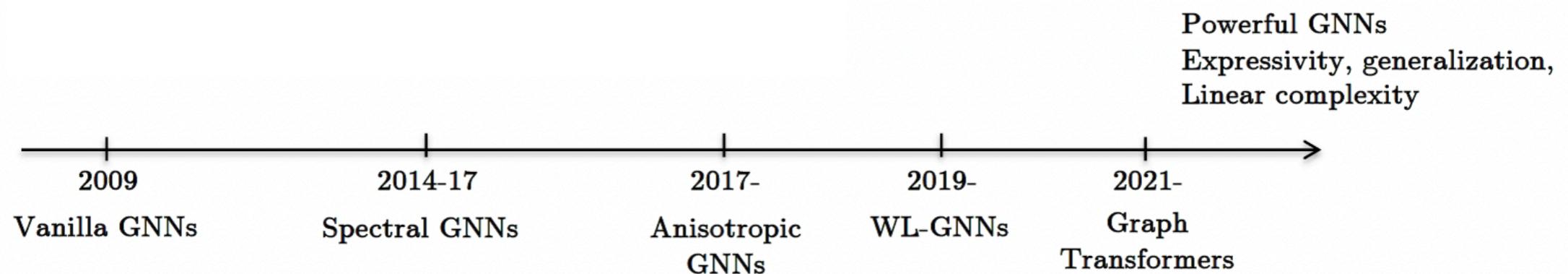


Graph neural networks: Advanced topics & Applications

Dr Dorina Thanou
May 6, 2025

From the last lecture:

- A brief history of GNNs



Outline

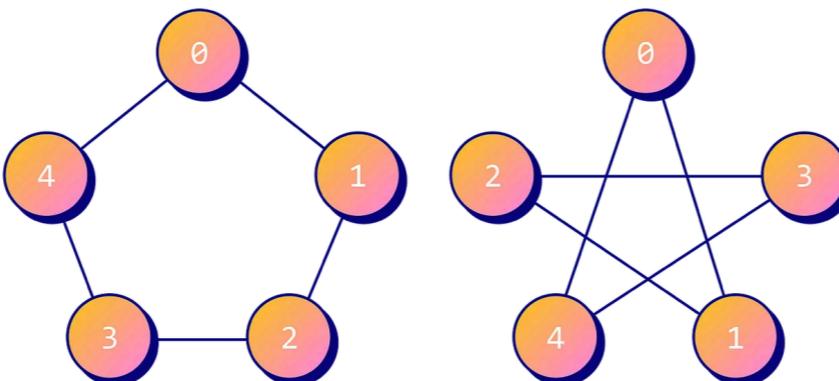
- Expressive power of GNNs
- Inferring the graph topology
- Dynamic graph models
- Learning with sparse labels
- Applications

Outline

- Expressive power of GNNs
- Inferring the graph topology
- Dynamic graph models
- Learning with sparse labels
- Applications

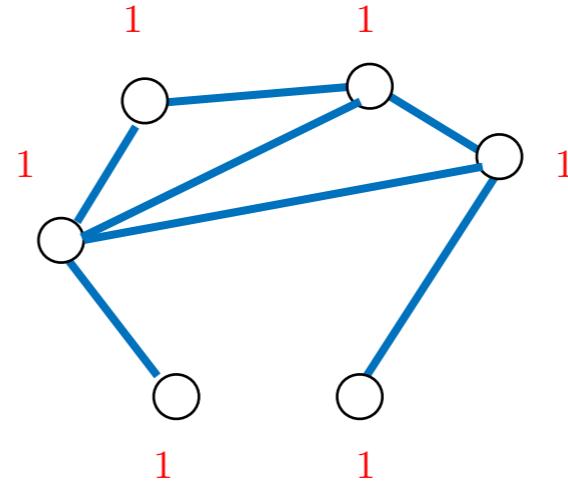
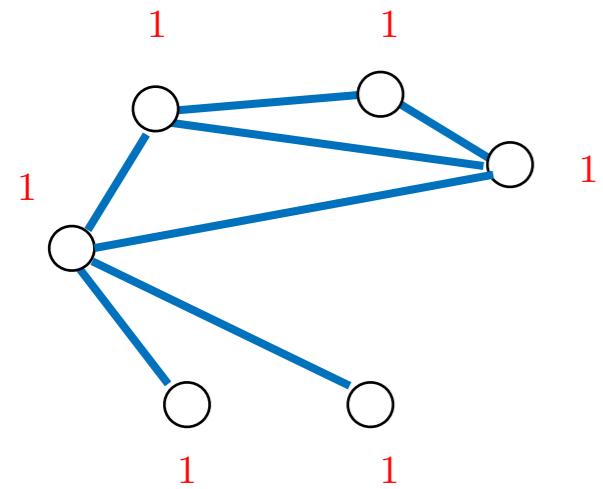
Representation power of MPNNs

- Typically done by analyzing how expressive a GNN is in learning to represent and **distinguish between different graph structures**
- This implies solving the graph isomorphism problem:
 - Two graphs are isomorphic if there exists an index permutation between the nodes that preserves node adjacencies (NP hard)
 - Isomorphic graphs should be mapped to the same representation and non-isomorphic ones to different representations



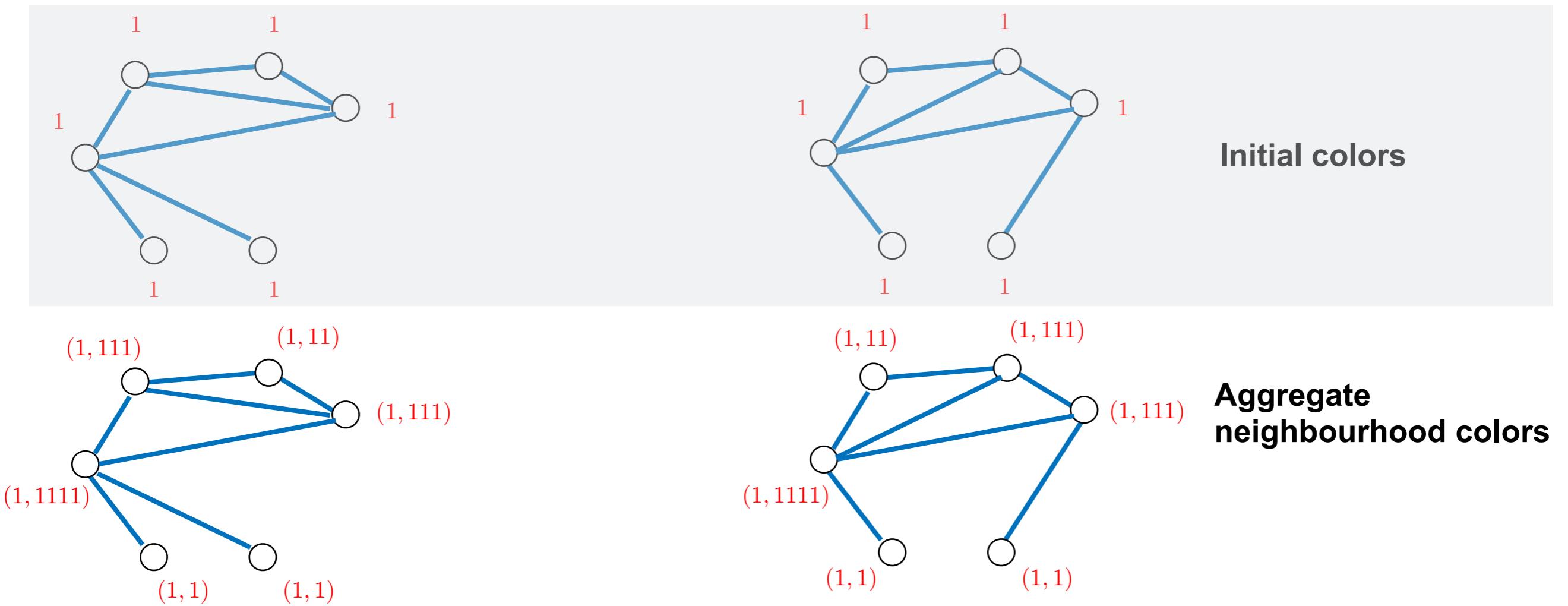
Example of two isomorphic graphs

Recall the WL kernel

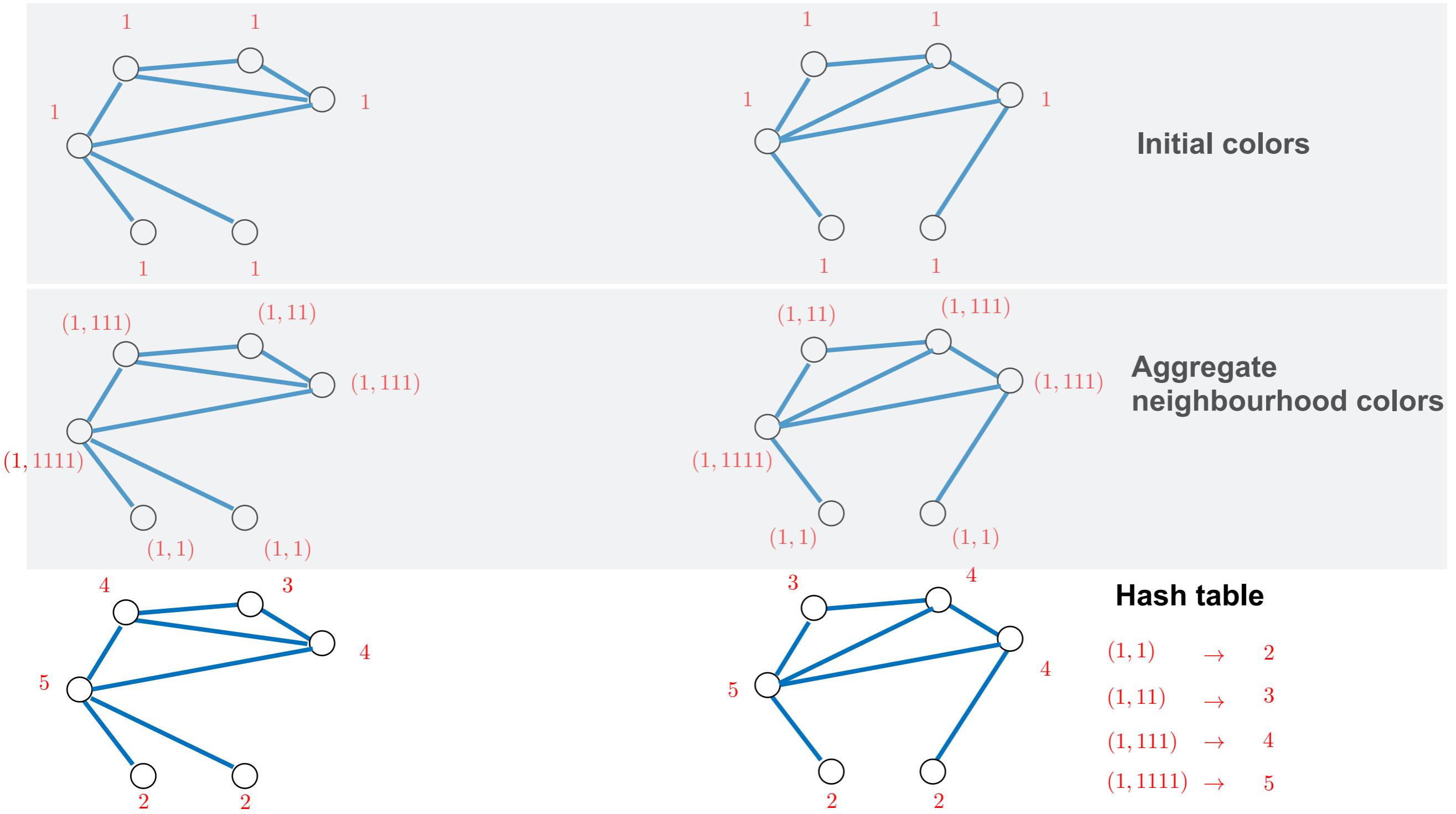


Initial colors

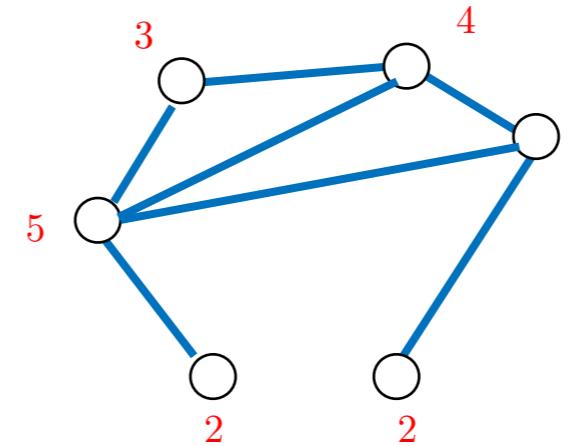
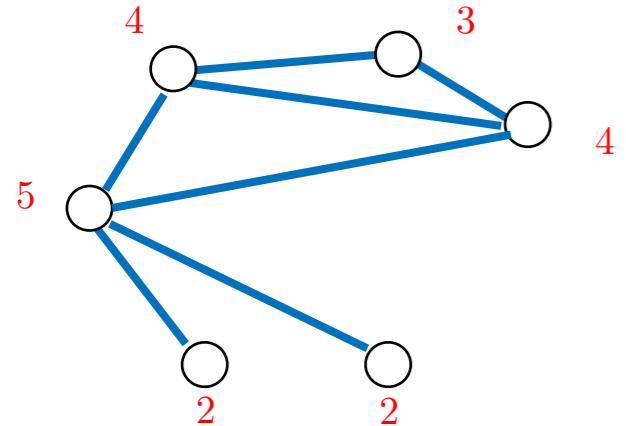
Recall the WL kernel



Recall the WL kernel



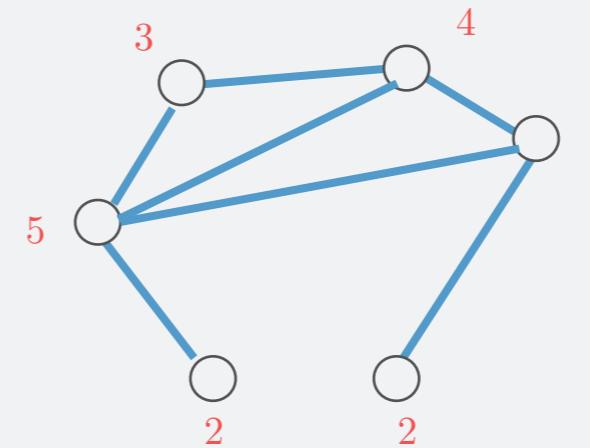
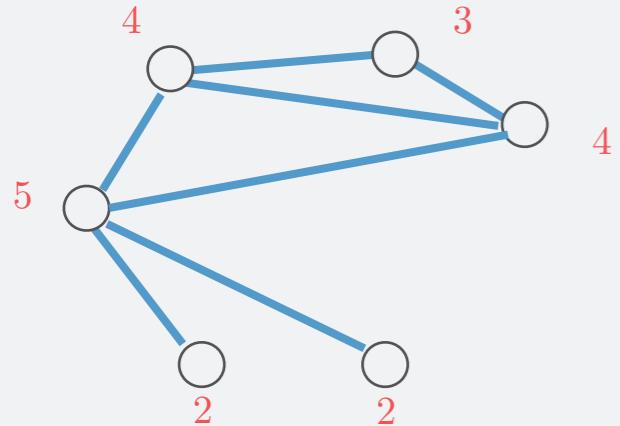
Recall the WL kernel



Hash table

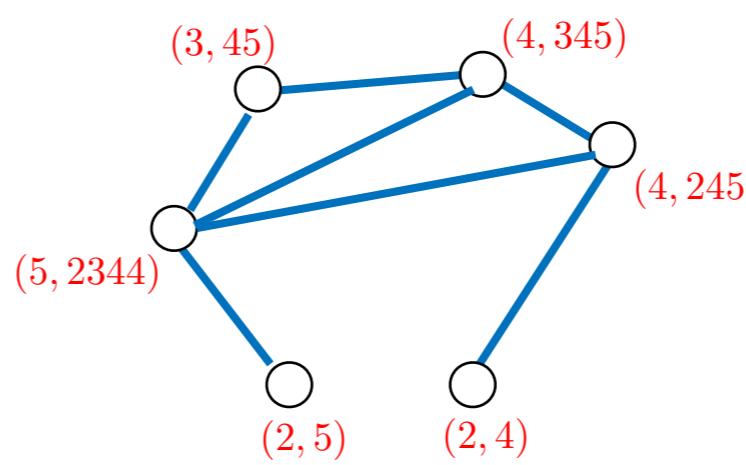
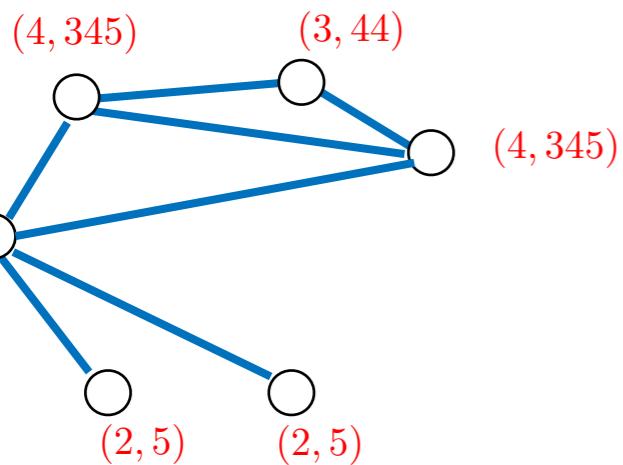
(1, 1)	→	2
(1, 11)	→	3
(1, 111)	→	4
(1, 1111)	→	5

Recall the WL kernel



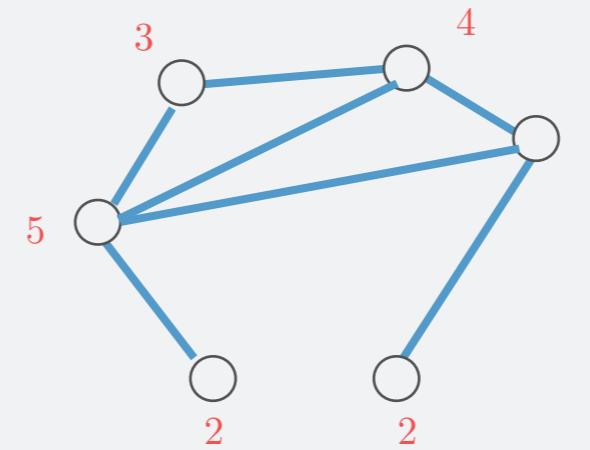
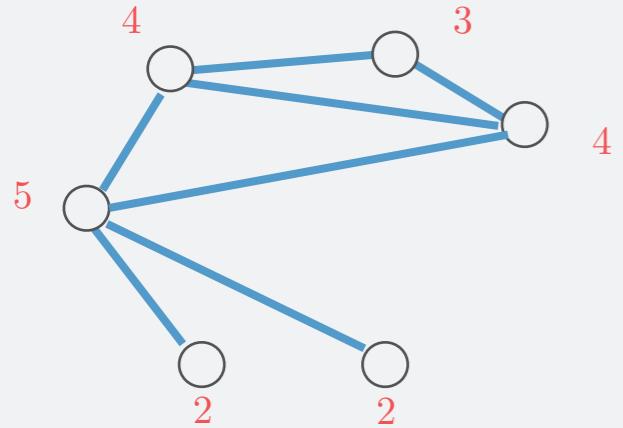
Hash table

(1, 1)	→	2
(1, 11)	→	3
(1, 111)	→	4
(1, 1111)	→	5



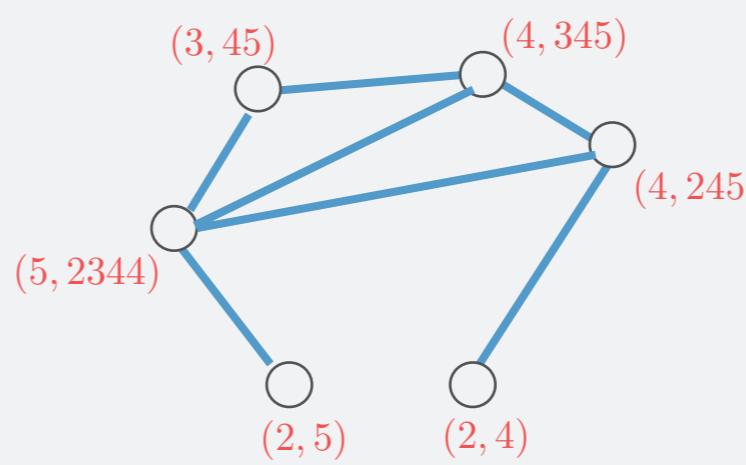
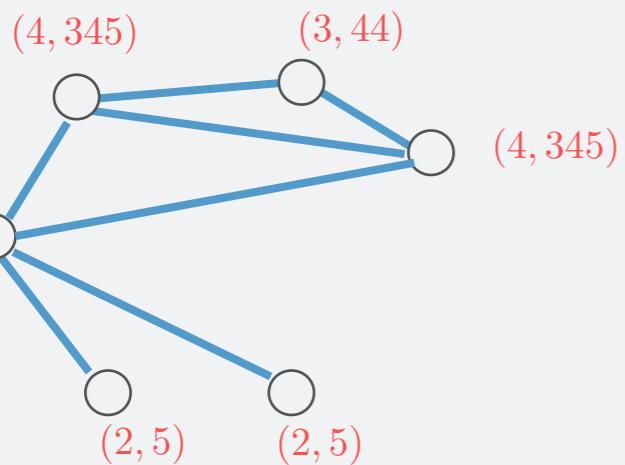
Aggregate neighbourhood colors

