Deep Learning

Graph Neural Networks

Fundamentals and Software Tools

Dr. Mohammed Salah Al-Radhi

(slides by: Dániel Unyi)

Copyright

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

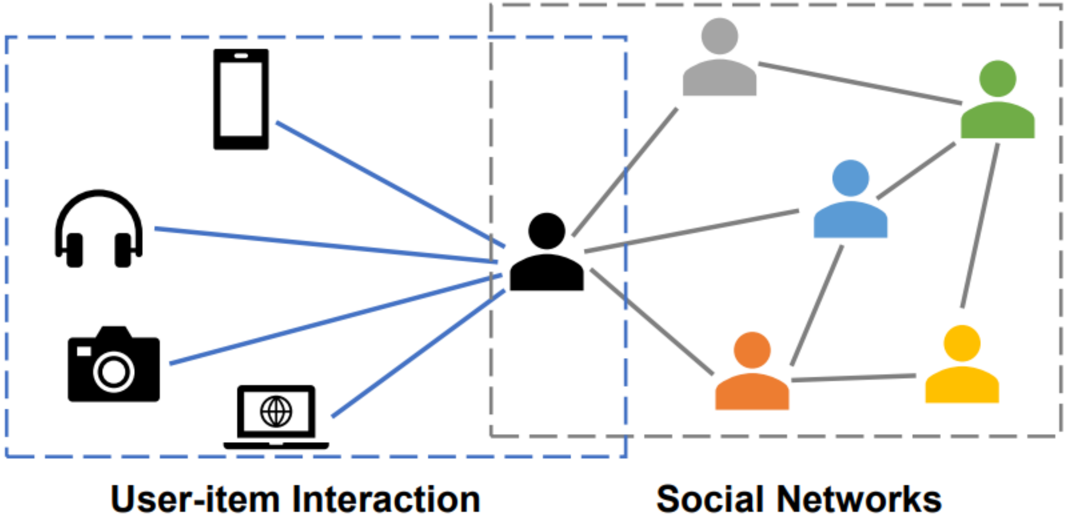
Real-world networks and GNN applications

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

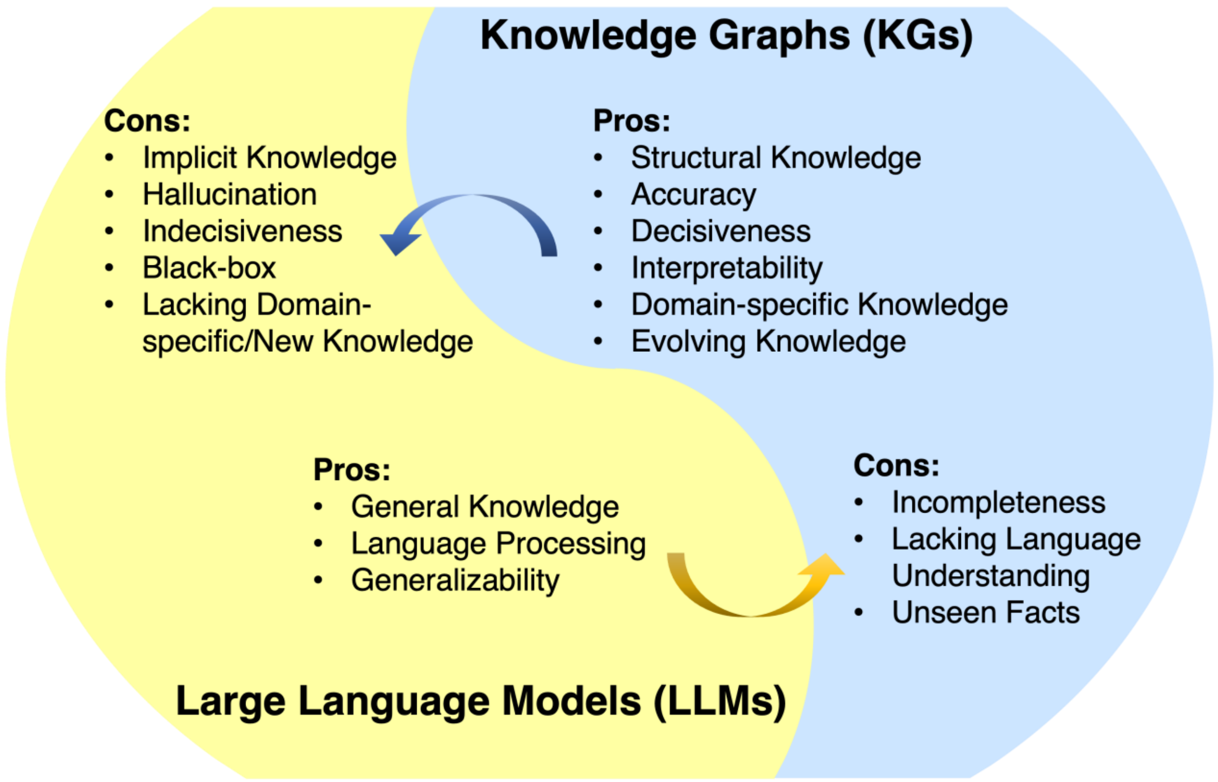
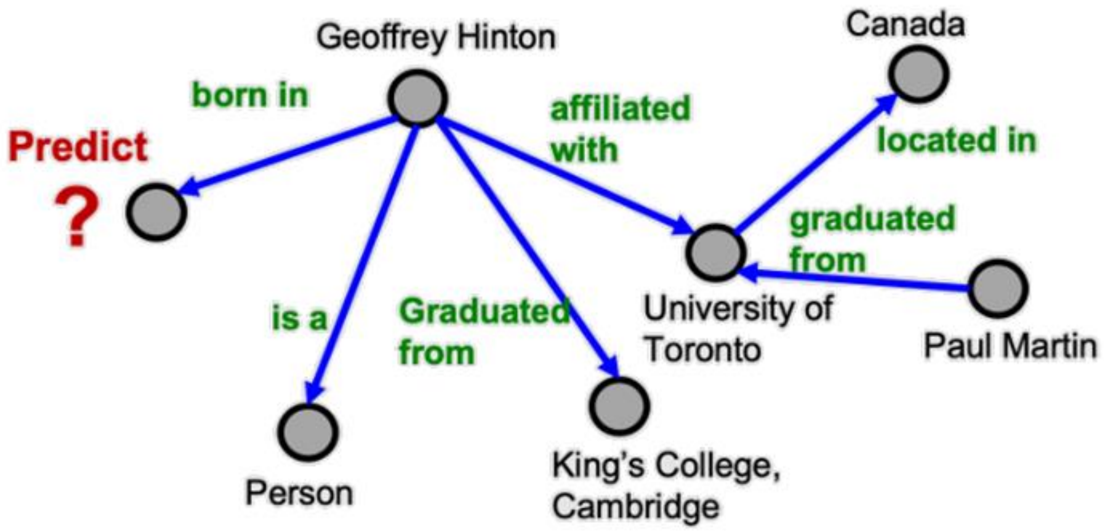


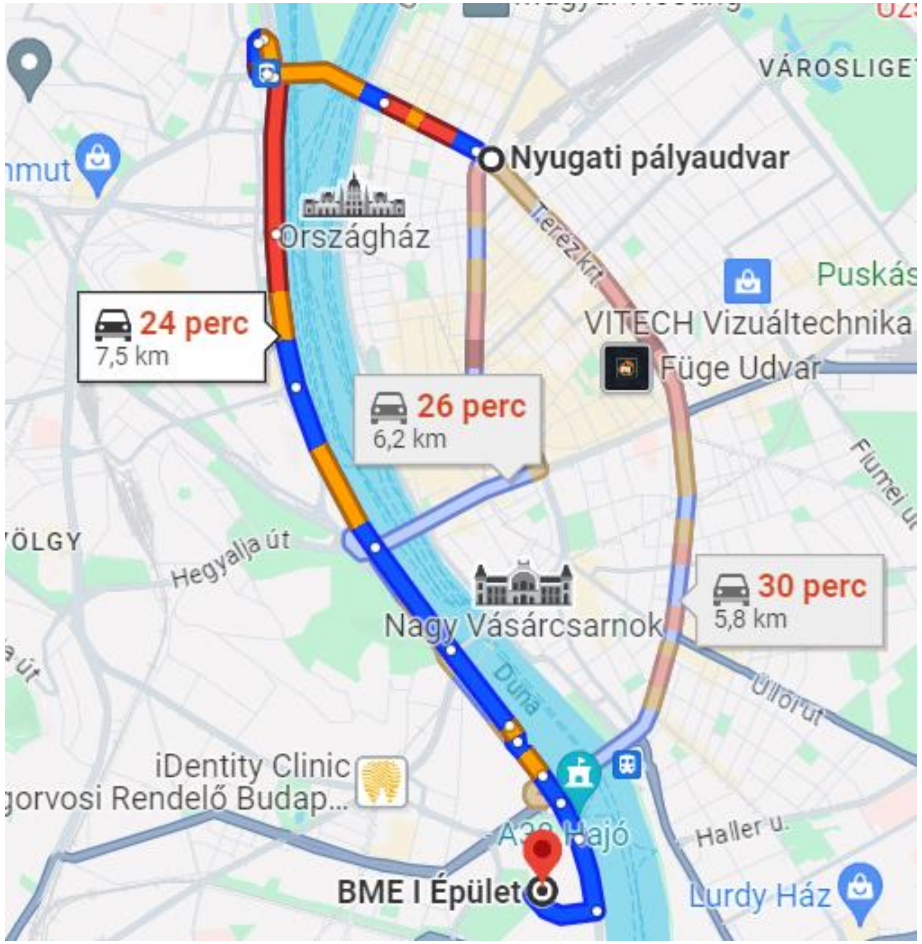
image source: [arxiv]



knowledge graphs

image source: [ogb.stanford.edu]

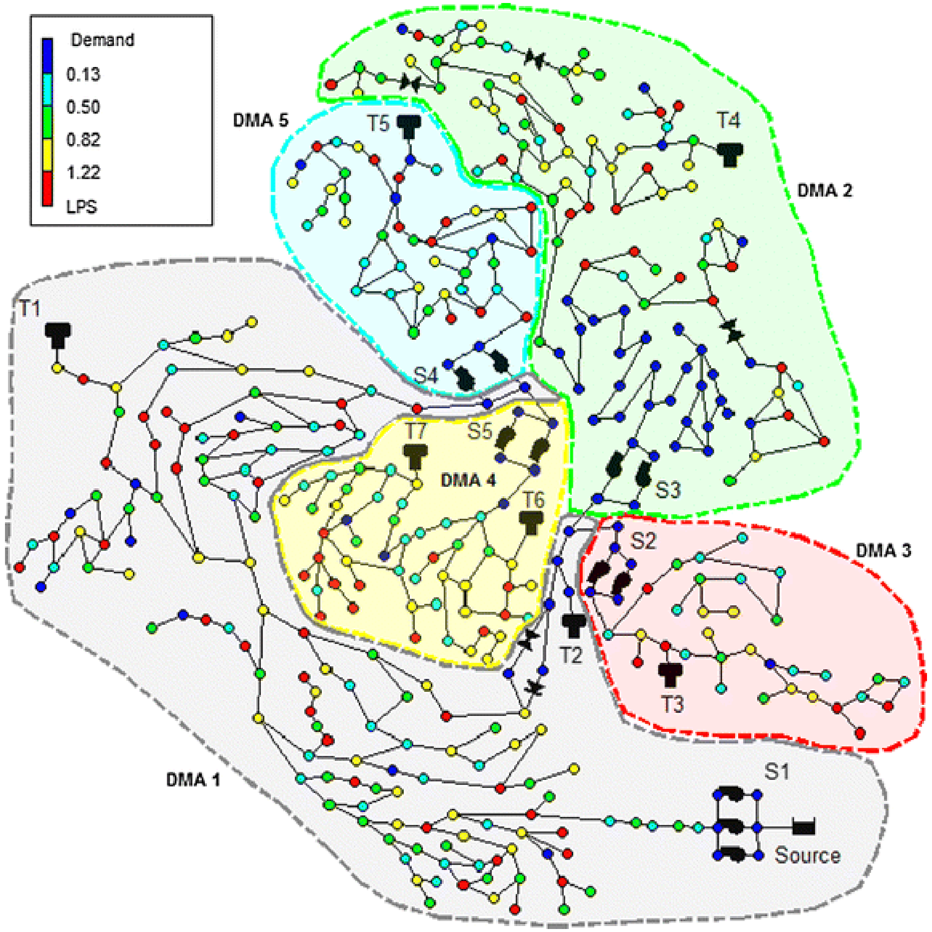
Real-world networks and GNN applications



