

Deep Learning

Graph Neural Networks

Fundamentals and Software Tools

Dr. Mohammed Salah Al-Radhi

(slides by: Dániel Unyi)



Copyright

Copyright © **Mohammed Salah Al-Radhi**, All Rights Reserved.

This presentation and its contents are protected by copyright law. The intellectual property contained herein, including but not limited to text, images, graphics, and design elements, are the exclusive property of the copyright holder identified above. Any unauthorized use, reproduction, distribution, or modification of this presentation or its contents is strictly prohibited without prior written consent from the copyright holder.

No Recordings or Reproductions: Attendees, viewers, and recipients of this presentation are expressly prohibited from making any audio, video, or photographic recordings, as well as screen captures, screenshots, or any form of reproduction, of this presentation, its content, or any related materials, whether during its live presentation or subsequent access. Violation of this prohibition may result in legal action.

For permissions, inquiries, or licensing requests, please contact: **malradhi@tmit.bme.hu**

Unauthorized use, distribution, or reproduction of this presentation may result in civil and criminal penalties. Thank you for respecting the intellectual property rights of the copyright holder.

Outline

1. Real-world networks and GNN applications
2. Representation of graph data
3. Message passing, pooling and mini-batching
4. Self-supervised learning
5. Further applications

A network graph with nodes and edges, overlaid with a semi-transparent white box containing the title text.

Real-world networks and GNN applications

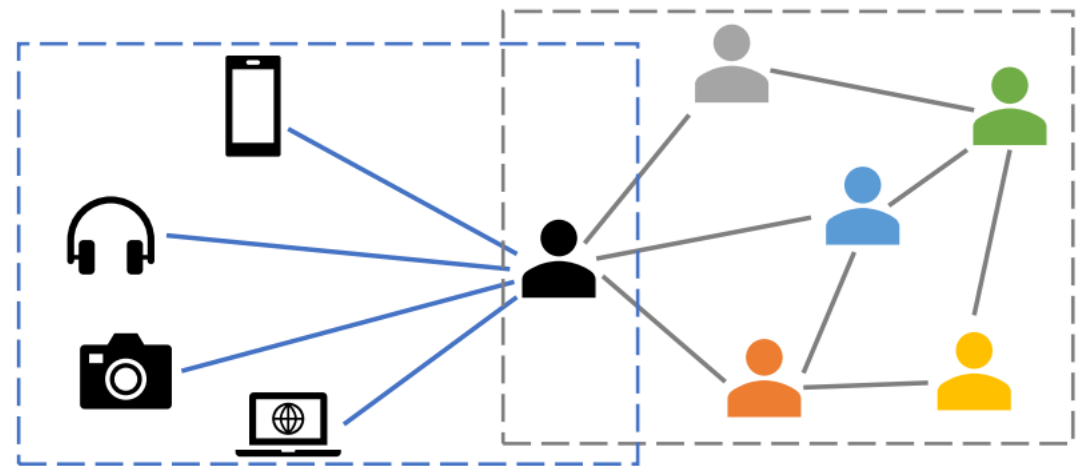
Real-world networks and GNN applications

- Graphs are very generic: objects and relations
- Representing complex systems: engineering, biological, social



social networks

image source: [Medium](#)



recommender systems

image source: [arxiv](#)

Real-world networks and GNN applications

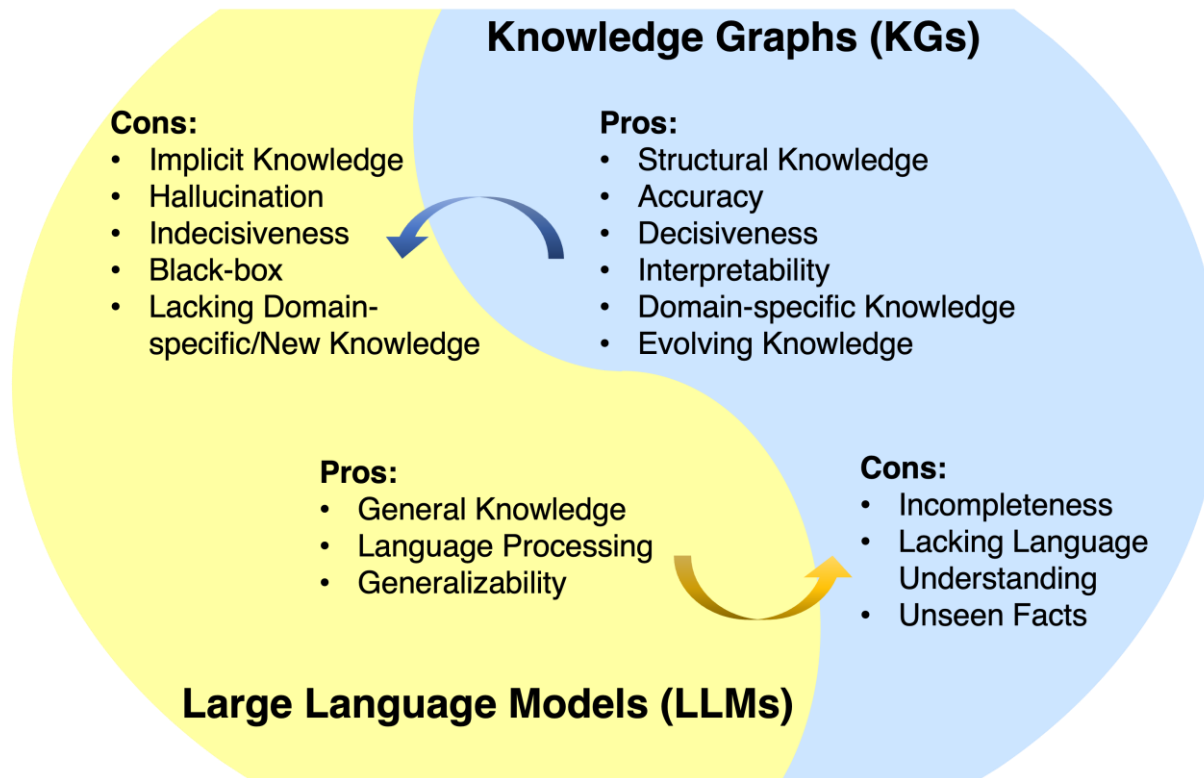
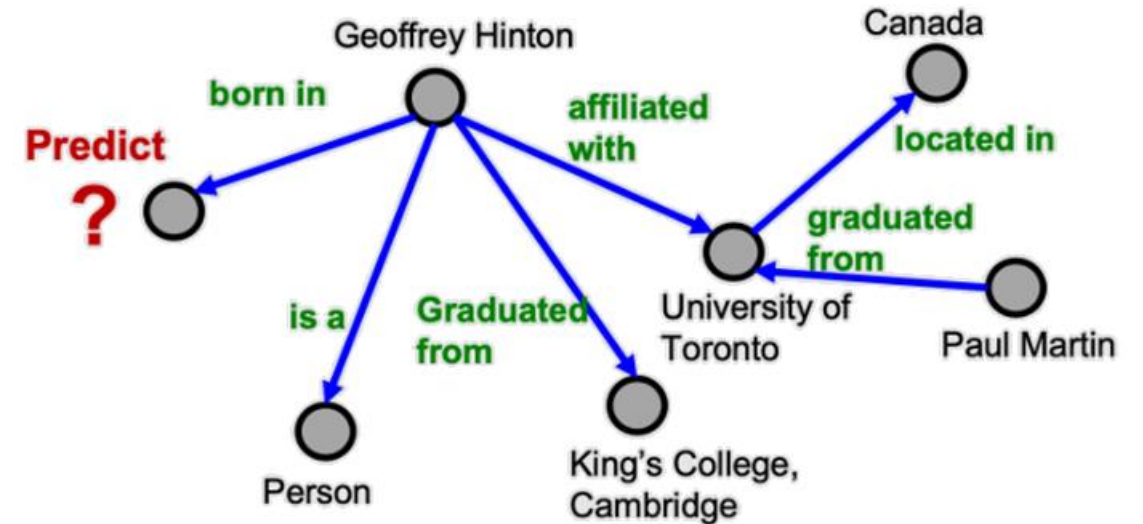


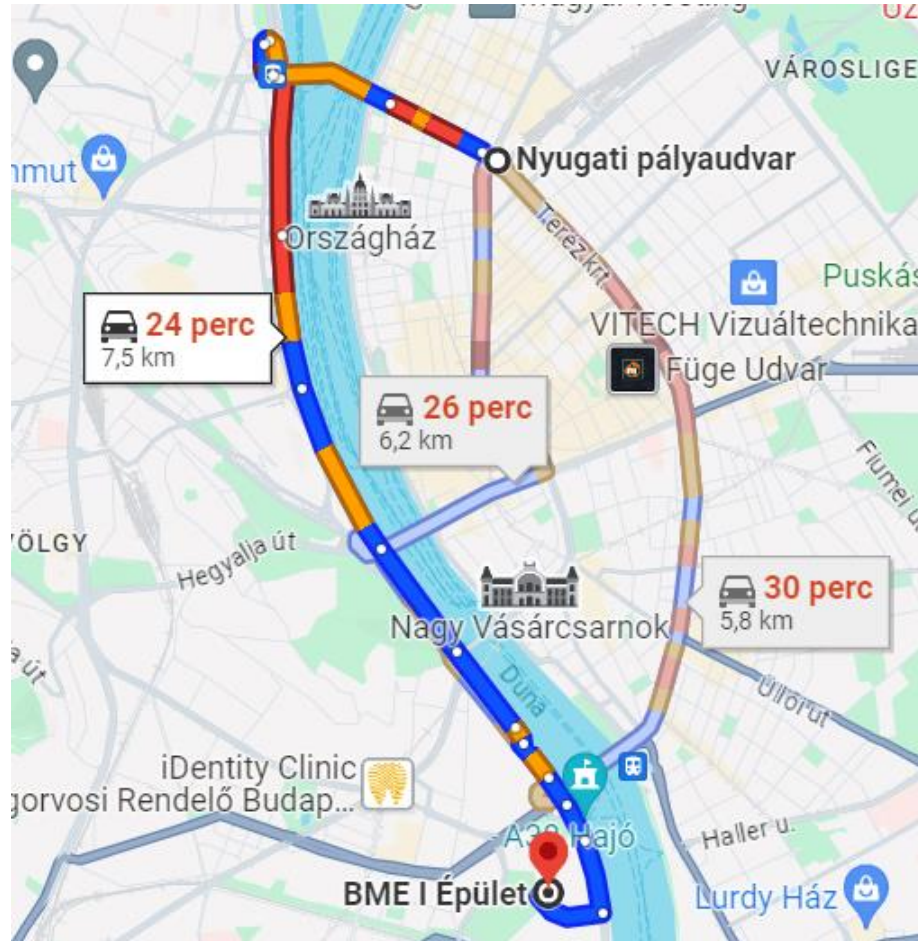
image source: [arxiv](https://arxiv.org/)



