

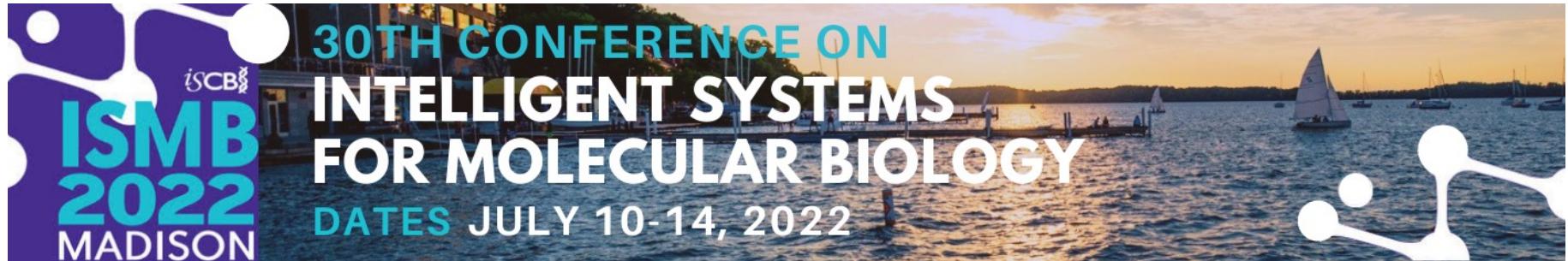
# Towards Precision Medicine with Graph Representation Learning

Michelle M. Li & Marinka Zitnik

Department of Biomedical Informatics  
Broad Institute of Harvard and MIT  
Harvard Data Science

[zitniklab.hms.harvard.edu/biomedgraphml](http://zitniklab.hms.harvard.edu/biomedgraphml)





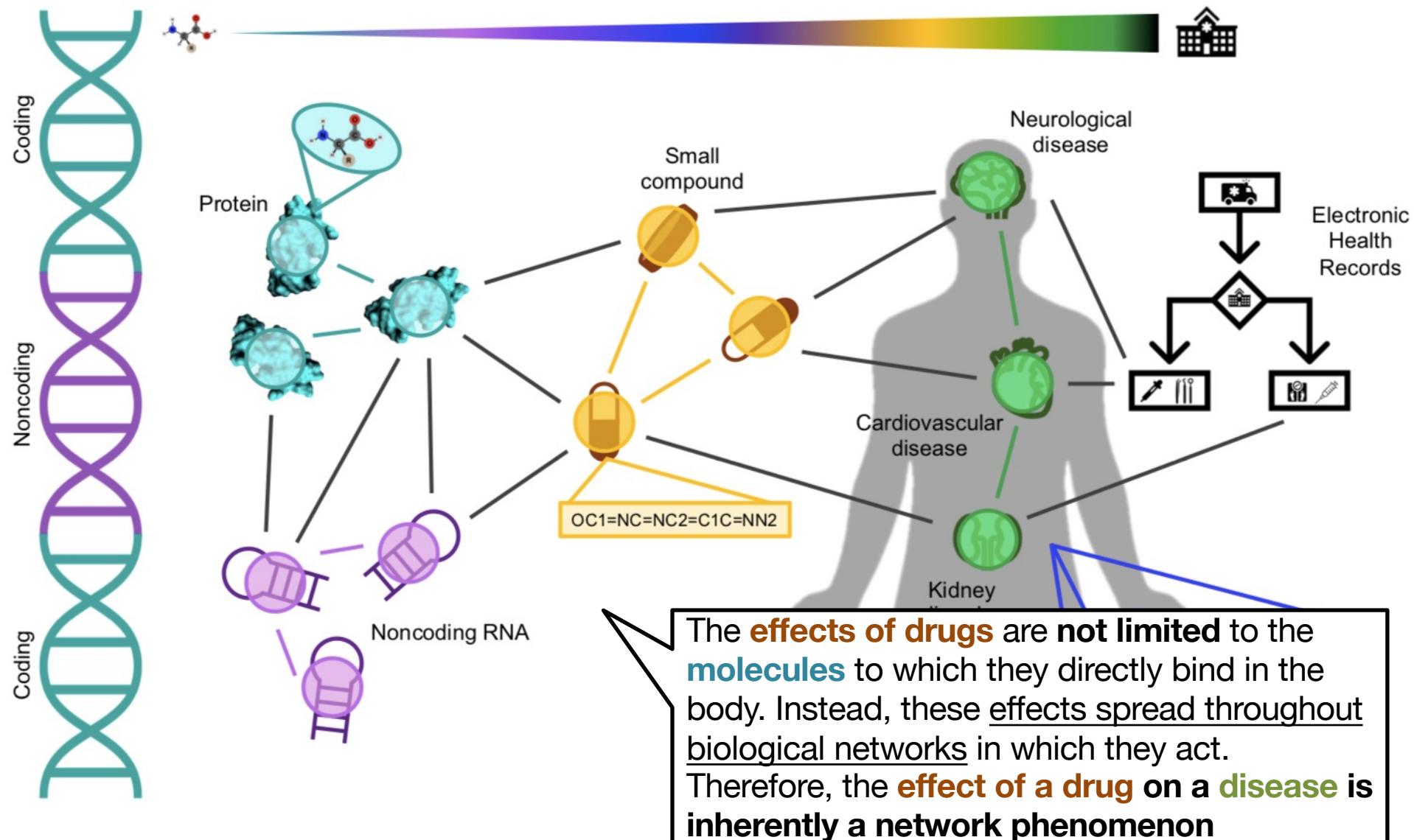
## Tutorial VT4

July 7, 2022 at 9am – 1pm CDT

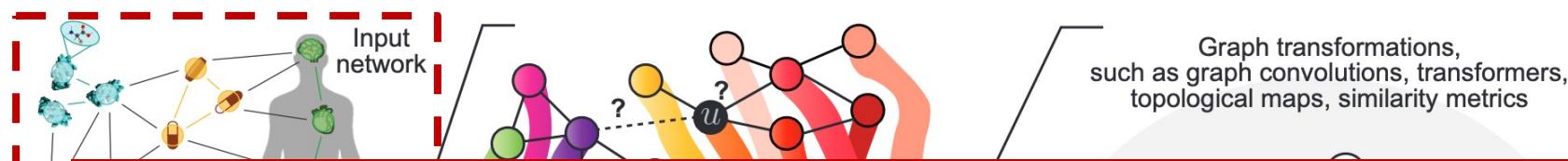


All tutorial materials are available at  
[zitniklab.hms.harvard.edu/biomedgraphml](http://zitniklab.hms.harvard.edu/biomedgraphml)

# Biology is interconnected



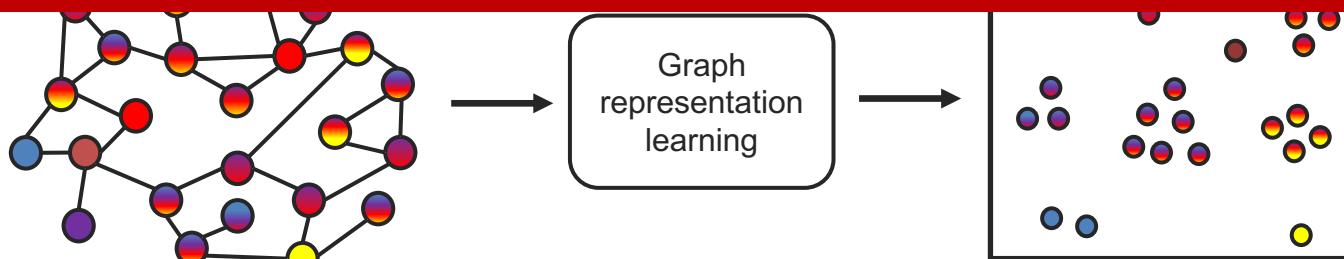
# Graph representation learning realizes key network principles for data-rich biomedicine



Cellular components associated with a specific disease (phenotype) show a tendency to cluster in the same network neighborhood



Deep graph representation learning methods are well-suited for the analysis of biological networks