Estimated time of arrival

(google maps)

https://arxiv.org/abs/2108.11482

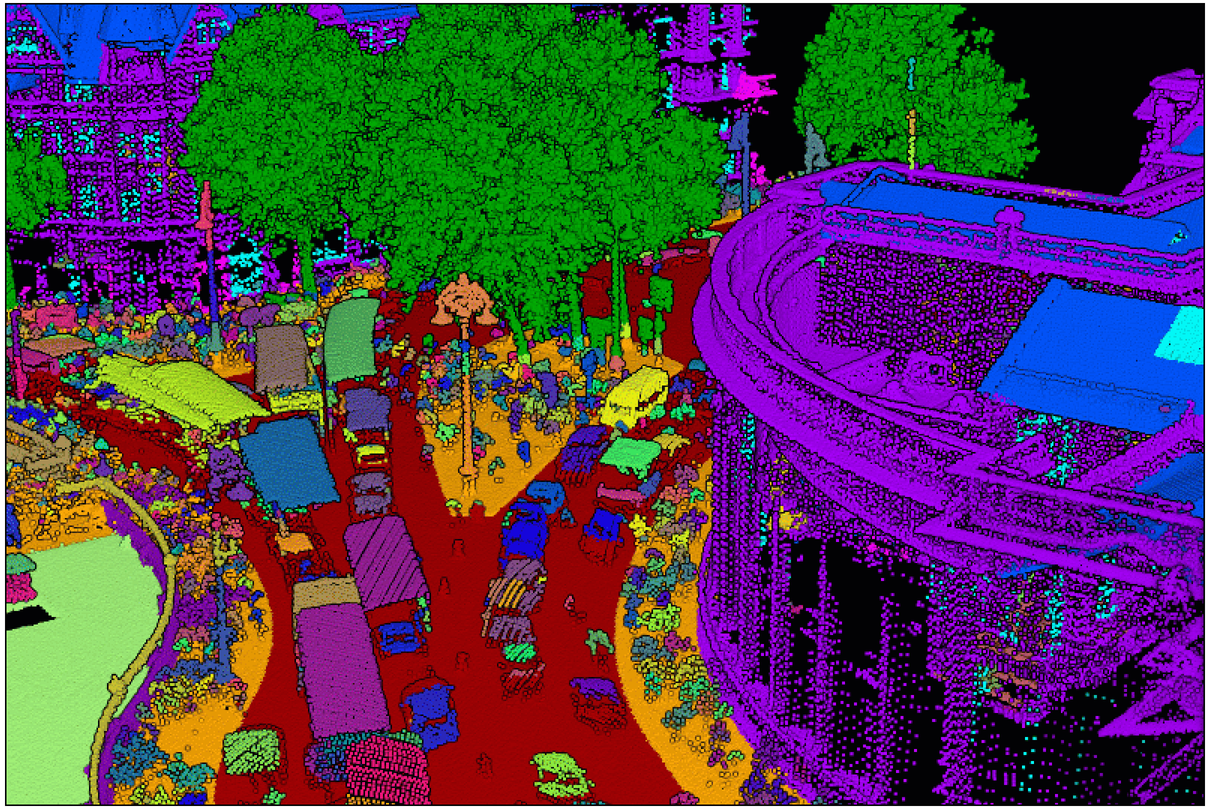


Water distribution systems

image source: springer

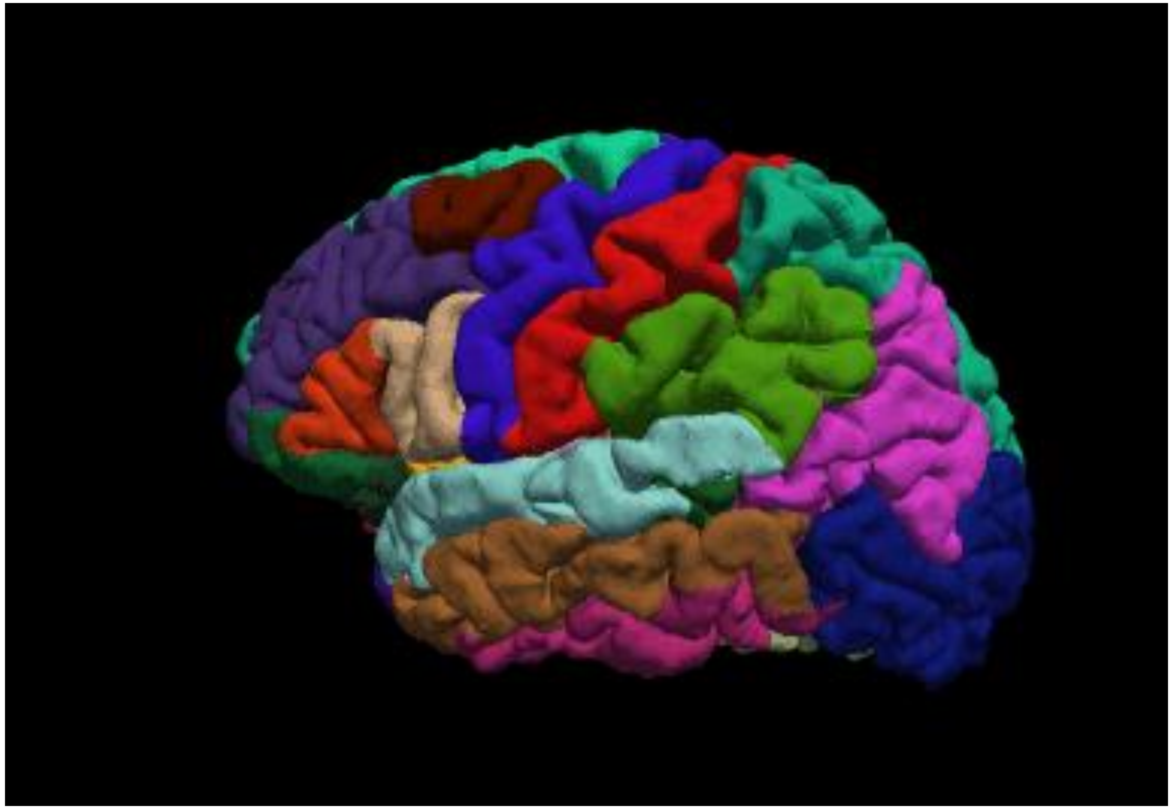
https://arxiv.org/abs/2104.13619v2

Real-world networks and GNN applications



Semantic segmentation

image source: [medium]

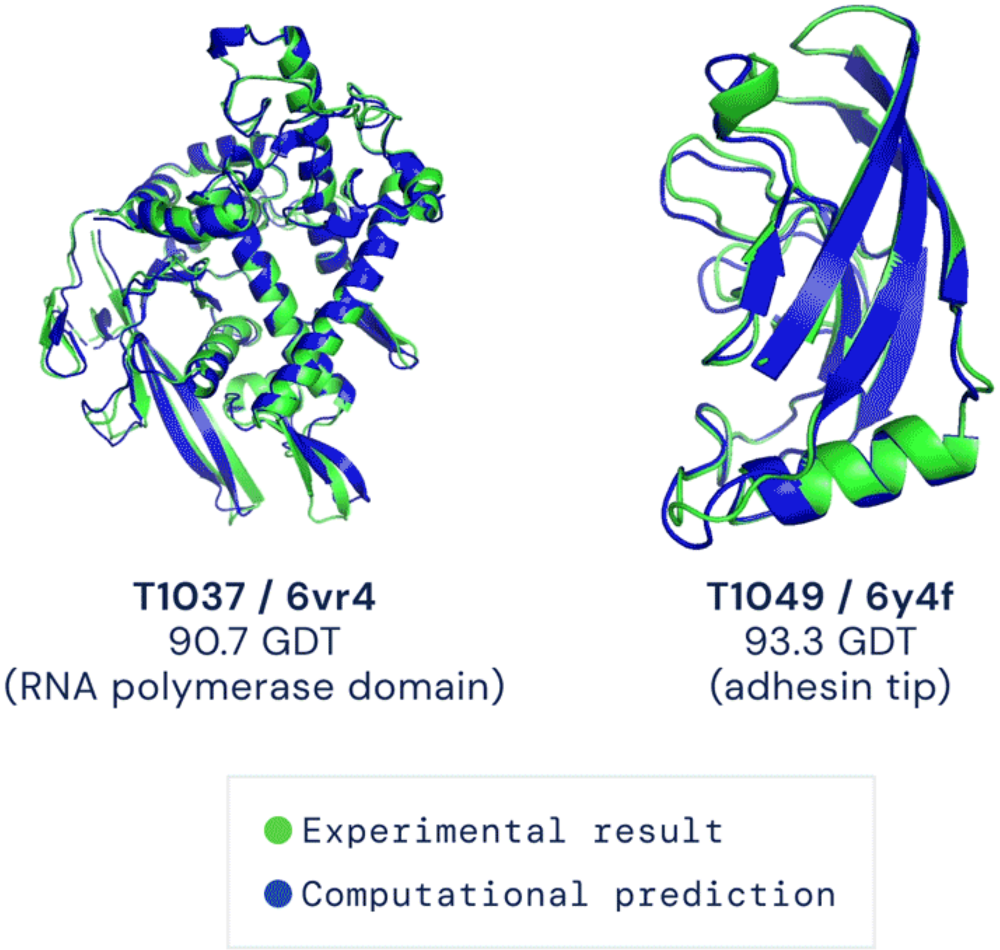


Semantic segmentation

in the medical domain

https://link.springer.com/chapter/10.1007/978-3-031-12053-4_31

Real-world networks and GNN applications



Alphafold 2.0 by DeepMind:

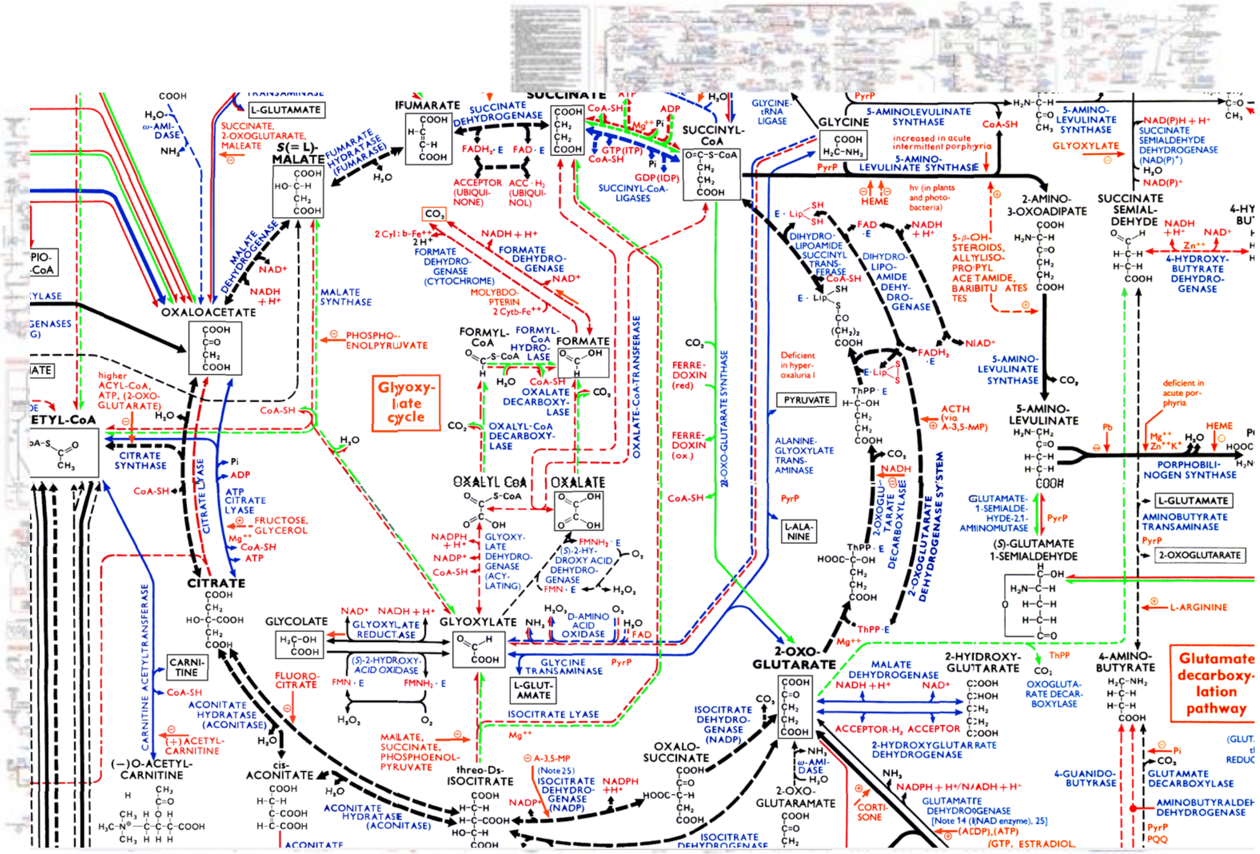
predicts a protein's 3D structure from its amino acid sequence

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

Professor Ewan Birney

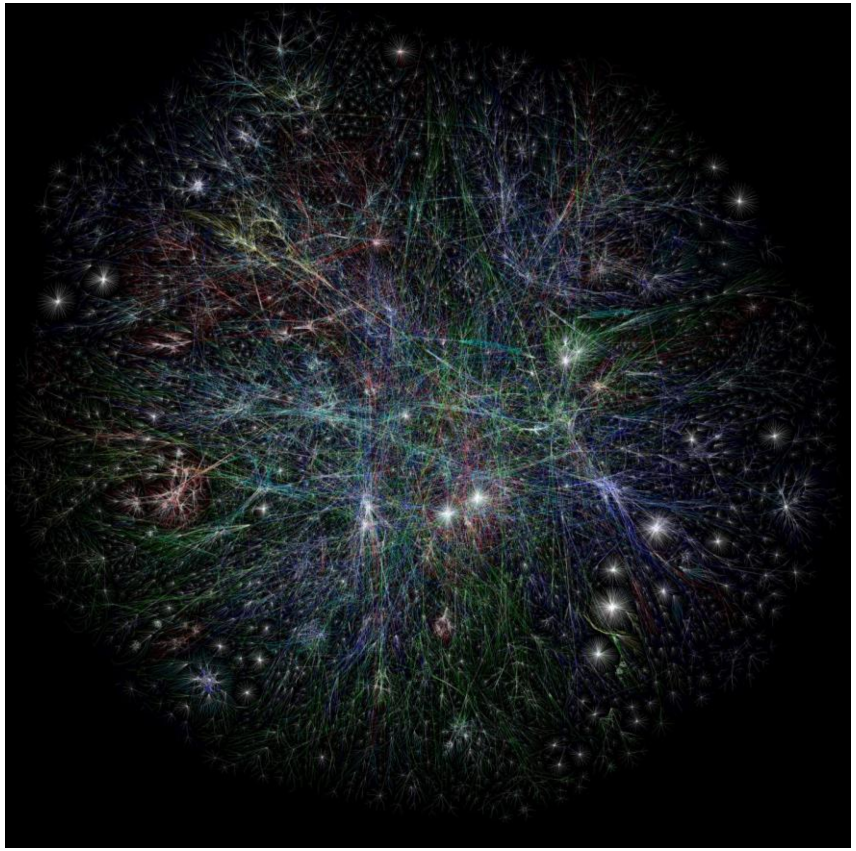
https://alphafold.ebi.ac.uk/

Real-world networks and GNN applications



Biochemical pathways in humans

image source: [metabolic_pathways]



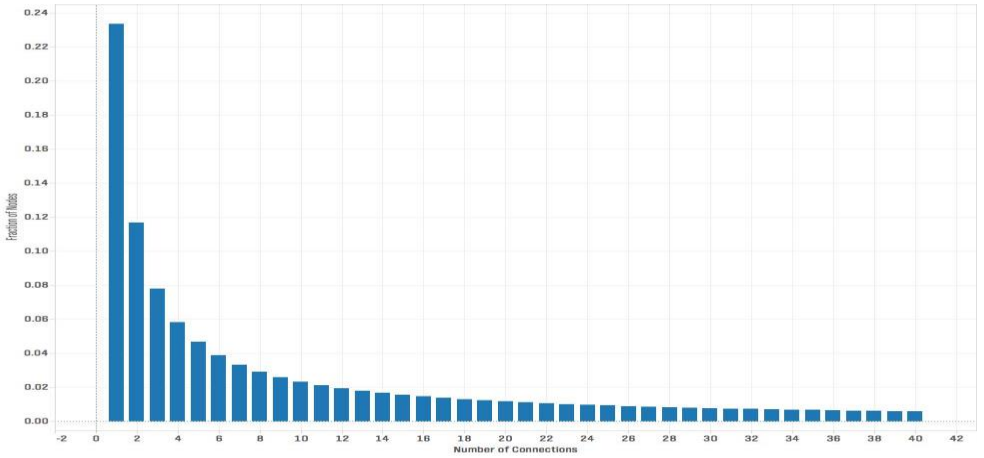
The Internet visualized

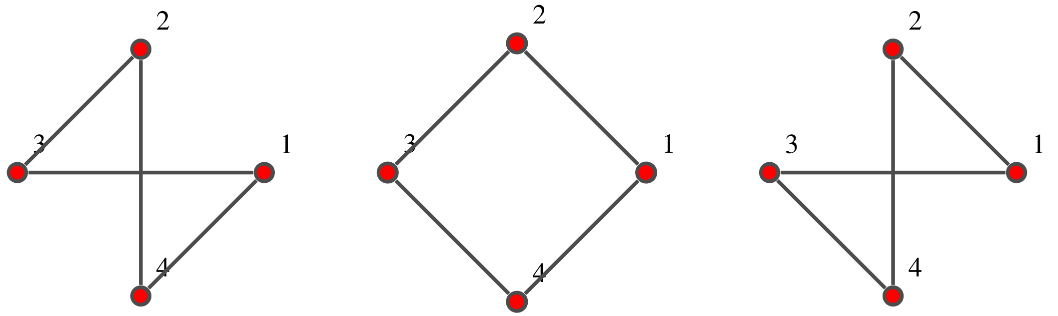
image source: [youtube]

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 x E
  - edge values 1 x E
- Undirected graphs
  - often desirable
  - symmetric adjacency matrix



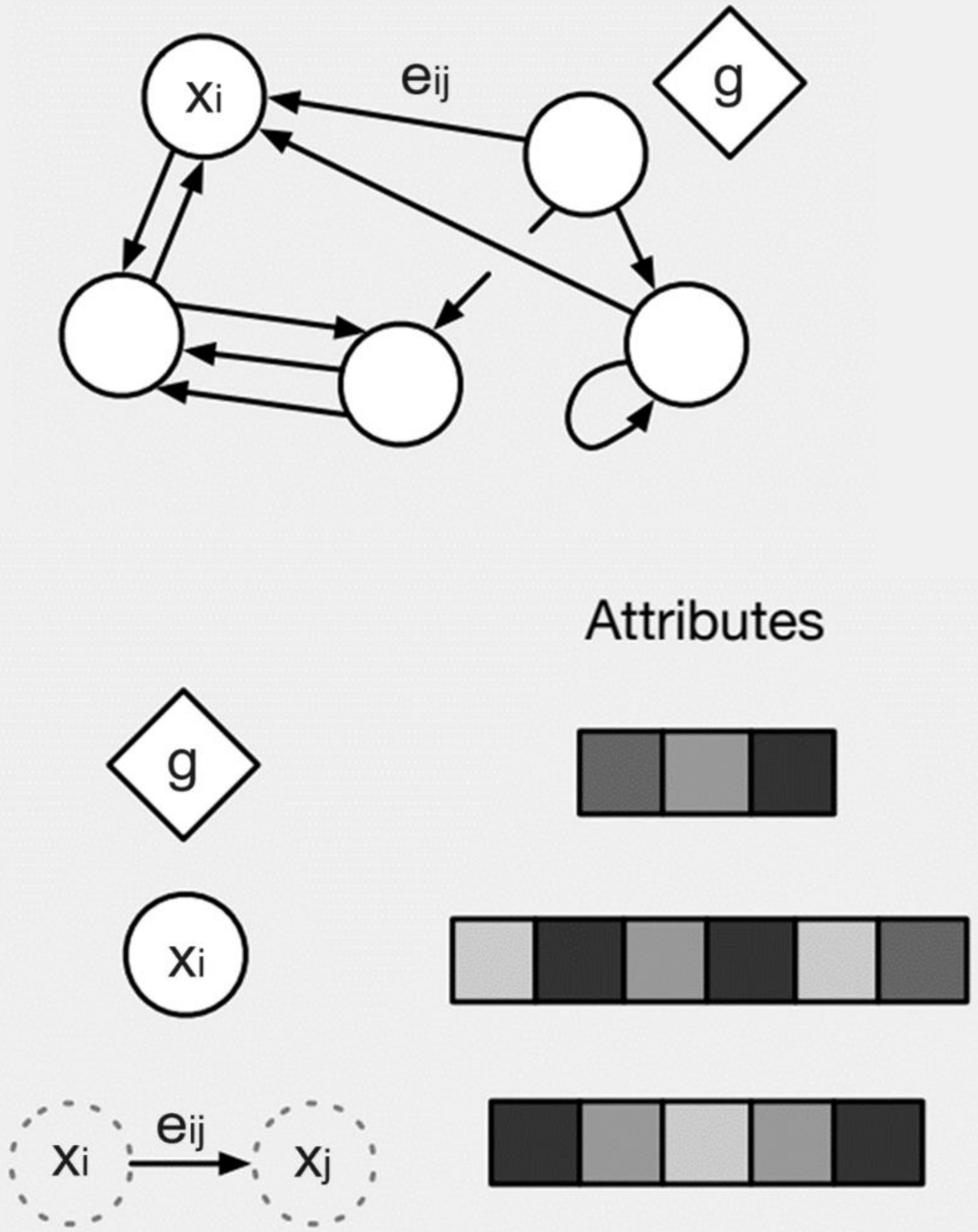


$$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Figure from: https://mathworld.wolfram.com/AdjacencyMatrix.html

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]