knowledge graphs

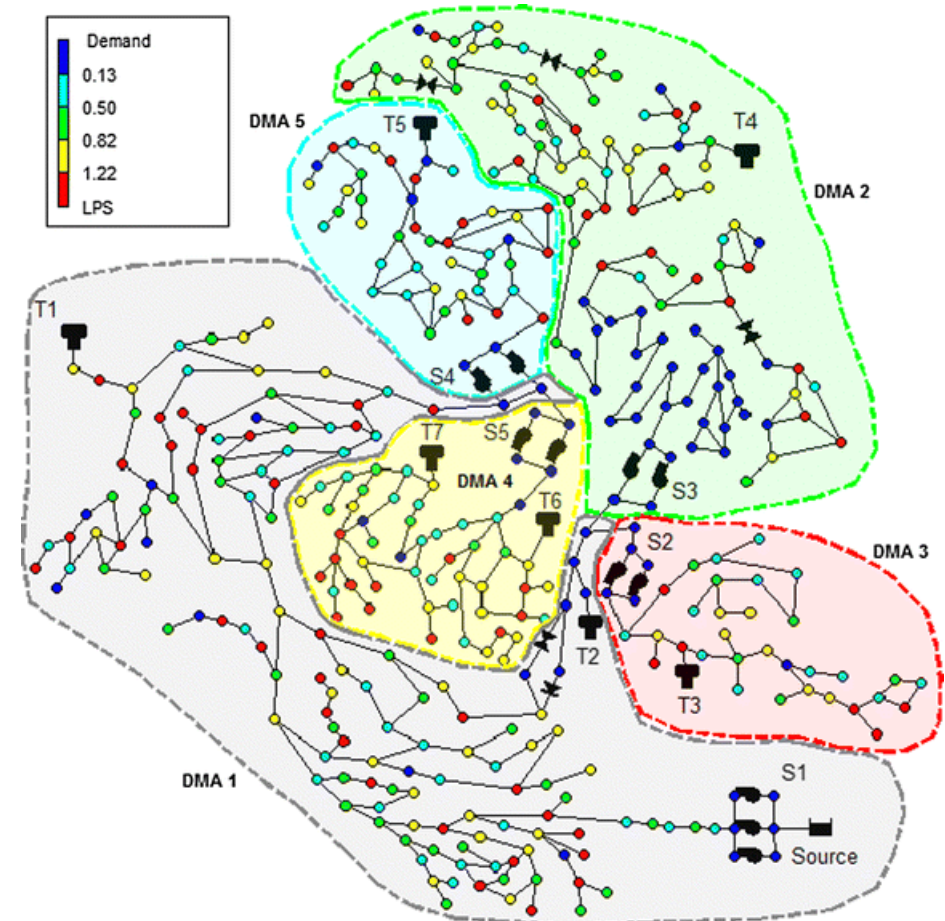
image source: ogb.stanford.edu

Real-world networks and GNN applications



Estimated time of arrival
(google maps)

<https://arxiv.org/abs/2108.11482>

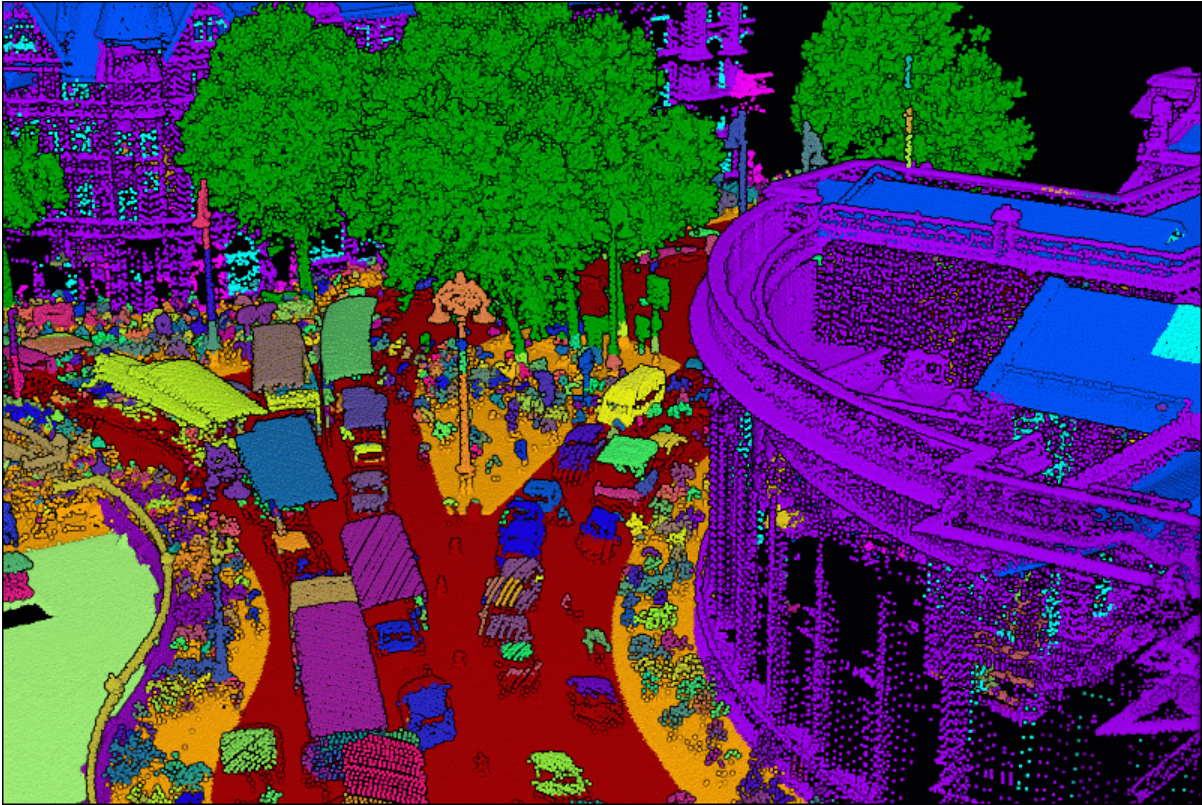


Water distribution systems

image source: [springer](#)

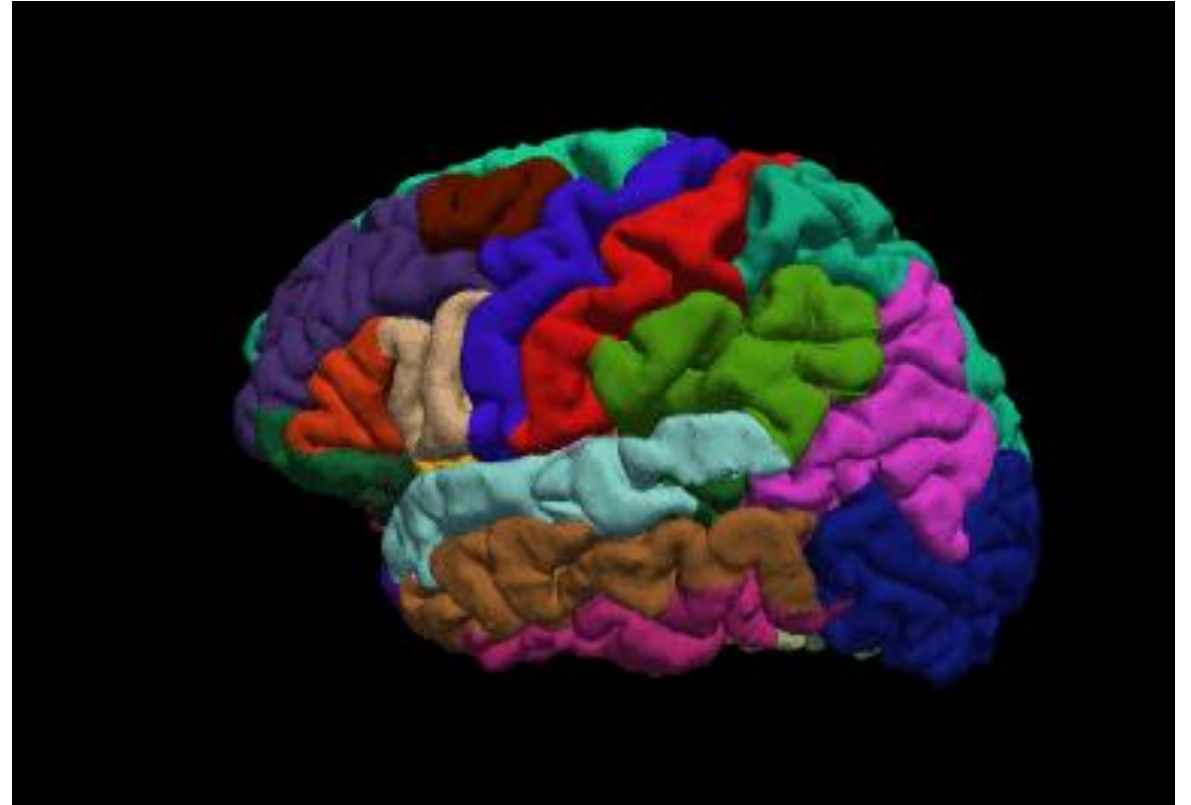
<https://arxiv.org/abs/2104.13619v2>

Real-world networks and GNN applications



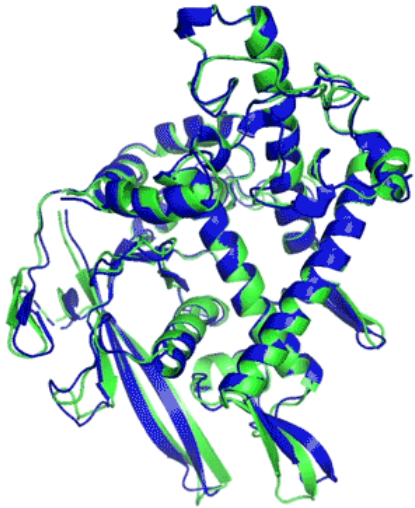
Semantic segmentation

image source: [medium](#)

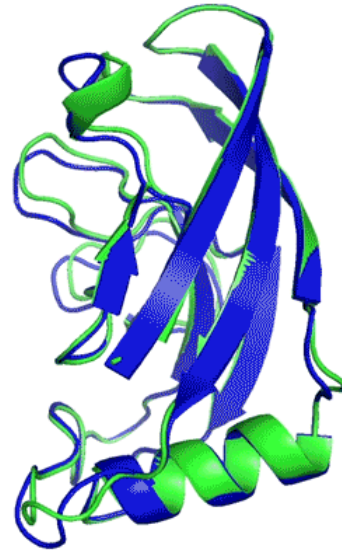


Semantic segmentation
in the medical domain

Real-world networks and GNN applications



T1037 / 6vvr4
90.7 GDT
(RNA polymerase domain)



T1049 / 6y4f
93.3 GDT
(adhesin tip)