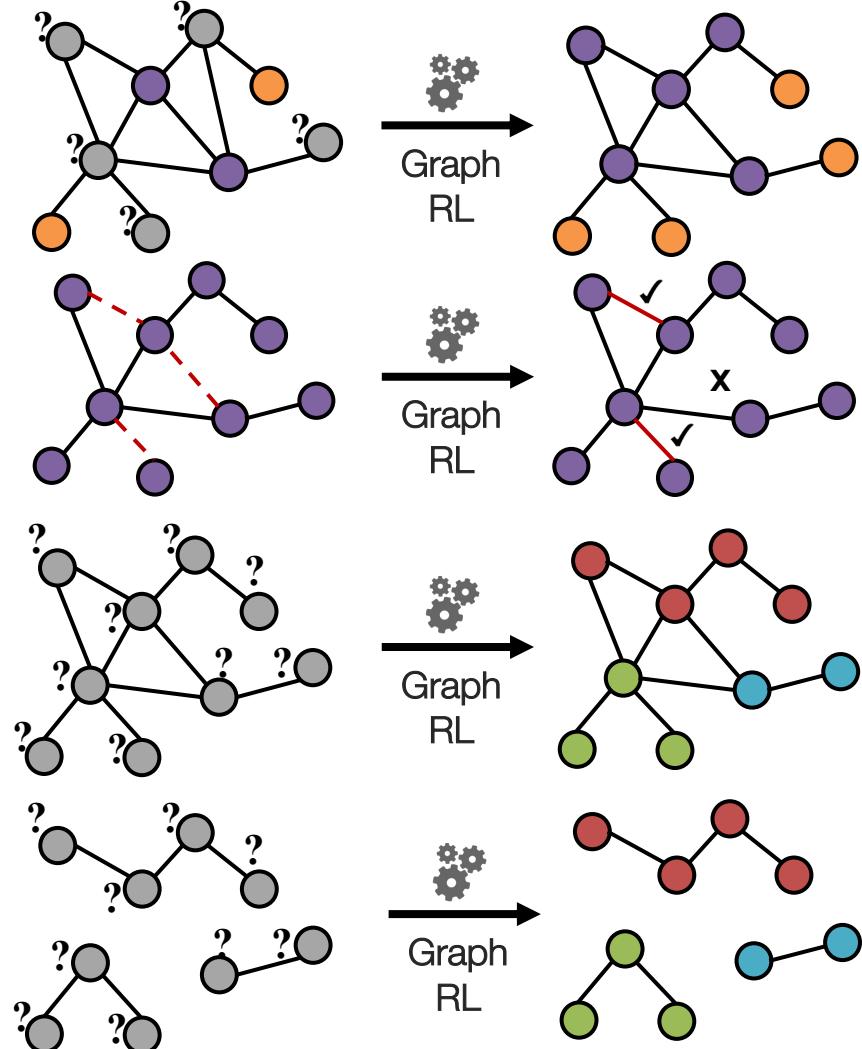
# This Tutorial



1. Methods: Network diffusion, shallow network embeddings, graph neural networks, equivariant neural networks
2. Applications: Fundamental biological discoveries and precision medicine
3. Hands-on exercises: Demos, implementation details, tools, and tips

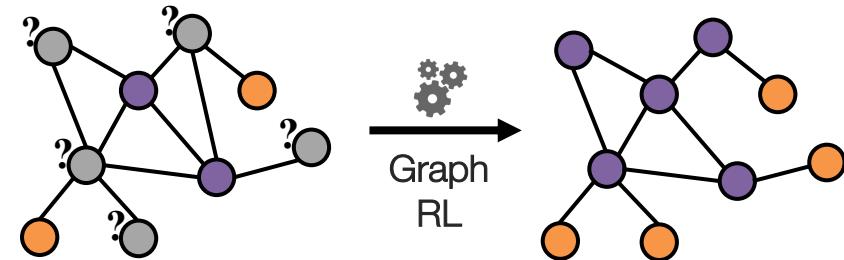
# Graph representation learning tasks

- Node-level
  - Characteristics of a given node
- Edge-level
  - Whether or how two nodes are connected
- Subgraph-level
  - How clusters of nodes interact with each other and the rest of the graph
- Graph-level
  - Similarity of graph to other graphs



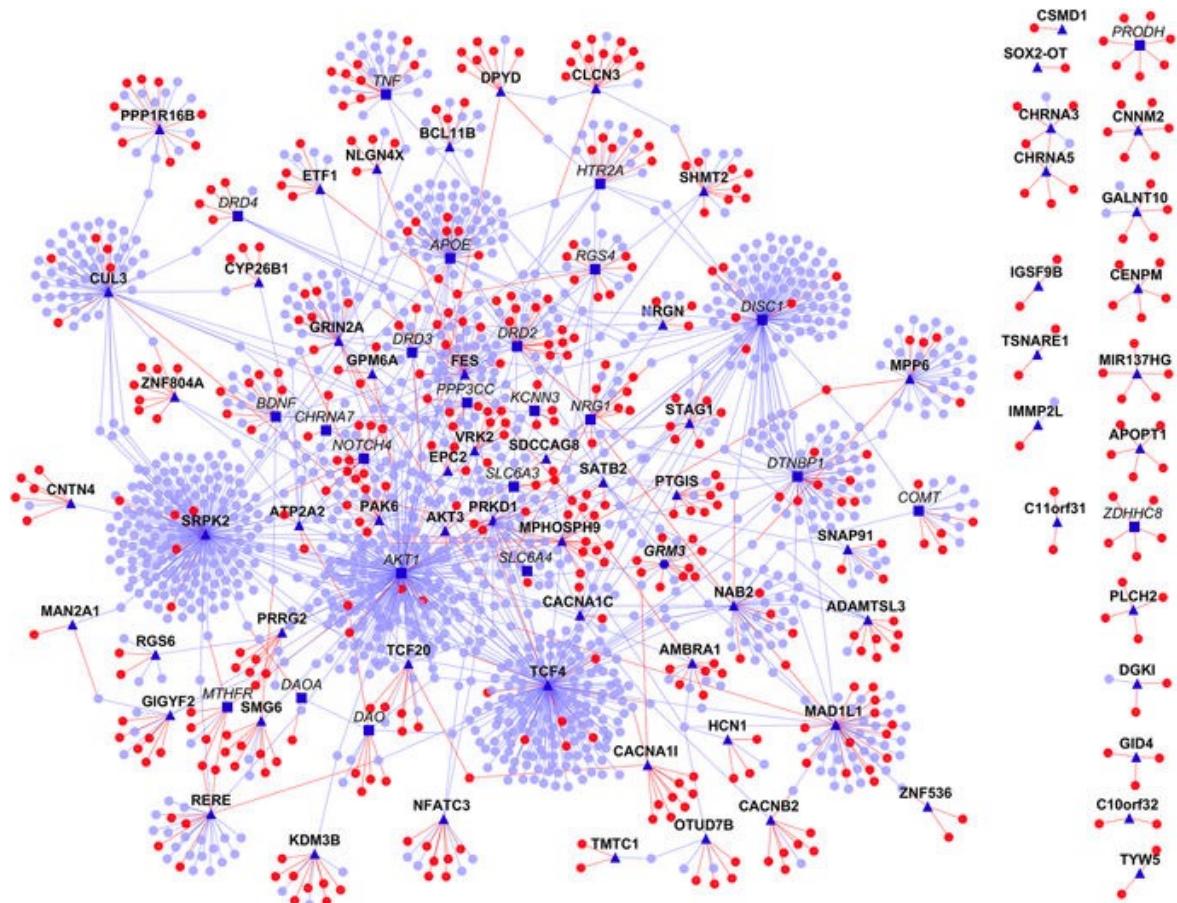
# Graph representation learning tasks

- Node-level
  - Characteristics of a given node



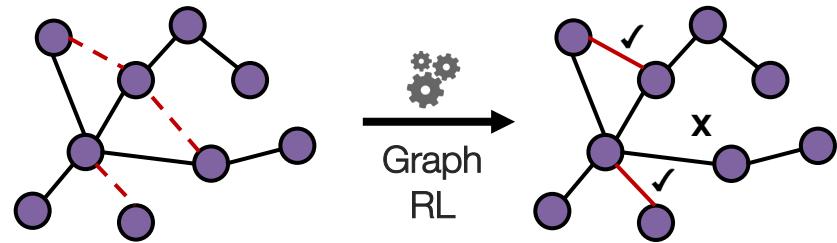
# Node classification: Example

Classifying the function of proteins in the interactome!