Recall the WL kernel



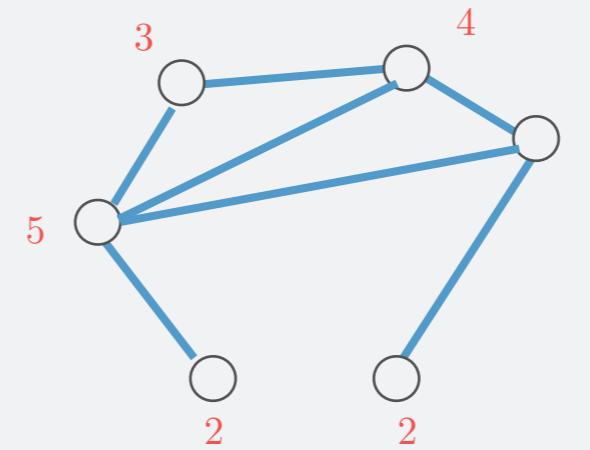
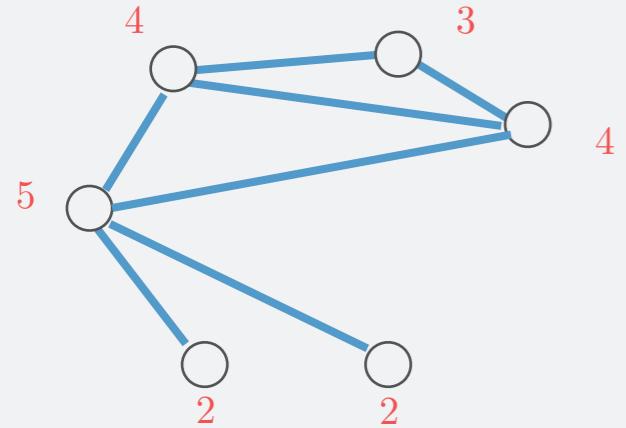
Hash table

(1, 1)	→	2
(1, 11)	→	3
(1, 111)	→	4
(1, 1111)	→	5

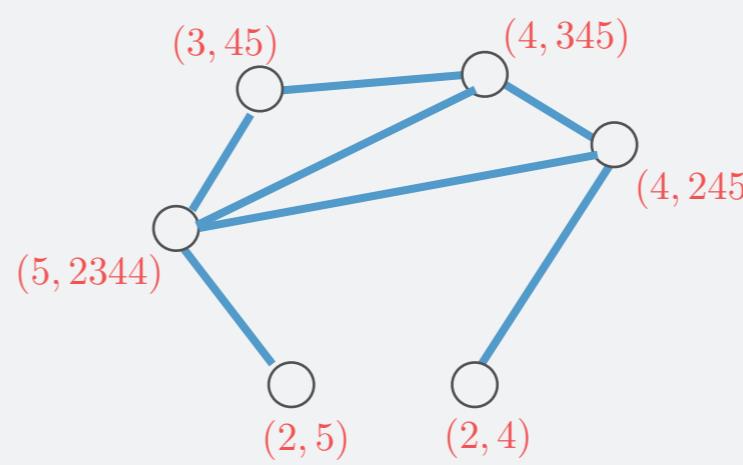
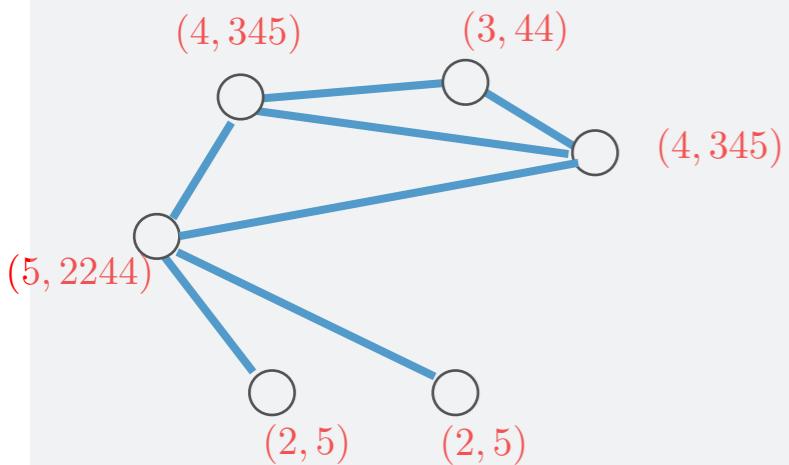


Aggregate neighbourhood colors

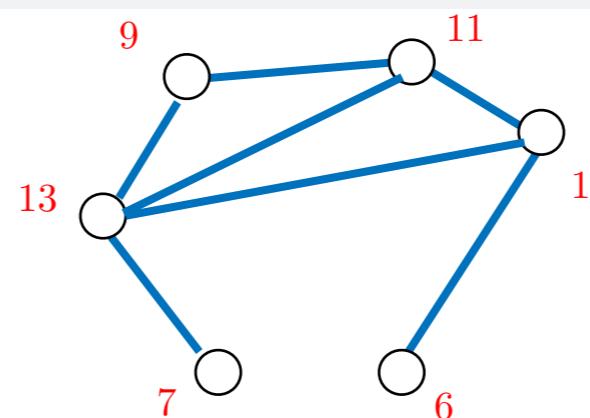
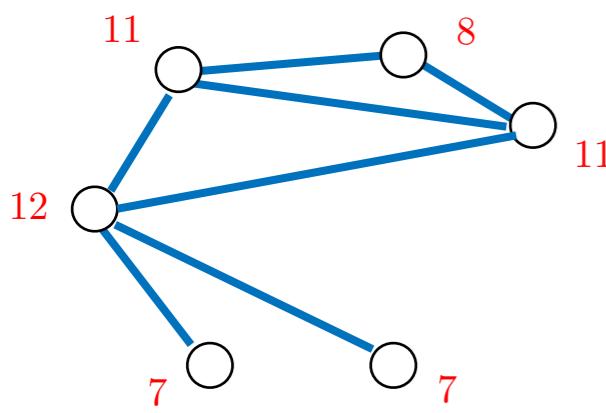
Recall the WL kernel



Hash table	
(1, 1)	→ 2
(1, 11)	→ 3
(1, 111)	→ 4
(1, 1111)	→ 5



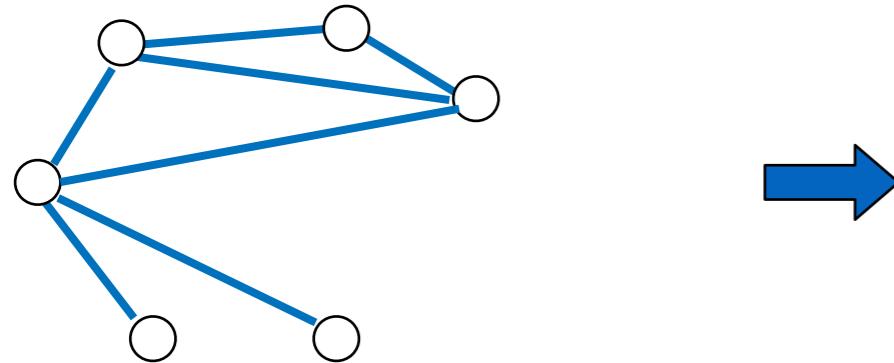
Aggregate neighbourhood colors



Hash table	
(2, 4)	→ 6
(2, 5)	→ 7
(3, 44)	→ 8
(3, 45)	→ 9
(4, 245)	→ 10
(4, 345)	→ 11
(5, 2244)	→ 12
(5, 2344)	→ 13

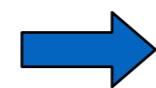
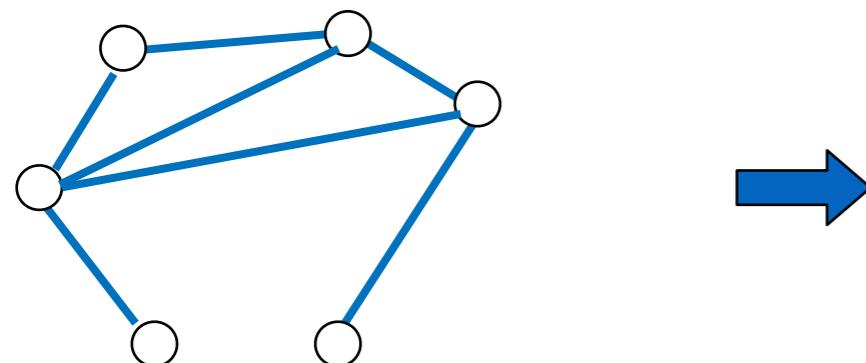
Recall the WL kernel

- After K iterations, the WL kernel computes the histogram of colors



1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

$$\phi(\mathcal{G}_1) = [6, 2, 1, 2, 1, 0, 2, 1, 0, 0, 2, 1, 0]$$



1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

$$\phi(\mathcal{G}_2) = [6, 2, 1, 2, 1, 1, 1, 0, 1, 1, 1, 0, 1]$$

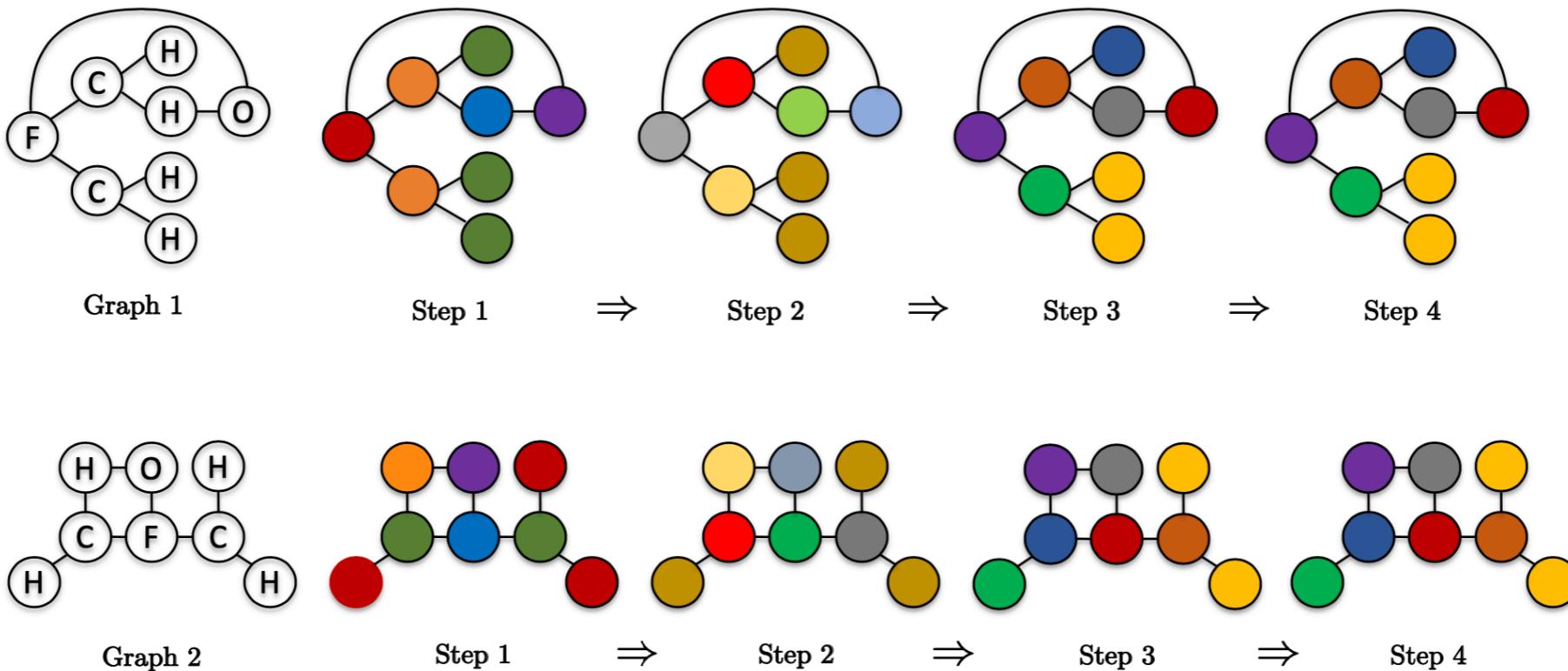


WL kernel

$$K(\mathcal{G}_1, \mathcal{G}_2) = \langle \phi(\mathcal{G}_1), \phi(\mathcal{G}_2) \rangle$$

Strong connections with 1-WL test (1)

- 1-Weisfeiler-Lehman (WL) is a classical algorithm for graph isomorphism
- It tells you if two graphs are not isomorphic, but it does not allow you to conclude if they are isomorphic (necessary but not sufficient condition)

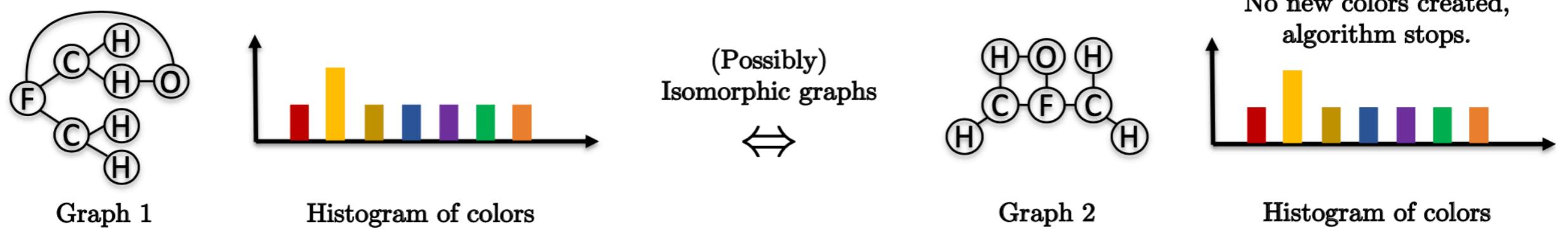


- By construction, GNNs can only be as powerful as the 1-WL test

[Illustrative example from X. Bresson]

Strong connections with 1-WL test (1)

- 1-Weisfeiler-Lehman (WL) is a classical algorithm for graph isomorphism
- It tells you if two graphs are not isomorphic, but it does not allow you to conclude if they are isomorphic (necessary but not sufficient condition)



- By construction, GNNs can only be as powerful as the 1-WL test

[Illustrative example from X. Bresson]

Strong connections with 1-WL test (2)

- MPNNs are equivalent to (at most as powerful as) the 1-WL isomorphism test
 - WL function should be injective: different inputs are mapped to different outputs
 - GINs are as powerful as the 1-WL test

1-WL test

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial color c_i^0 (e.g., node degree) to each node i of \mathcal{V}
- For each iteration $l + 1$ refine node colors as
$$c_i^{l+1} = \text{HASH}(\{c_i^l, \{c_j^l\}_{j \in \mathcal{N}_i}\})$$
- Until stable node coloring is reached
- **Output:** The node colors $\{c_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

MPNN

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial embedding h_i^0 (e.g., node attribute) to each node i of \mathcal{V}
- For each layer $l + 1$ refine node embeddings as
$$h_i^{l+1} = U_l(h_i^l, \bigoplus_{j \in \mathcal{N}_i} M_l(h_j^l, h_i^l))$$
- Until maximum number of layers is reached
- **Output:** The node embeddings $\{h_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

- Over discrete features, GNNs can only be as powerful as the 1-WL test

[Xu et al., How powerful are graph neural networks, ICLR 2019]

Strong connections with 1-WL test (2)

- MPNNs are equivalent to (at most as powerful as) the 1-WL isomorphism test
 - WL function should be injective: different inputs are mapped to different outputs
 - GINs are as powerful as the 1-WL test

1-WL test

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial color c_i^0 (e.g., node degree) to each node i of \mathcal{V}
- For each iteration $l + 1$ refine node colors as
$$c_i^{l+1} = \text{HASH}(\{c_i^l, \{c_j^l\}_{j \in \mathcal{N}_i}\})$$
- Until stable node coloring is reached
- **Output:** The node colors $\{c_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

MPNN

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial embedding h_i^0 (e.g., node attribute) to each node i of \mathcal{V}
- For each layer $l + 1$ refine node embeddings as
$$h_i^{l+1} = U_l(h_i^l, \bigoplus_{j \in \mathcal{N}_i} M_l(h_j^l, h_i^l))$$
- Until maximum number of layers is reached
- **Output:** The node embeddings $\{h_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

- Over discrete features, GNNs can only be as powerful as the 1-WL test

[Xu et al., How powerful are graph neural networks, ICLR 2019]

Strong connections with 1-WL test (2)

- MPNNs are equivalent to (at most as powerful as) the 1-WL isomorphism test
 - WL function should be injective: different inputs are mapped to different outputs
 - GINs are as powerful as the 1-WL test

1-WL test

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial color c_i^0 (e.g., node degree) to each node i of \mathcal{V}
- For each iteration $l + 1$ refine node colors as
$$c_i^{l+1} = \text{HASH}\left(\{c_i^l, \{c_j^l\}_{j \in \mathcal{N}_i}\}\right)$$
- Until stable node coloring is reached
- **Output:** The node colors $\{c_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

MPNN

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial embedding h_i^0 (e.g., node attribute) to each node i of \mathcal{V}
- For each layer $l + 1$ refine node embeddings as
$$h_i^{l+1} = U_l\left(h_i^l, \bigoplus_{j \in \mathcal{N}_i} M_l(h_j^l, h_i^l)\right)$$
- Until maximum number of layers is reached
- **Output:** The node embeddings $\{h_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

- Over discrete features, GNNs can only be as powerful as the 1-WL test

[Xu et al., How powerful are graph neural networks, ICLR 2019]

Strong connections with 1-WL test (2)

- MPNNs are equivalent to (at most as powerful as) the 1-WL isomorphism test
 - WL function should be injective: different inputs are mapped to different outputs
 - GINs are as powerful as the 1-WL test

1-WL test

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial color c_i^0 (e.g., node degree) to each node i of \mathcal{V}
- For each iteration $l + 1$ refine node colors as
$$c_i^{l+1} = \text{HASH}\left(\{c_i^l, \{c_j^l\}_{j \in \mathcal{N}_i}\}\right)$$
- Until stable node coloring is reached
- **Output:** The node colors $\{c_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

MPNN

- **Input:** a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$
- Assign an initial embedding h_i^0 (e.g., node attribute) to each node i of \mathcal{V}
- For each layer $l + 1$ refine node embeddings as
$$h_i^{l+1} = U_l\left(h_i^l, \bigoplus_{j \in \mathcal{N}_i} M_l(h_j^l, h_i^l)\right)$$
- Until maximum number of layers is reached
- **Output:** The node embeddings $\{h_i^{l_{max}}\}_{i=\{1,2,\dots,N\}}$

- Over discrete features, GNNs can only be as powerful as the 1-WL test

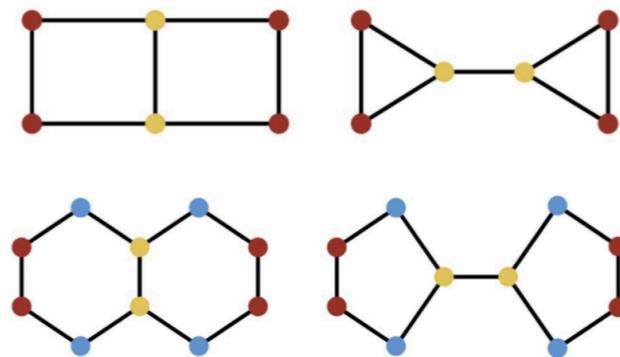
[Xu et al., How powerful are graph neural networks, ICLR 2019]

Limitations of MPNNs

- A lot of works have shown that MPNNs have some limitations, e.g., [Morris'19, Xu'18, Chen'20]
 - They fail to capture long run interactions