- Experimental result
- Computational prediction

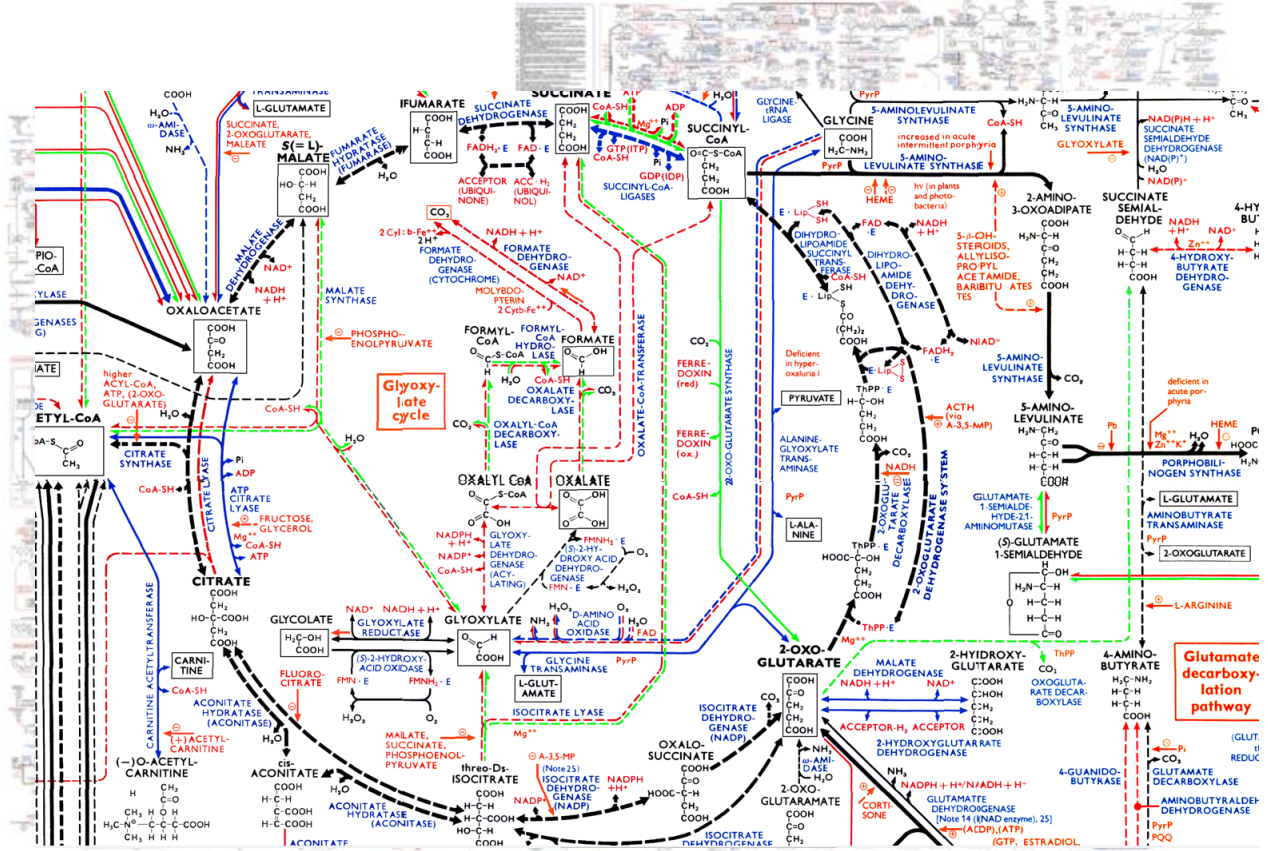
Alphafold 2.0 by DeepMind:
predicts a protein's 3D structure
from its amino acid sequence

“This will be one of the most
important datasets since the
mapping of the Human Genome.”

Professor Ewan Birney

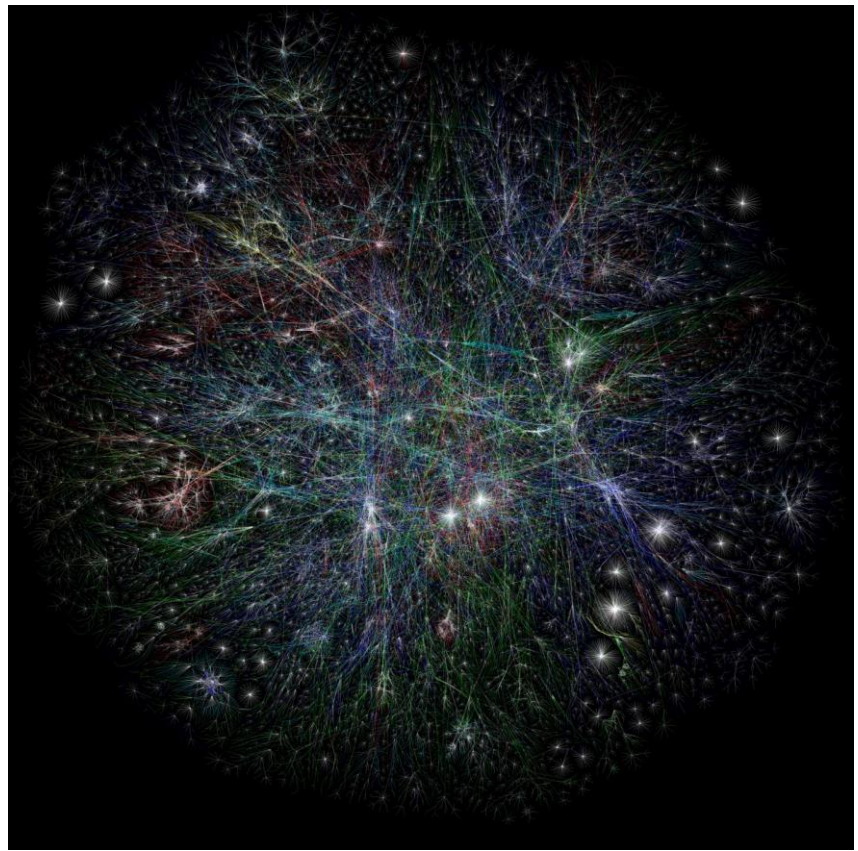
<https://alphafold.ebi.ac.uk/>

Real-world networks and GNN applications



Biochemical pathways in humans

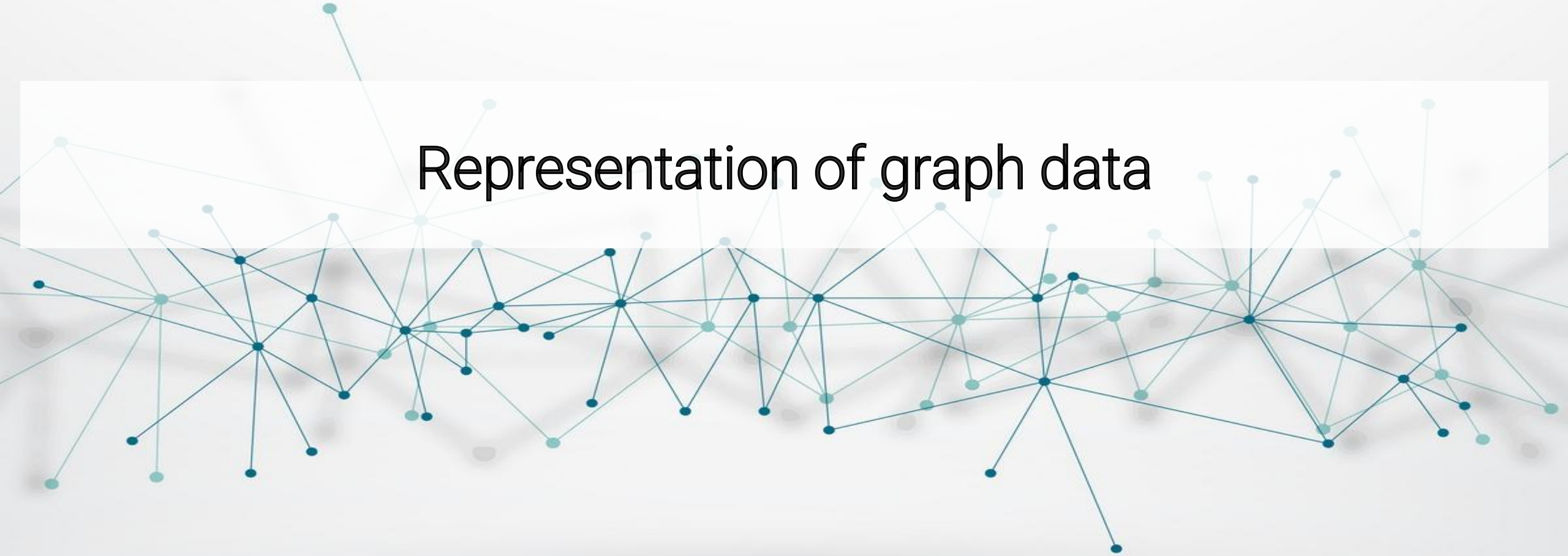
image source: [metabolic_pathways](https://www.metabolic-pathways.com/)



The Internet visualized

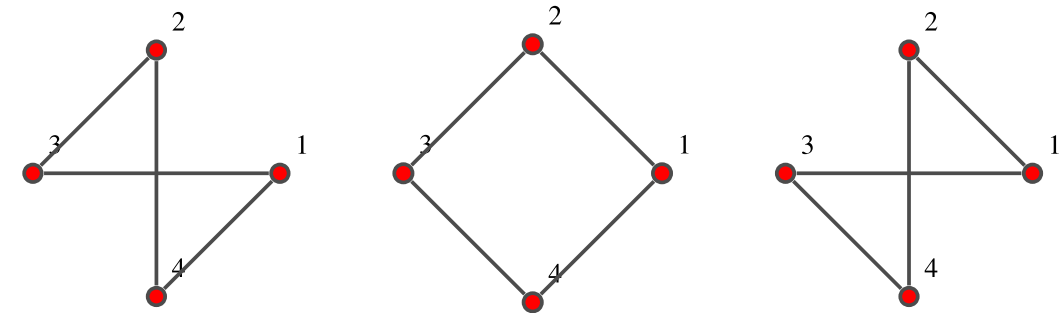
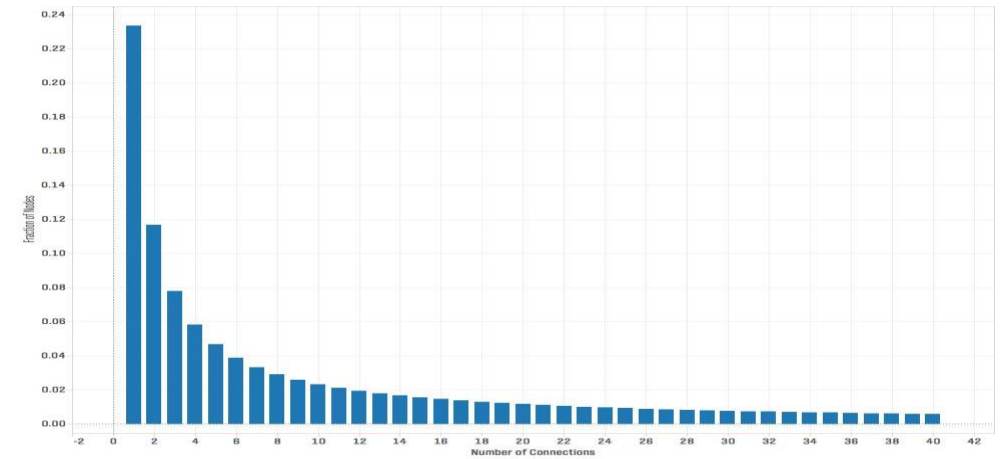
image source: [youtube](https://www.youtube.com/watch?v=...)

Representation of graph data



How to store a graph?

- Adjacency matrix: $O(N^2)$ memory
- Most real-world networks are sparse
- Sparse matrix format: $O(E)$ memory
 - supported by [SciPy](#) & [PyTorch](#)
 - edge indices $2 \times E$
 - edge values $1 \times E$
- Undirected graphs
 - often desirable
 - symmetric adjacency matrix