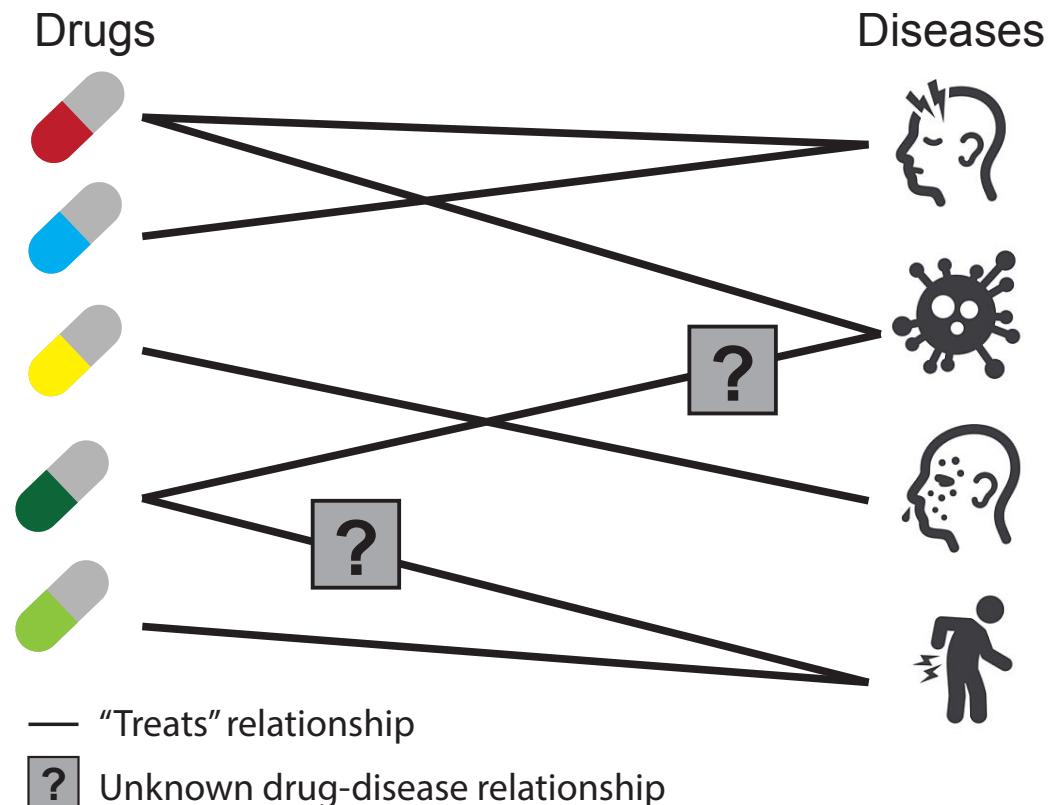
# Graph representation learning tasks

- Edge-level
  - Whether or how two nodes are connected



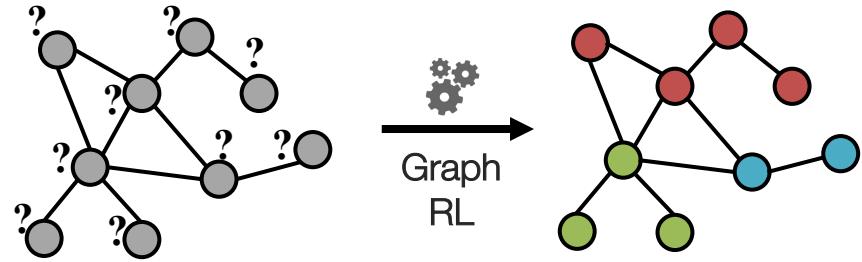
# Link prediction: Example

Predicting  
which  
diseases a  
new molecule  
might treat!



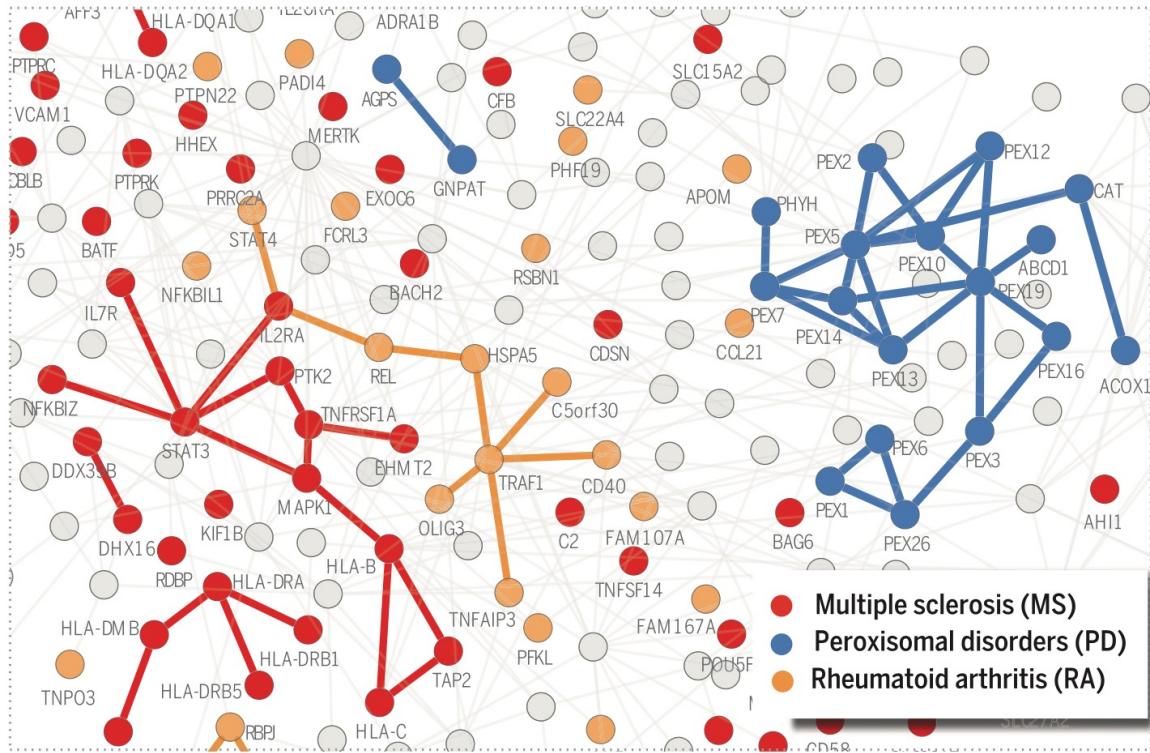
# Graph representation learning tasks

- Subgraph-level
  - How clusters of nodes interact with each other and the rest of the graph



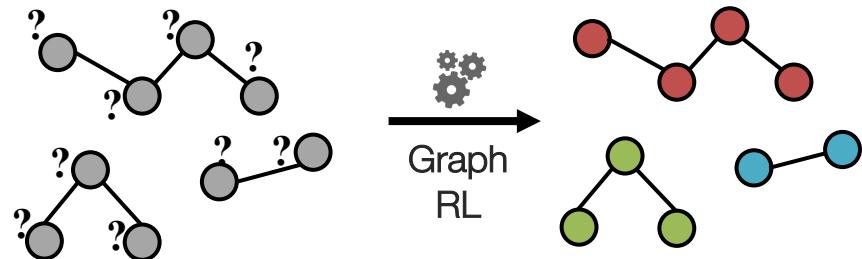
# Subgraph classification: Example

Identifying  
disease  
proteins in the  
interactome!