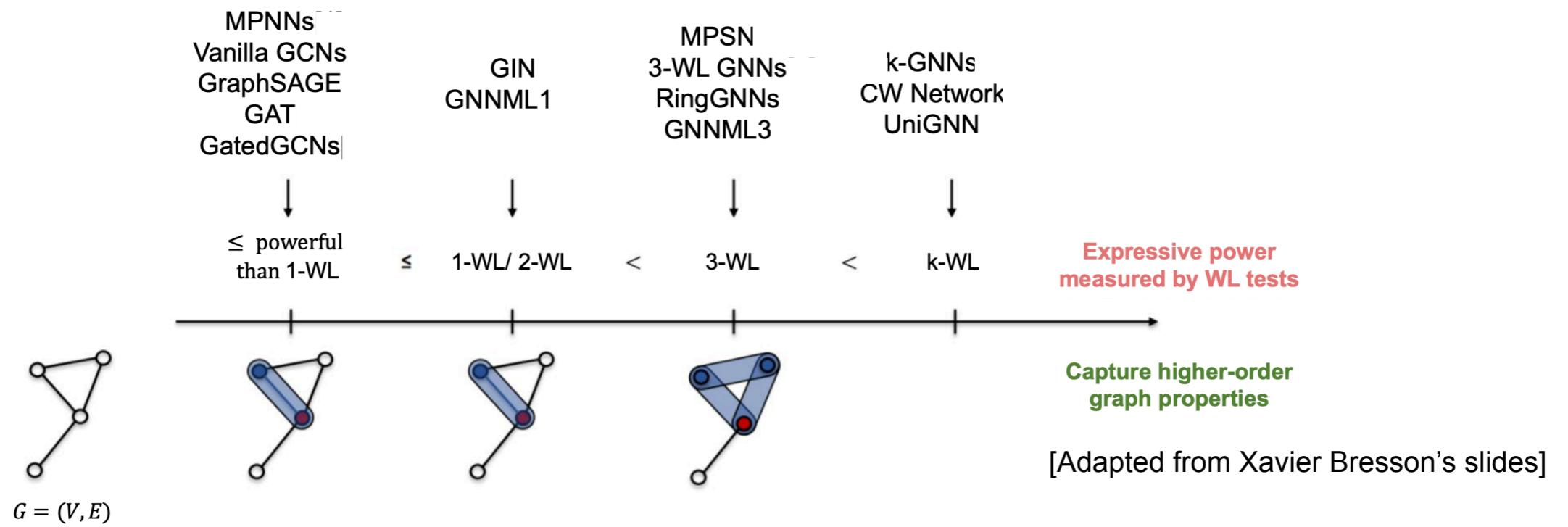
- They fail to distinguish higher-order structures (similar to WL)



- There are non-isomorphic subgraphs that are considered equivalent by MPNNs

Improvements of expressiveness of MPNNs

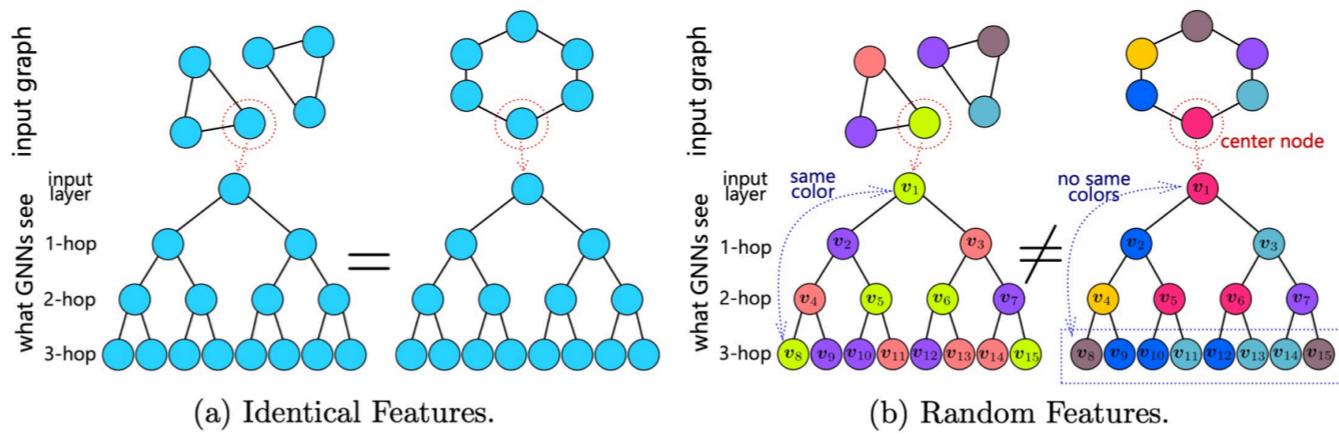
- **Intuition:** Design algorithms inspired from higher-dimensional isomorphism tests (k-WL induced GNNs)



- ✓ These models are more expressive
 - They (often) loose the advantage of locality and linear complexity of MPNNs

Improvements of expressiveness of MPNNs

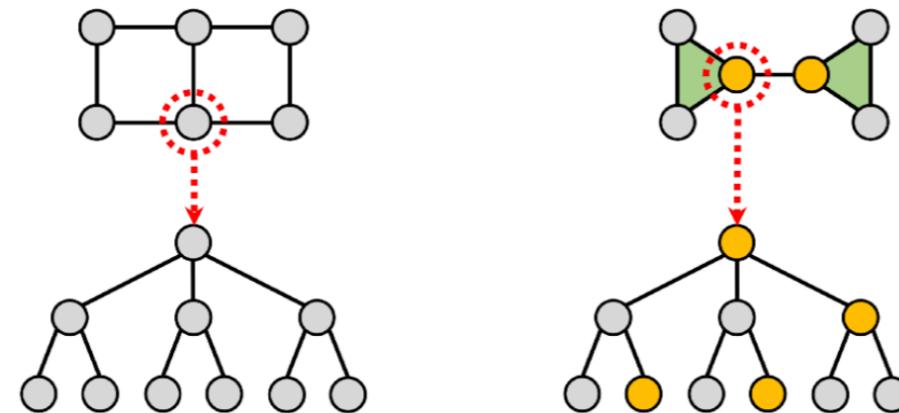
- **Intuition:** Exploit the identity of each node or a neighborhood around it to increase the expressive power of MPNNs
 - Augment nodes with randomized/positional features or local context (positional encoding)
 - Examples: [Loukas'20, Vignac'20, Sato'21, Abboud'21]



- ✓ These models are more expressive
 - They are either not permutation equivariant or computationally costly

Improvements of expressiveness of MPNNs

- **Intuition:** If the network cannot detect a pattern, we can count this pattern and add the extra count as an additional feature
 - Augment nodes with handcrafted subgraph-based features (structural encoding)
 - Examples: [Bouritsas'20]



- ✓ These models are at least as expressive as 1-WL and they can distinguish some non-isomorphic graphs that 1-WL fails
 - They require expert knowledge on what features are relevant for a task

Still MPNNs remain the most widely-used framework in practice!

Outline

- Expressive power of GNNs
- **Inferring the graph topology**
- Dynamic graph models
- Learning with sparse labels
- Applications

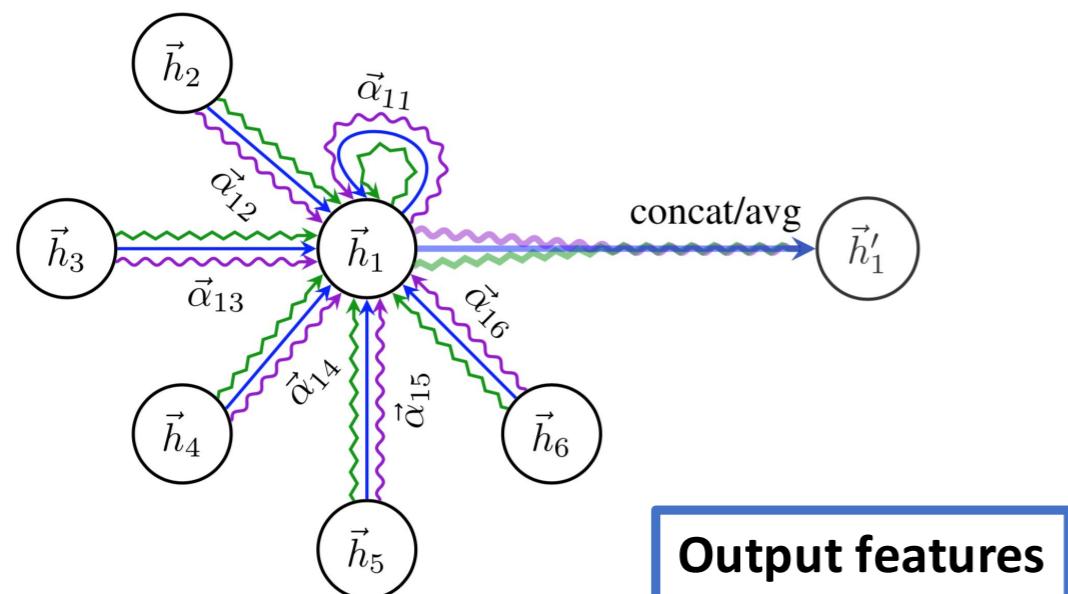
What if the graph is not given

- All techniques so far require a graph to be provided as input
- In many applications, the graph is not given
- A fully connected graph could be designed, but:
 - It does not capture important interactions between nodes
 - It does not scale

How can we learn a graph from data in a GNN?

Graph attention can be considered as an example of graph inference

- For a given connectivity matrix, attentional GNNs learn the weights of each edge
 - Different weights are attributed to different nodes in a neighbourhood
 - Dependence on the global graph structure can be removed



Updated embedding

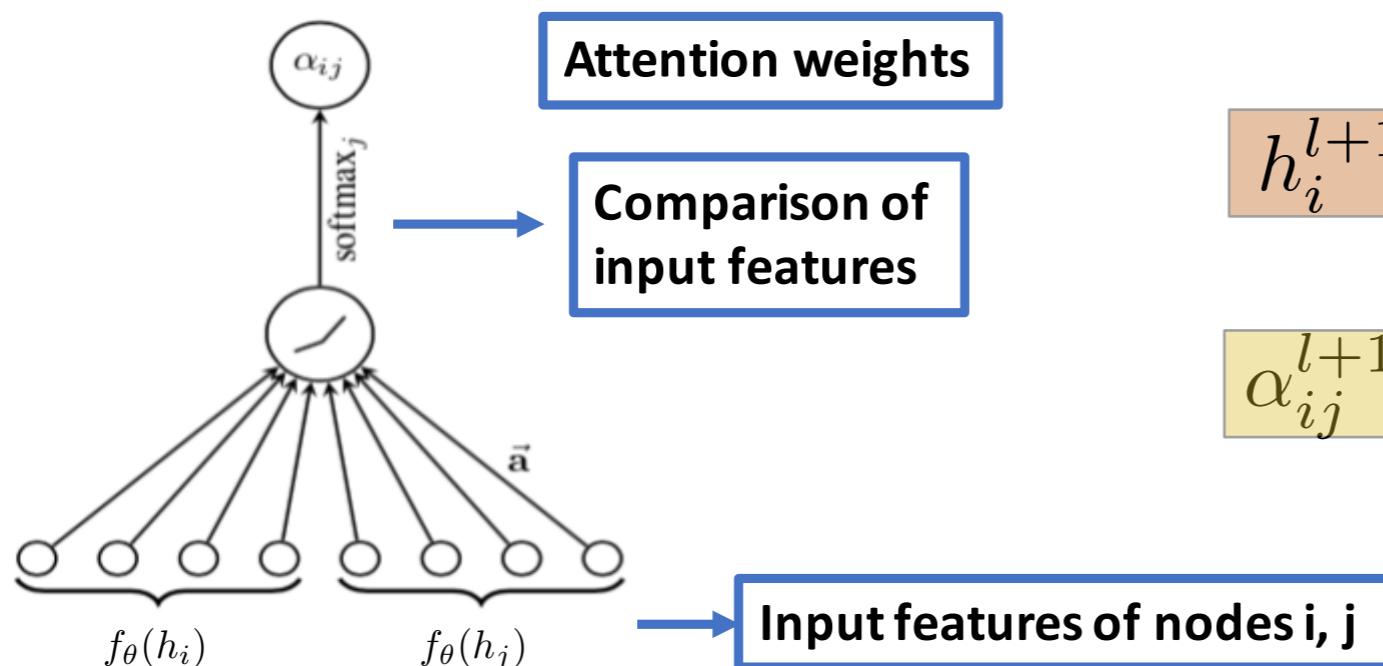
$$\vec{h}_i^{l+1} = \sigma(\alpha_{ii}^l \theta_0^l \vec{h}_i^l + \sum_{j \in \mathcal{N}_i} \alpha_{ij}^l \theta_1^l \vec{h}_j^l)$$

Output features

[Velickovic et al., Graph Attention Networks, ICLR 2018]

Graph attention can be considered as an example of graph inference

- For a given connectivity matrix, attentional GNNs learn the weights of each edge
 - Different weights are attributed to different nodes in a neighbourhood
 - Dependence on the global graph structure can be removed



Updated embedding

$$h_i^{l+1} = \sigma(\alpha_{ii}^l \theta_0^l h_i^l + \sum_{j \in \mathcal{N}_i} \alpha_{ij}^l \theta_1^l h_j^l)$$

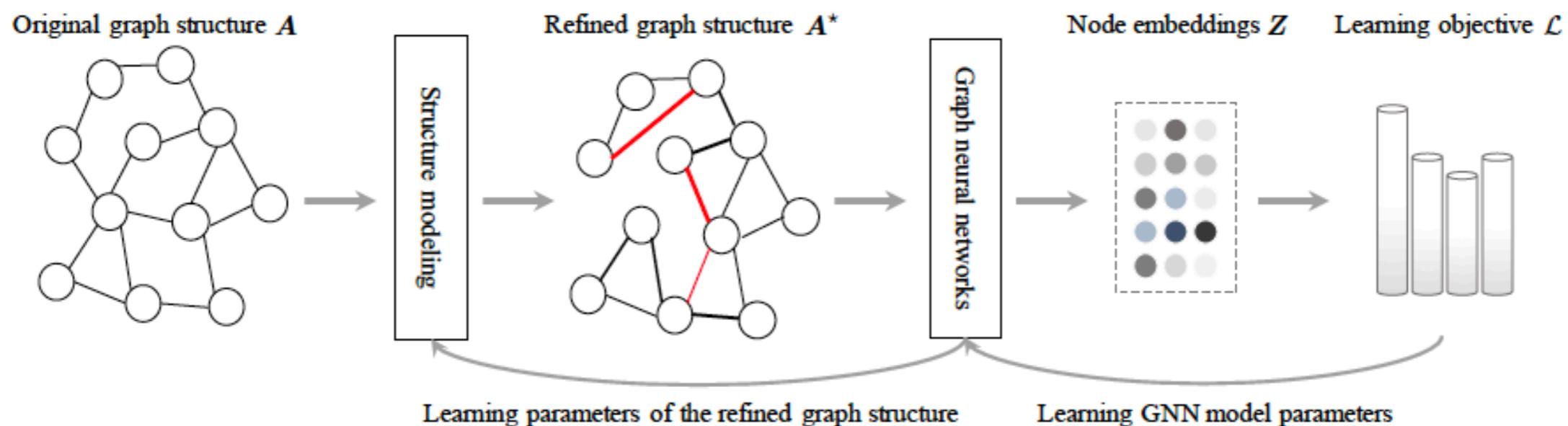
$$\alpha_{ij}^{l+1} = \alpha(f_\theta(h_i^{l+1}), f_\theta(h_j^{l+1}))$$

Updated edge attention

[Velickovic et al., Graph Attention Networks, ICLR 2018]

Learning the connectivity

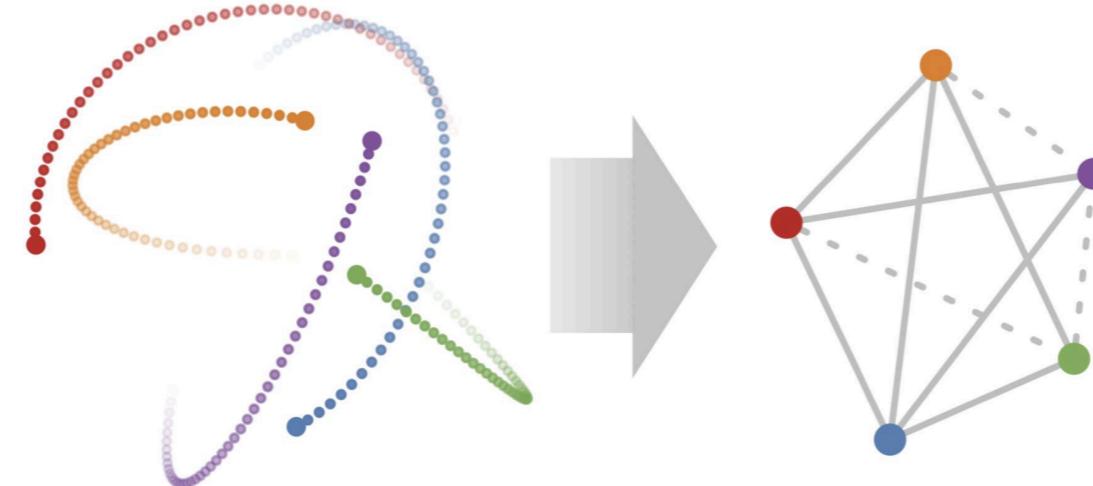
- We can infer simultaneously the graph structure and the node representations by optimising a downstream task



[Zhu et al., Deep Graph Structure Learning for Robust Representations: A Survey, arXiv, 2021]

Example: Neural relational inference

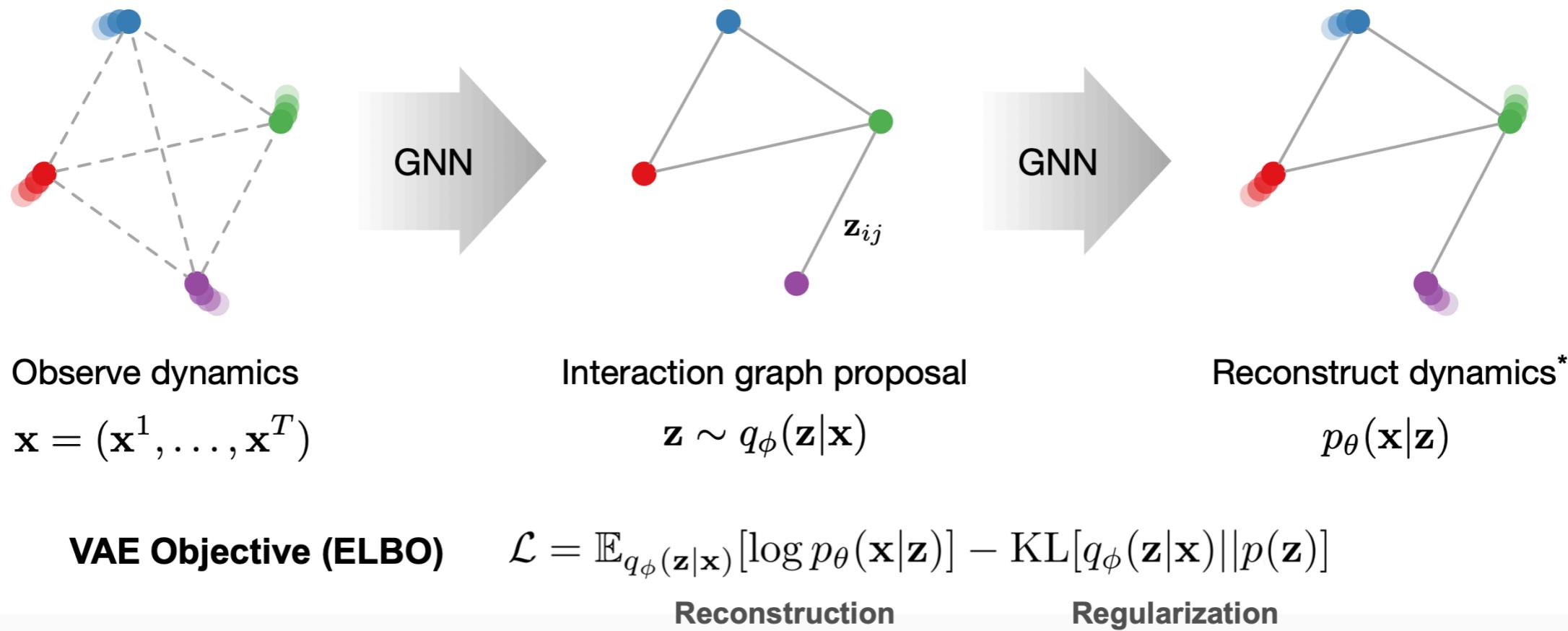
- Objective: Infer the interaction graph from observed dynamics without knowing structure of interactions



- Intuition: Use a downstream task to drive graph construction
 - Infer the graph through an encoder
 - Use the graph to make predictions
 - Optimize the entire system end-to-end