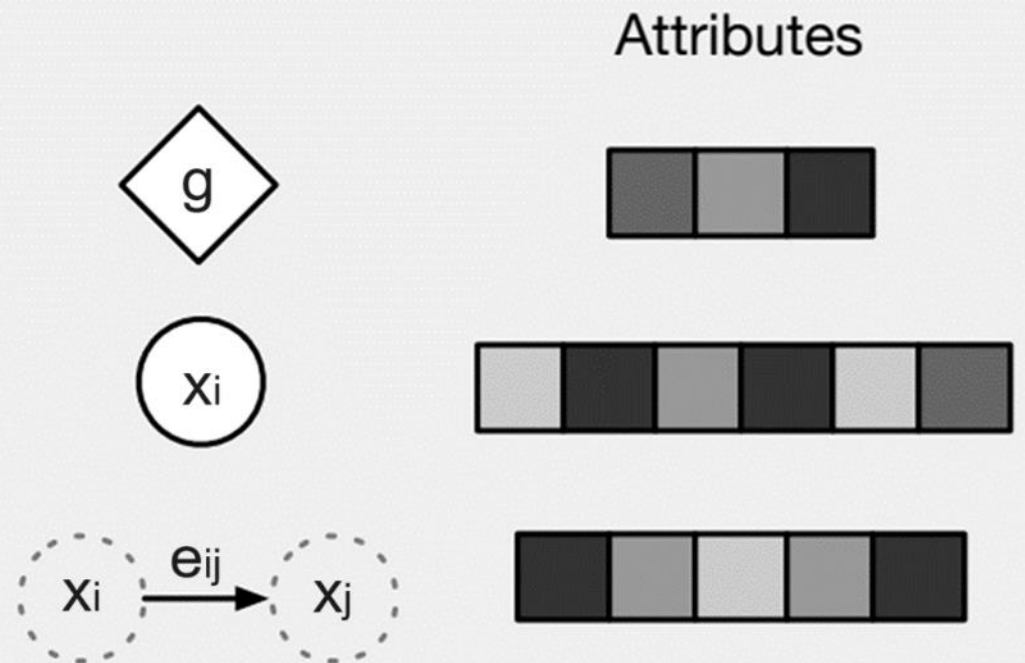
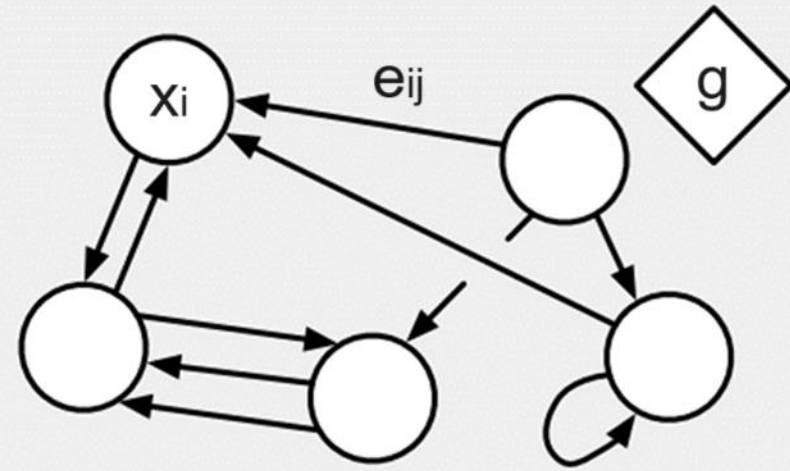
$$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

How to store a graph?

- \mathbf{x}_i node-level features
- \mathbf{e}_{ij} edge-level features
- \mathbf{g} graph-level features
- labels and predictions:
 - predict an unknown property of nodes
 - predict an unknown property of edges
 - predict an unknown property of the whole graph



How to store a graph?

- Nodes:
 - drug
 - protein
- Edges:
 - drug-drug (side effect)
 - drug-protein (interaction)
 - protein-protein (interaction)
- Node level features
- Application:
 - [drug repurposing for COVID-19](https://academic.oup.com/bioinformatics/article/34/13/i457/5045770)

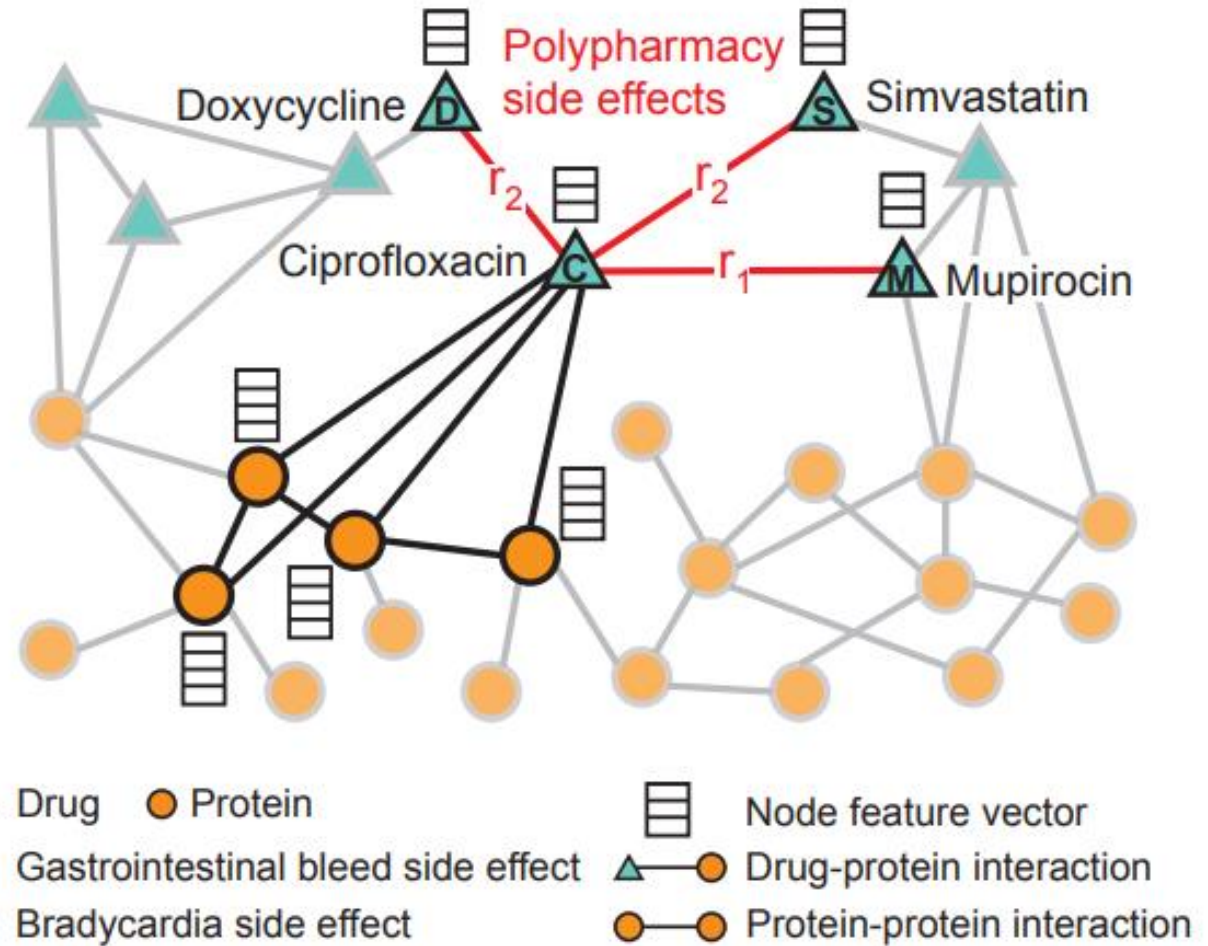
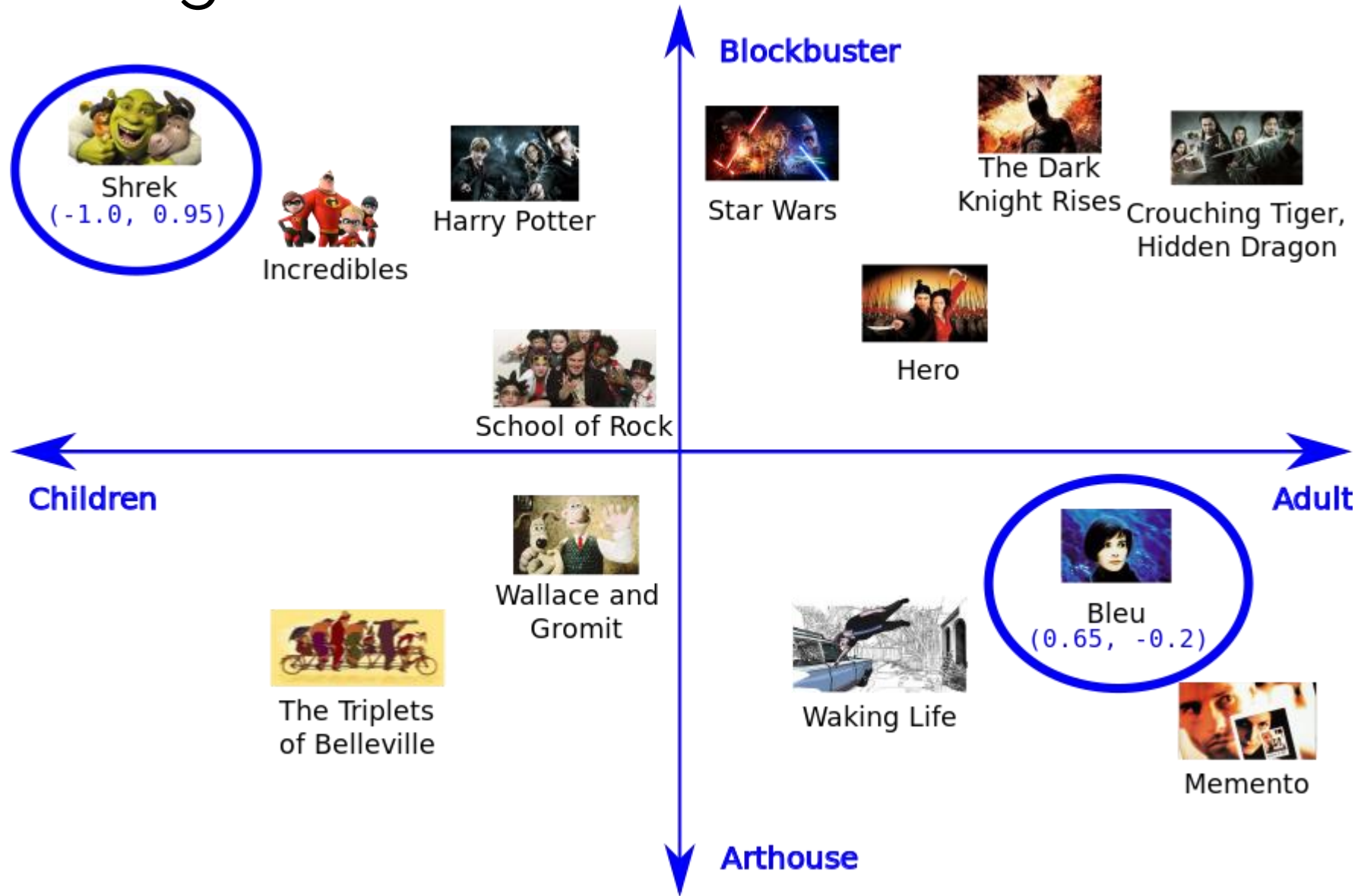


Figure from: <https://academic.oup.com/bioinformatics/article/34/13/i457/5045770>