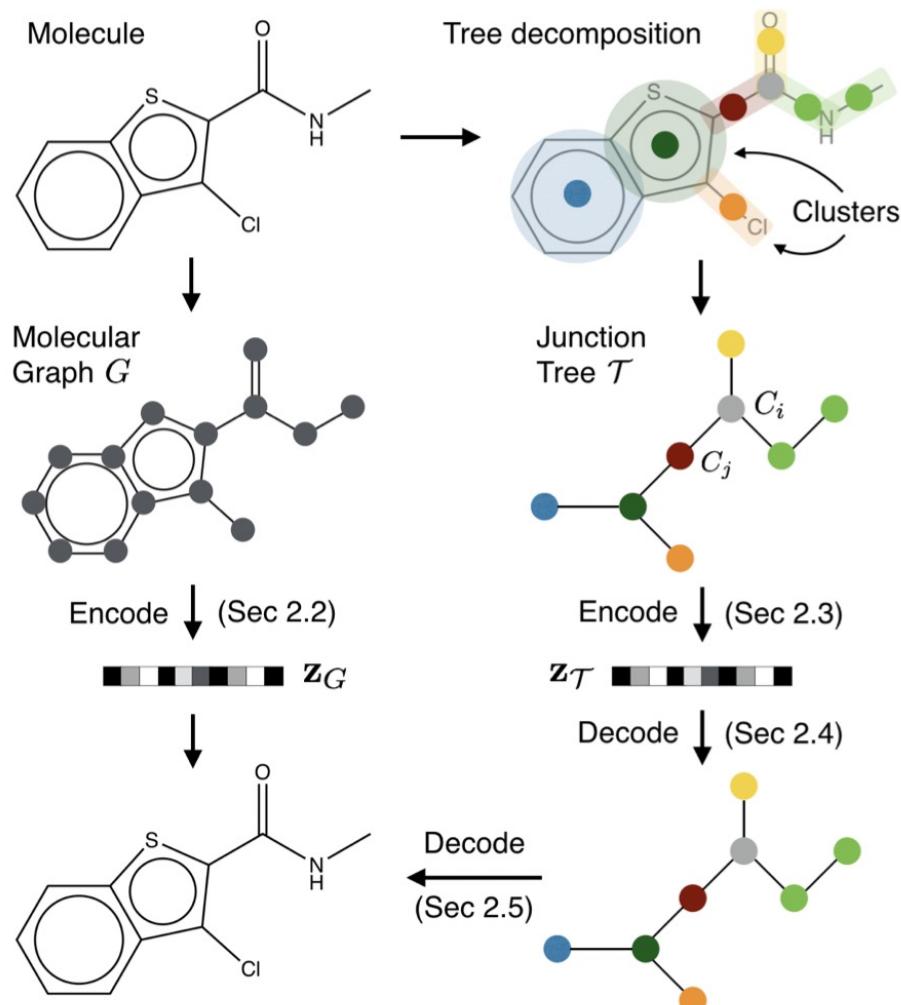
# Graph representation learning tasks

- Graph-level
  - Similarity of graph to other graphs



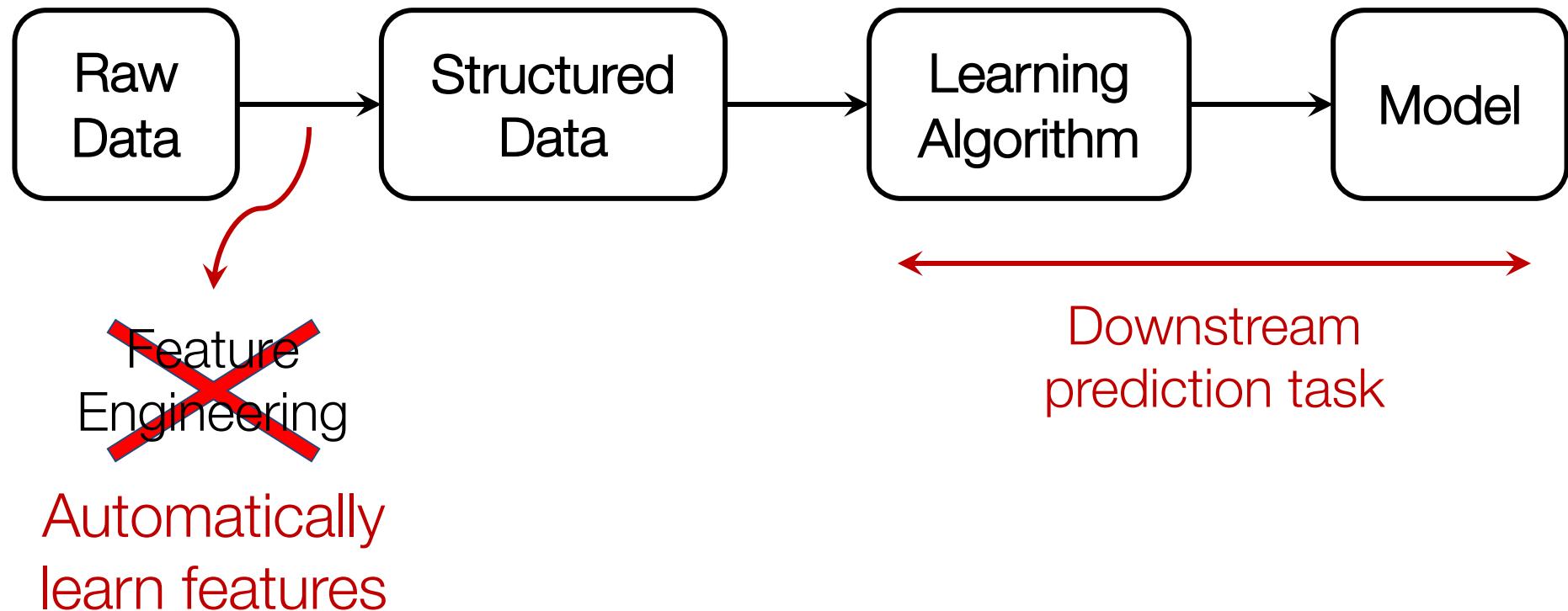
# Graph classification: Example

Designing new  
small molecule  
compounds to  
treat a disease!



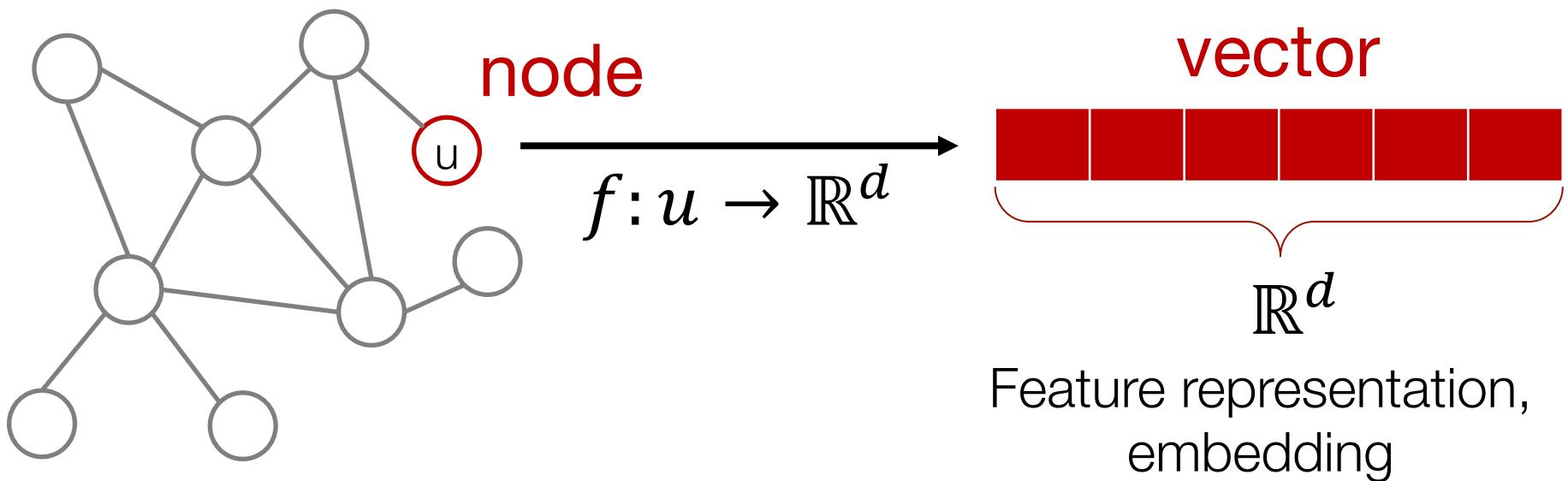
# Predictive modeling lifecycle

(Supervised) Machine learning lifecycle: This feature, that feature. **Every single time!**



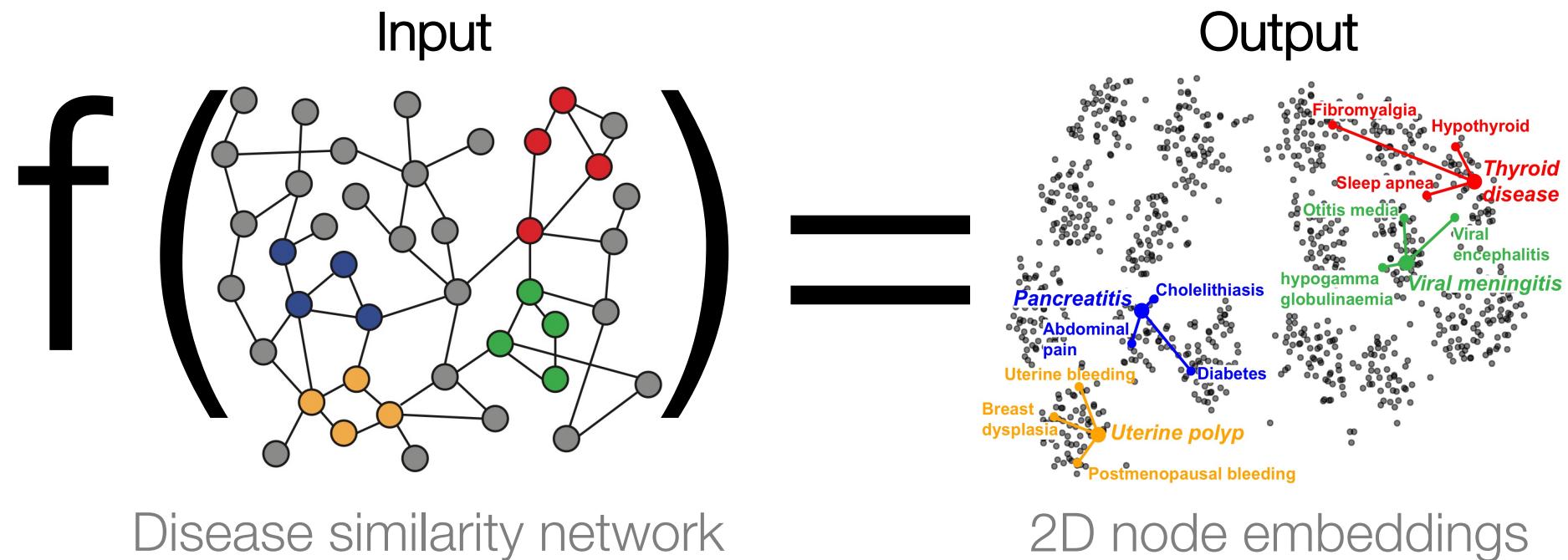
# Feature learning in graphs

Goal: Efficient task-independent feature learning for machine learning in networks!