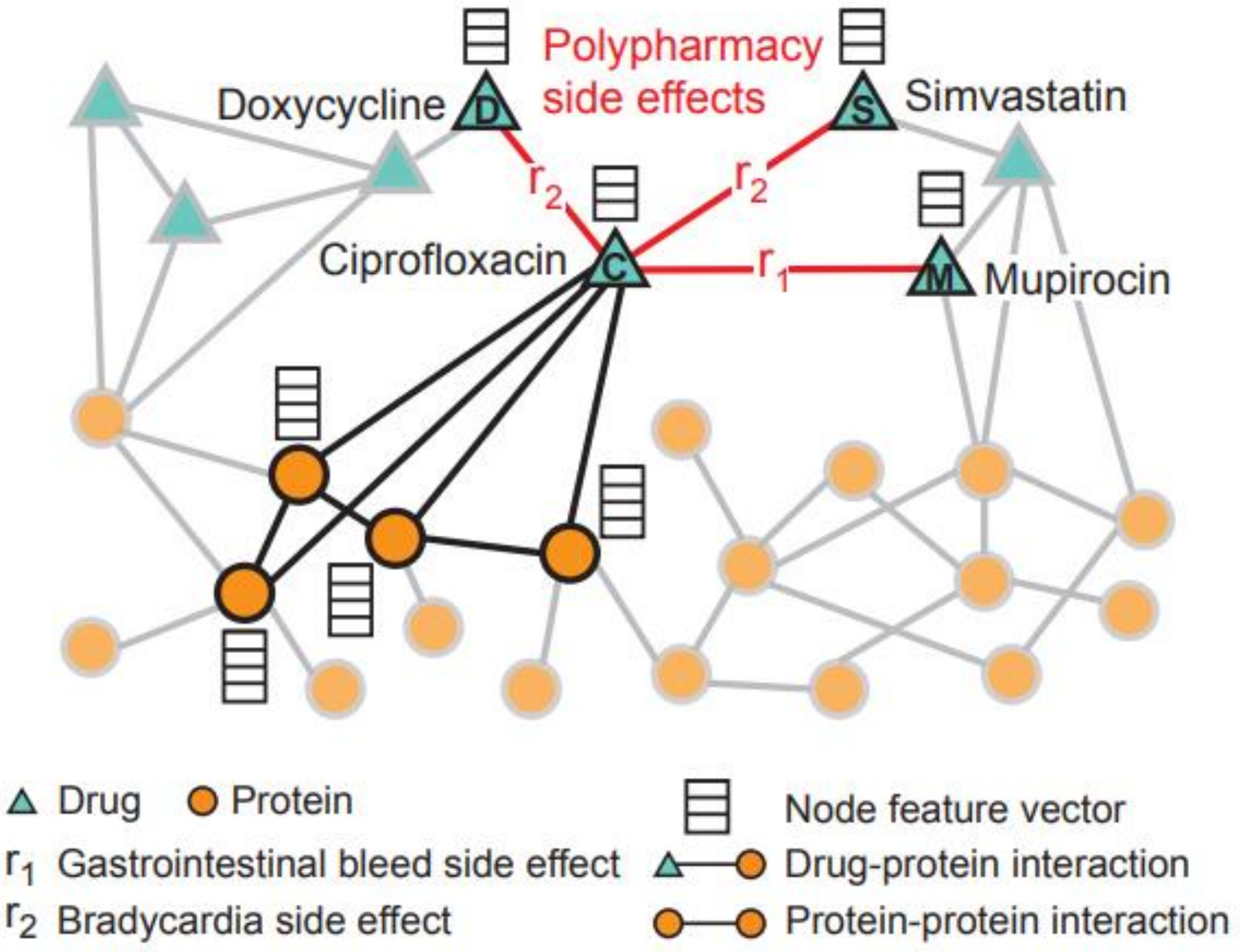
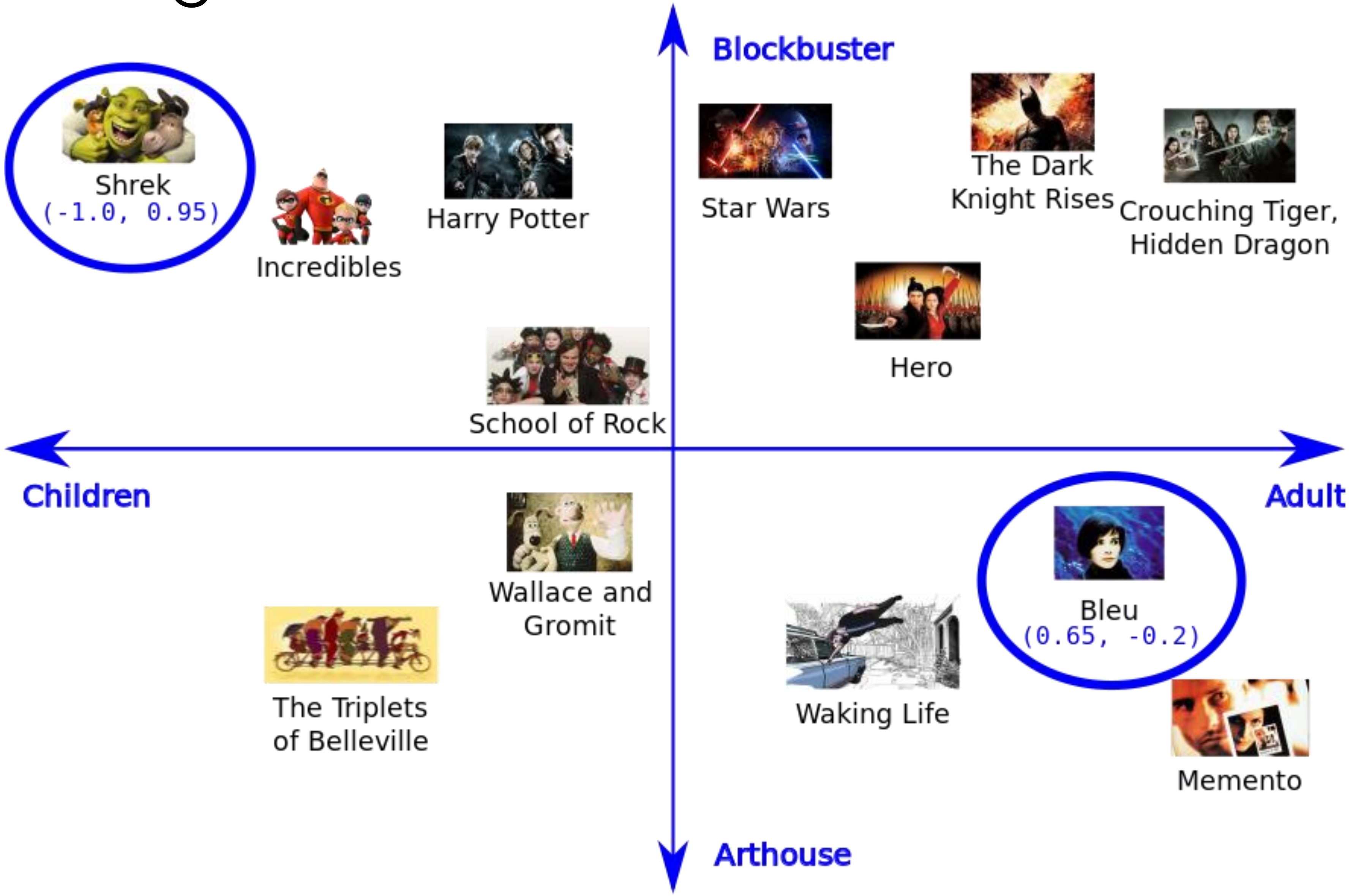


Figure from: https://academic.oup.com/bioinformatics/article/34/13/i457/5045770

Message passing and mini-batching

Embeddings



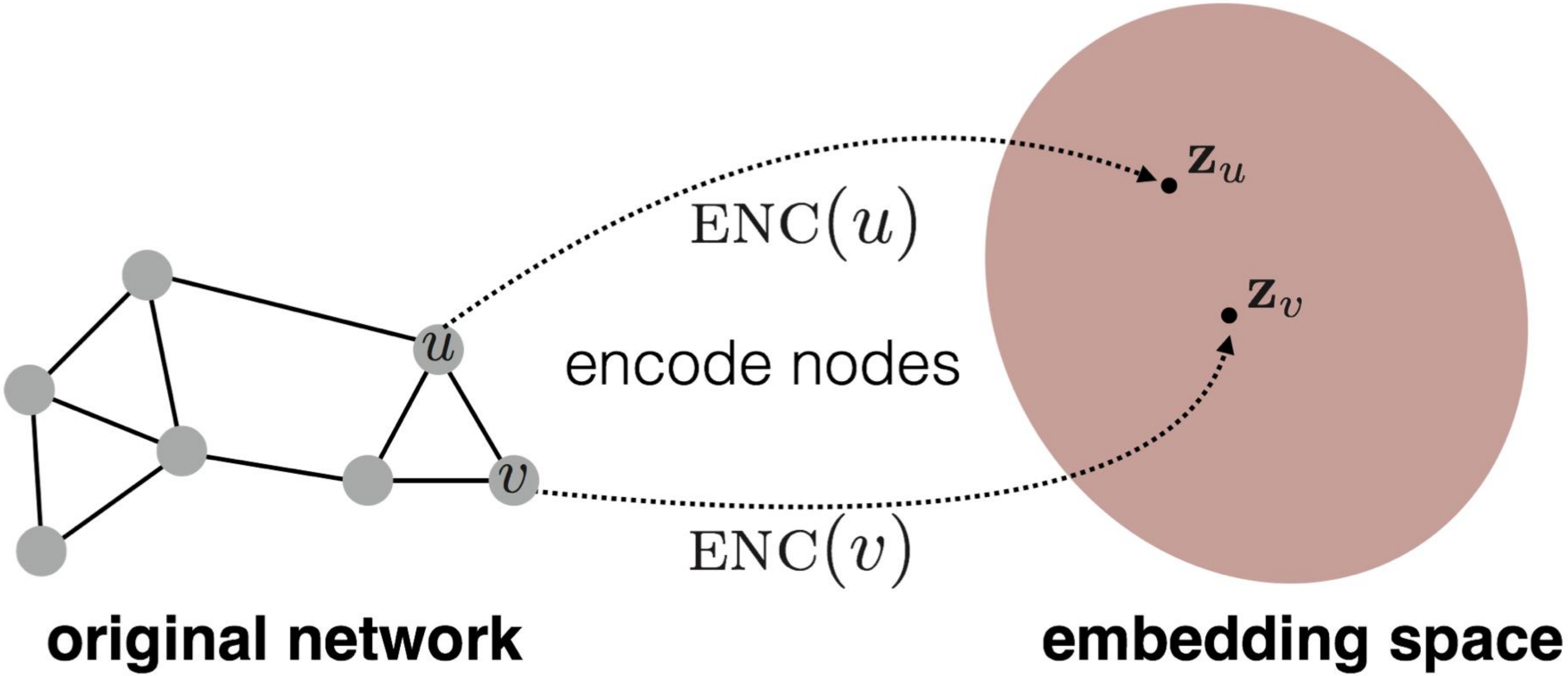
https://developers.google.com/machine-learning/crash-course/images/Embedding2dWithLabels.svg

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



https://snap-stanford.github.io/cs224w-notes/assets/img/node_embeddings.png?style=centerme