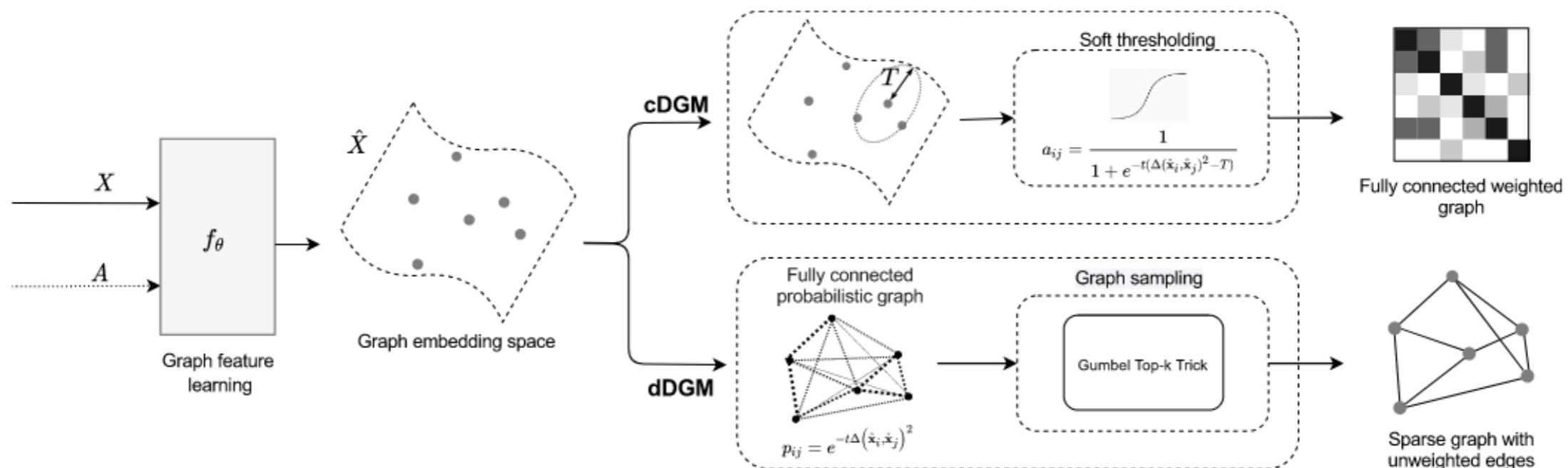
[Kipf et al., Neural relational inference for interacting systems, ICML 2018]

Example: Neural relational inference



Example: Differential graph module

- DGM projects nodes to a latent space and uses a Gaussian kernel to obtain a probability for each node pair
- The learnable function predicting the probability is optimized for a downstream task



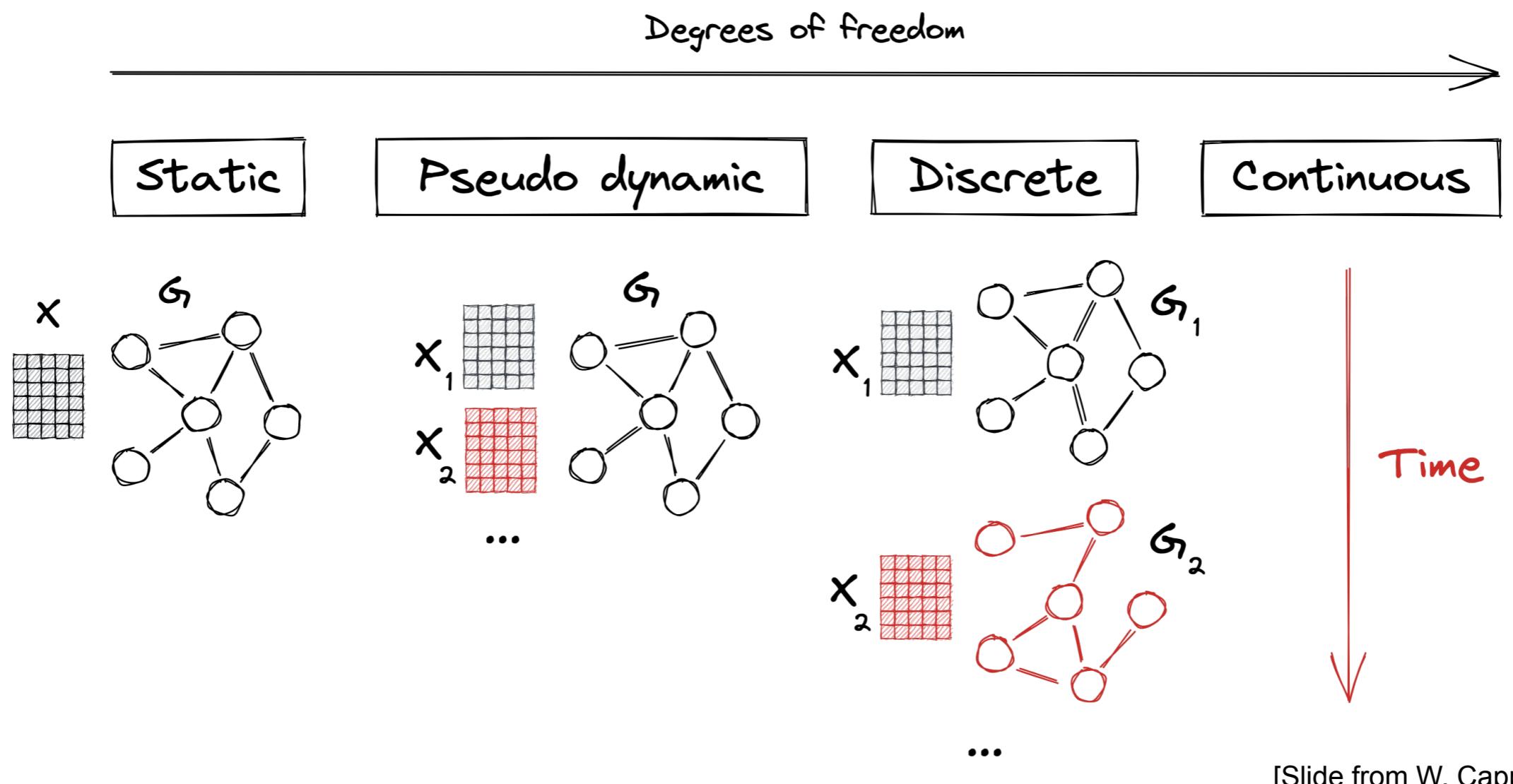
[Kazi et al., Differentiable graph module for graph convolutional networks, ICML, 2020]

Outline

- Expressive power of GNNs
- Inferring the graph topology
- **Dynamic graph models**
- Learning with sparse labels
- Applications

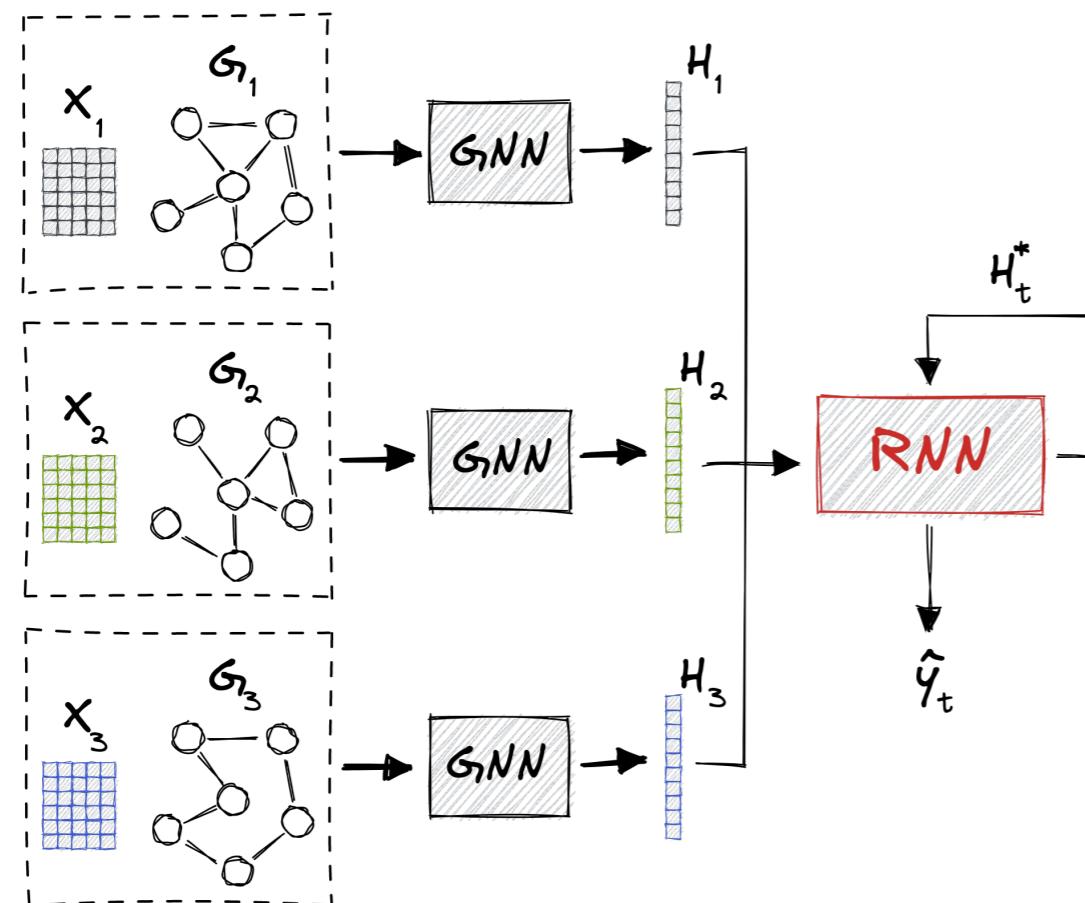
Spatial-temporal GNNs

- Dynamic networks are graphs where over time links and nodes may appear and disappear, and features may change



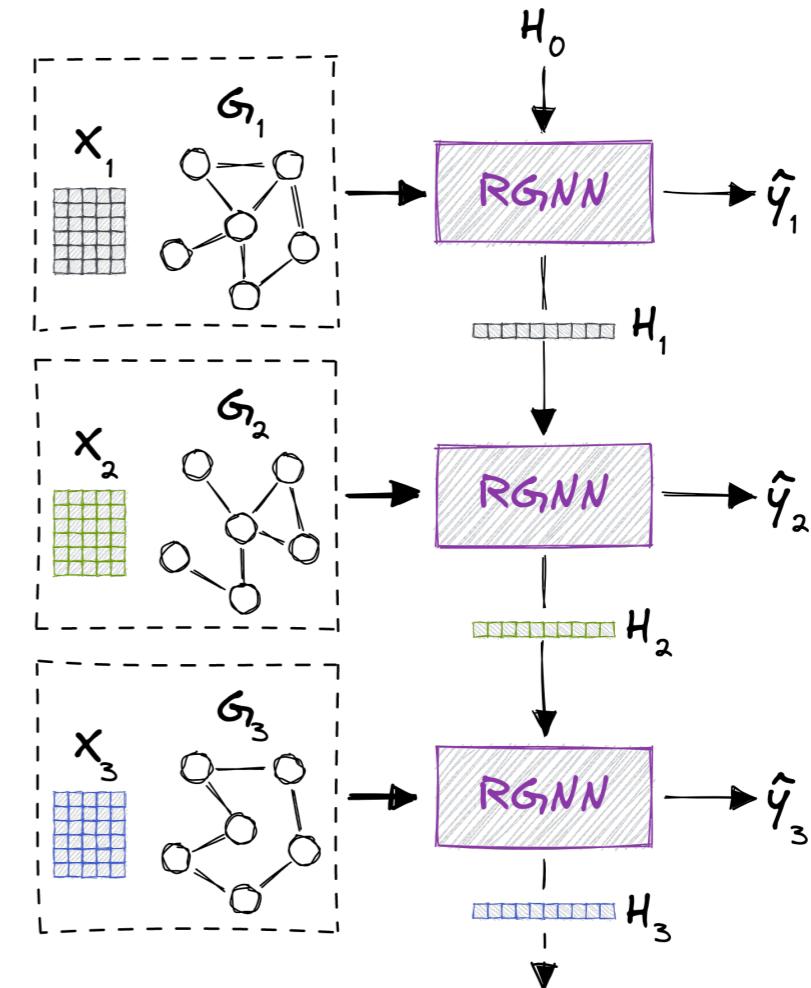
[Slide from W. Cappelletti]

Temporal GNNs follow two paradigms



Spatial then temporal:

Node embeddings for each time step are fed to a temporal model



Spatio-temporal:

A single model for embedding spatial and temporal information

[Skarding et al., Foundations and Modeling of Dynamic Networks Using Dynamic Graph Neural Networks: A Survey, IEEE Access, 2021]
[Longa et al., Graph Neural Networks for Temporal Graphs: State of the art, open challenges and opportunities, TMLR, 2023]

Outline

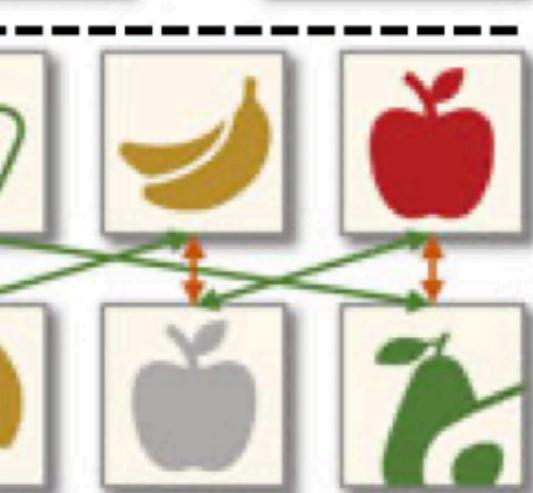
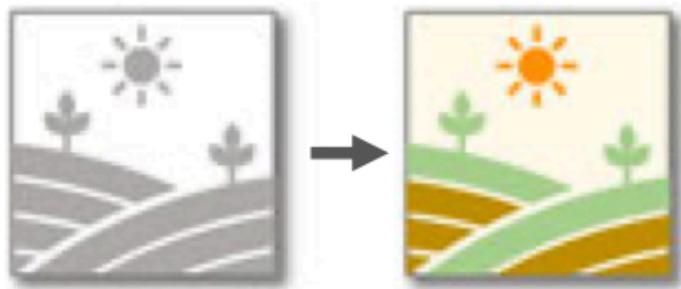
- Expressive power of GNNs
- Inferring the graph topology
- Dynamic graph models
- **Learning with sparse labels**
- Applications

Unsupervised/self-supervised learning with GNNs

- So far, we have assumed that labels are available
 - They are used to design a loss function
- What if the labels are not available or limited?
 - Loss function should depend on the information provided by the graph itself:
e.g., input features, graph topology
 - Some examples: DeepWalk, node2vec, SDNE (they consider only the graph structure)
- In self-supervised learning (SSL), models are learned by solving a series of hand-crafted auxiliary tasks (so-called pretext tasks)
 - Supervision signals are acquired from data itself, without the need for annotation

Illustrative examples of pretext tasks

Computer vision



Natural Language Processing

[MASK] is impossible.

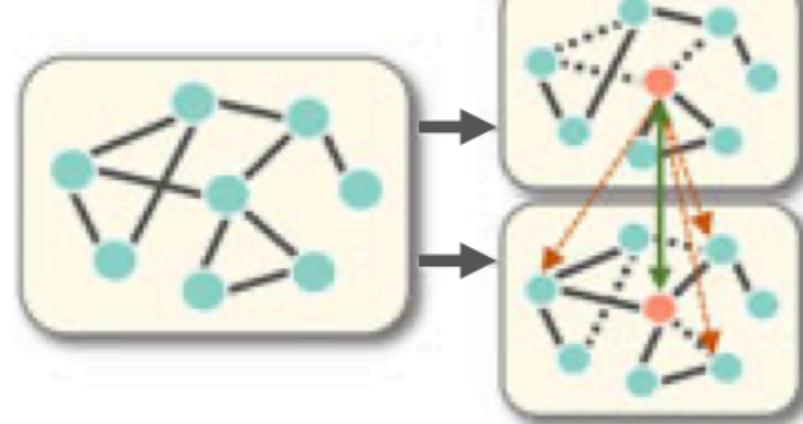
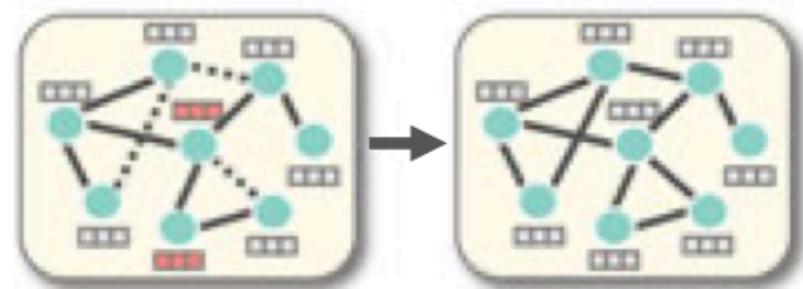
Nothing is impossible.

I'm going outside.

I'll be back soon.

I got up late.

Graph machine learning



- A. Image colorisation

- B. Image contrastive learning

- A. Masked language modelling

- B. Next sentence prediction

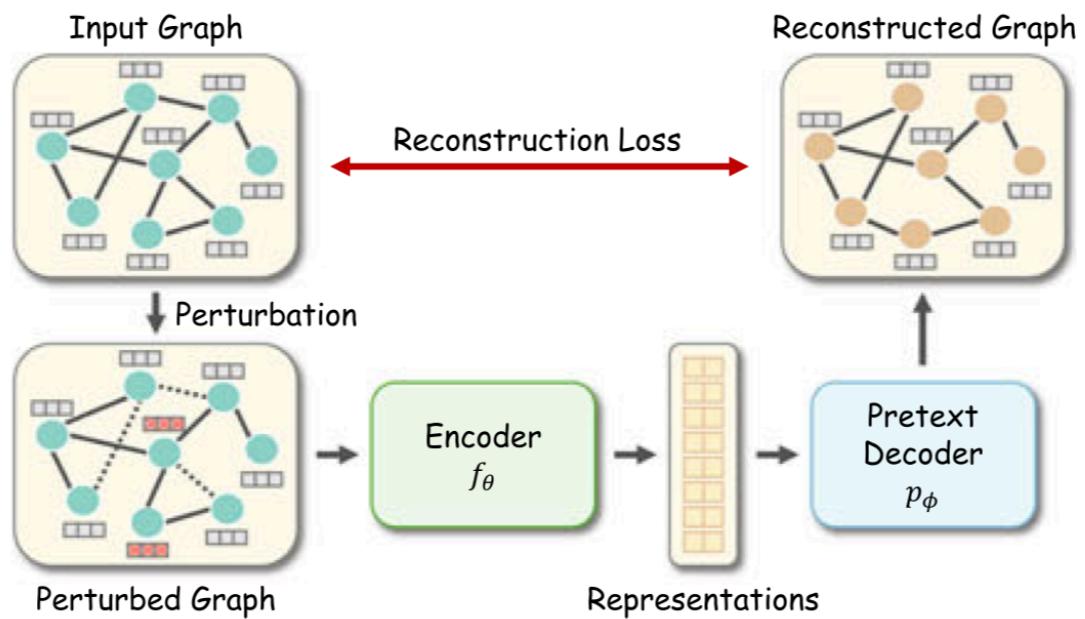
- A. Masked graph generation

- B. Node contrastive learning

[Liu et al., Graph Self-Supervised learning: A survey, IEEE TKDE 2023]

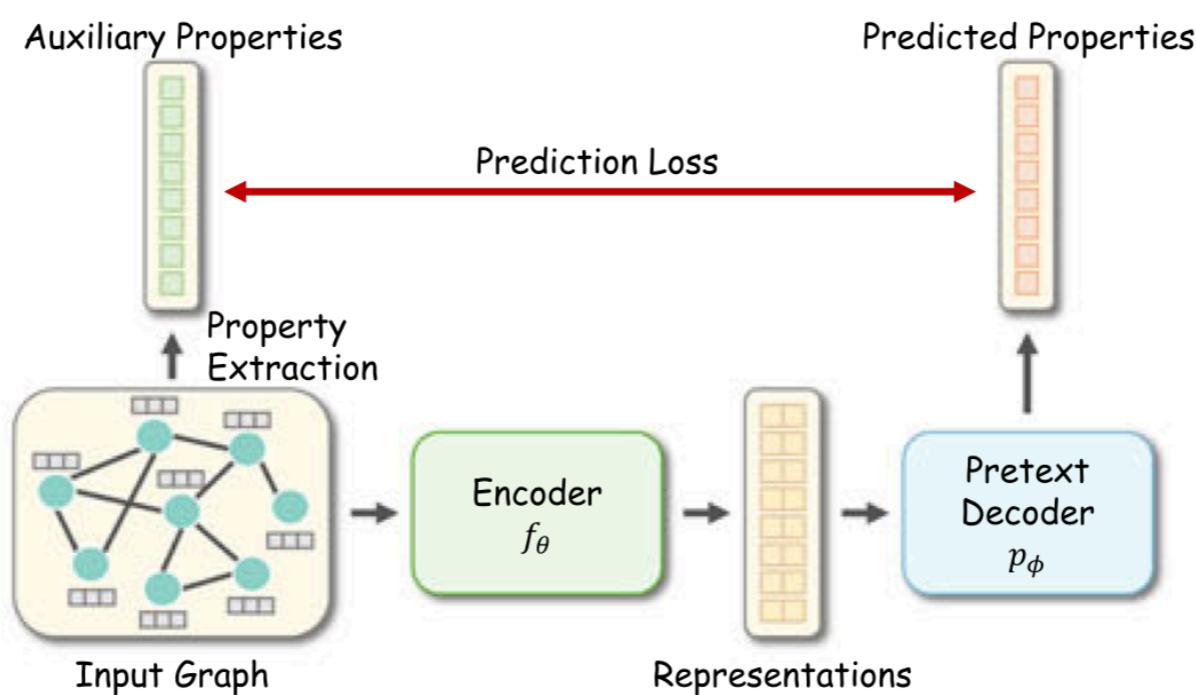
Generation-based SSL

- Pretext tasks are defined as a graph data reconstruction problem of either the graph structure or the node attributes
- GNNs are typically used to encode nodes in the graph (encoder)
- The decoder reconstructs the adjacency matrix or the node attributes (pretext task)
- The loss is defined to measure the difference between the reconstructed and the original graph data (e.g., GAE)