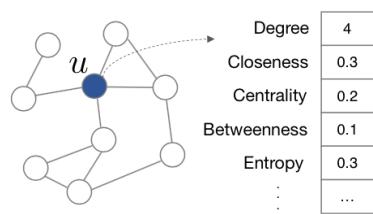
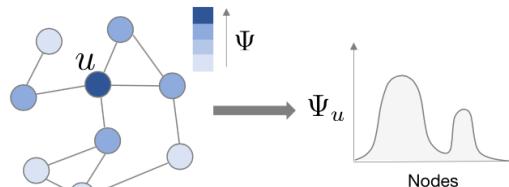
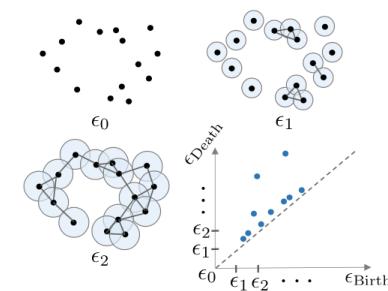
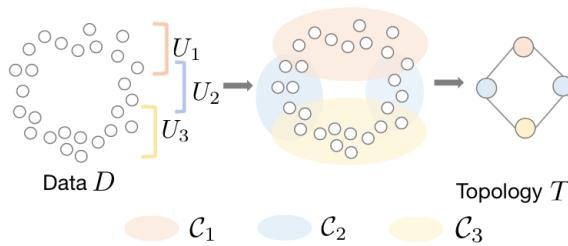
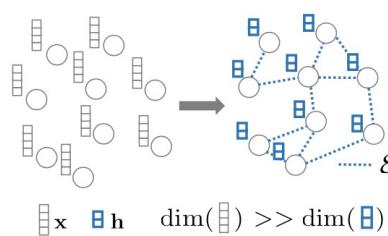
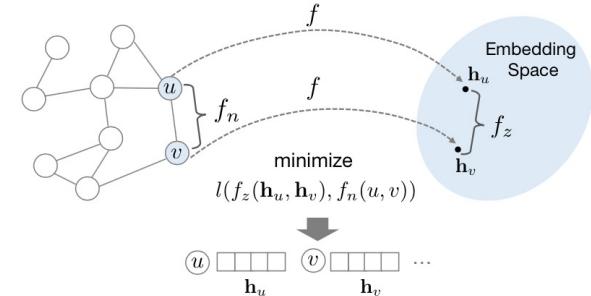
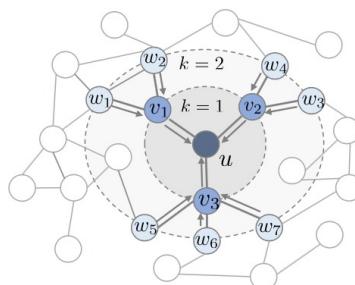
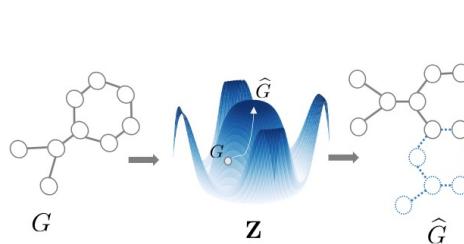
# Embedding nodes

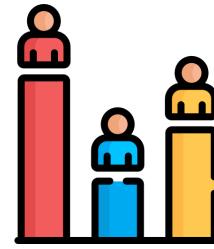
Intuition: Map nodes to embeddings such that similar nodes in graph are embedded close by



How to learn mapping function  $f$ ?

# Predominant graph learning paradigms

**a** Graph theoretic techniques**b** Random walks and diffusion**c** Persistent homology**d** Geometrical and topological representations**e** Manifold learning**f** Shallow network embeddings**g** Graph neural networks**h** Graph generative models



Time for a poll question about...

# GRAPH MACHINE LEARNING TASKS

1. Which of the following is an example of a node-level task? *Multiple choice*
2. Which of the following is an example of a graph-level task? *Multiple choice*

# Poll Question

Which of the following is an example of a node-level task? *Multiple choice*

1. Identifying a collection of disease proteins in the interactome
2. Predicting which diseases a new molecule might treat
3. Designing new small molecule compounds to treat a disease
4. Classifying the function of proteins in the interactome

# Poll Question

Which of the following is an example of a node-level task? *Multiple choice*

1. Identifying a collection of disease proteins in the interactome
2. Predicting which diseases a new molecule might treat
3. Designing new small molecule compounds to treat a disease
4. Classifying the function of proteins in the interactome

# Poll Question

Which of the following is an example of a graph-level task? *Multiple choice*

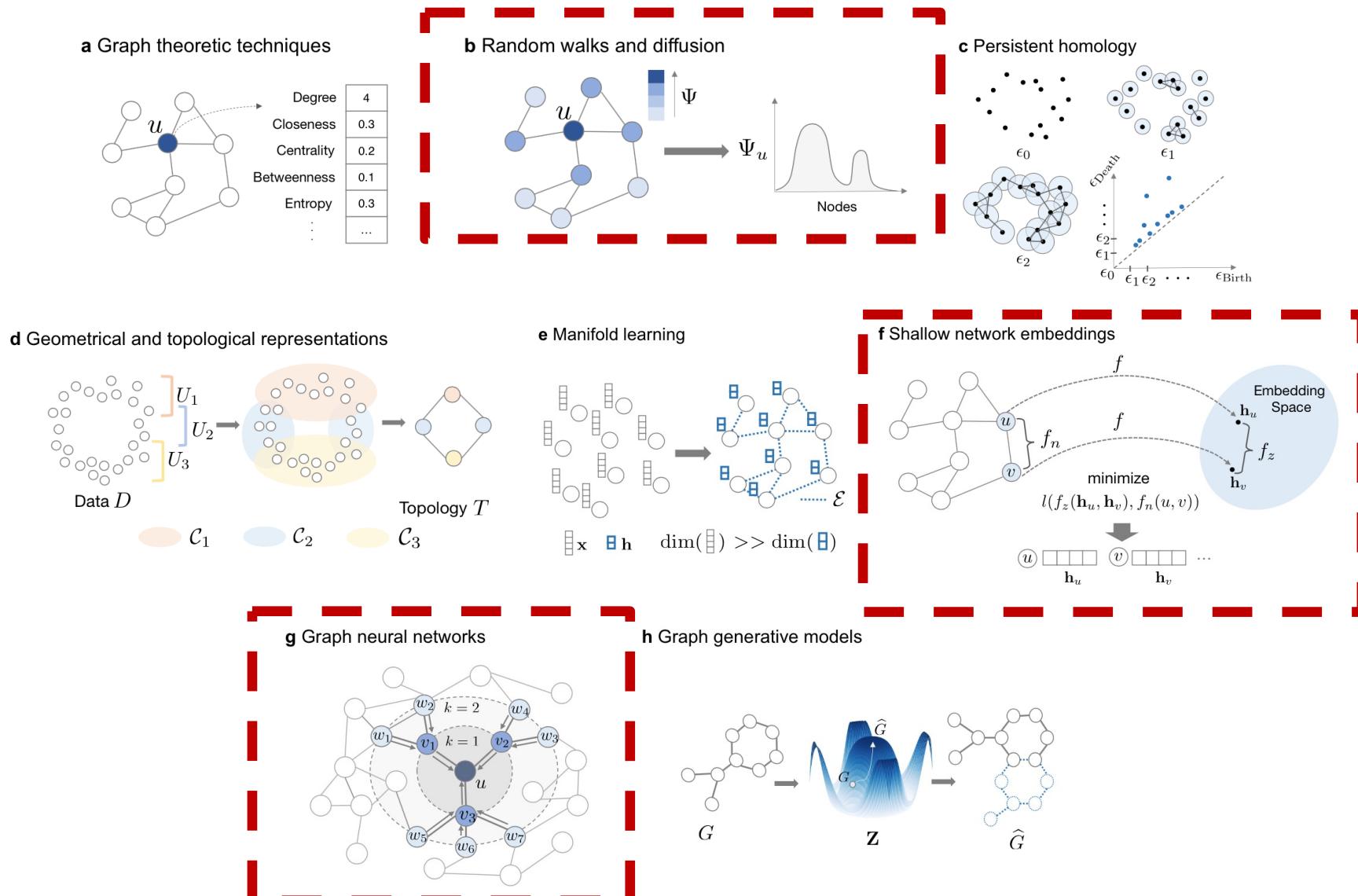
1. Identifying a collection of disease proteins in the interactome
2. Predicting which diseases a new molecule might treat
3. Designing new small molecule compounds to treat a disease
4. Classifying the function of proteins in the interactome

# Poll Question

Which of the following is an example of a graph-level task? *Multiple choice*

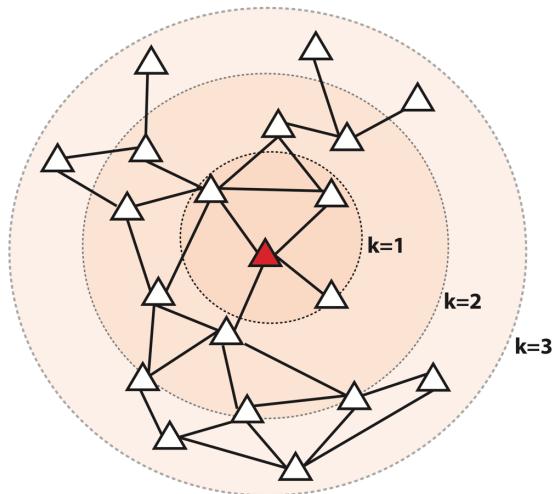
1. Identifying a collection of disease proteins in the interactome
2. Predicting which diseases a new molecule might treat
3. Designing new small molecule compounds to treat a disease
4. Classifying the function of proteins in the interactome