Message passing and mini-batching

Embeddings



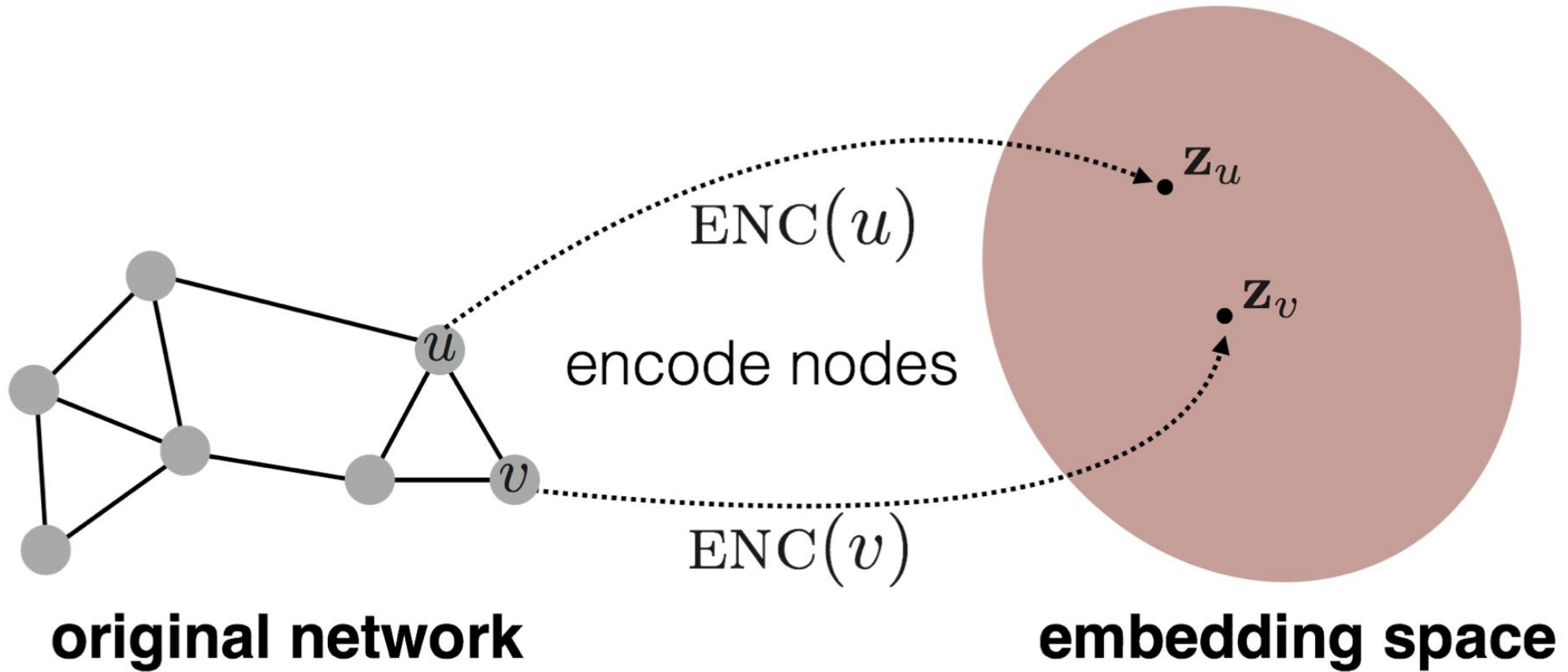
Embeddings

- Graph neural networks produce node embeddings
- Similar nodes should have similar embeddings (according to a distance metric)
- A and B are positionally close, while A and C are structurally close:



- Edge embeddings: from node embeddings e.g. pairwise averaging
- Graph embeddings: from node embeddings e.g. global averaging
 - Permutation invariant operators

Nodes in embedding space



Embeddings

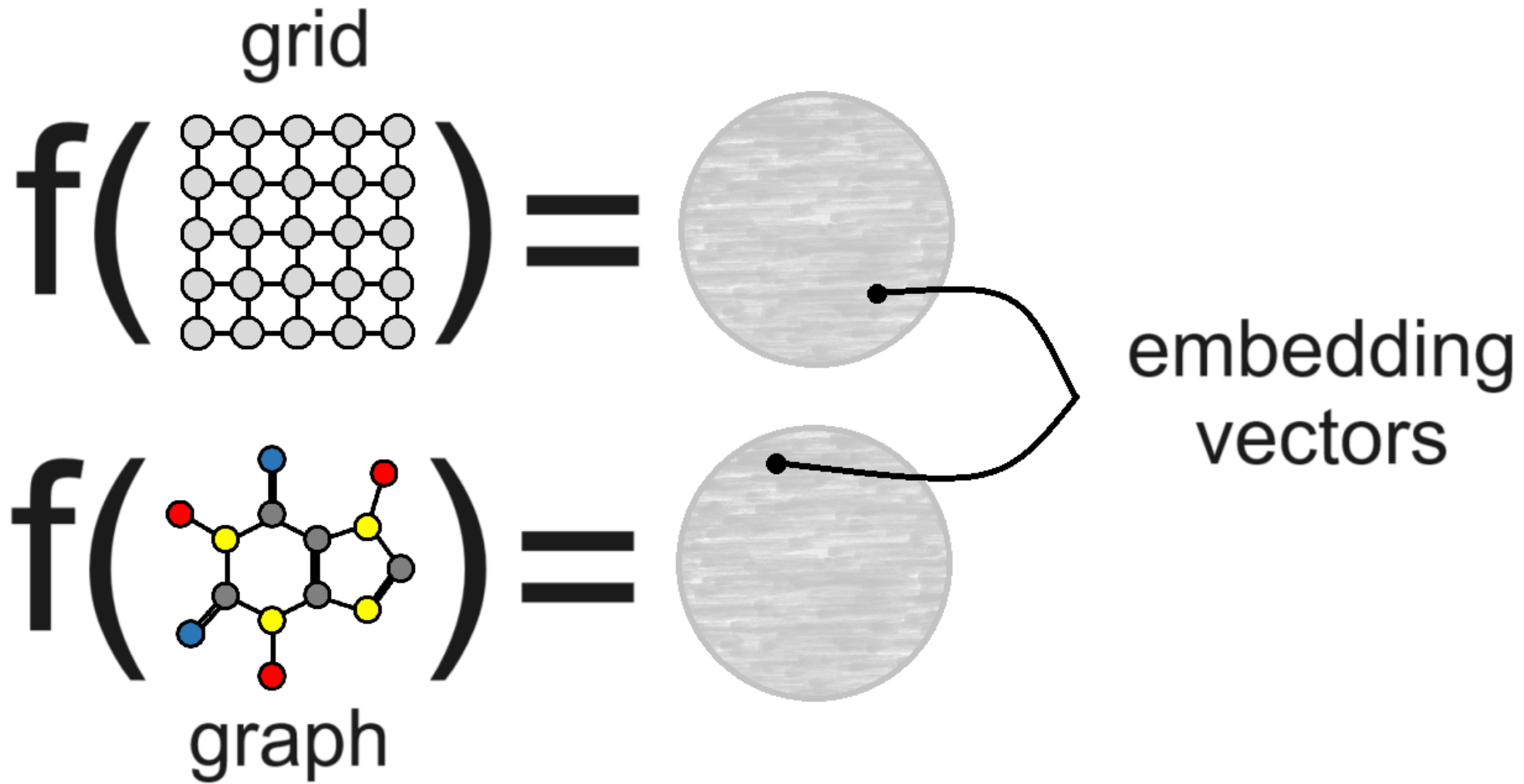
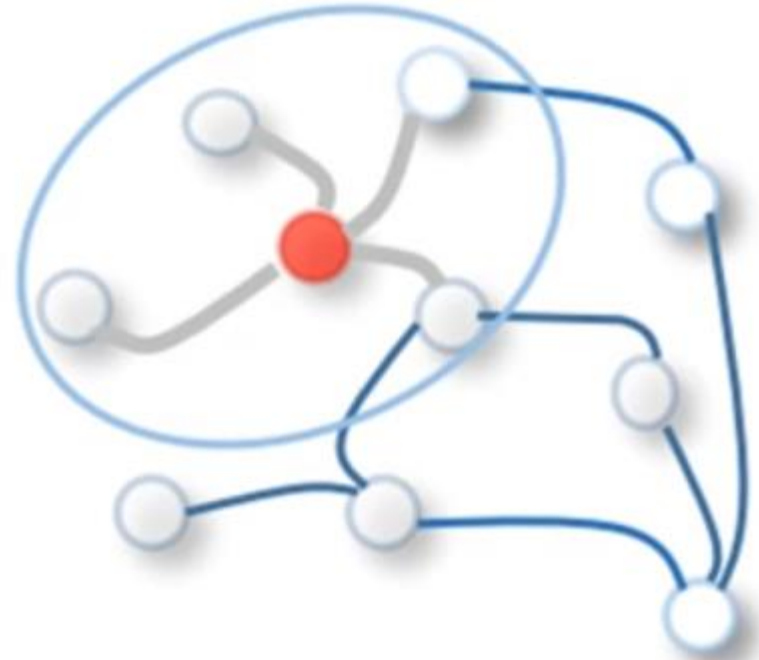
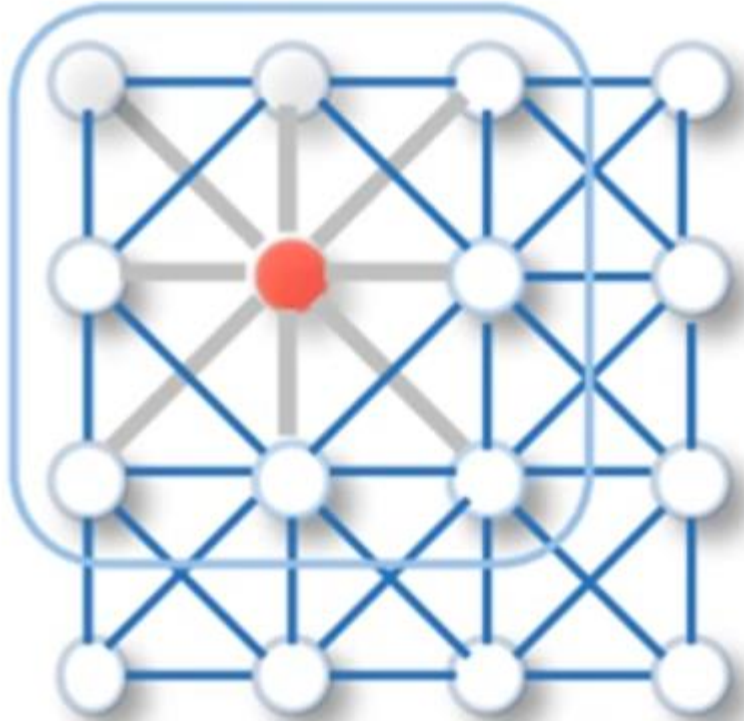