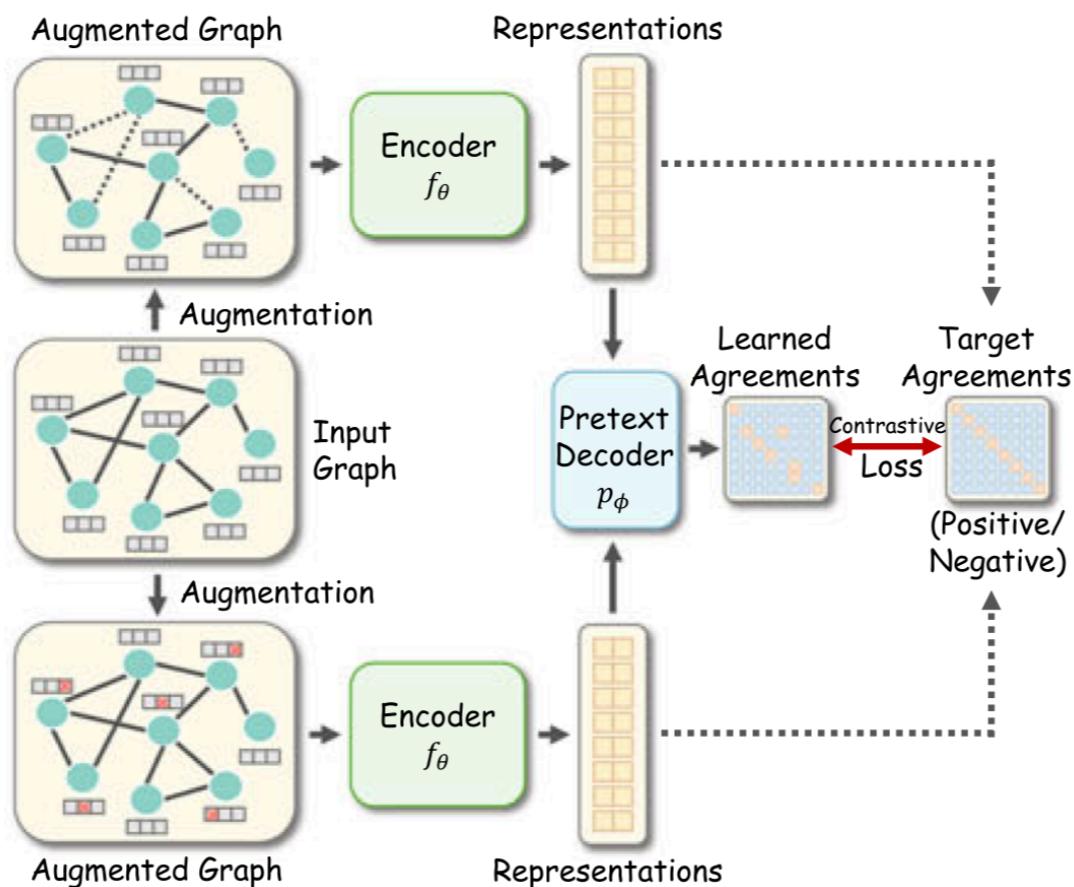
Auxiliary property-based SSL

- Pretext tasks are defined as auxiliary properties of the data (e.g., graph partition, node degree, cluster index)
- It often requires domain knowledge
- The loss is defined to measure the difference between the estimated and the original auxiliary property (e.g., mean square error, cross-entropy)



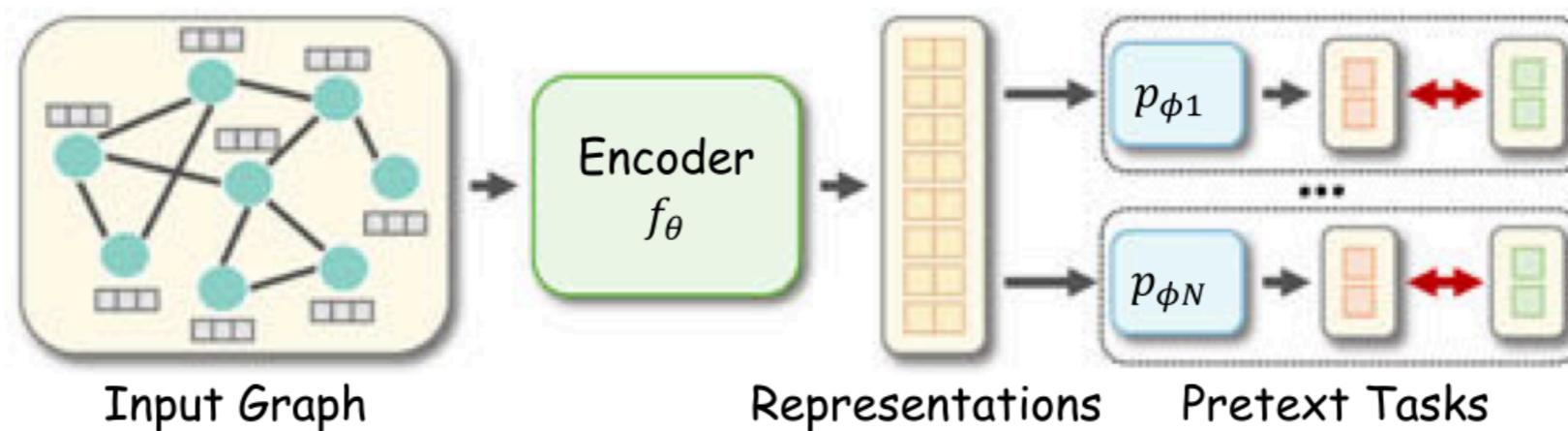
Contrastive-based SSL

- Pretext tasks are defined based on the concepts of mutual information (MI) maximization
- The loss is defined by maximizing the estimated MI between augmented instances of the same object (e.g., node, subgraph, graph)



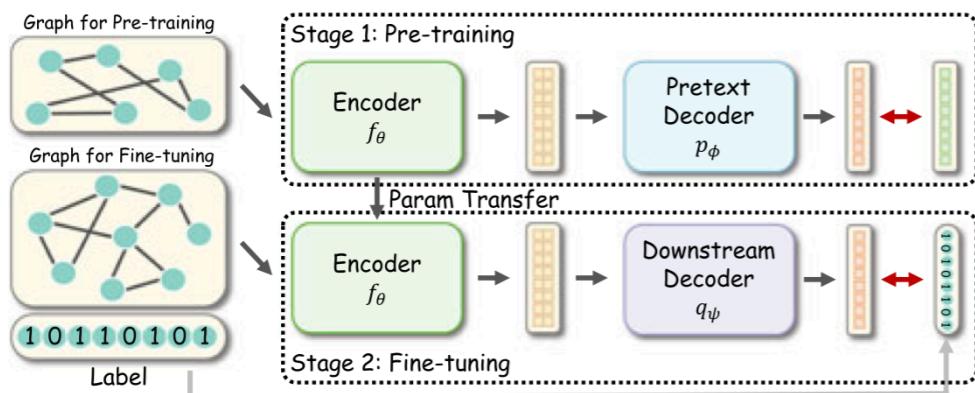
Hybrid methods for SSL

- Pretext tasks are defined based on a combination of multiple objective
- Balancing the different pretext tasks is often challenging

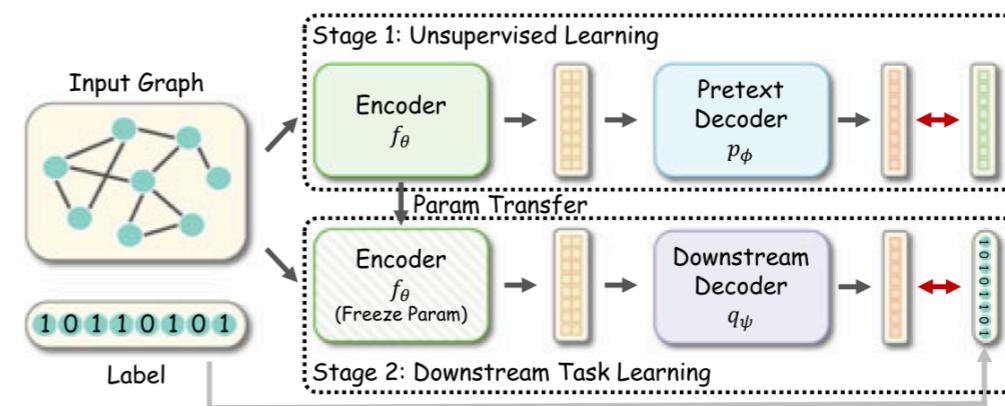


How do I use my labels in SSL?

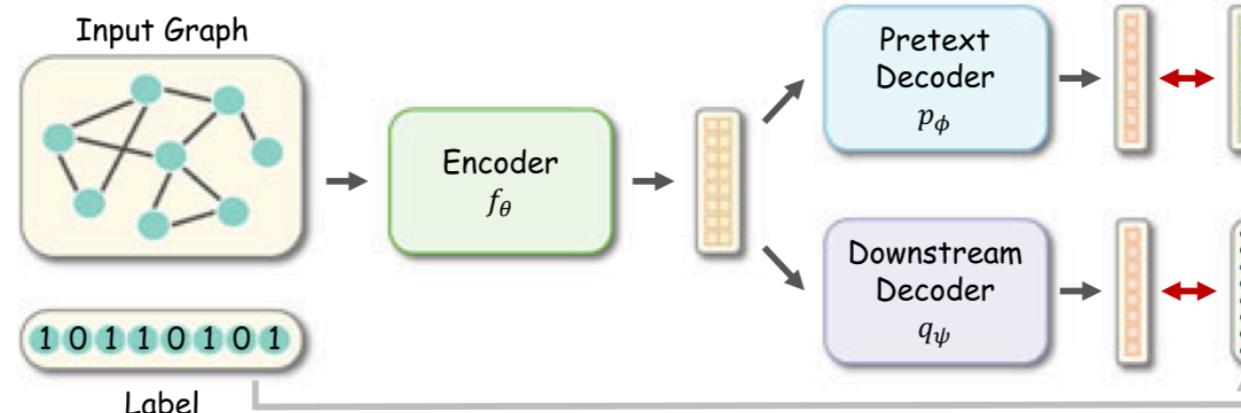
- If labelled data are available, they can be combined with the pretext task(s) to improve the models:



Stage 1: Pre-training on pretext
Stage 2: Fine-tuning on labeled data



Stage 1: Pre-training on pretext
Stage 2: Fine-tuning on labeled data with frozen decoder

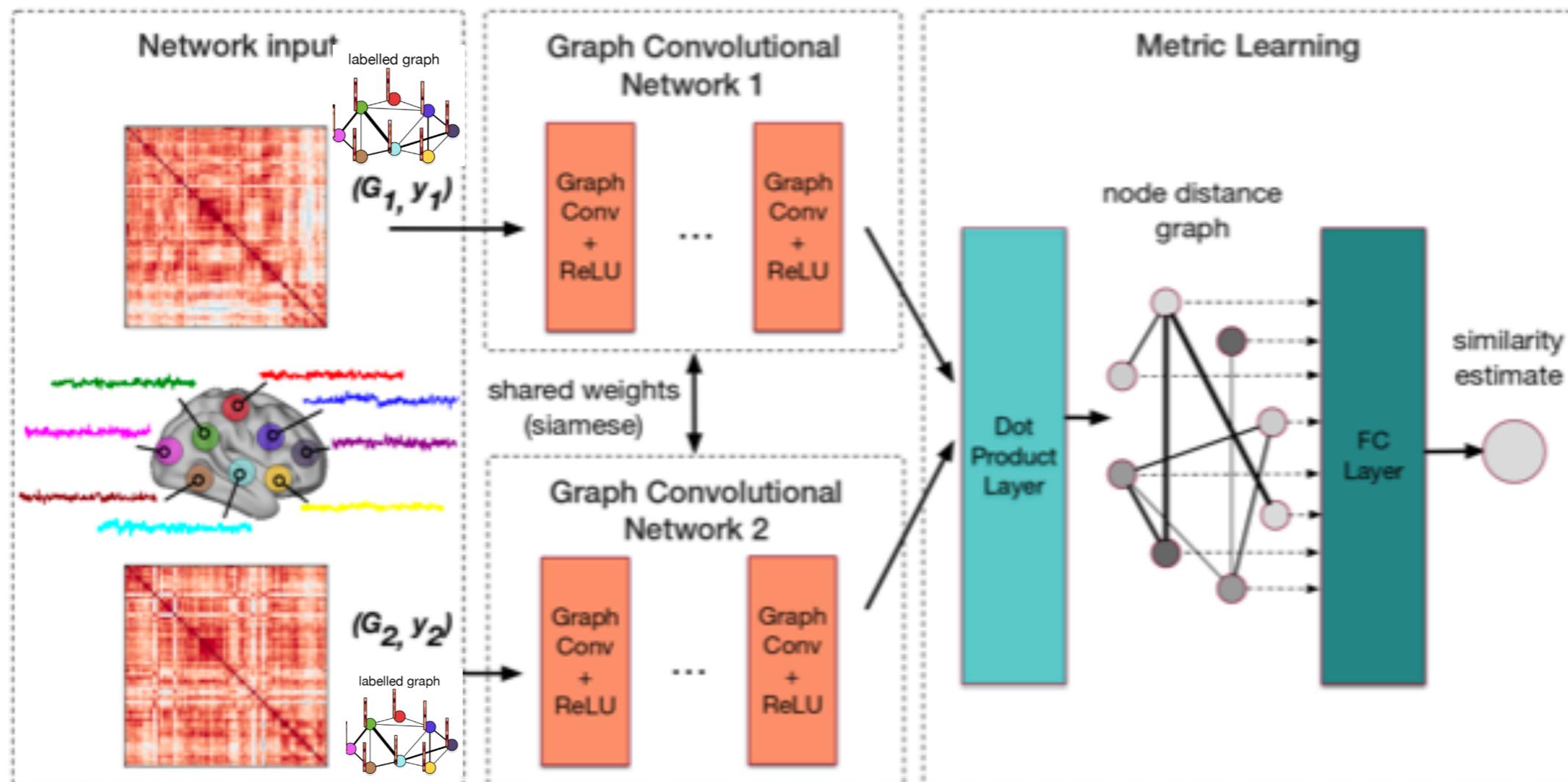


Multitask learning: Learn jointly pretext and downstream task

Outline

- Expressive power of GNNs
- Inferring the graph topology
- Dynamic graph models
- Learning with sparse labels
- Applications

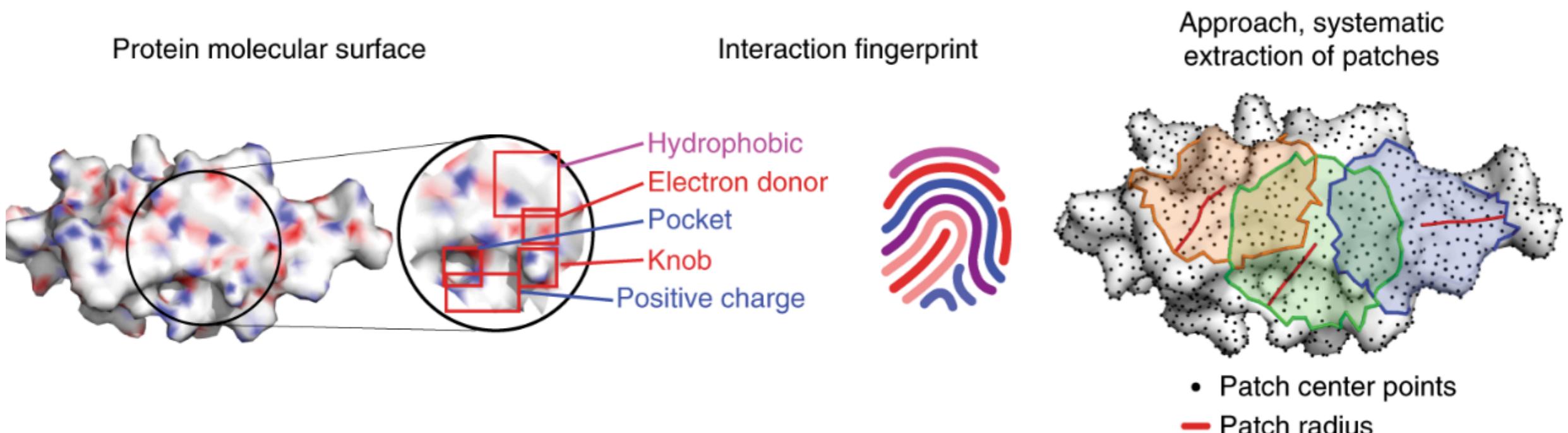
Neuroscience: learn to compare brain networks



[Ktena et al., Metric learning with spectral graph convolutions on brain connectivity networks, NeuroImage, 2018]

Protein-protein interactions

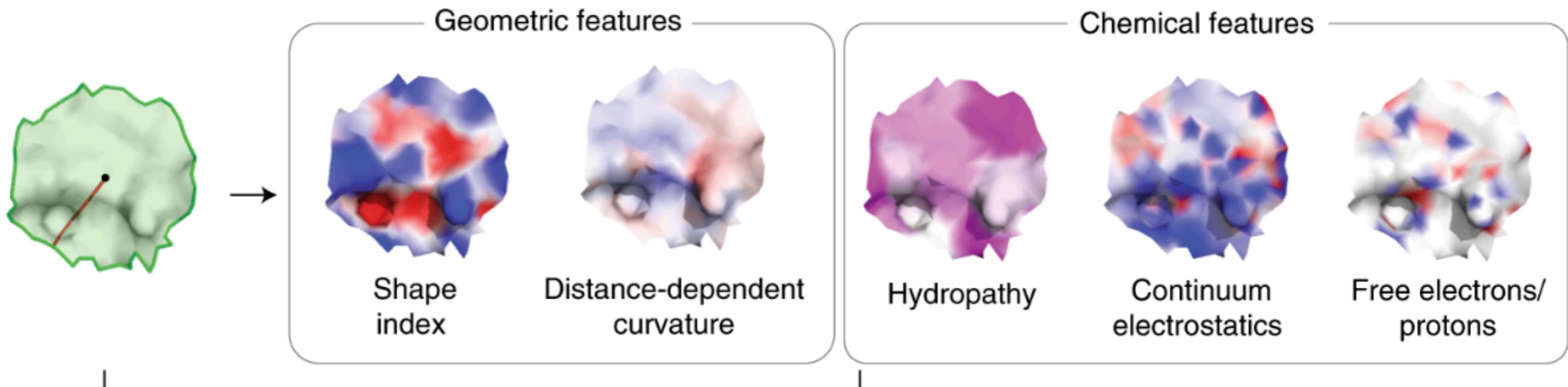
- Predicting interactions between proteins and other biomolecules solely based on structure remains a challenge in biology
- Exploit GNNs to learn interaction fingerprints in protein molecular surfaces that determine protein interactions



[Gainza et al, Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning, Nature methods, 2019]

Protein-protein interactions

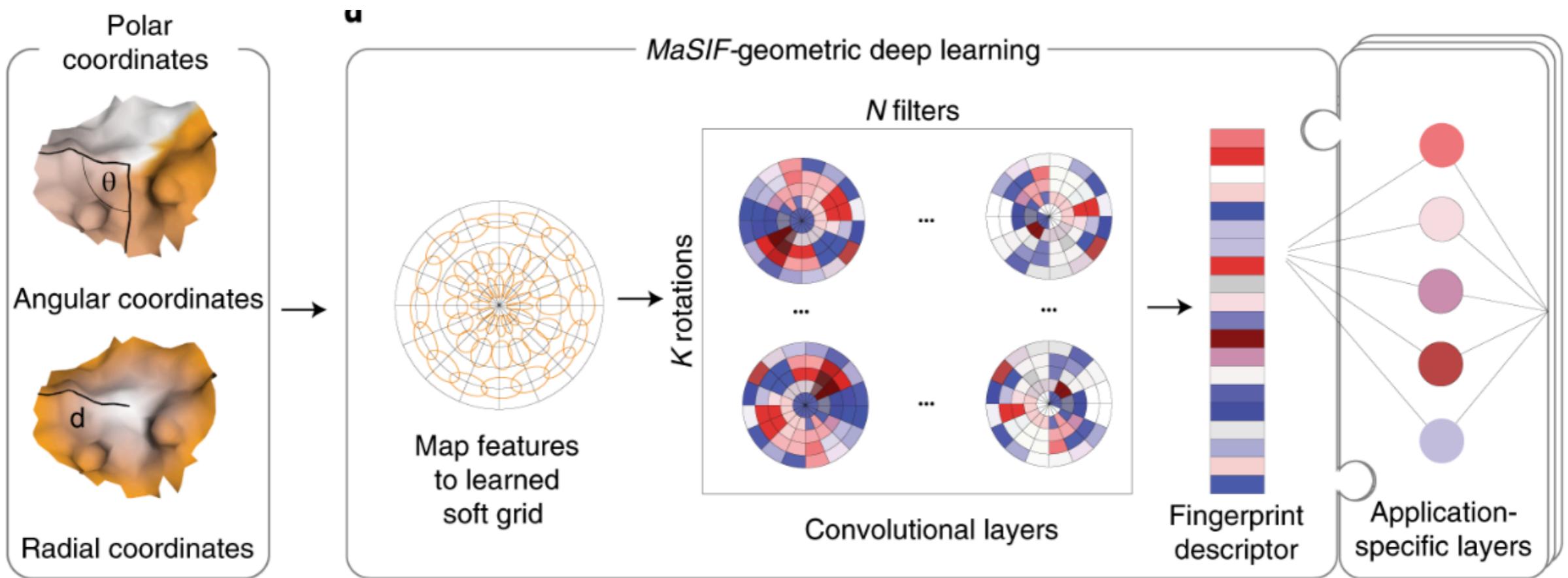
- Predicting interactions between proteins and other biomolecules solely based on structure remains a challenge in biology
- Exploit GNNs to learn interaction fingerprints in protein molecular surfaces that determine protein interactions