# Predominant graph learning paradigms



# Random walks and diffusion

- Nodes in a graph **influence** each other along paths
- **Diffusion** measures these spreads of influences

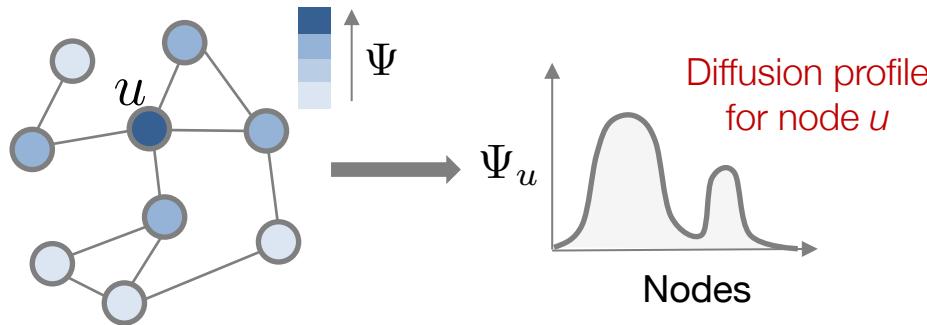


**Red:** Target node  
 $k = 1$ : 1-hop neighbors  
 $k = 2$ : 2-hop neighbors  
 $k = 3$ : 3-hop neighbors

- Intuition
  - Capture the local connectivity patterns for each node
  - Define node similarity function based on higher-order neighborhoods

# Random walks and diffusion

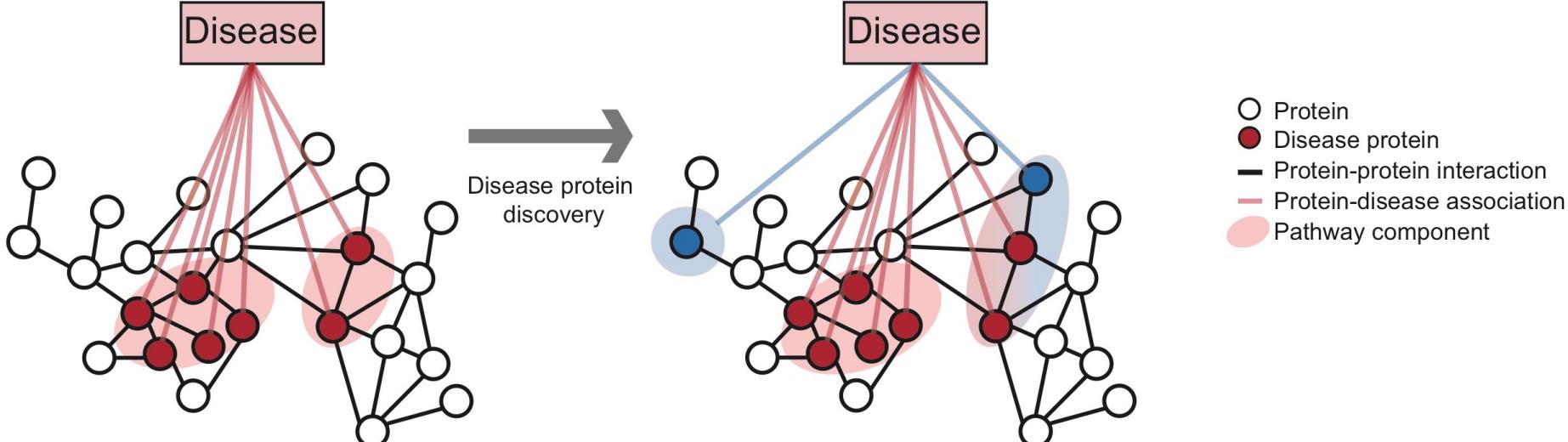
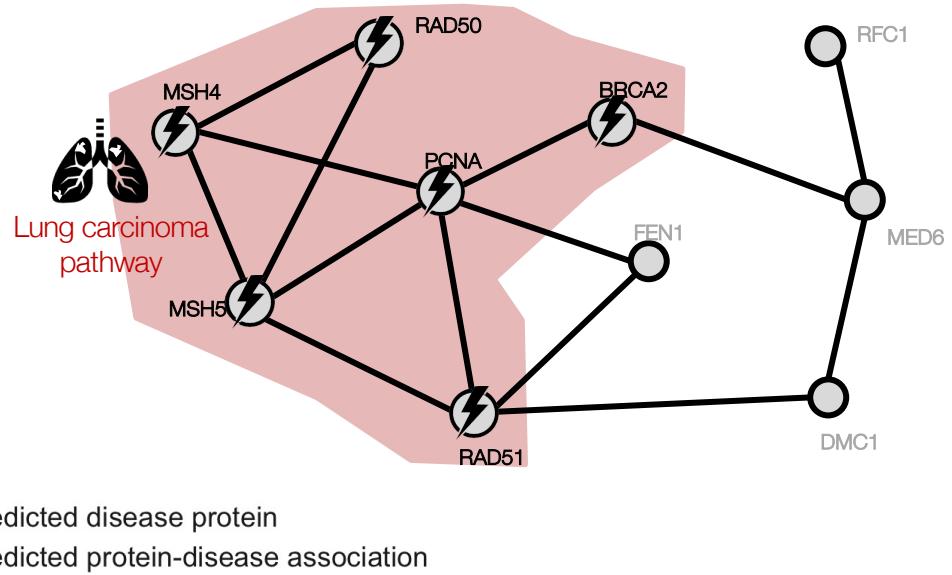
- Method: Diffusion state distance (DSD)
  - Simulate random walks from source node  $u$
  - Count the number of random walks of length  $k$  that start at  $u$  and visit a destination node  $v$
- Node representation: Each node  $u$  has vector  $\Psi_u$  that represents its influence on its  $k$ -hop neighborhood



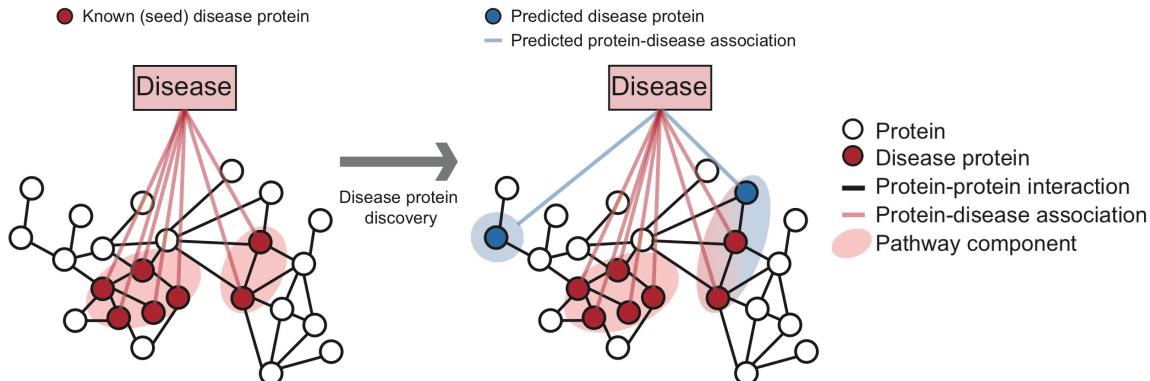
- Comparison: Calculate whether nodes  $u$  and  $v$  have similar local connectivity by  $DSD(u, v) = \|\Psi_u - \Psi_v\|_1$

# Application: Identify disease pathways

- **Pathway:** Subnetwork of interacting proteins associated with a disease

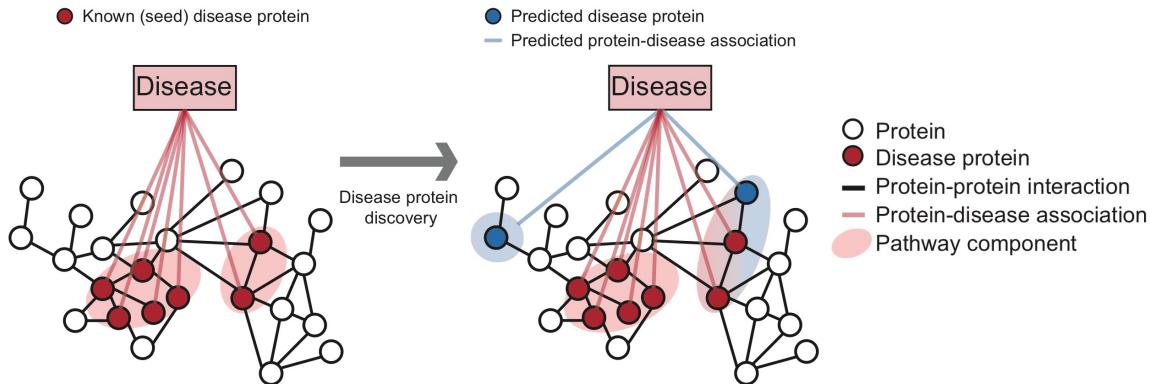


# Disease pathway dataset

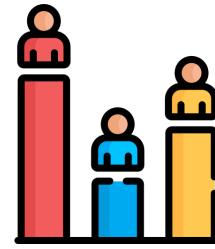


- Protein-protein interaction (PPI) network culled from 15 knowledge databases:
  - 350k physical interactions
    - Examples: metabolic enzyme-coupled interactions, signaling interactions, protein complexes
  - All protein-coding human genes (21k)
- Protein-disease associations
  - 21k associations split among 519 diseases
- Multi-label node classification
  - Every node (i.e. protein) can have 0, 1, or more labels (i.e. disease associations)

# Experimental setup



- Two main stages:
  1. Take the PPI network and use DSD to compute a **vector representation for every node**
  2. For each disease, fit a logistic regression **classifier** that predicts disease proteins based on the vector representations:
    - Train the classifier using training proteins
    - Predict disease proteins in the test set (i.e. the **probability** that a particular protein is associated with the disease)



Time for a poll question about...

