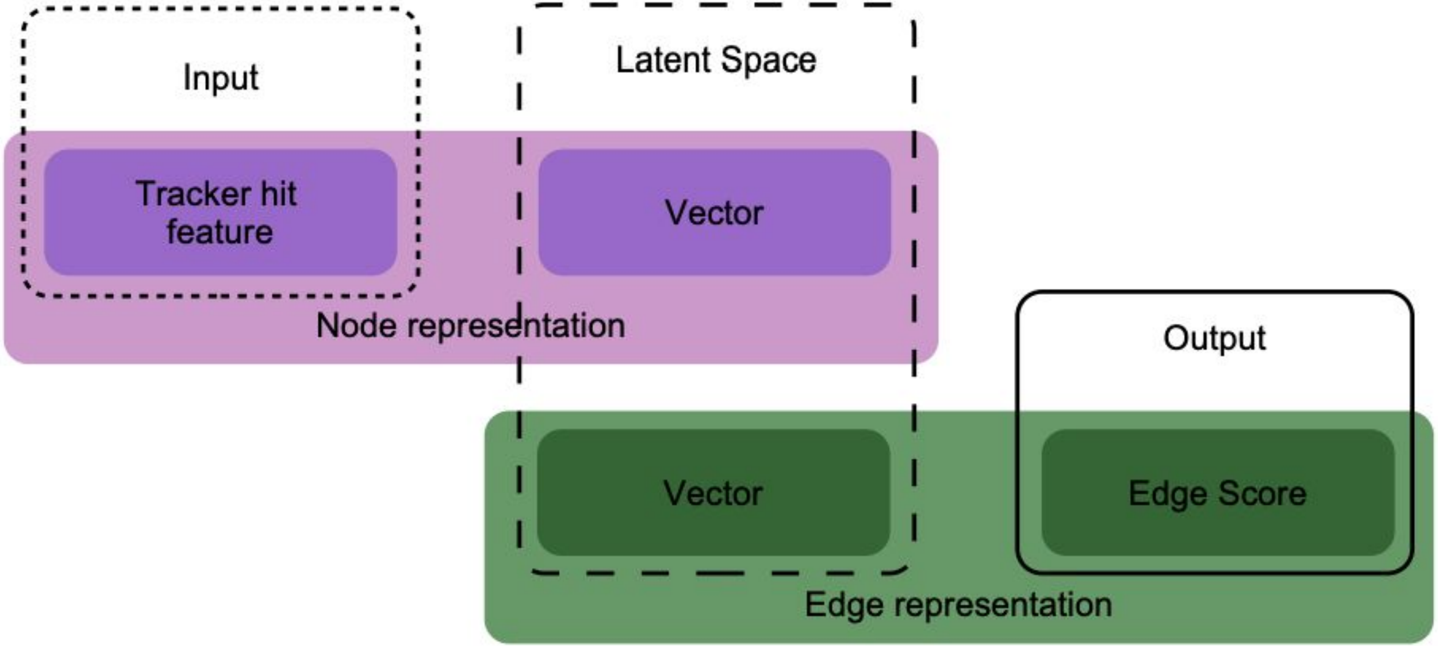


Graph construction
➢ One tracker hit ≡ one node
➢ Sparse edges constructed from geometrical consideration
➔ Edge classification ≡ reconstructing the trajectory of particles

Edge Classification

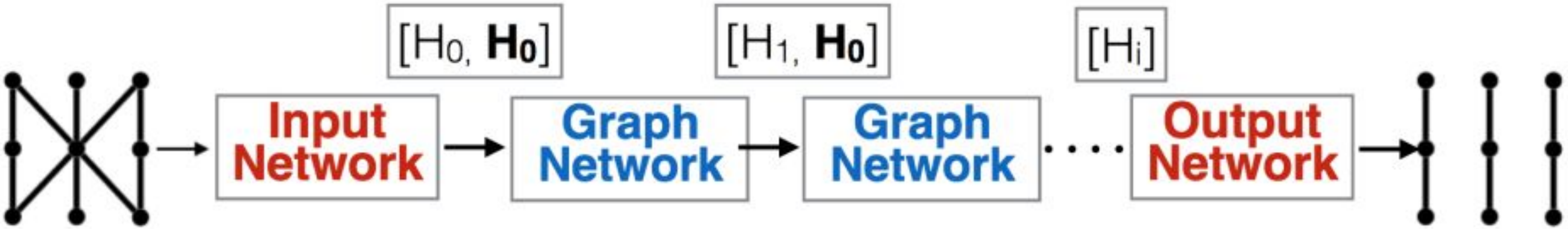


Node & Edge representation



Edge representation is not the edge score.
Final edge score extracted from the latent edge representation.