[Gainza et al, Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning, Nature methods, 2019]

Protein-protein interactions

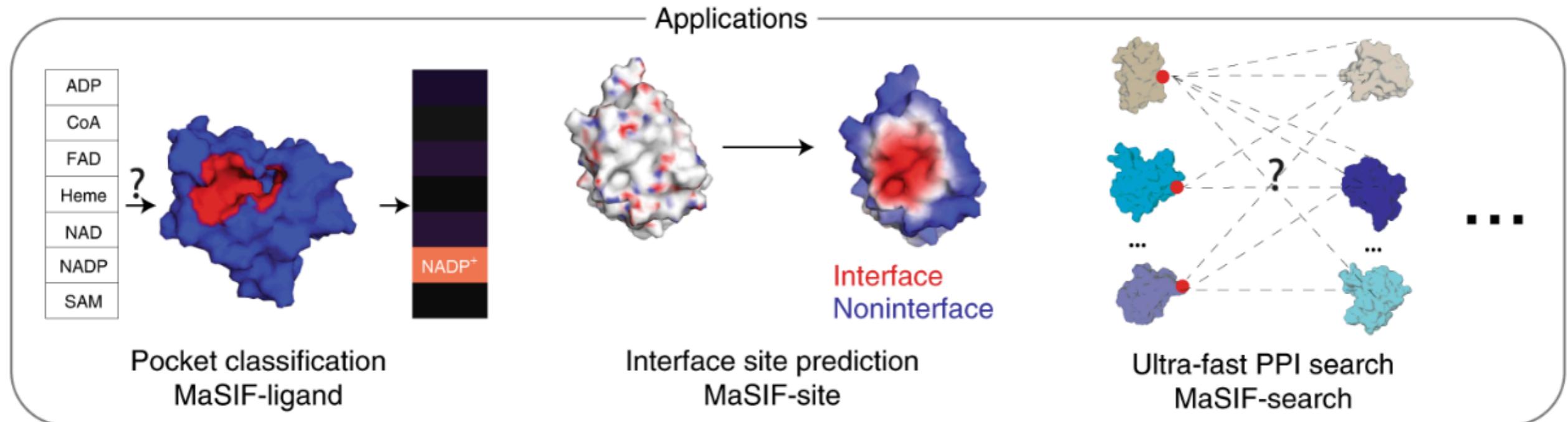
- Predicting interactions between proteins and other biomolecules solely based on structure remains a challenge in biology
- Exploit GNNs to learn interaction fingerprints in protein molecular surfaces that determine protein interactions



[Gainza et al, Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning, Nature methods, 2019]

Protein-protein interactions

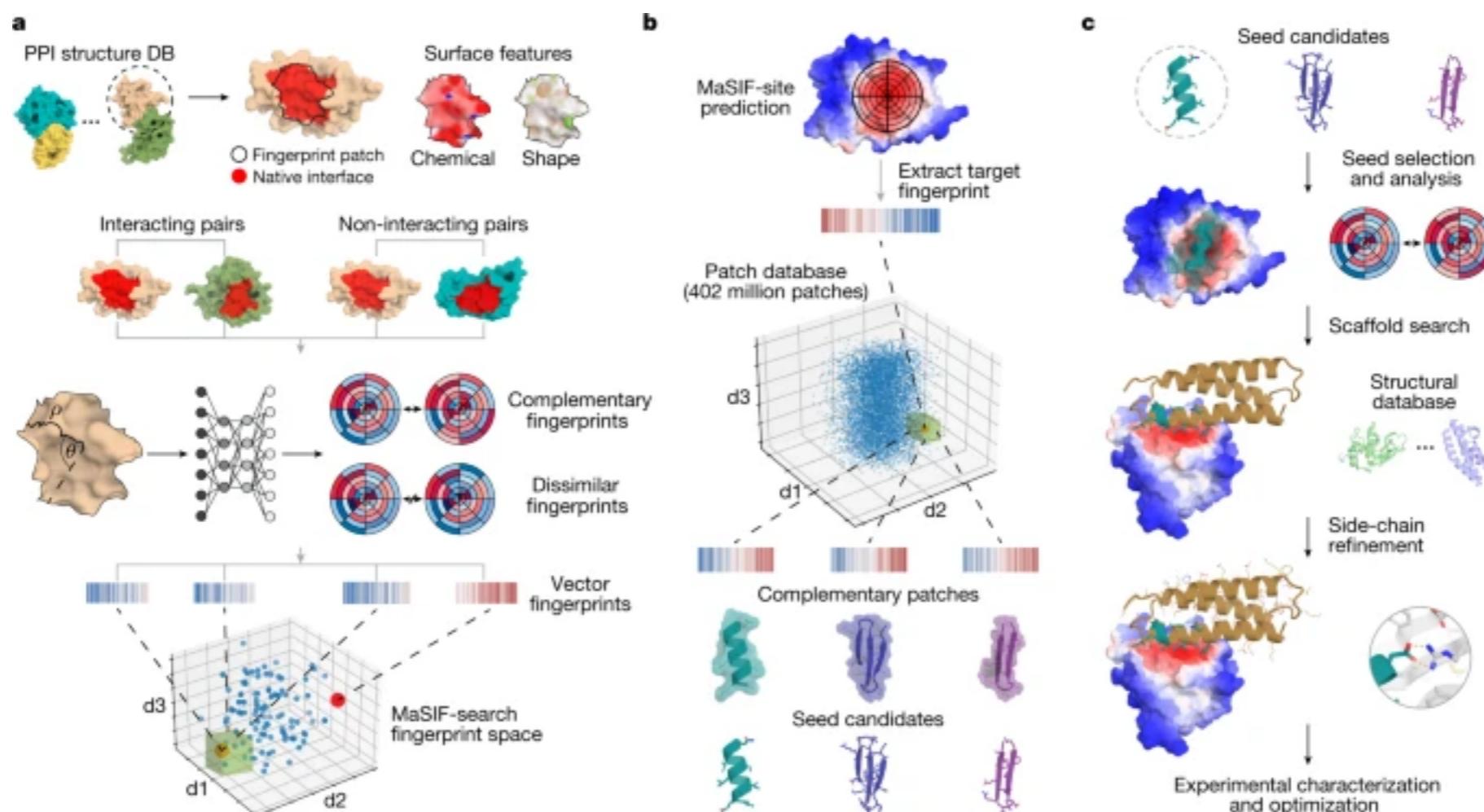
- Predicting interactions between proteins and other biomolecules solely based on structure remains a challenge in biology
- Exploit GNNs to learn interaction fingerprints in protein molecular surfaces that determine protein interactions



[Gainza et al, Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning, Nature methods, 2019]

De novo design of protein-protein interactions

- Design PPIs by targeting sites using only structural information from the target protein



[Gainza et al, De novo design of protein interactions with learned surface fingerprints, Nature, 2023]

Protein folding

AlphaFold: a solution to a 50-year-old grand challenge in biology

SHARE

AUTHORS

TAt The AlphaFold team

Proteins are essential to life, supporting practically all its functions. They are large complex molecules, made up of chains of amino acids, and what a protein does largely depends on its unique 3D structure. Figuring out what shapes proteins fold into is known as the "protein folding problem", and has stood as a grand challenge in biology for the past 50 years. In a major scientific advance, the latest version of our AI system AlphaFold has been recognised as a solution to this grand challenge by the organisers of the biennial Critical Assessment of protein Structure Prediction (CASP). This breakthrough demonstrates the impact AI can have on scientific discovery and its potential to dramatically accelerate progress in some of the most fundamental fields that explain and shape our world.

Graph Transformers

[Jumper et al., High protein structure prediction with AlphaFold, Nature, 2021]

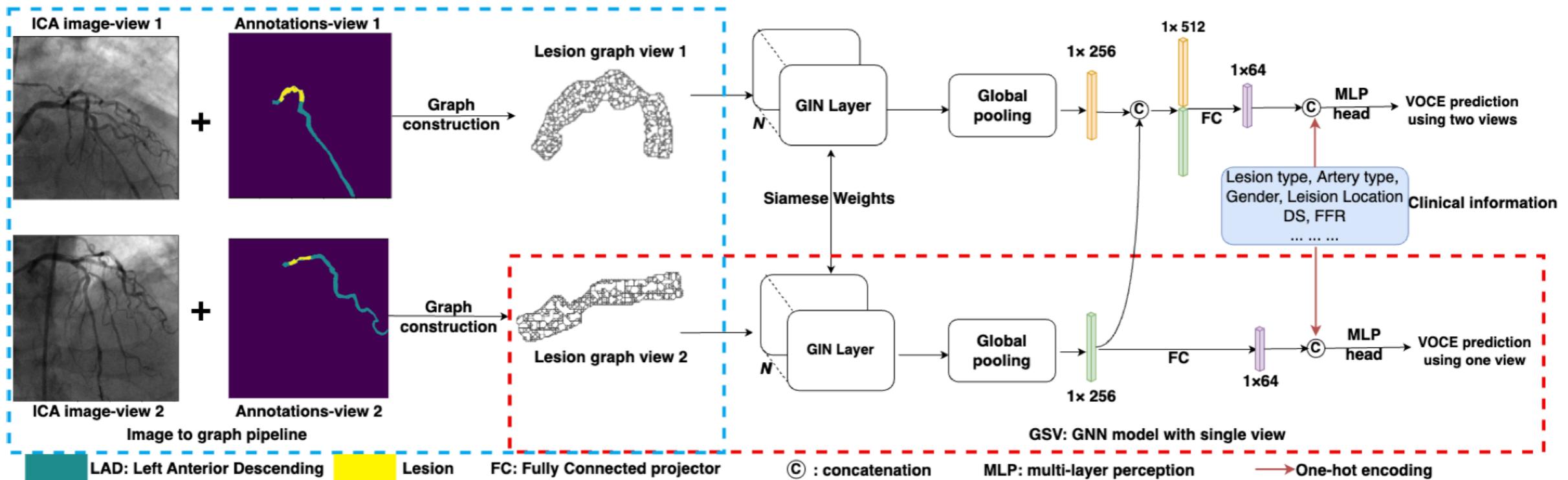
EPFL

Network Machine Learning - EE452
Dr Dorina Thanou
Prof. Pascal Frossard

37

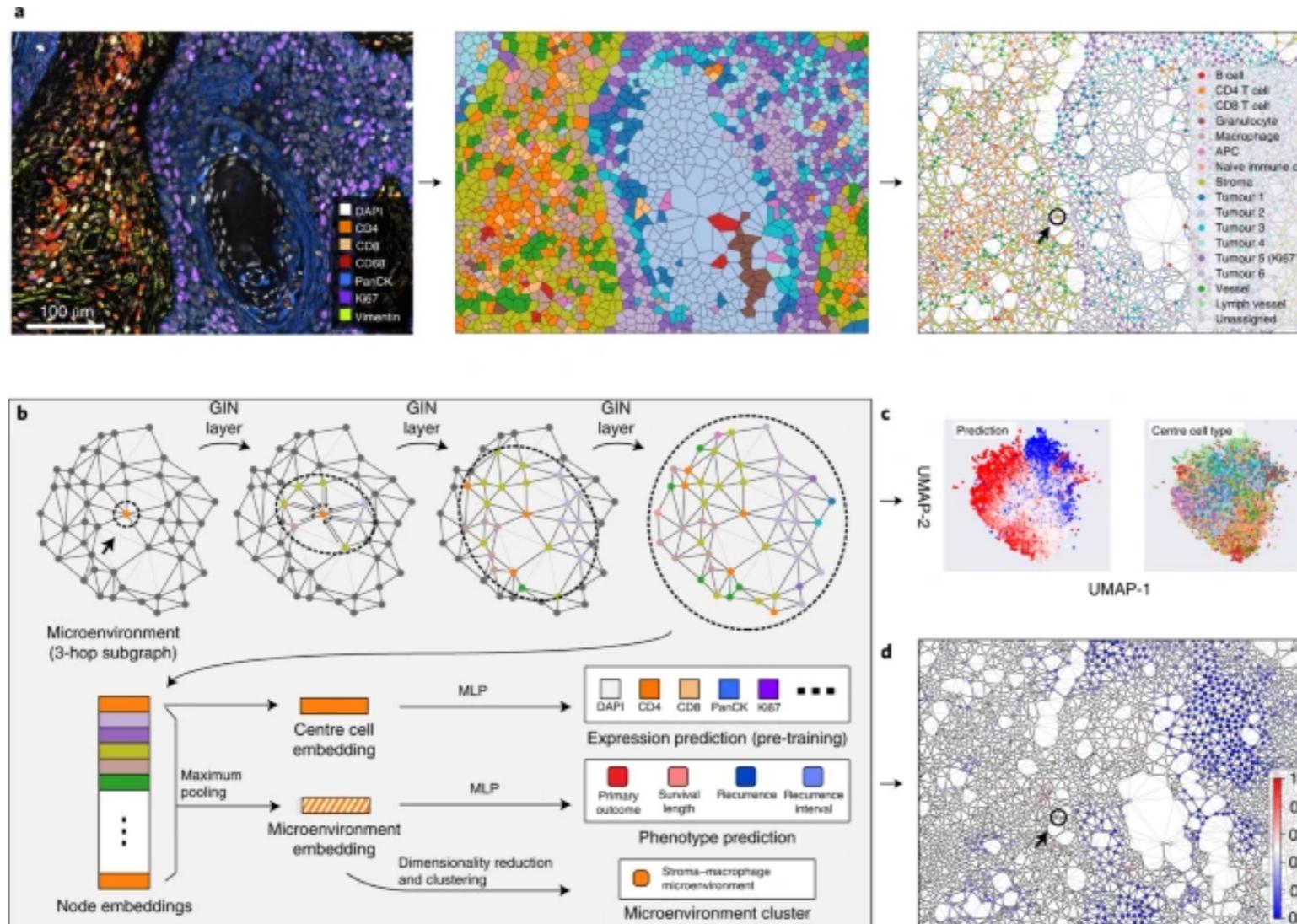
Future cardiovascular events prediction

- Graphs are used to model the geometry of the arteries
- Graph neural networks extract predictive features from invasive coronary artery images



[Sun et al., Graph neural network based future clinical events prediction from invasive coronary angiography, ISBI, 2024]

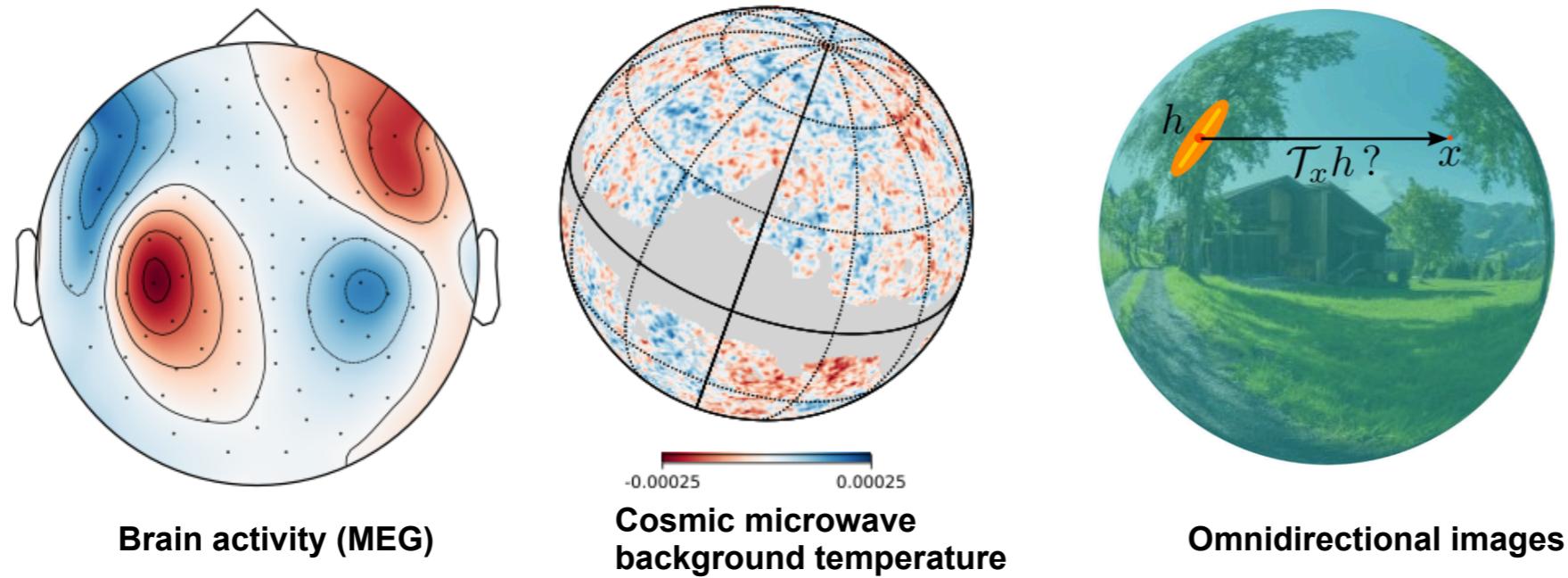
Geometric deep learning models cellular microenvironment



[Wu et al, Graph deep learning for the characterization of tumour microenvironments from spatial protein profiles in tissue specimens, Nature Biomedical Engineering, 2022]

Spherical imaging

- Spherical data has specific spatial and statistical properties that cannot be captured by regular CNN models



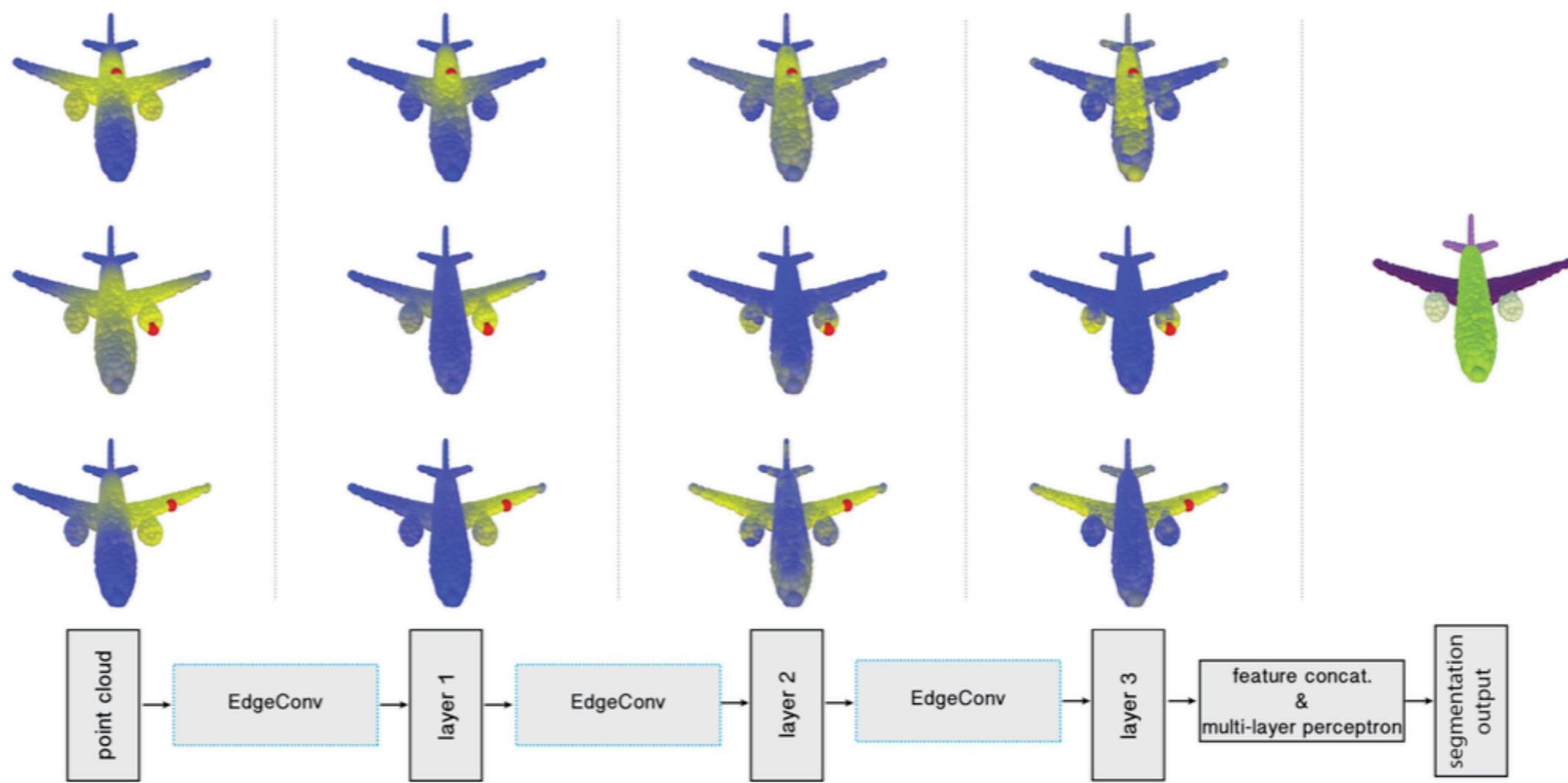
- Sphere is modelled as a graph and classical operation (convolution, translation, pooling...) are performed on the graph