# RANDOM WALKS & DIFFUSION

1. Which of the following describes how random walks can be used to generate representations of nodes? *Multiple choice*

# Poll Question

Which of the following describes how random walks can be used to generate representations of nodes?

*Multiple choice*

Each element (denoted by  $v$ , corresponding to node  $v$ ) of the embedding for node  $u$  represents the \_\_\_\_\_ between  $u$  and  $v$ .

1. ...number of hops...
2. ...connection...
3. ...collaboration...
4. ...influence...

# Poll Question

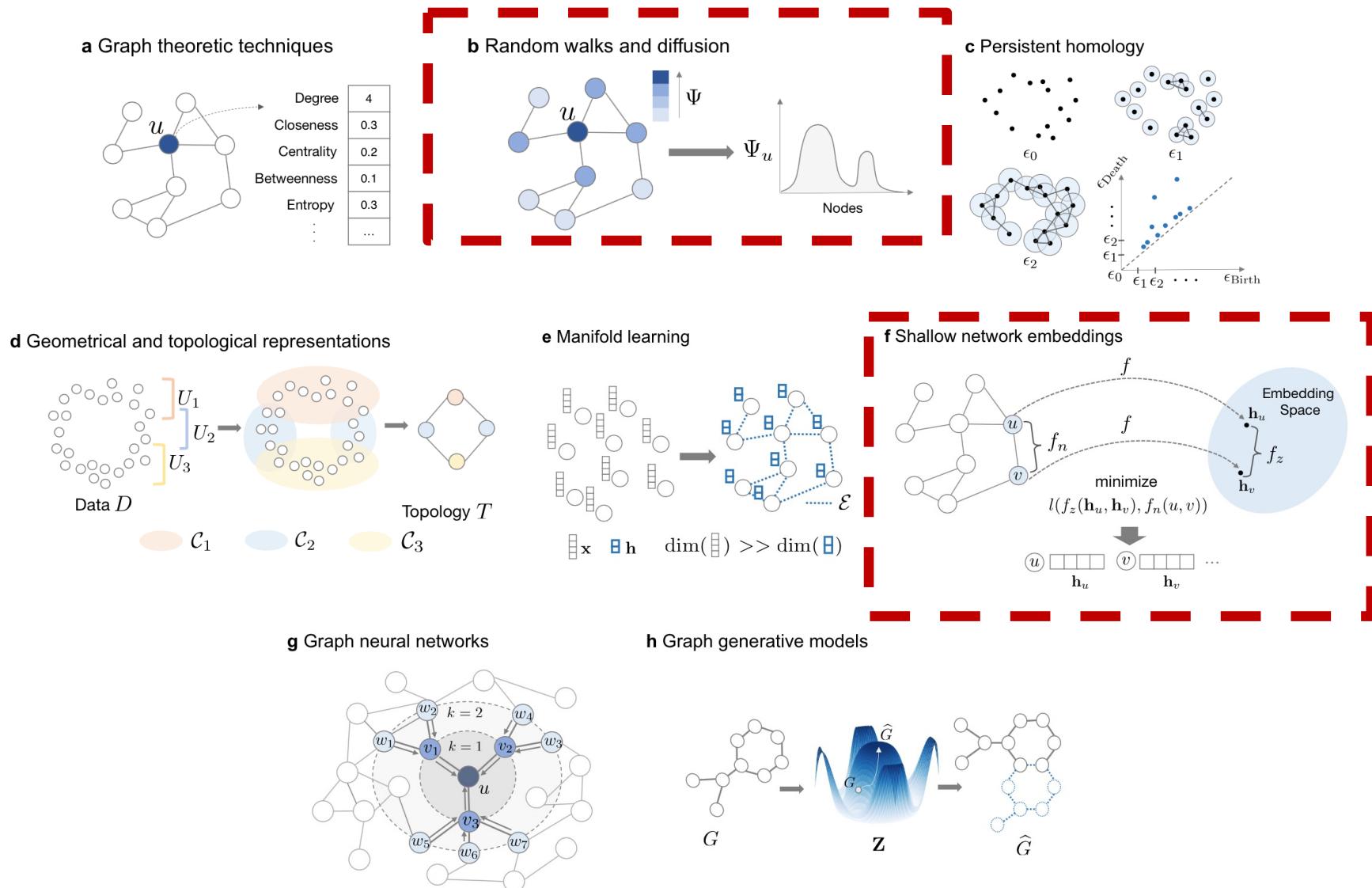
Which of the following describes how random walks can be used to generate representations of nodes?

*Multiple choice*

Each element (denoted by  $v$ , corresponding to node  $v$ ) of the embedding for node  $u$  represents the \_\_\_\_\_ between  $u$  and  $v$ .

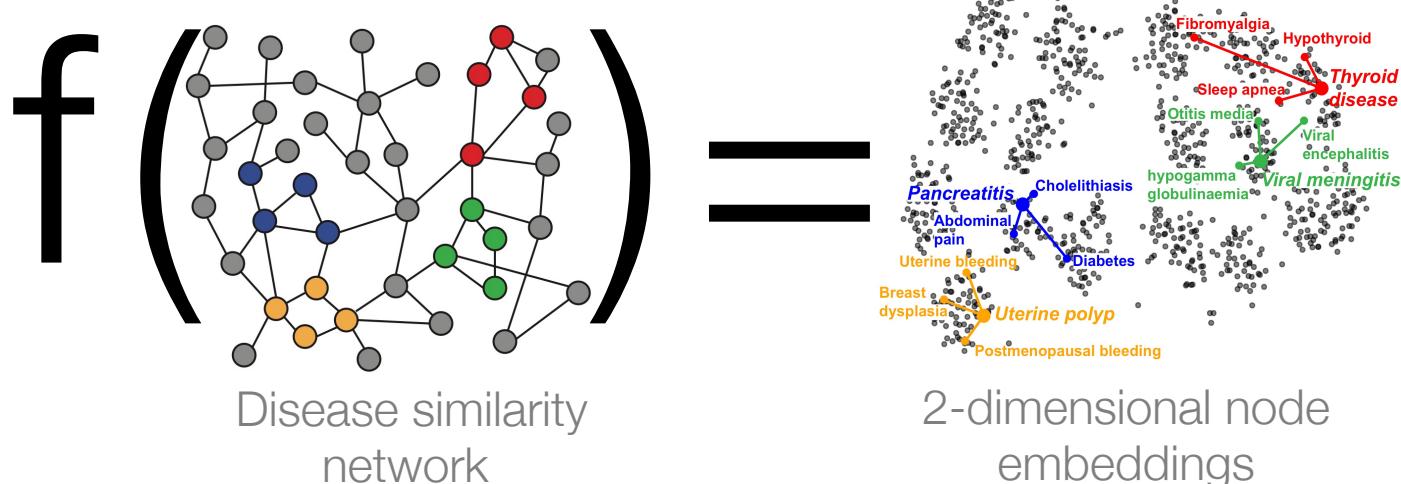
1. ...number of hops...
2. ...connection...
3. ...collaboration...
4. ...influence...