Message Passing Model



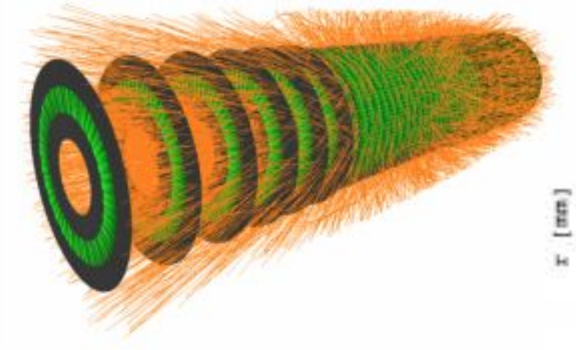
- Graph is sparsely connected from consecutive layers
- Edge representation computed from features at the ends
- Node representation computed from the sum over all connected edges
- Correlates hits information through multiple (8) iterations of (Graph Network)

Charged Particle Tracking

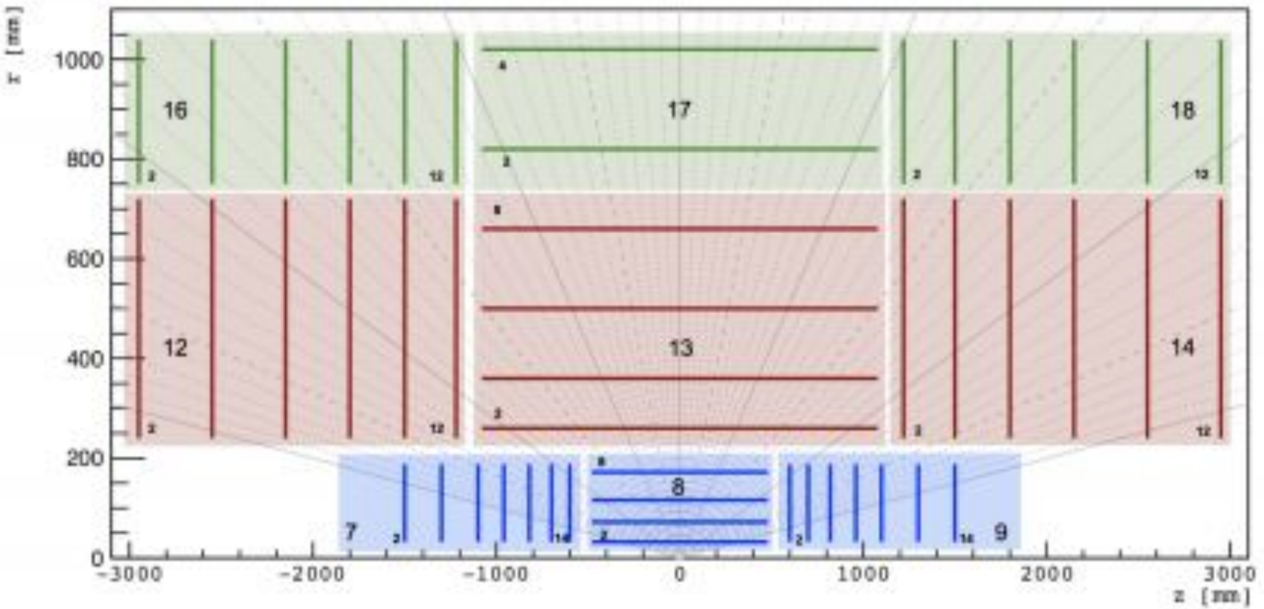
arXiv:1810.06111

- Edge Classification
  - Each node represents one hit, with edge constructed between pairs of hits with geometrically plausible relations.
  - Multiple updates of the node representation and edge weight over the graph (using the edge weight as attention)
  - Model learns what are the edges truly connecting hits belonging to the same track.
  - transforms the clustering problem into an edge classification

TrackML Detector Geometry



https://arxiv.org/pdf/2103.06995.pdf

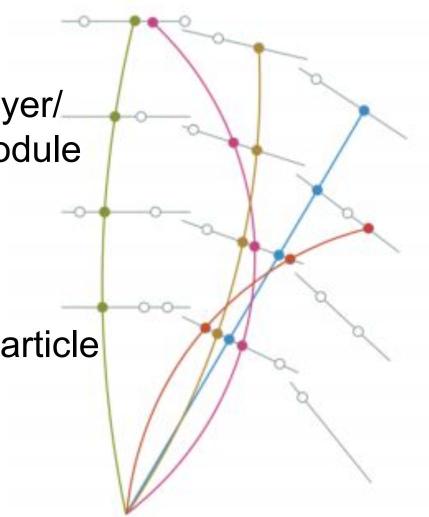


TrackML Particle Tracking Challenge

Hit

Layer/
Module

Particle



	hit_id	x	y	z	volume_id	layer_id	module_id
16873	16874	-32.554401	-3.648710	-469.864990	8	2	1
16874	16875	-33.153702	-1.934740	-423.516998	8	2	1
16875	16876	-26.362400	-18.423700	-461.375000	8	2	2
16876	16877	-12.034100	-29.649799	-464.428009	8	2	3
16877	16878	-15.127200	-28.159300	-439.713989	8	2	3

	particle_id	vx	vy	vz	px	py	pz	q	nhits
0	4503668346847232	-0.009288	0.009861	-0.077879	-0.055269	0.323272	-0.203492	-1	8
1	4503737066323968	-0.009288	0.009861	-0.077879	-0.948125	0.470892	2.010060	1	11
2	4503805785800704	-0.009288	0.009861	-0.077879	-0.886484	0.105749	0.683881	-1	0
3	4503874505277440	-0.009288	0.009861	-0.077879	0.257539	-0.676718	0.991616	1	12
4	4503943224754176	-0.009288	0.009861	-0.077879	16.439400	-15.548900	-39.824902	1	3

https://arxiv.org/pdf/2012.01249.pdf

Reference

- J. Shlomi, P. Battaglia, J.-R. Vlimant, “Graph Neural Network in Particle Physics” https://arxiv.org/abs/2007.13681
- Battaglia, et. al. “Relational inductive biases, deep learning, and graph networks” https://arxiv.org/pdf/1806.01261.pdf