Image convolution vs graph convolution

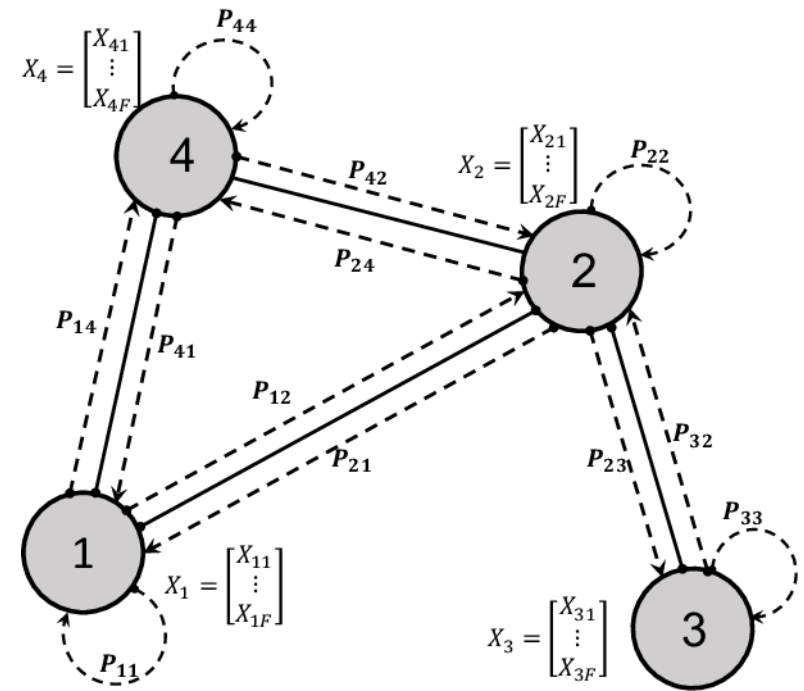


Message passing layers

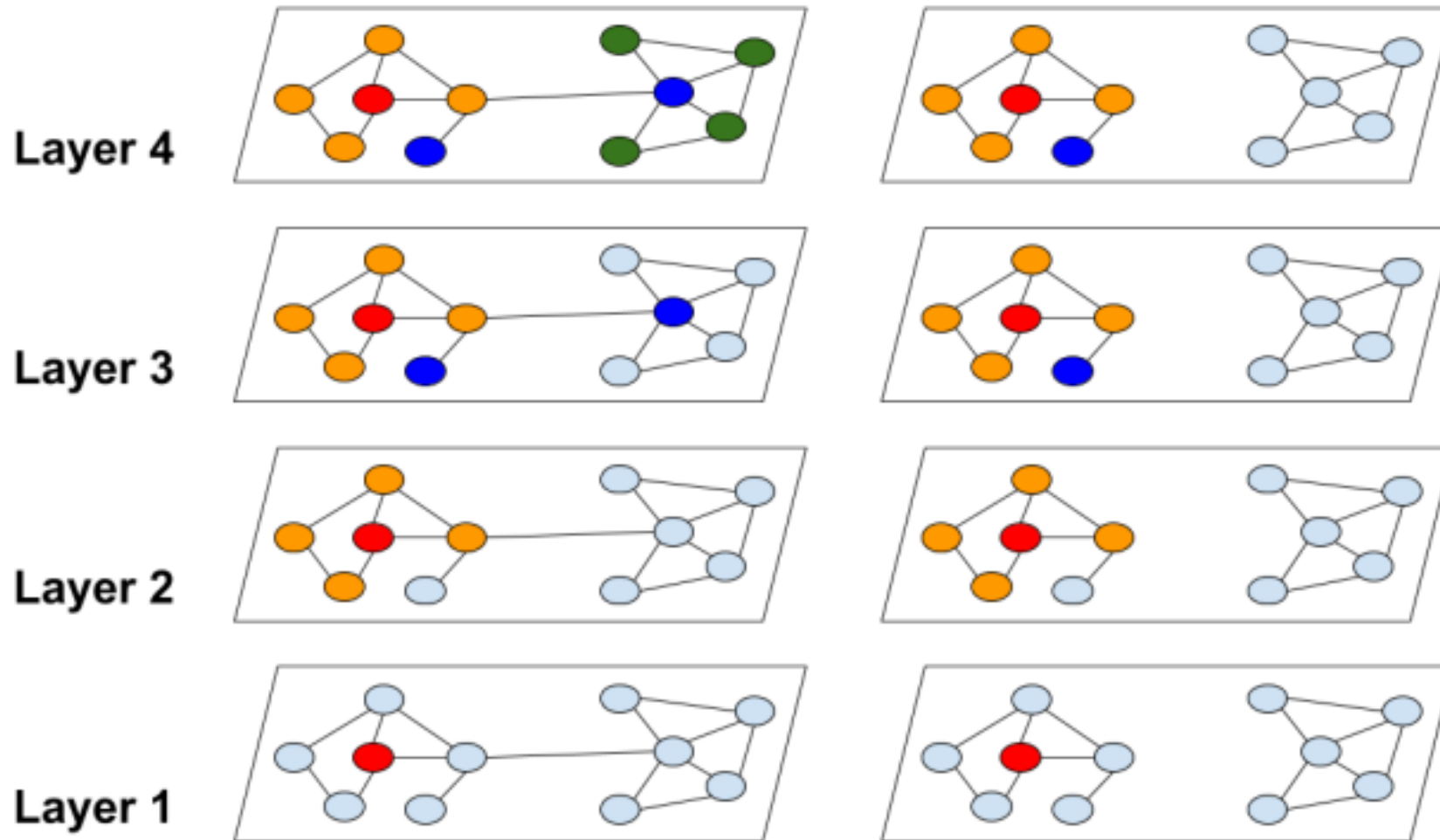
- In each layer, we aggregate each node's own features with its neighbors' features
- Graph Isomorphism Network (GIN)
 - Aggregation is summation
 - In a layer, a node's own features are summed with its neighbor's features, and then the expression is put into an MLP:

$$x'_i = MLP(x_i + \sum_{j \in \mathcal{N}(i)} x_j)$$

- Implemented as a sparse-dense matrix multiplication: $X' = MLP(X + AX)$ where A is the $N \times N$ adjacency matrix and X is the $N \times F$ feature matrix
- If the Weisfeiler-Lehman test can distinguish two graphs, a GIN also can → with node features, GINs are more powerful



Message passing layers



Message passing layers

- GIN cannot handle edge features
- A more general message passing layer:

$$x'_i = \phi^x \left(x_i, \sum_{j \in \mathcal{N}(i)} \phi^e(x_i, x_j, e_{ij}) \right)$$

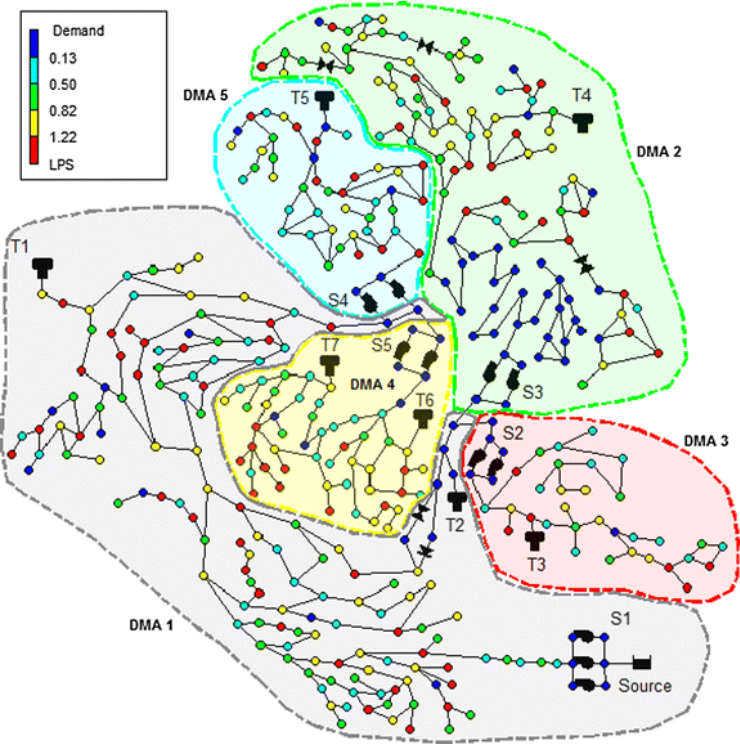
where ϕ^x and ϕ^e are learnable functions (MLPs)

- Edge features can also be updated:

$$e'_{ij} = \phi^e(x_i, x_j, e_{ij})$$

$$x'_i = \phi^x \left(x_i, \sum_{j \in \mathcal{N}(i)} e'_{ij} \right)$$

Examples - again



Water distribution systems

image source: [springer](https://www.springer.com)



social networks

image source: [Medium](https://www.medium.com)

$$f(\text{graph}) = \text{graph}$$

The diagram shows a chemical structure graph on the left, followed by an equals sign and a circular graph on the right, illustrating a function f that maps a graph to another graph.

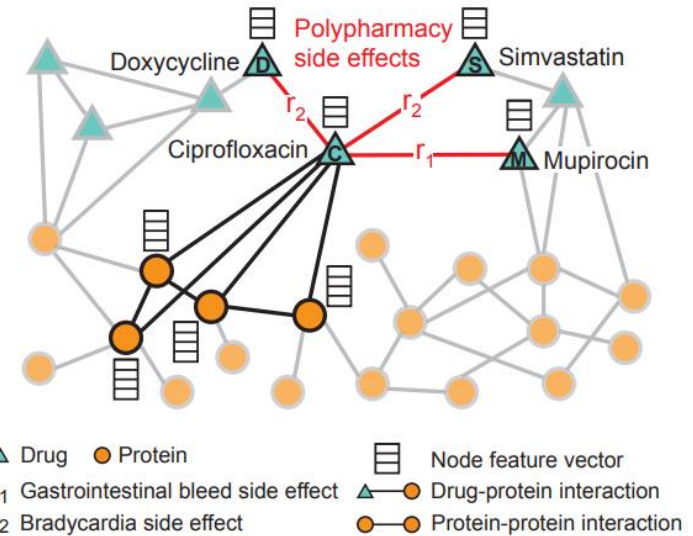
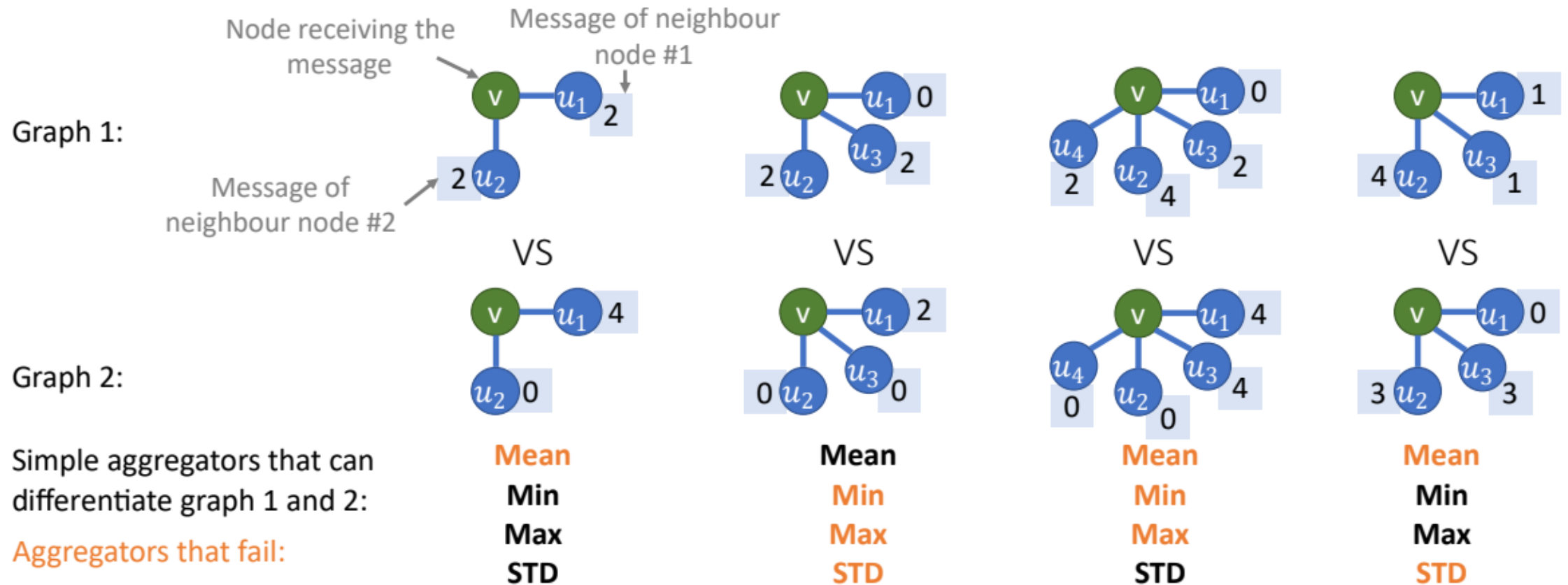


Figure from:

<https://academic.oup.com/bioinformatics/article/34/13/i457/5045770>

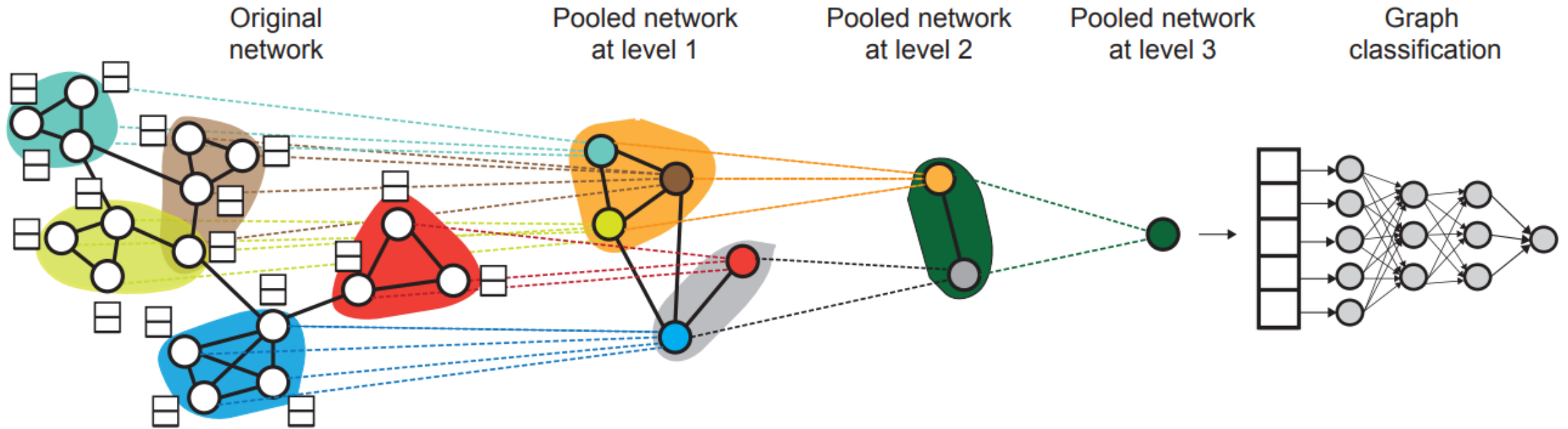
Message passing layers

- Other permutation invariant operations instead of summation:



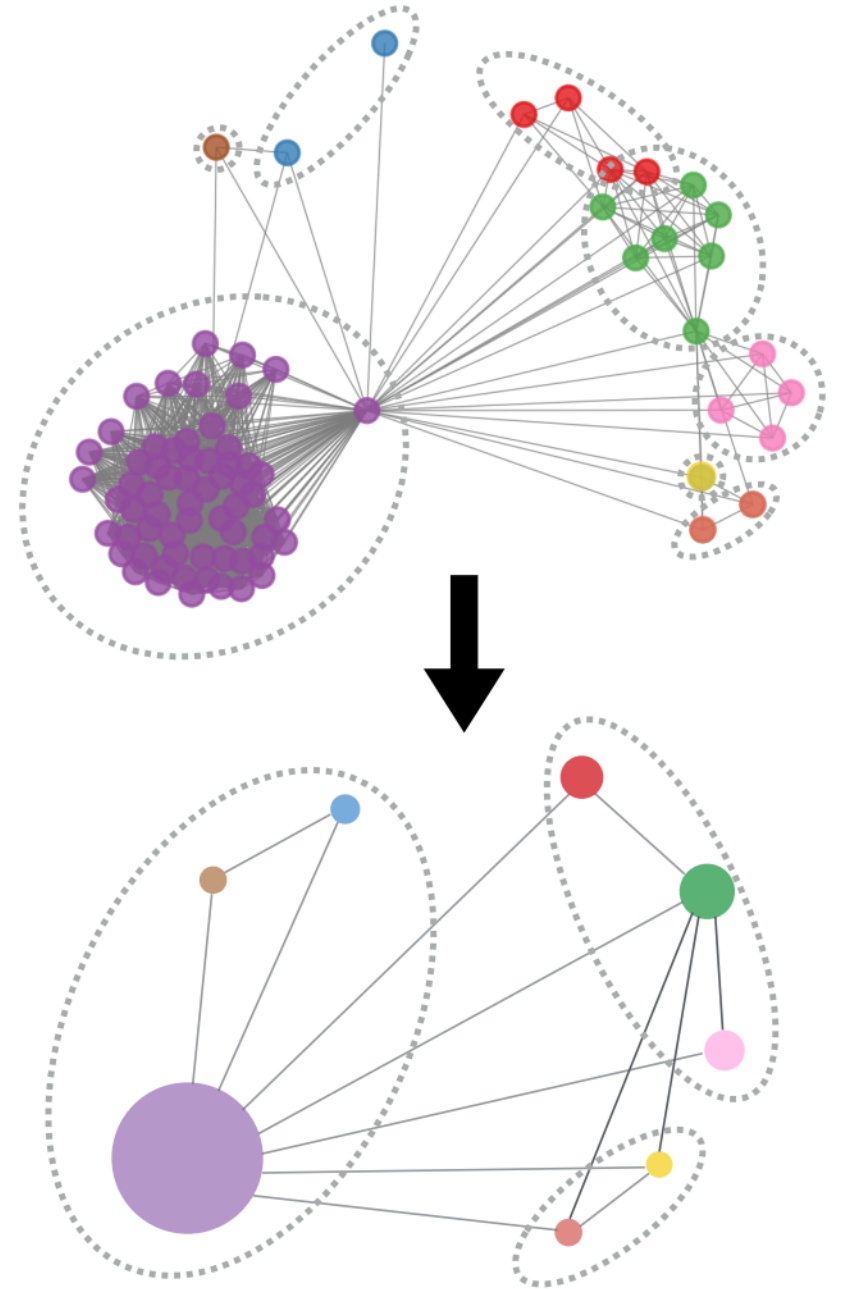
$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}$$

Pooling layers



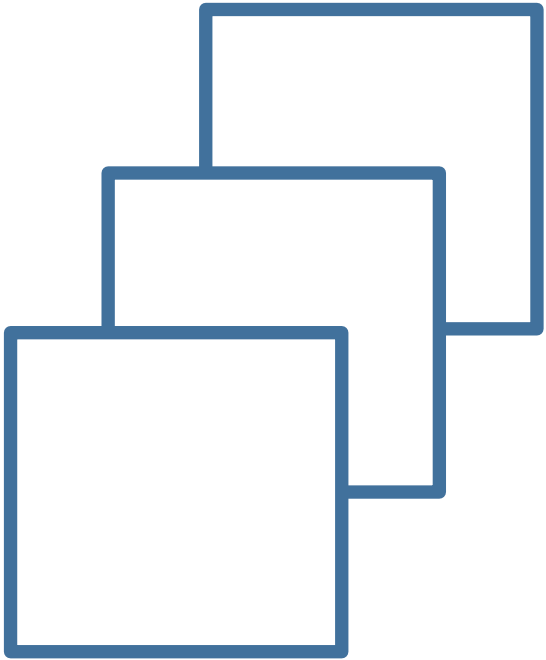
Pooling layers

- $Z = GNN(X, A)$ $N \times F$ matrix
- $S = \text{softmax}(GNN(X, A))$ $N \times M$ matrix
- “soft cluster assignments” $M < N$
- $X' = S^T Z$ $M \times F$ matrix
- $A' = S^T A S$ $M \times M$ matrix

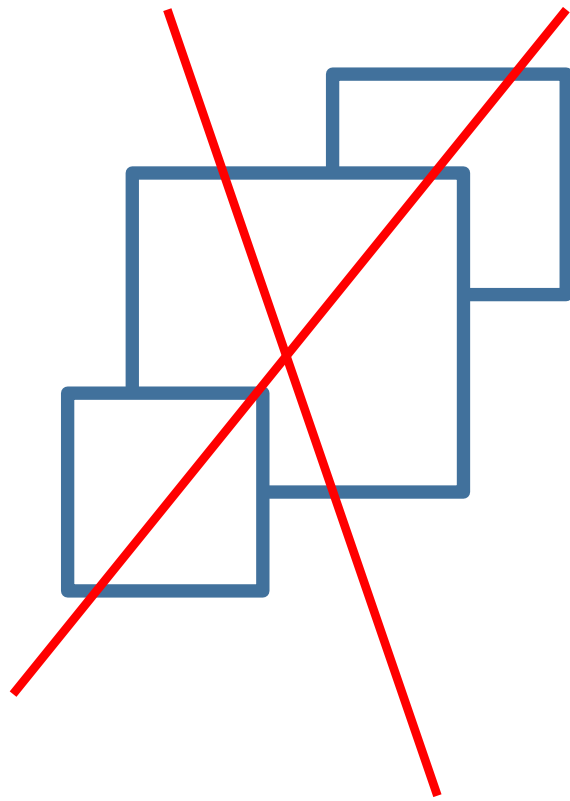


Mini-batching

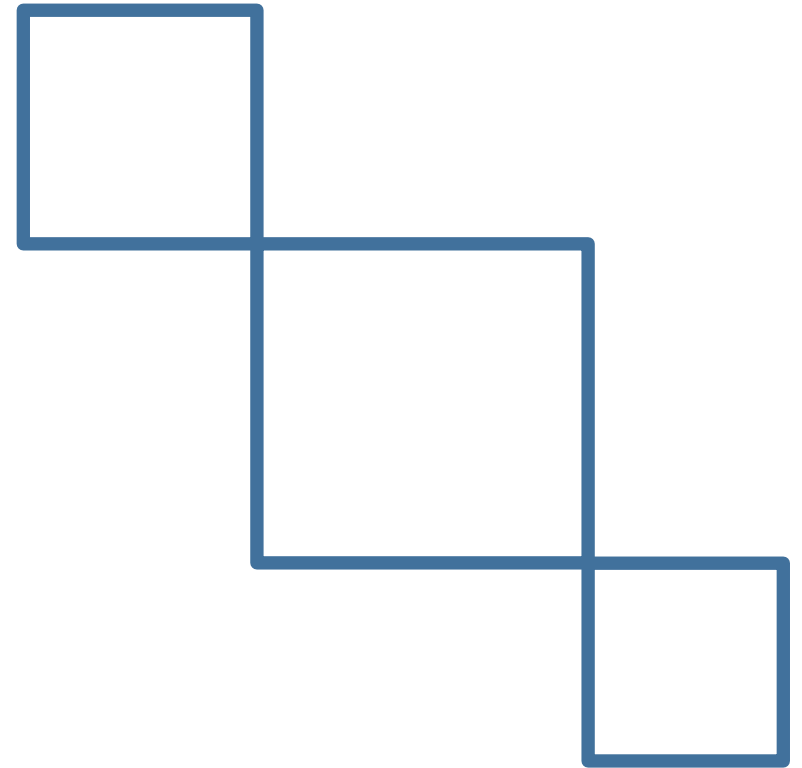
Adjacency matrices,
Graphs with same size



Adjacency matrices,
Graphs with different sizes



Adjacency matrix,
Blockdiagonal



Mini-batching

- Multiple graphs in a mini-batch:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & \\ & \ddots & \\ & & \mathbf{A}_n \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \vdots \\ \mathbf{X}_n \end{bmatrix} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \vdots \\ \mathbf{Y}_n \end{bmatrix}$$

- Very large graphs: a graph partitioning algorithm is used (e.g. [METIS](#)), and multiple clusters are randomly selected to create a mini-batch

Another interesting application

Simulation of complex physics with graph neural networks:

<https://sites.google.com/view/learning-to-simulate>

Water



The image features a network graph with numerous nodes and edges, rendered in a light teal color. The nodes are small circles, and the edges are thin lines connecting them. The graph is spread across the width of the image. A semi-transparent white rectangular box is positioned horizontally across the middle of the image, containing the text "Self-supervised learning" in a bold, black, sans-serif font. The background is a light, neutral color with a subtle gradient.