Embeddings

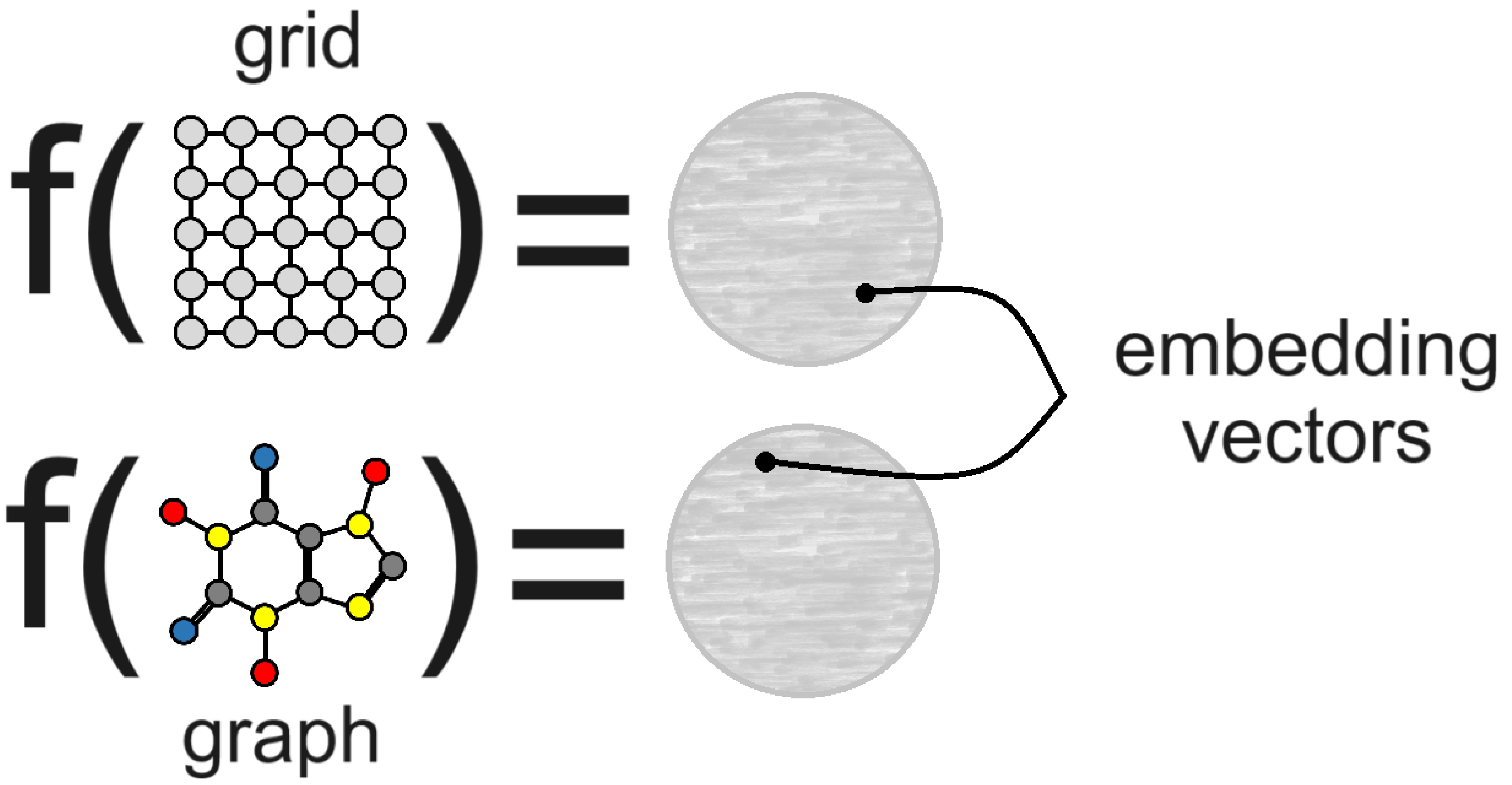
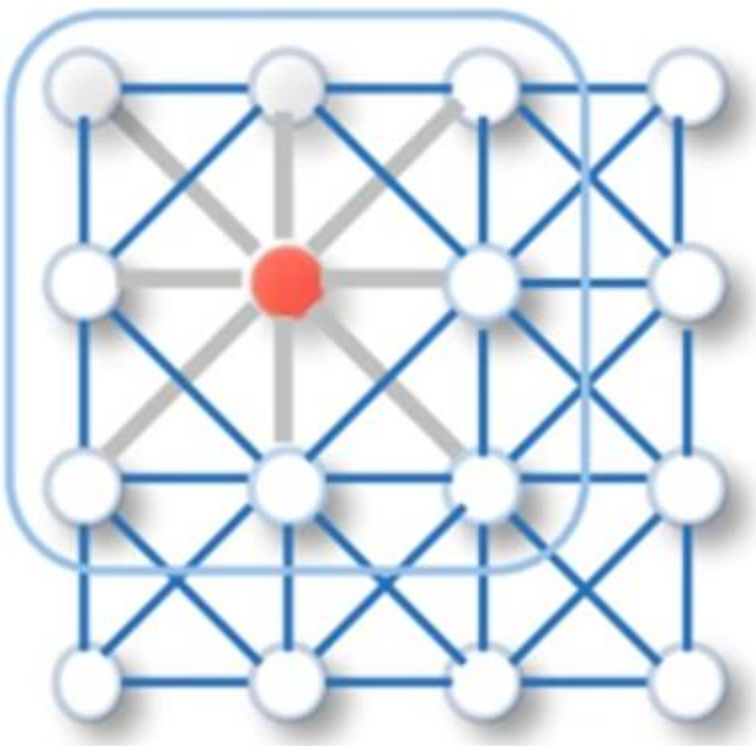
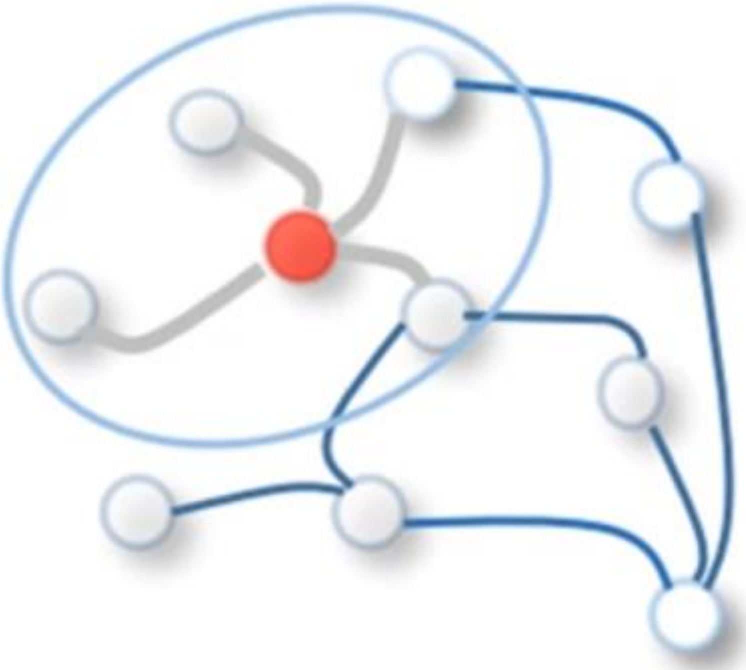


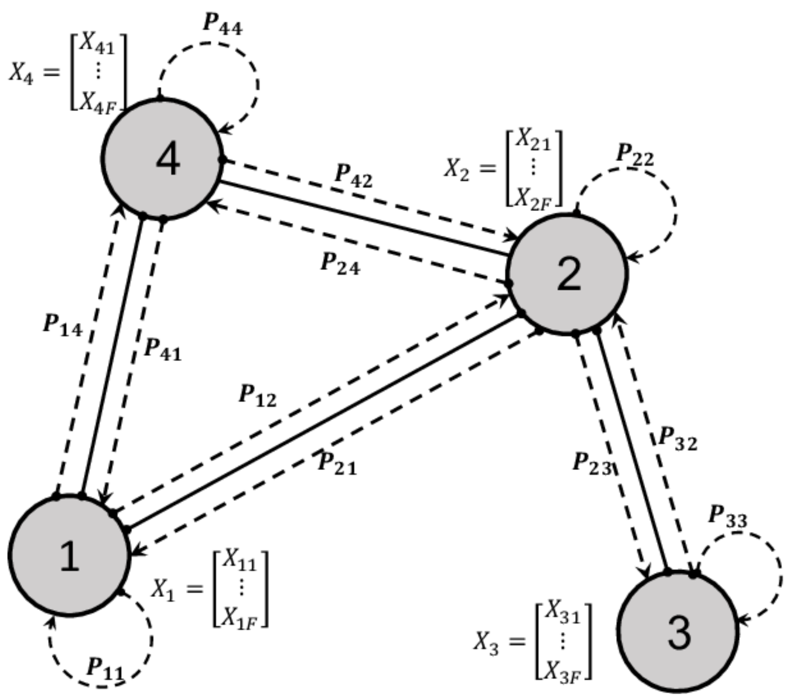
Image convolution vs graph convolution





https://arxiv.org/abs/1901.00596

Message passing layers



- In each layer, we aggregate each node's own features with its neighbors' features
- <u>G</u>raph <u>I</u>somorphism <u>N</u>etwork (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:

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

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

Reference: https://arxiv.org/pdf/1810.00826.pdf

Message passing layers

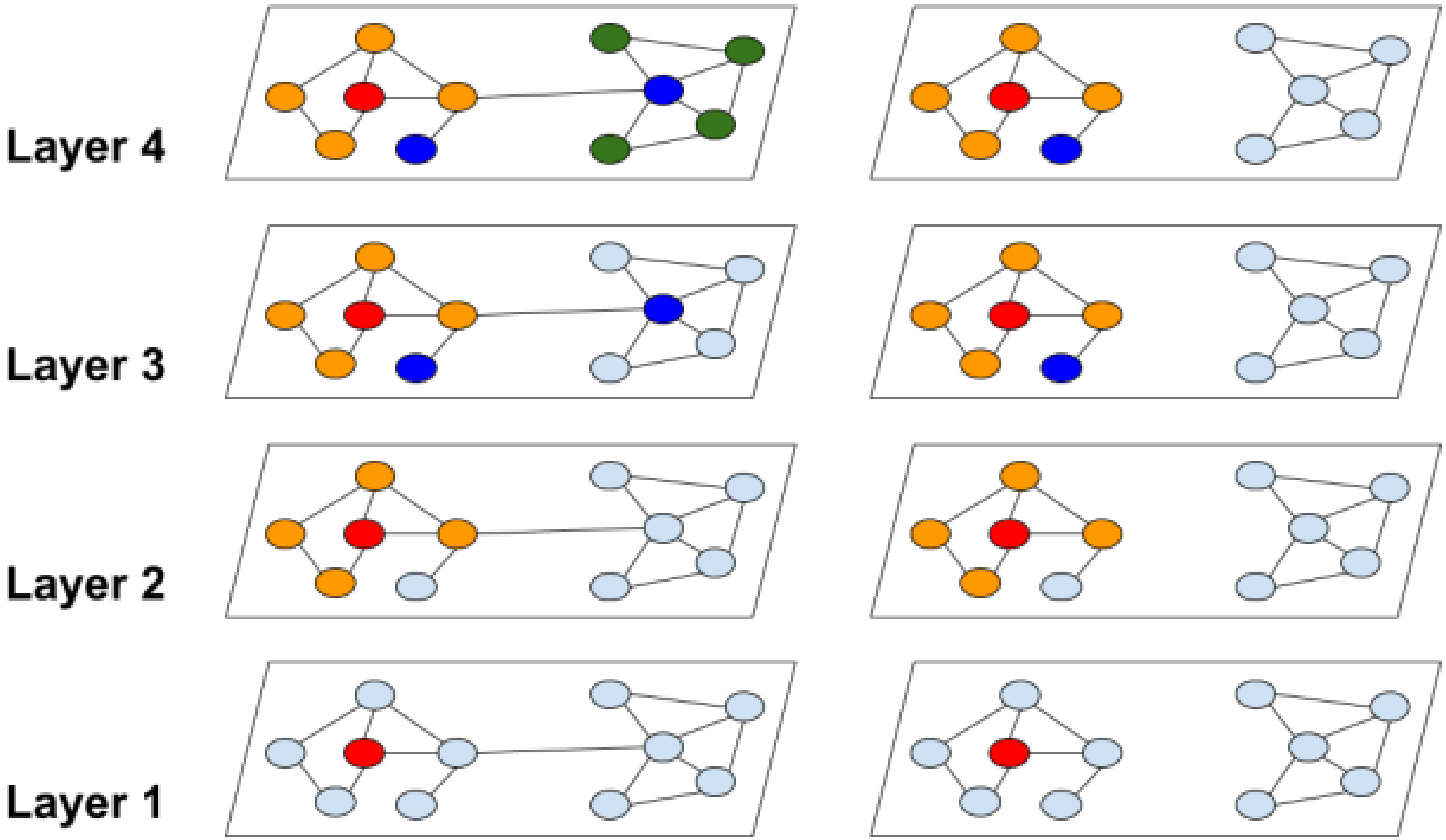


Figure from: https://arxiv.org/pdf/1905.07953.pdf

Message passing layers

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

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

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

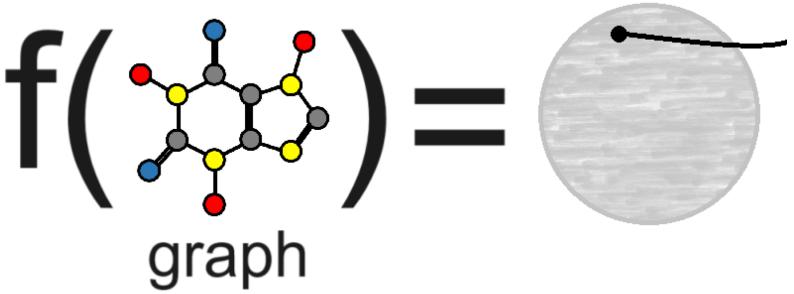
- Edge features can also be updated:

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

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

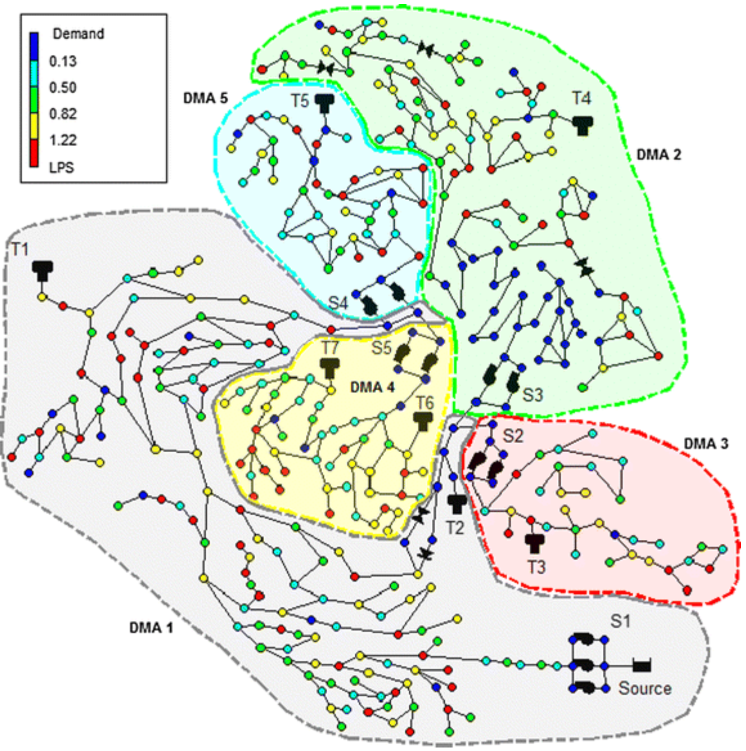
Reference: https://arxiv.org/pdf/1806.01261.pdf

Examples - again



social networks

image source: [Medium]



Water distribution systems

image source: [springer]

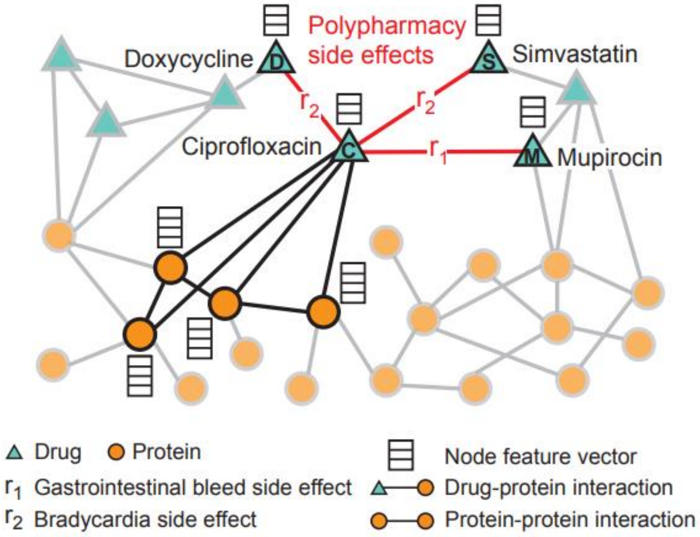
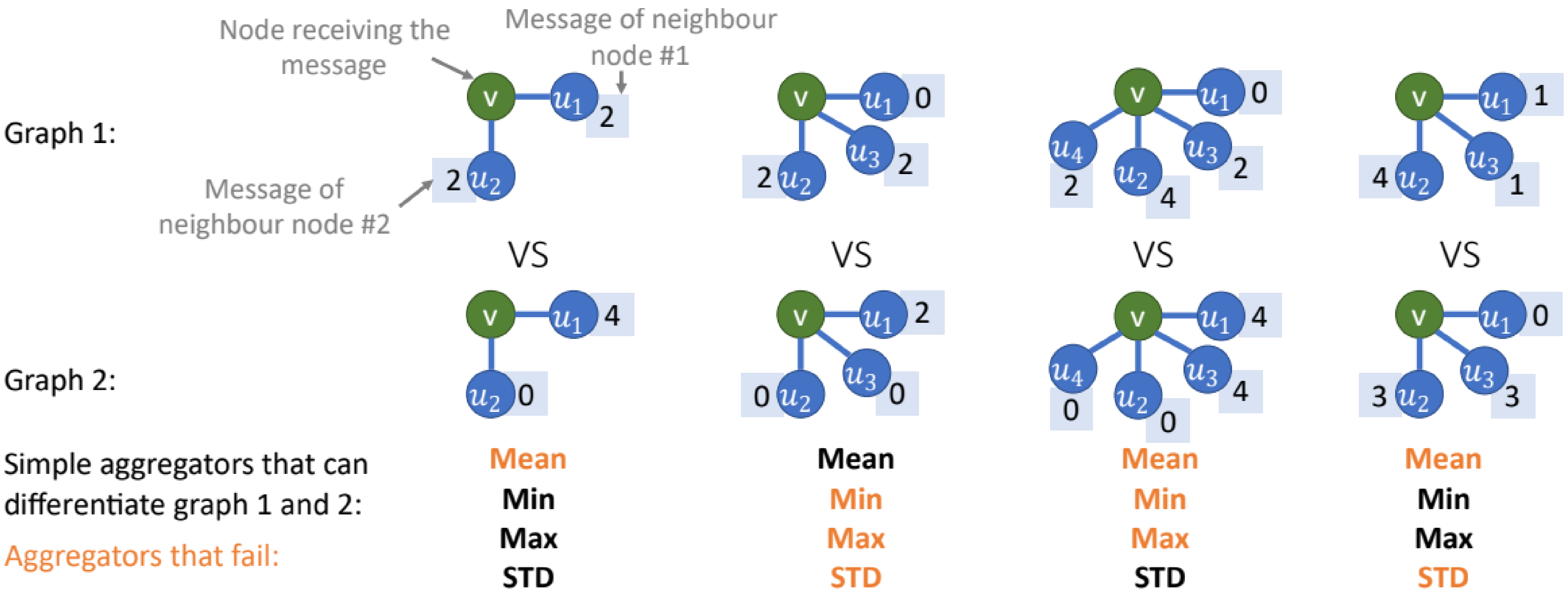


Figure from:
https://academic.oup.com/bioinformatics/article/34/13/i457/5045770

Message passing layers

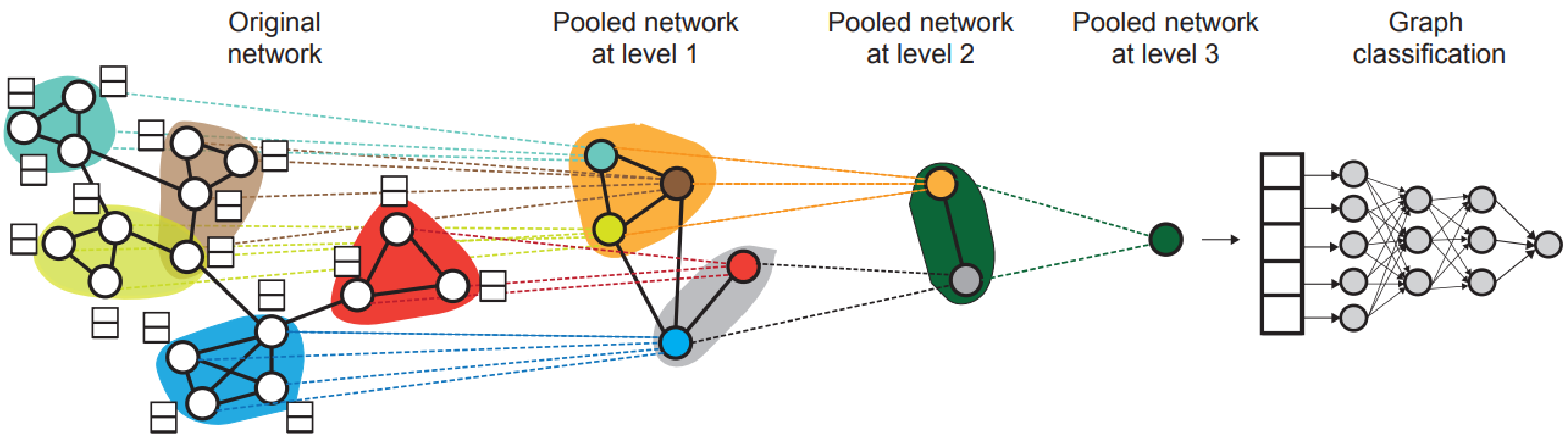
- Other permutation invariant operations instead of summation:



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

Reference: https://arxiv.org/pdf/2004.05718.pdf

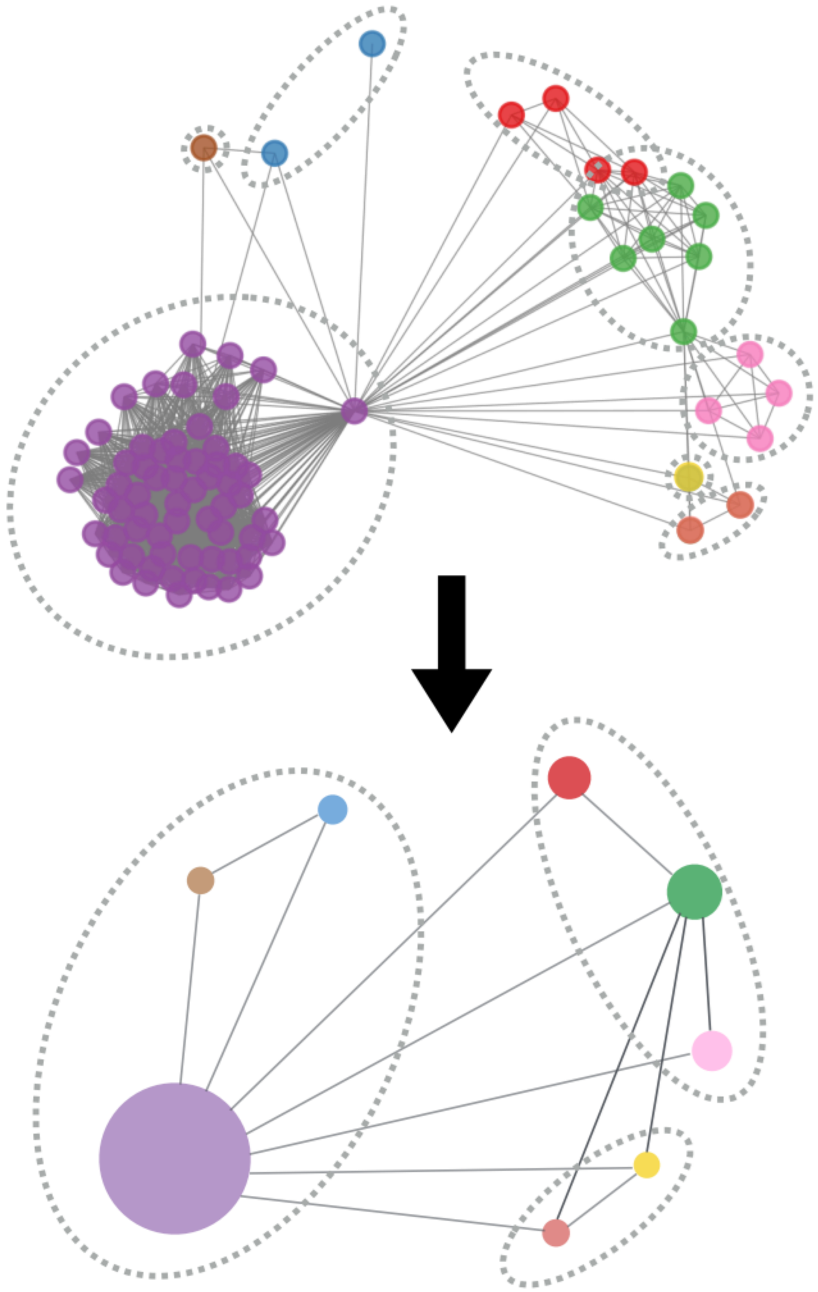
Pooling layers



reference: https://arxiv.org/pdf/1806.08804.pdf

Pooling layers

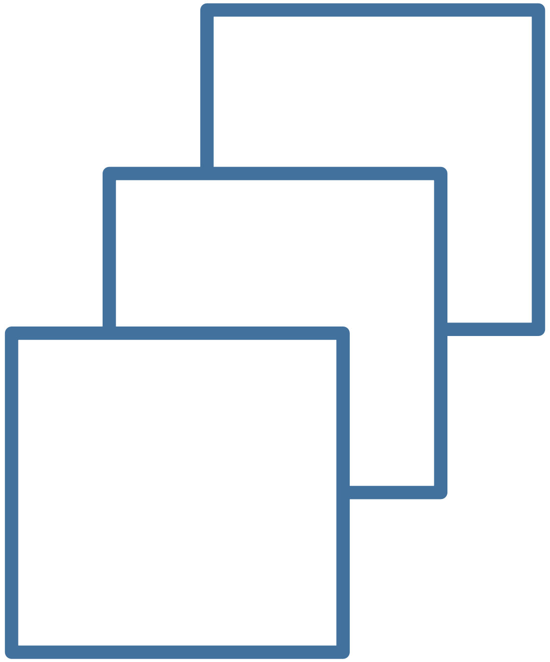
- $Z = GNN(X, A)$ N x F matrix
- $S = softmax\left(GNN(X, A)\right)$ N x M matrix
- "soft cluster assignments" M < N
- $X' = S^T Z$ M x F matrix
- $A' = S^T A S$ M x M matrix



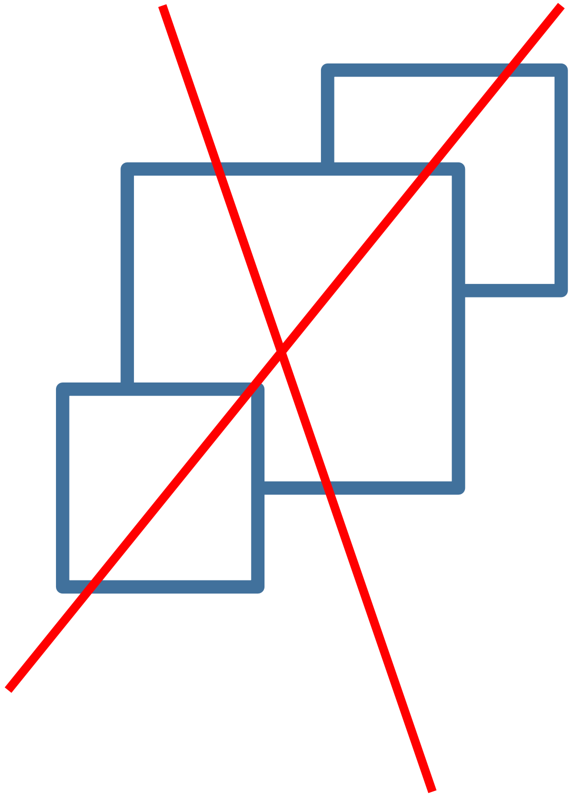
reference: https://arxiv.org/pdf/1806.08804.pdf

Mini-batching

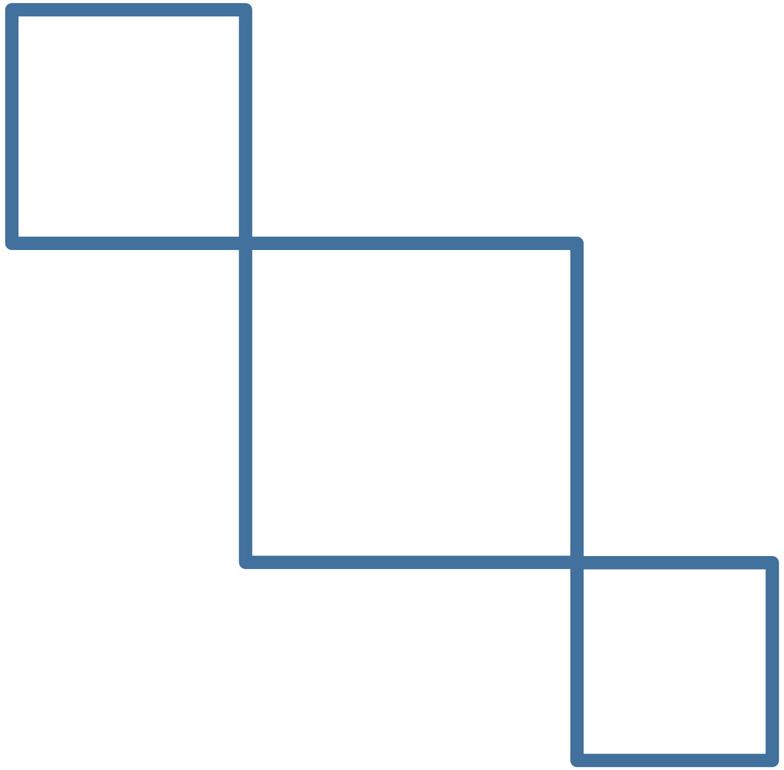
Adjacency matrices,
Graphs with same size



Adjacency matrices,
Graphs with different sizes



Adjacency matrix,
Blockdiagonal



Mini-batching

- Multiple graphs in a mini-batch:

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

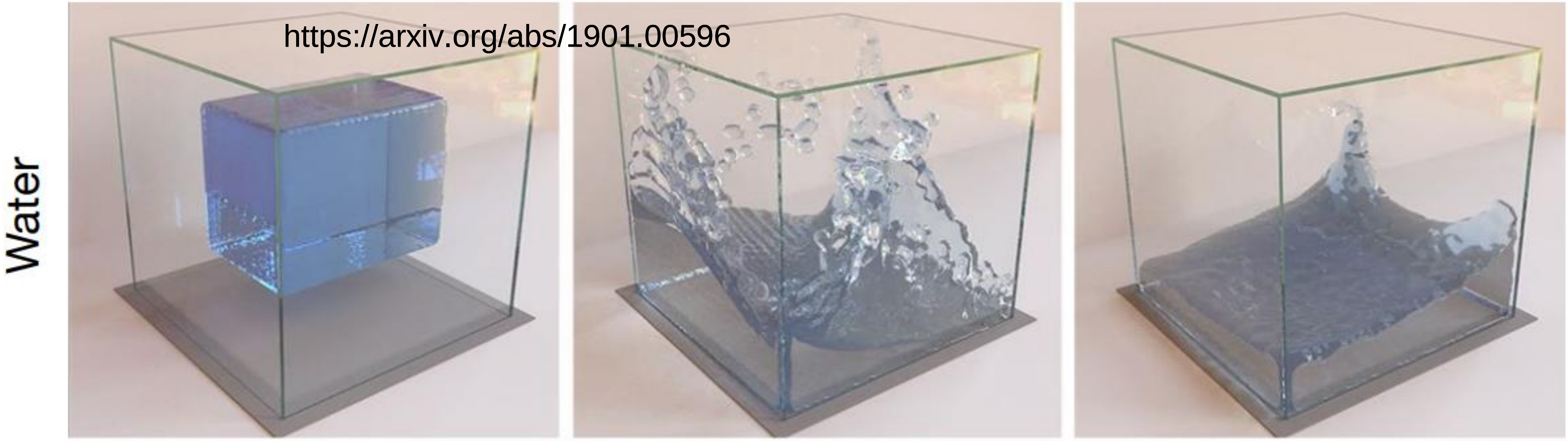
- Very large graphs: a graph partitioning algorithm is used (e.g. METIS), and multiple clusters are randomly selected to create a mini-batch

https://pytorch-geometric.readthedocs.io/en/latest/advanced/batching.html

Another interesting application

Simulation of complex physics with graph neural networks:

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



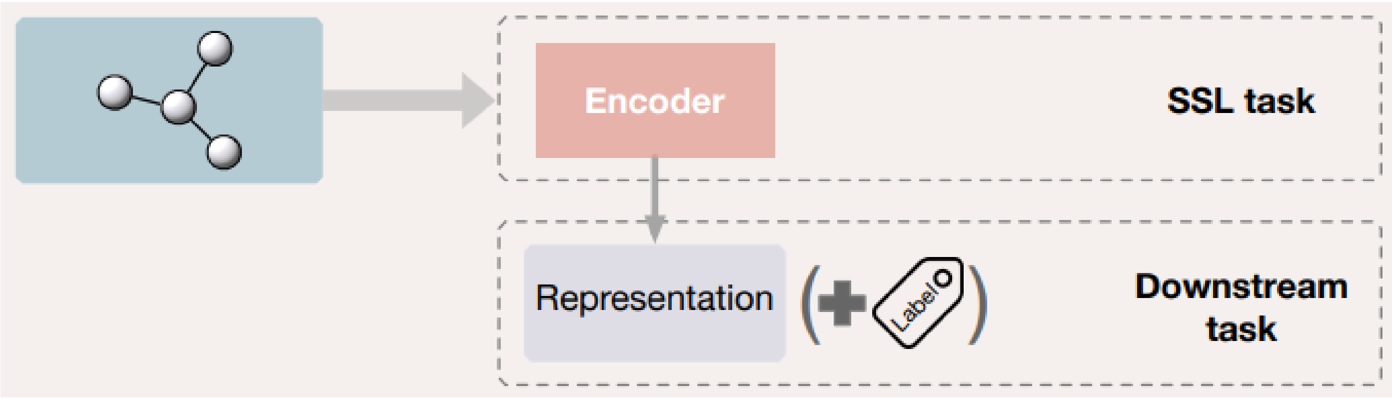
Self-supervised learning

Self-supervised learning

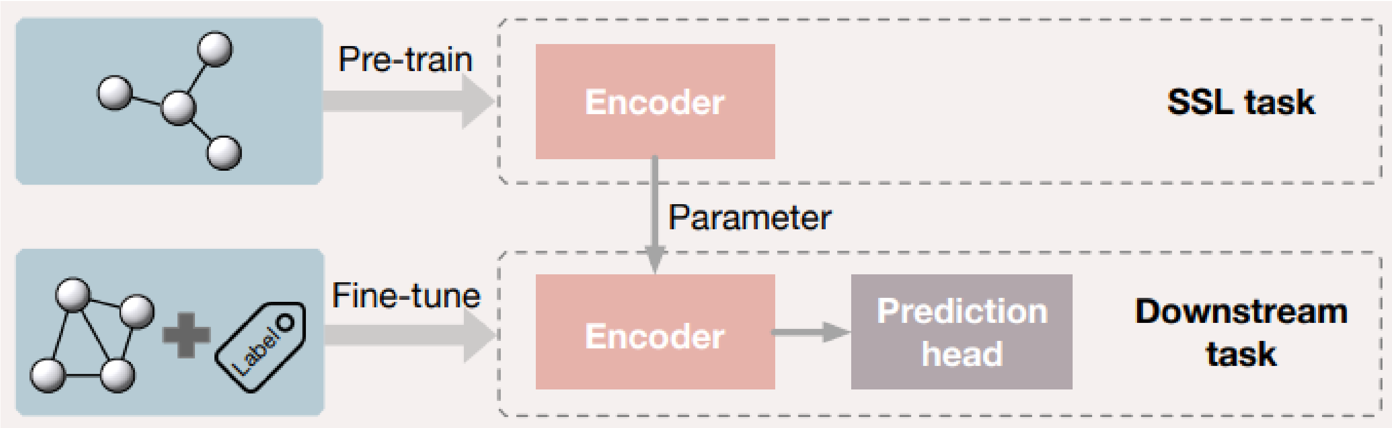
- No pretrained models or foundation models: very different domains → very different graph structures and features
- Pretraining works well <u>within the same domain</u>
- Labeled data is expensive, and often requires domain knowledge
- But networks are everywhere around us → self-supervised learning