[Perraudin et al., “DeepSphere”, Astronomy and Computing, 2019]

[Bidgoli et al, OSLO: On-the-Sphere Learning for Omnidirectional images and its application to 360-degree image compression, arXiv, 2021]

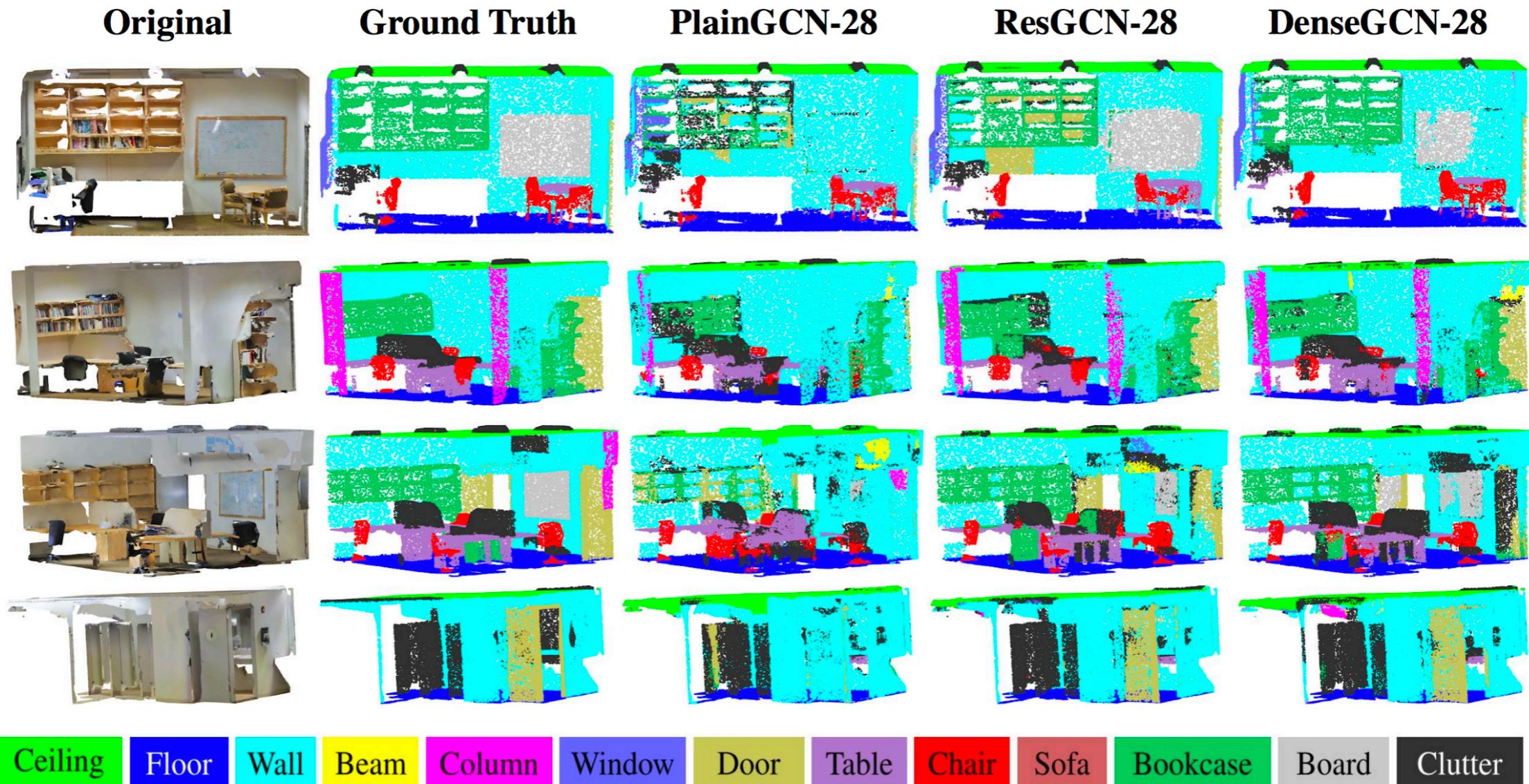
Point cloud semantic segmentation

- Graph attention convolution are successful in capturing specific shapes that adapt to the structure of an object



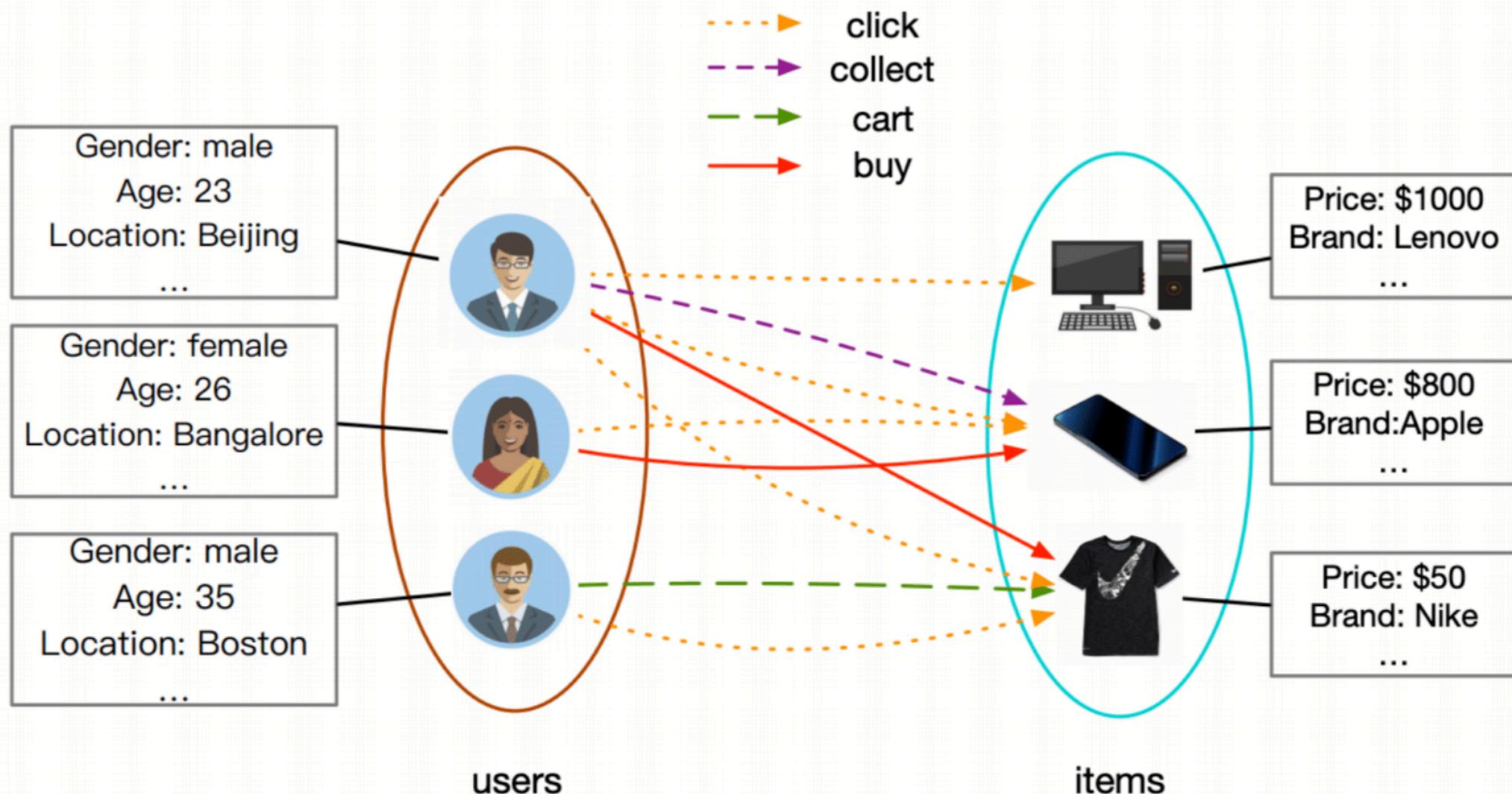
[Wang et al., Graph Attention Convolution for Point Cloud Semantic Segmentation, CVPR, 2019]

Point clouds semantic segmentation



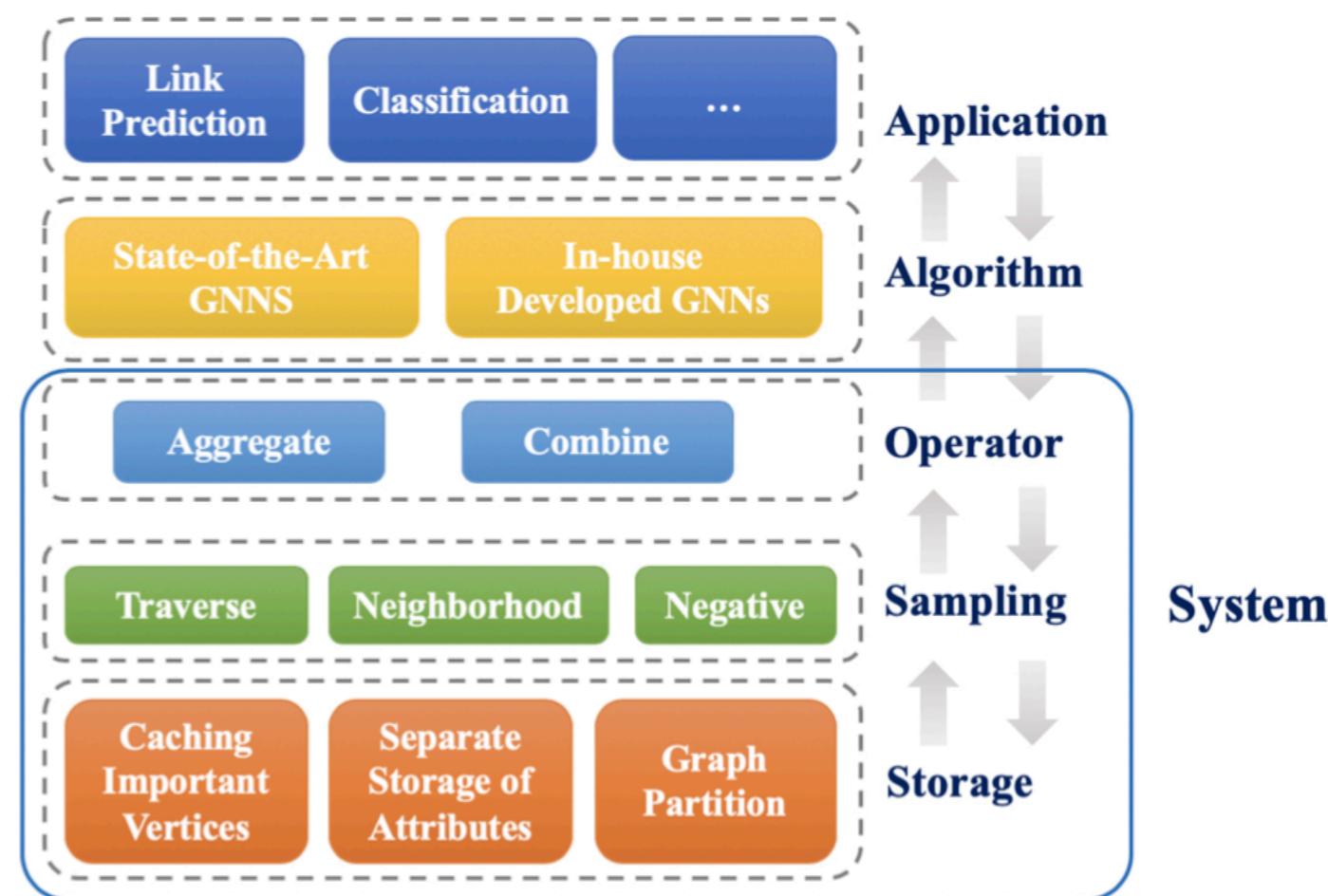
[Li et al., DeepGCNs: Can GCNs Go as Deep as CNNs?, ICCV 2019]

Recommender systems



Recommender systems: Aligraph

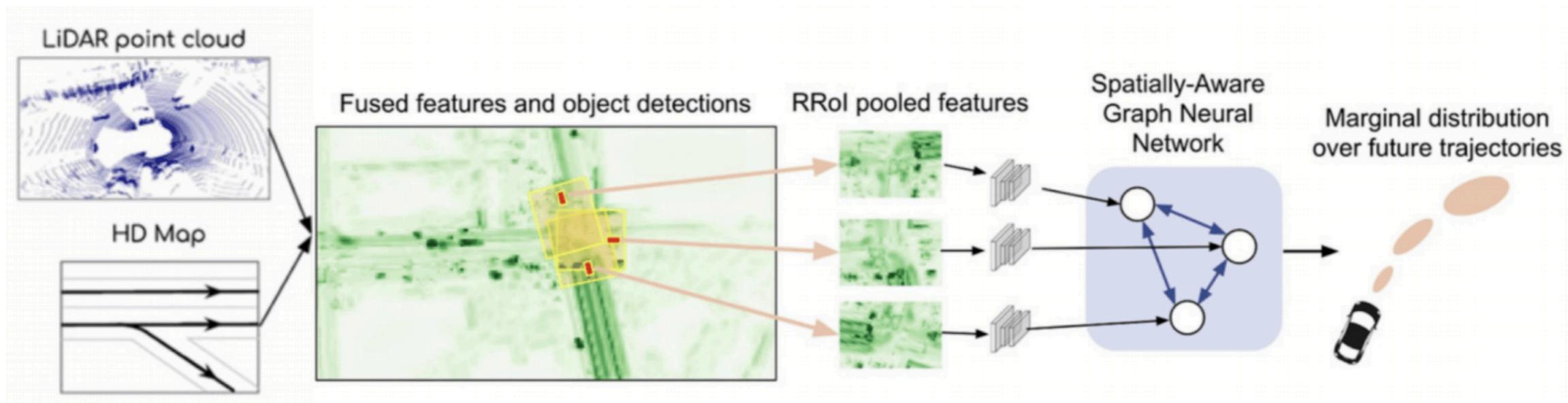
- The system is currently deployed at Alibaba to support product recommendation and personalized search at Alibaba's E-Commerce platform



[Zhu et al., *AliGraph: A Comprehensive Graph Neural Network Platform*, 2019]

Self driving cars

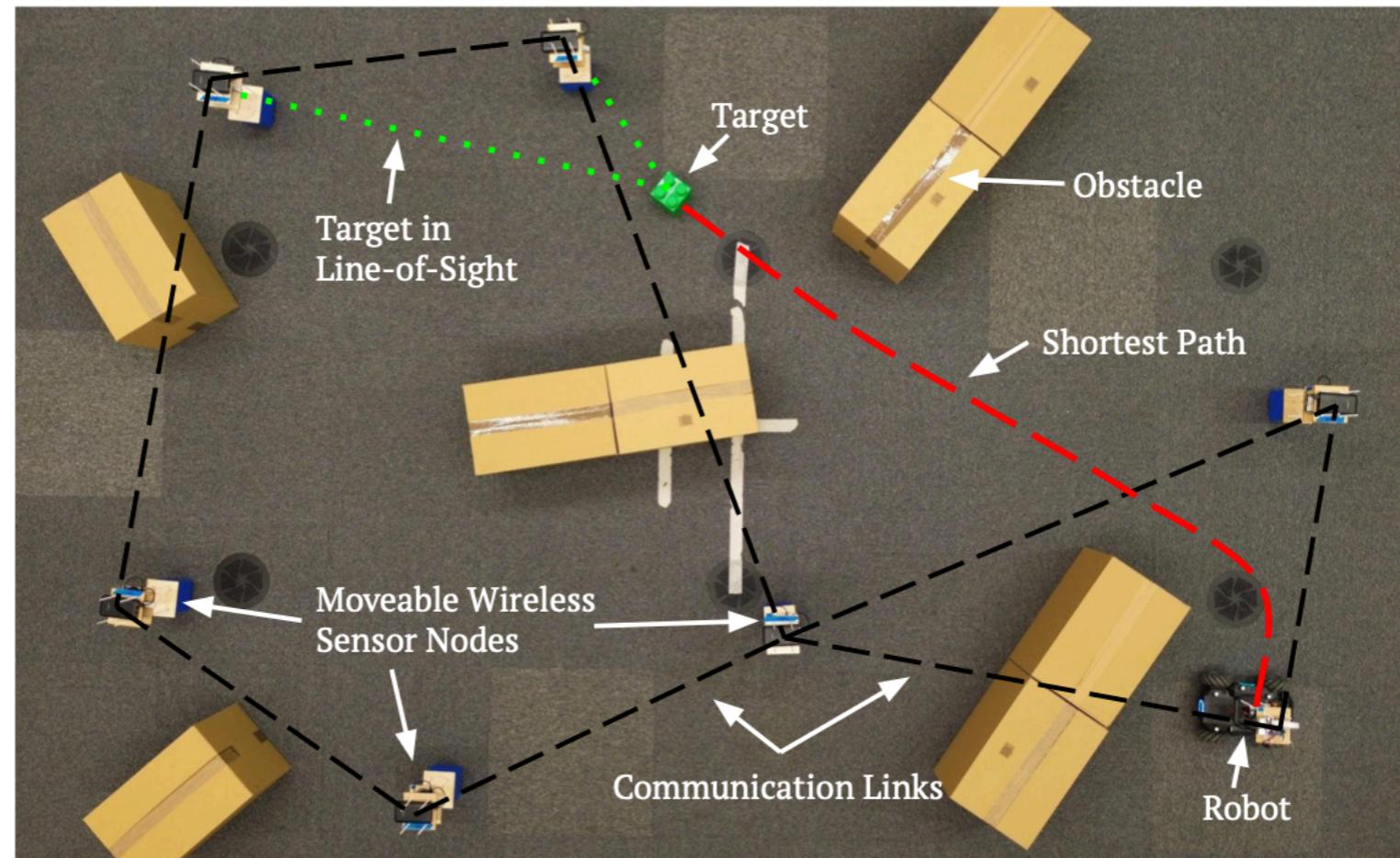
- GNNs provide probabilistic estimates of future trajectories
 - A CNN detects objects
 - A GNN captures interactions between objects and predicts behaviors



[Casas et al, Spatially aware graph neural networks for relational behaviour forecasting for sensor data, ICRA, 2020]