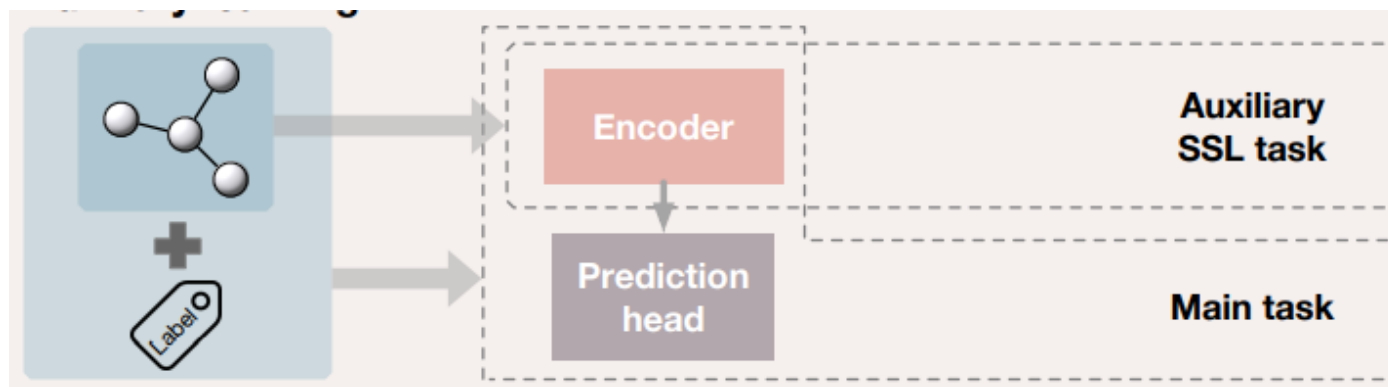
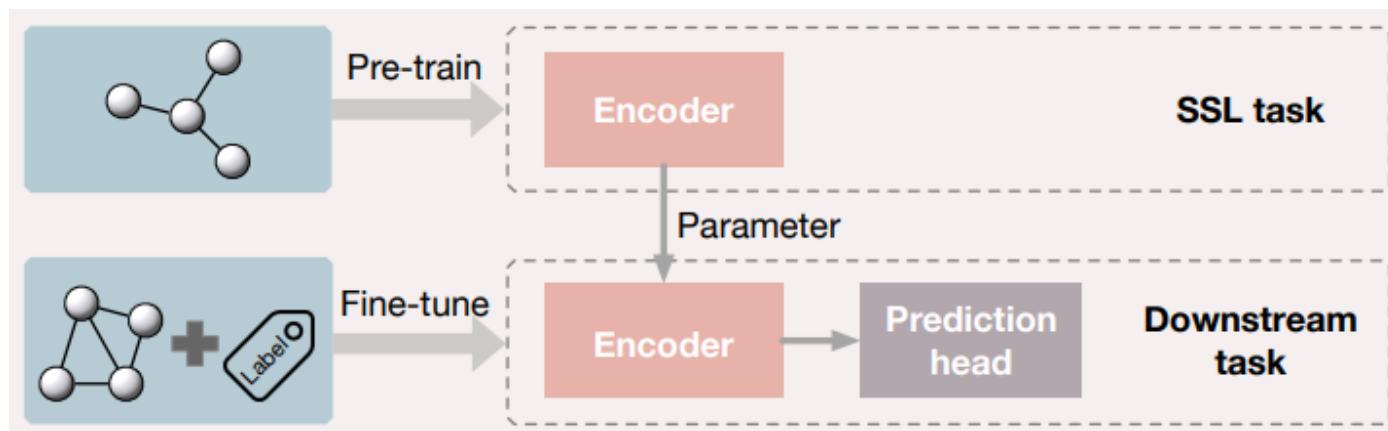
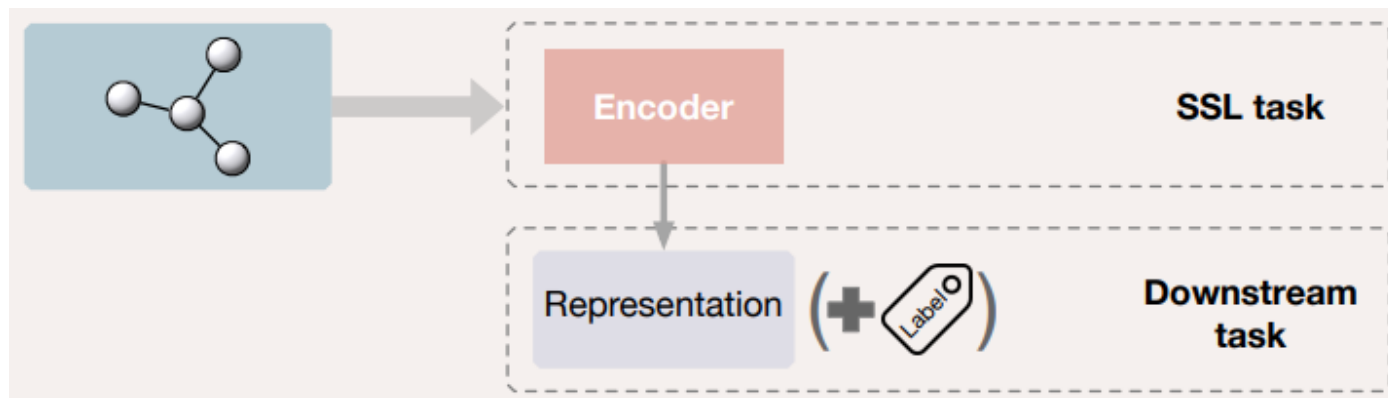
Self-supervised learning

Self-supervised learning

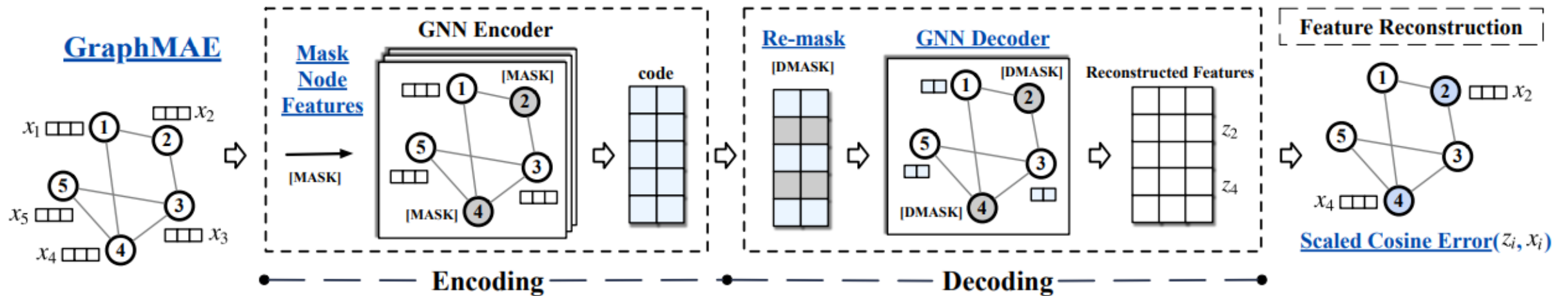
- No pretrained models or foundation models: very different domains → very different graph structures and features
- Pretraining works well within the same domain
- Labeled data is expensive, and often requires domain knowledge
- But networks are everywhere around us → self-supervised learning

Self-supervised learning

- 1) Unsupervised representation learning
- 2) Unsupervised pretraining:
- 3) Auxiliary learning:



Self-supervised learning: an example

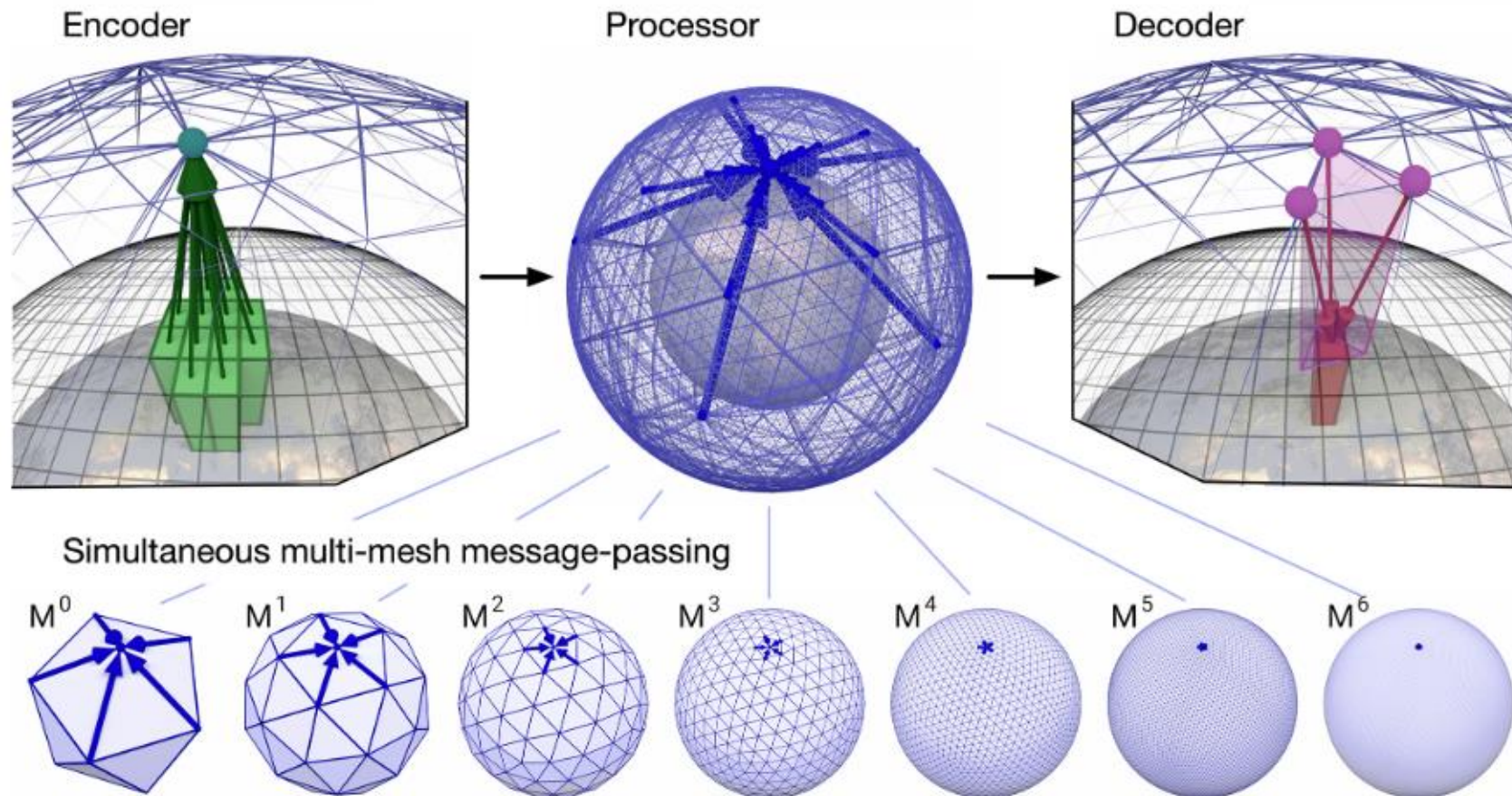


- Features of the selected nodes are masked using a mask token
- The graph is encoded with a GNN encoder
- The codes of the selected nodes are masked using another mask token
- The codes are decoded with a GNN decoder \rightarrow feature reconstruction loss

Once another interesting application

AI model for faster and more accurate global weather forecasting

https://charts.ecmwf.int/products/graphcast_medium-mslp-wind850



A network graph visualization with numerous nodes and edges. The nodes are small circles, and the edges are thin lines connecting them. The graph is overlaid with a semi-transparent white rectangular box in the center. The word "References" is written in a simple, black, sans-serif font within this box.

References

References

- Book
 - William L. Hamilton (2020). Graph representation learning. Morgan & Claypool Publishers.
- Videos
 - <https://geometricdeeplearning.com/lectures/>
 - <https://www.youtube.com/watch?v=blZB1hIJ4u8>
 - <https://www.youtube.com/watch?v=jAGluobLp60>
 - <https://sites.google.com/view/learning-to-simulate>
 - <https://sites.google.com/view/meshgraphnets>
- Website, blog posts
 - <https://web.stanford.edu/class/cs224w/>
 - <https://distill.pub/2021/gnn-intro/>
 - <https://distill.pub/2021/understanding-gnns/>

Please, don't forget
to send feedback:

<https://bit.ly/bme-dl>



Thank you for your attention

Dr. Mohammed Salah Al-Radhi

malradhi@tmit.bme.hu

26 November 2024