Self-supervised learning

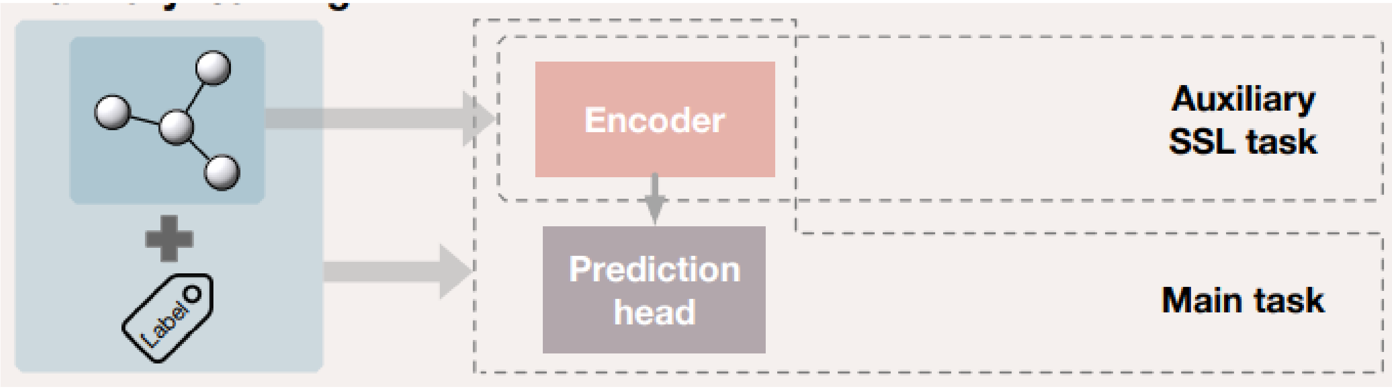
- 1) Unsupervised representation learning



- 2) Unsupervised pretraining:

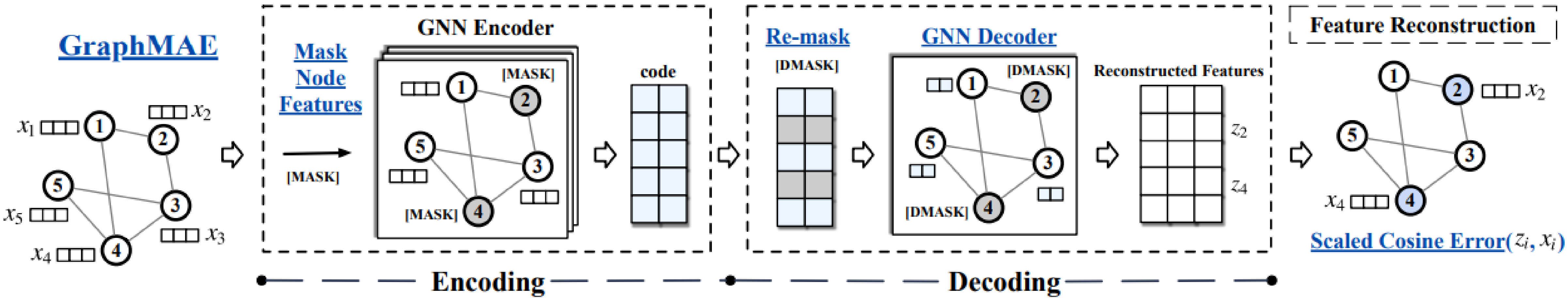


- 3) Auxiliary learning:



Reference & figures: https://arxiv.org/pdf/2102.10757.pdf

Self-supervised learning: an example



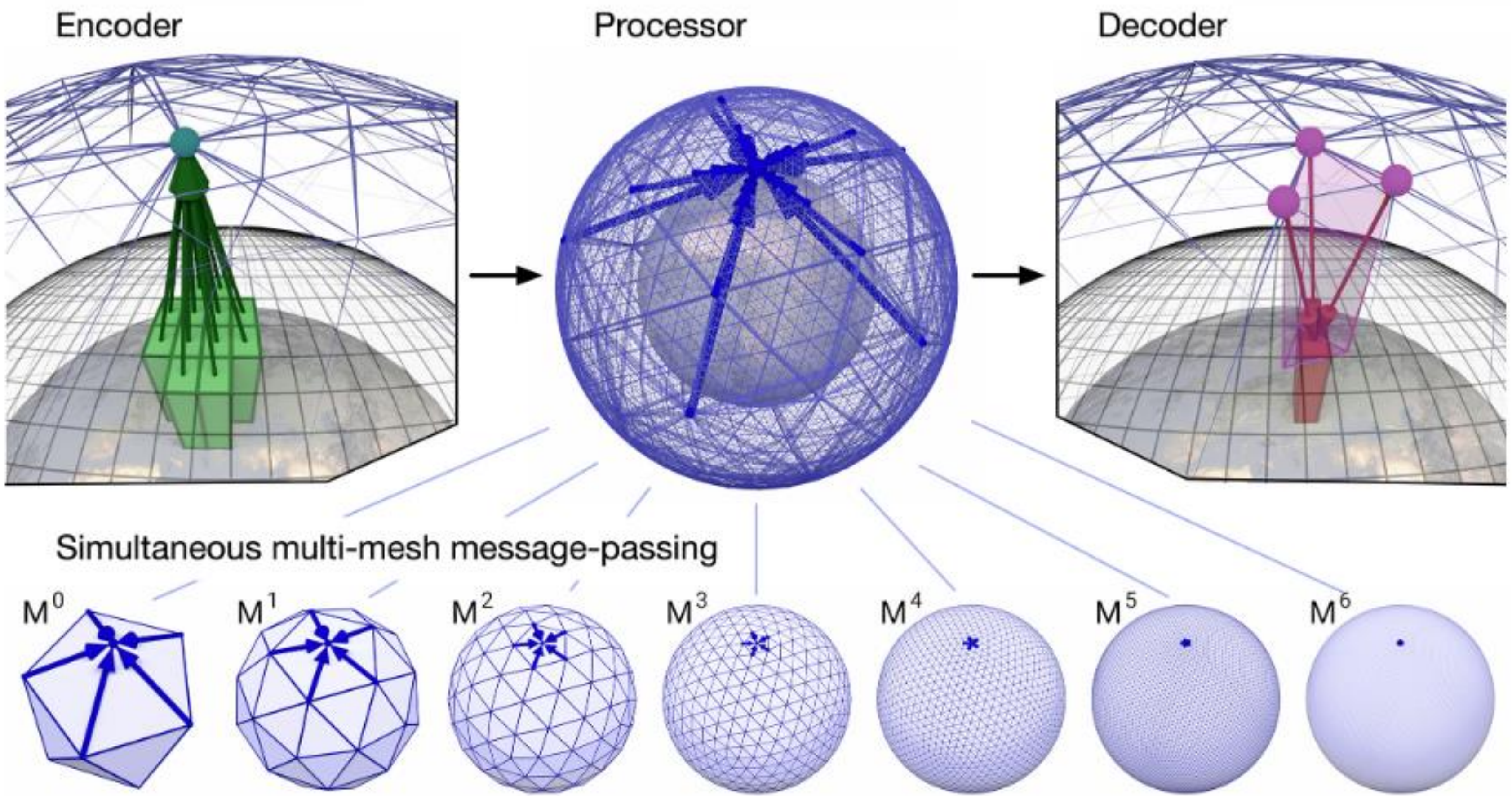
- Features of the selected nodes are masked using a mask token
- The graph is encoded with a GNN encoder
- The codes of the selected nodes are masked using another mask token
- The codes are decoded with a GNN decoder → feature reconstruction loss

Reference & figure: https://arxiv.org/pdf/2205.10803.pdf

Once another interesting application

AI model for faster and more accurate global weather forecasting

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



References

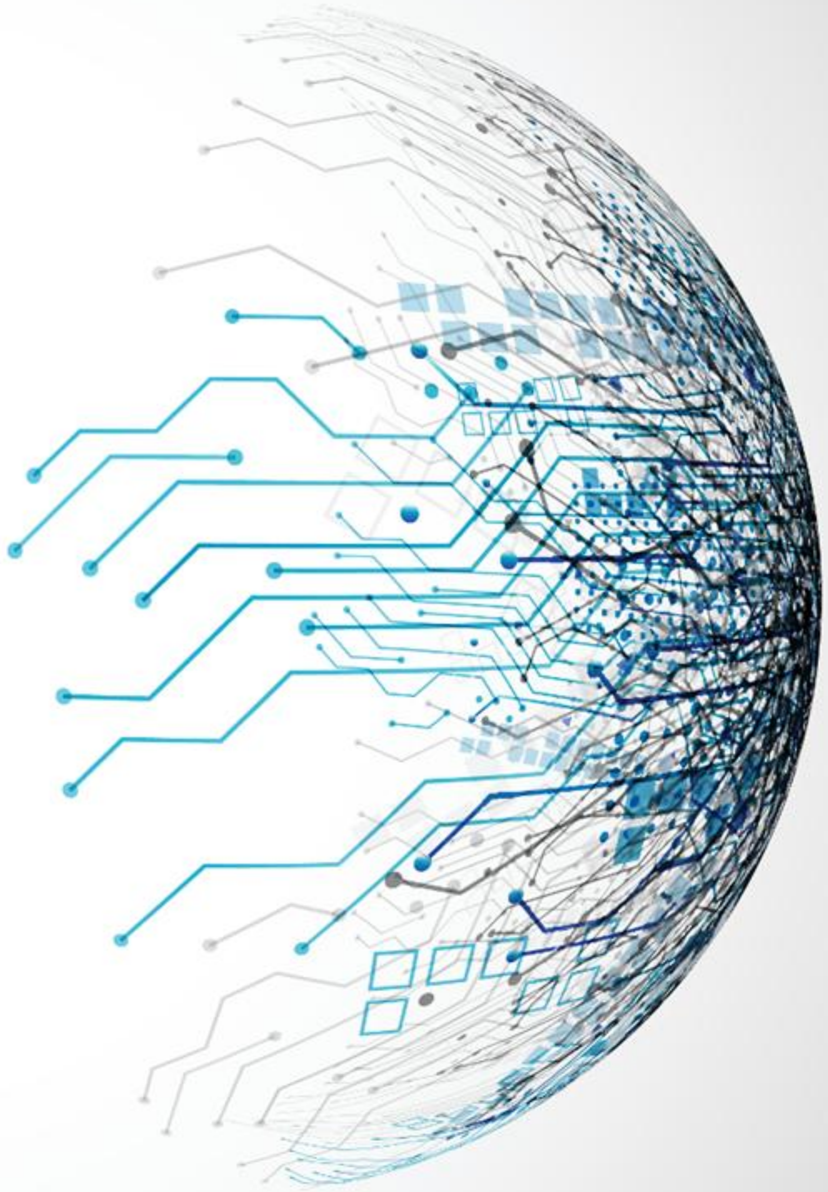
References

- Book
  - William L. Hamilton (2020). Graph representation learning. Morgan & Claypool Publishers.
- Videos
  - https://geometricdeeplearning.com/lectures/
  - https://www.youtube.com/watch?v=bIZB1hIJ4u8
  - https://www.youtube.com/watch?v=jAGIuobLp60
  - 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