Learning cooperative perception

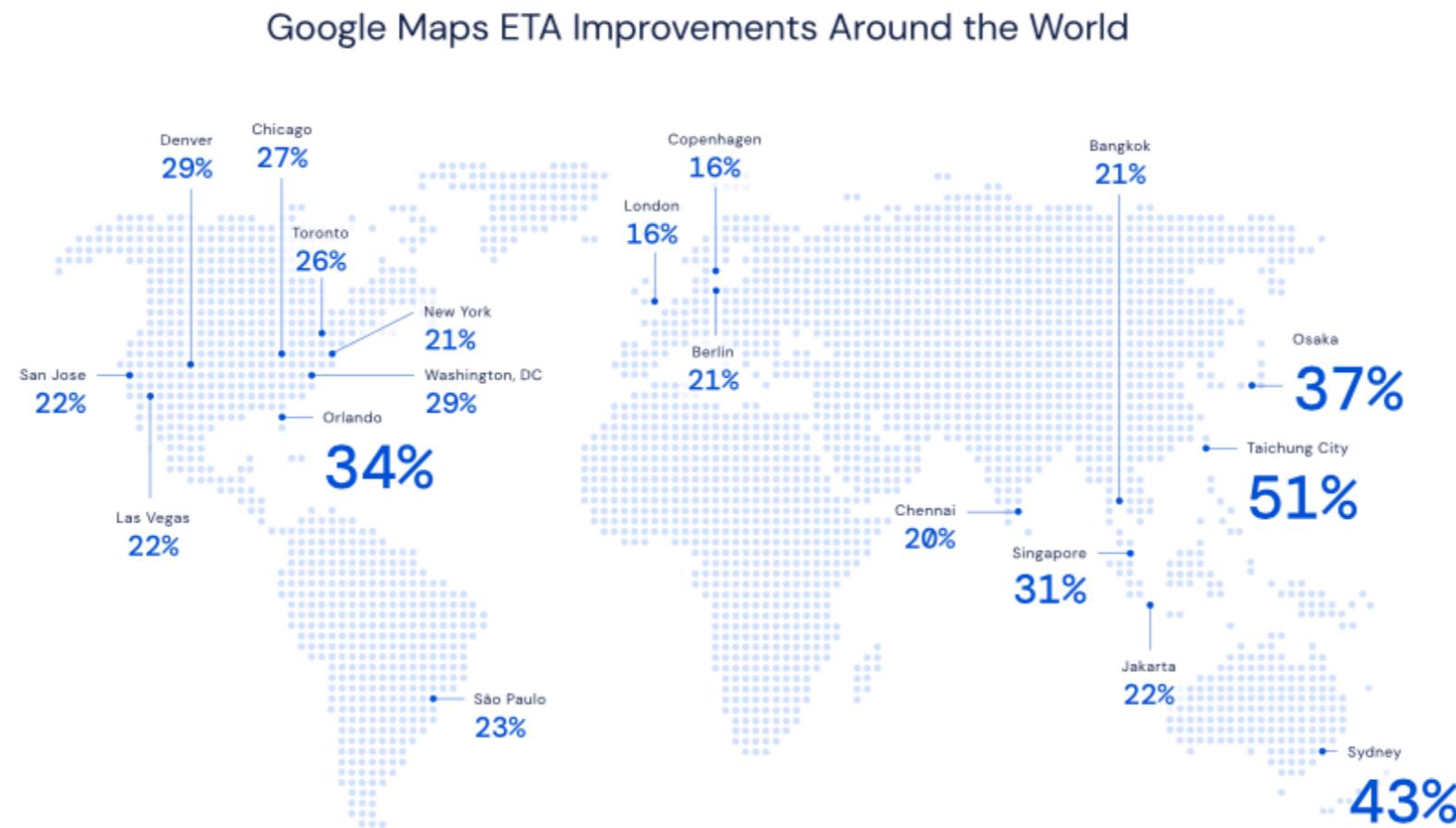
- GNNs can learn to guide the robot to its target under the guidance of a visual sensor network



[Blumenkamp et al, See What the Robot Can't See: Learning Cooperative Perception for Visual Navigation, arXiv 2023]

Traffic prediction

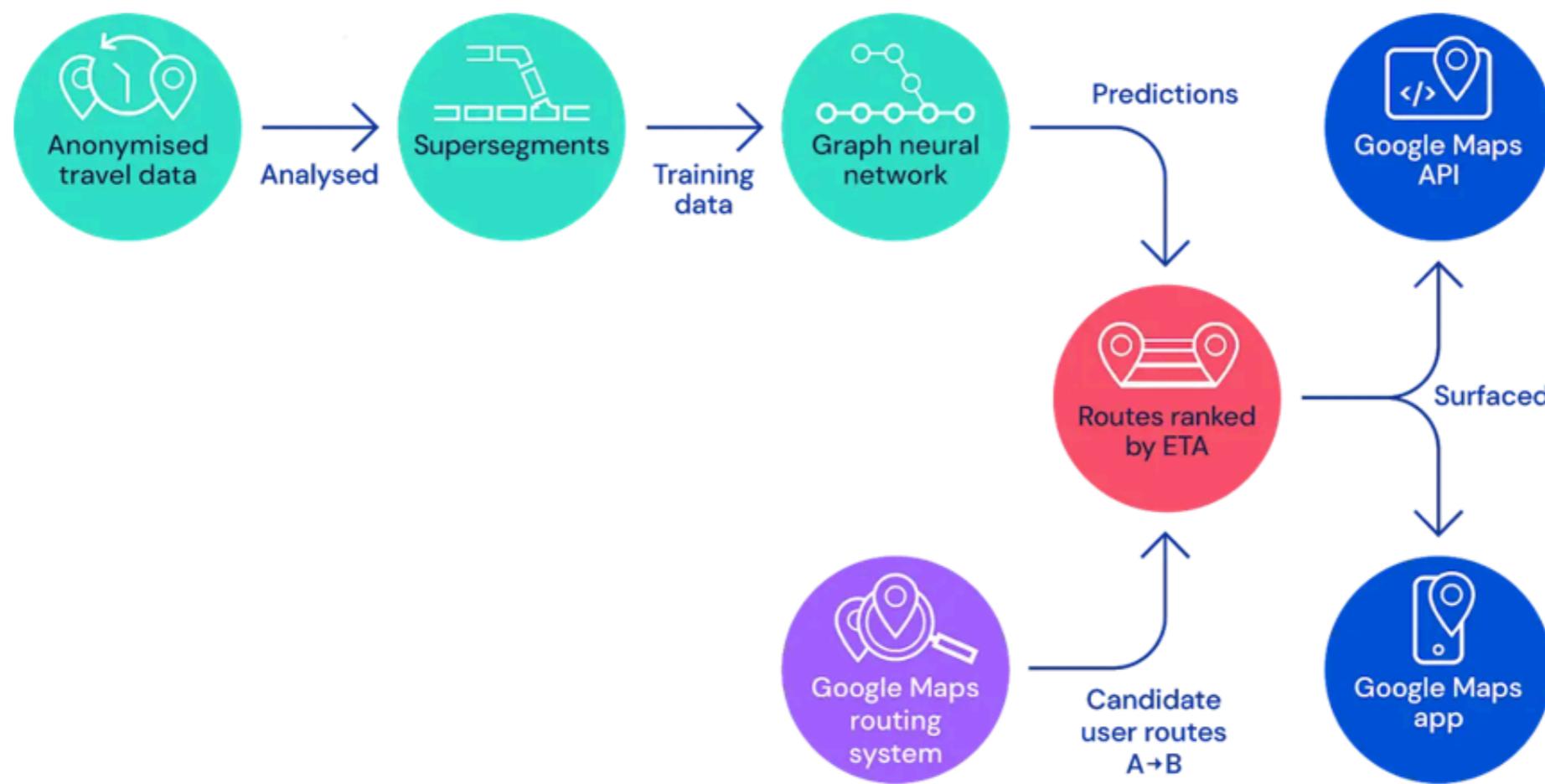
- As the road network is naturally modelled by a graph of road segments and intersections, ETA prediction can be improved with graph representation learning



[Derrow-Pinion et al., ETA Prediction with Graph Neural Networks in Google Maps, 2021]

Traffic prediction

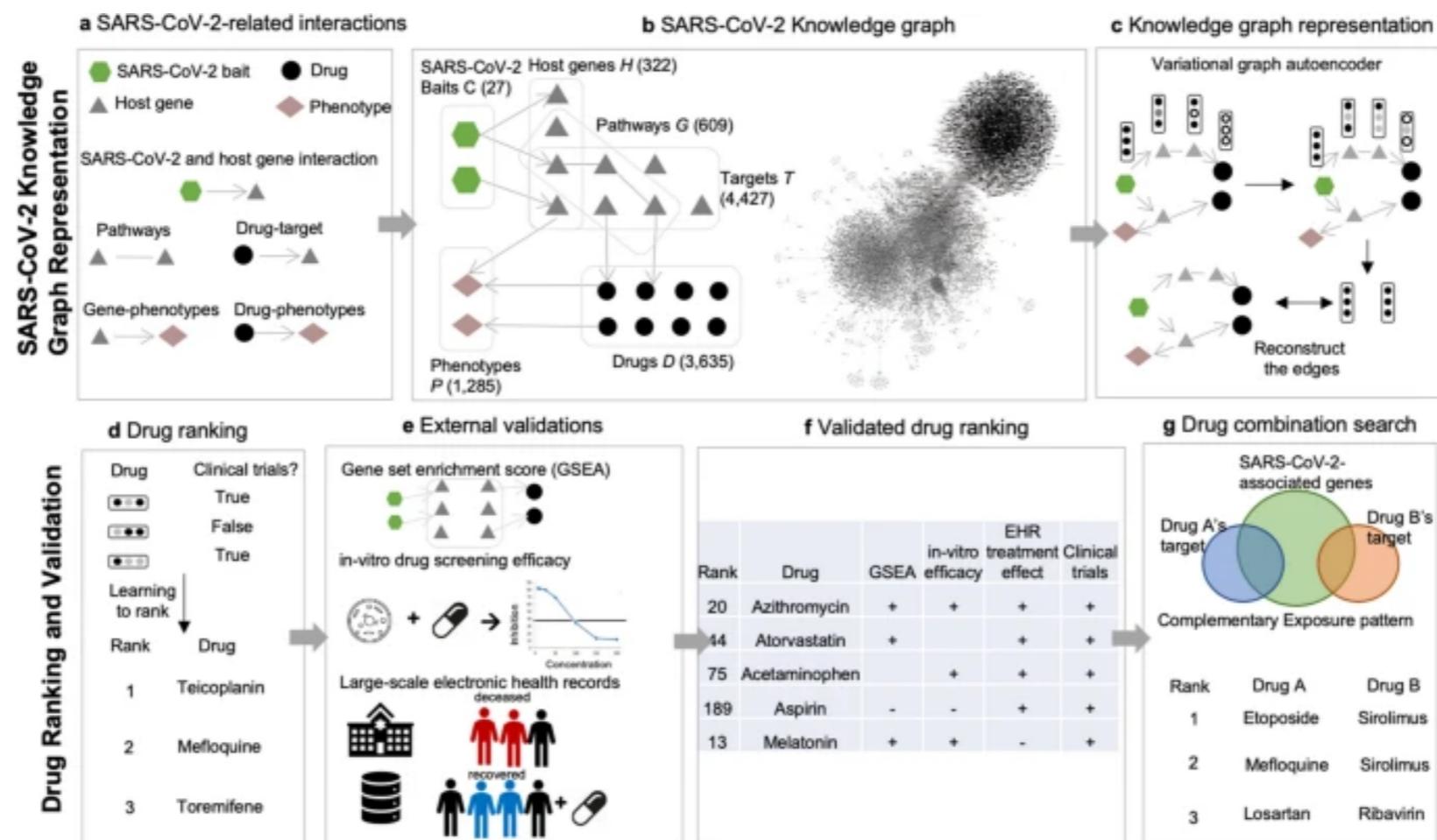
- As the road network is naturally modelled by a graph of road segments and intersections, ETA prediction can be improved with graph representation learning



[Derrow-Pinion et al., ETA Prediction with Graph Neural Networks in Google Maps, 2021]

Drug repurposing for COVID-19

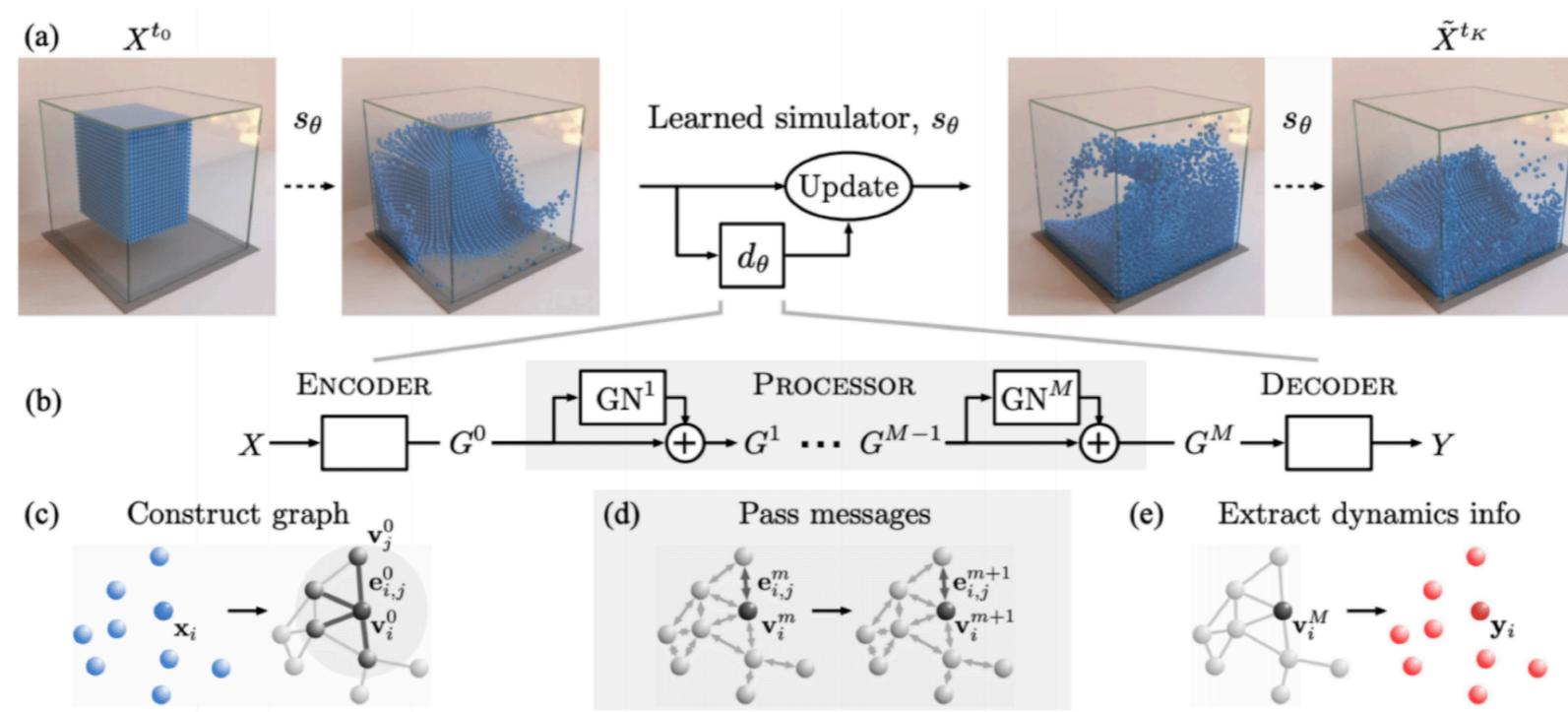
- Deep GNN approaches have been used to derive the candidate drug's representation based on the biological interactions



[Hsieh et al, Drug repurposing for COVID-19 using graph neural network and harmonizing multiple evidence, Nature Sc. Rep., 2021]

Learning physical simulations

- Mesh-based simulations are central to modeling complex physical systems
- High-dimensional scientific simulations are very expensive
- GNNs have been used to learn mesh-based simulations and predict the dynamics of physical systems



[Sanchez-Gonzales et al, Learning to Simulate Complex Physics with Graph Networks, ICML 2020]
[Pfaff et al, Learning mesh-based simulation with Graph Networks, ICML 2021]

Learning physical simulations

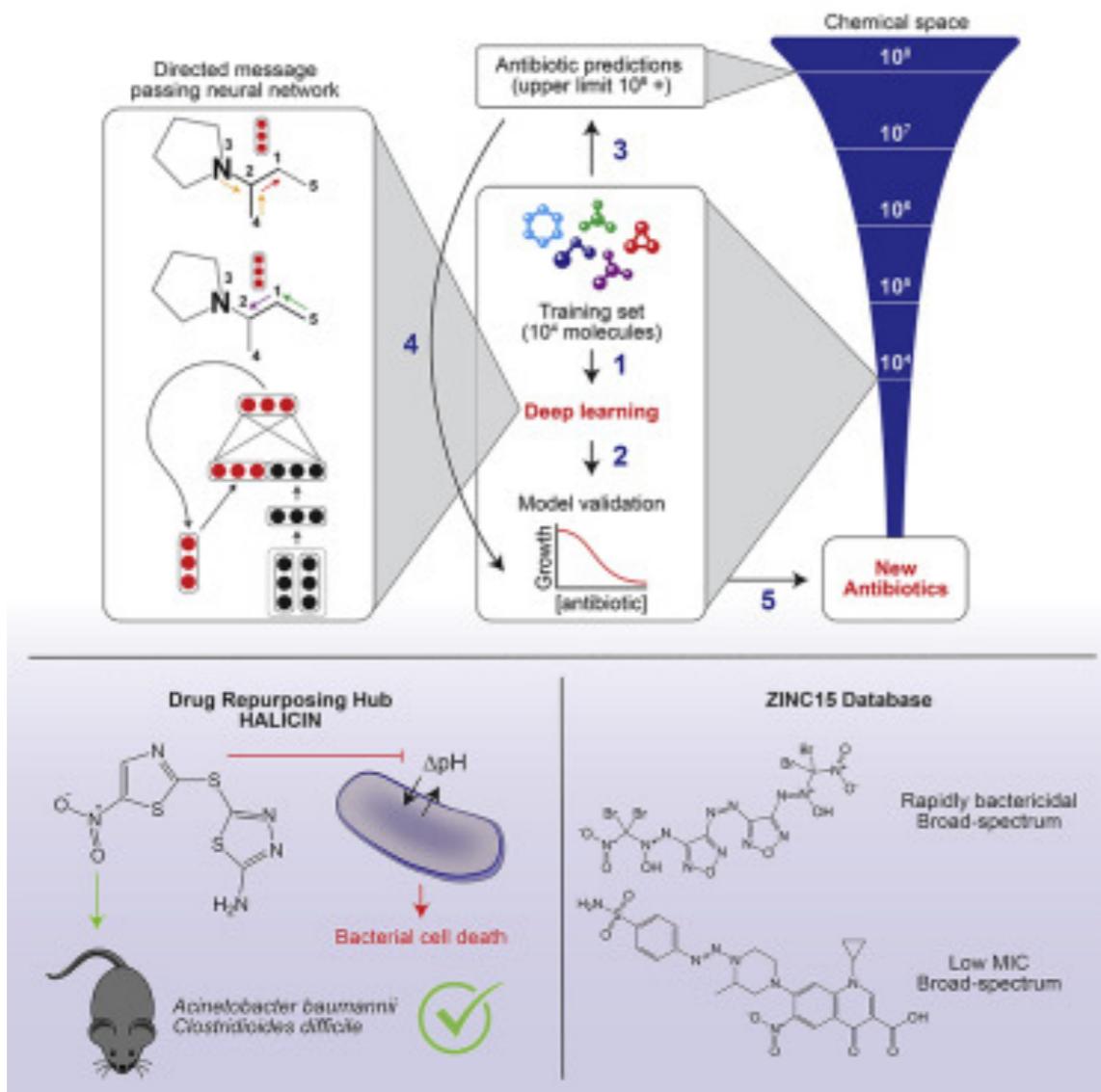
- Some examples:

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

<https://sites.google.com/view/meshgraphnets>

Molecular graph generation

- Recent advances in antibiotic discovery ...



BBC NEWS

Home | Coronavirus | Climate | Video | World | UK | Business | Tech | Science | Stories | Entertainment & Arts

Health | Coronavirus

Do we ask too much of our planet?

Scientists discover powerful antibiotic using AI

© 21 February 2020

Support the Guardian

Available for everyone, funded by readers

Contribute → Subscribe →

News **Opinion** **Sport** **Culture** **Lifestyle** More ▾

Education Schools Teachers Universities Students

Antibiotics

This article is more than 1 year old

Powerful antibiotic discovered using machine learning for first time

Team at MIT says halicin kills some of the world's most dangerous strains

Ian Sample Science editor
Thu 20 Feb 2020 16.28 GMT

Chemical Structures: Two petri dish images showing bacterial growth inhibition around antibiotic spots.

[Simonovsky et al, 2017, De Cao et al 2018, Stokes et al 2020]

Summary

- The expressive power of GNN architectures has strong connections with the WL kernel
- Building more expressive architectures is still a challenge
- Self-supervised learning on graphs is a promising learning paradigm when labels are limited
- The choice of the pretext task is crucial; It needs to be correlated with the main task
- Many emerging applications!

Recall: Useful resources

- **Toolboxes**
 - https://github.com/rusty1s/pytorch_geometric
 - <https://github.com/dmlc/dgl>
 - <https://github.com/deepmind/jraph>
 - <https://github.com/tensorflow/gnn>
- **Datasets**
 - DGL datasets: <https://docs.dgl.ai/api/python/dgl.data.html>
 - PyG datasets: <https://pytorch-geometric.readthedocs.io/en/latest/modules/datasets.html>
 - OGB datasets: <https://ogb.stanford.edu>
 - <https://chrsmrrs.github.io/datasets/>
 - <https://chrsmrrs.github.io/datasets/>

References

1. Graph representation learning (chap 5), William Hamilton
 - https://www.cs.mcgill.ca/~wlh/grl_book/files/GRL_Book.pdf
2. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, Bronstein et al, 2022
 - <https://arxiv.org/abs/2104.13478>
3. A Comprehensive Survey on Graph Neural Networks, Wu et al, 2021
 - <https://arxiv.org/abs/1901.00596>
4. Graph neural networks: A review of methods and applications, Zhou et al, 2020.
 - <https://arxiv.org/pdf/1812.08434.pdf>

References

5. How powerful are graph neural networks? Xu et al, 2019
 - <https://arxiv.org/pdf/1810.00826.pdf>
6. Graph Self-Supervised learning: A survey, Liu et al, 2023
 - <https://arxiv.org/pdf/2103.00111.pdf>
7. Self-supervised learning of graph neural networks: A unified review, Xie et al., 2022
 - <https://arxiv.org/pdf/2102.10757.pdf>