# Predominant graph learning paradigms



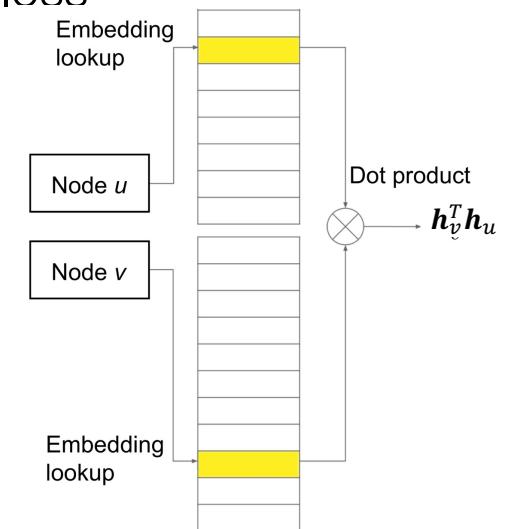
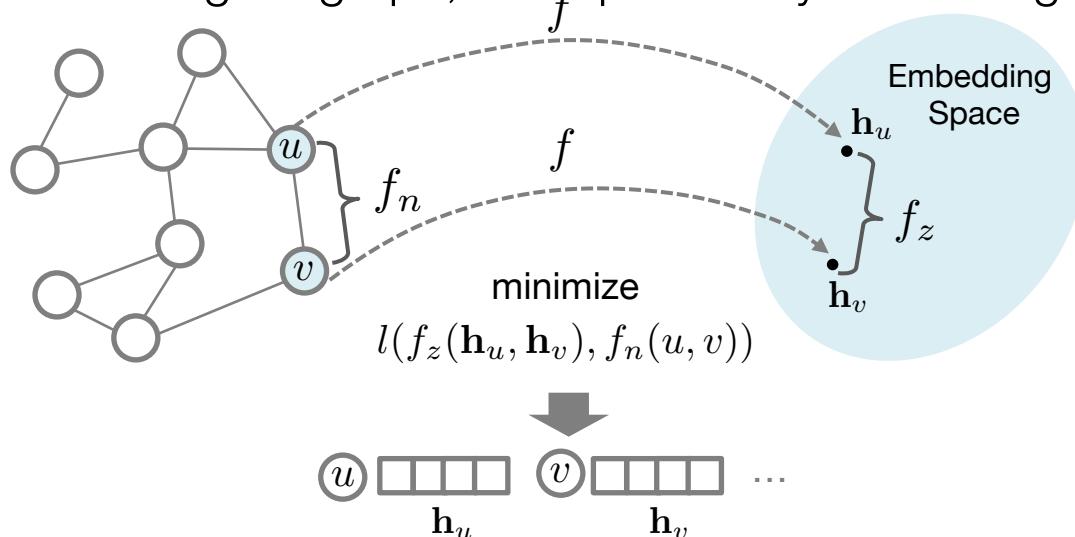
# Shallow network embeddings

- **Intuition:** Map nodes to  $d$ -dimensional embeddings such that similar nodes in the graph are embedded close together
- Assume we have a graph  $G$ :
  - $V$  is the vertex set
  - $A$  is the adjacency matrix (assume binary)
  - No node features or extra information is used!



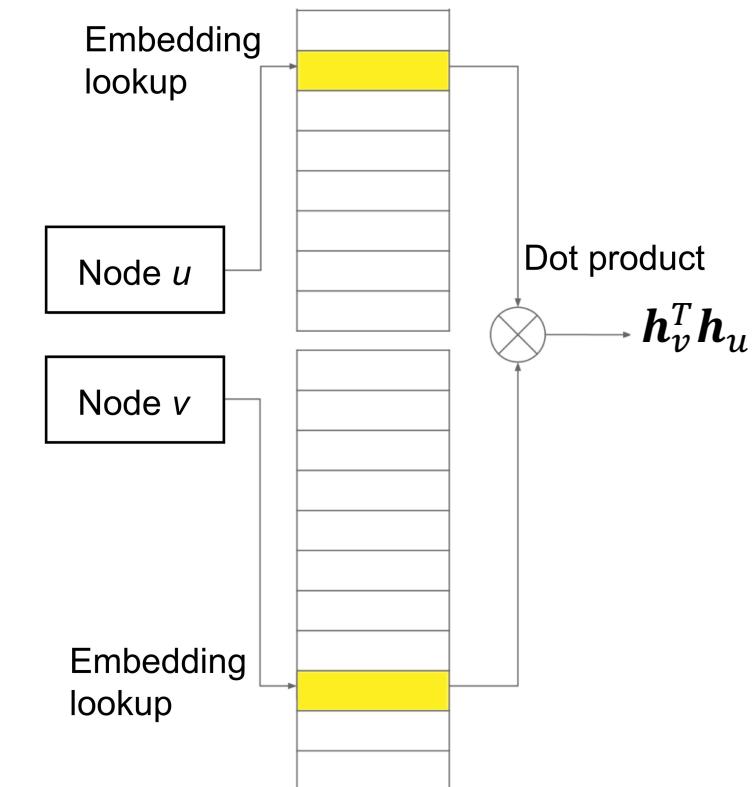
# Shallow network embeddings

- **Goal:** Similarity in the embedding space approximates similarity in the network
- **Three main stages:**
  1. Given a pair of nodes  $u, v$  in network  $G$ , obtain function  $f$  to map these nodes to an embedding space to generate  $\mathbf{h}_u$  and  $\mathbf{h}_v$
  2. Define network similarity  $f_n(u, v)$  and embedding similarity  $f_z(\mathbf{h}_u, \mathbf{h}_v)$
  3. Define loss  $l$  to measure whether embedding preserves distance in original graph, and optimize by minimizing the loss



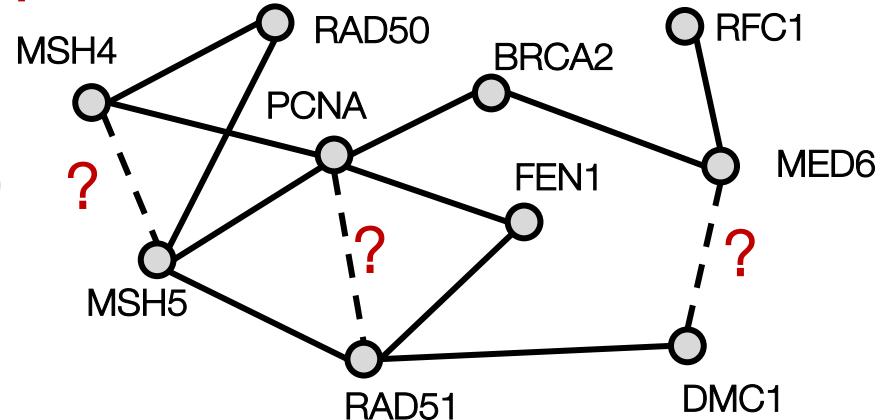
# Shallow network embeddings

- Summary:
  - One-layer of data transformation
  - A single hidden layer maps node  $u$  to embedding  $\mathbf{h}_u$  via function  $f$
- Limitations:
  - $O(|V|)$  parameters are needed:
    - No sharing of parameters between nodes
    - Every node has its own unique embedding
  - Inherently “transductive”
    - Cannot generate embeddings for nodes **not** seen during training
  - Do not incorporate node features
    - Many graphs have features that we can and should leverage



# Application: Predict protein interactions

- Human PPI network:
  - Experimentally validated physical protein-protein interactions (BioGRID)
- Link prediction: Given two proteins, predict probability that they interact



How to address tasks involving pairs of nodes (e.g., link prediction)?

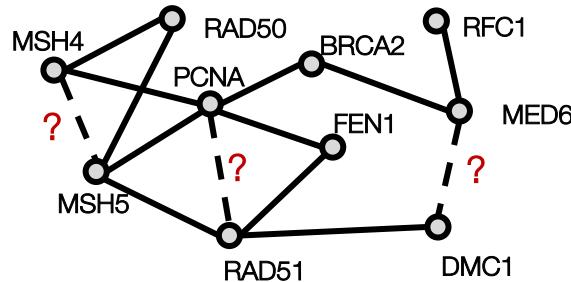
- Given  $u$  and  $v$ , define an operator  $g$  that generates an embedding for pair  $(u, v)$ :

$$\mathbf{h}_{(u,v)} = g(u, v)$$

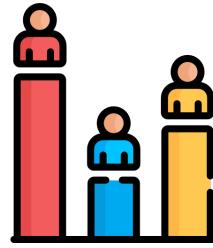
- Examples of choices for  $g$

Scoring node pairs	Definition
(a) Average	$[\mathbf{z}_u \boxplus \mathbf{z}_v]_i = \frac{\mathbf{z}_u(i) + \mathbf{z}_v(i)}{2}$
(b) Hadamard	$[\mathbf{z}_u \boxdot \mathbf{z}_v]_i = \mathbf{z}_u(i) * \mathbf{z}_v(i)$
(c) Weighted-L1	$\ \mathbf{z}_u \cdot \mathbf{z}_v\ _{\bar{1}i} =  \mathbf{z}_u(i) - \mathbf{z}_v(i) $
(d) Weighted-L2	$\ \mathbf{z}_u \cdot \mathbf{z}_v\ _{\bar{2}i} =  \mathbf{z}_u(i) - \mathbf{z}_v(i) ^2$

# Experimental Setup



- We are given a PPI network with a certain fraction of edges removed:
  - Remove about 50% of edges
  - Randomly sample an equal number of node pairs that have no edge connecting them
  - Explicitly removed edges and non-existent (or false) edges form a balanced test data set
- Two main stages:
  1. Learn an embedding for every node in the filtered PPI network
  2. Predict a score for every protein pair in the test set based on the embeddings



Time for a poll question about...

# SHALLOW NETWORK EMBEDDINGS

1. Which of the following are true about shallow network embeddings? *Select many*

# Poll Question

Which of the following are true about shallow network embeddings? *Select many*

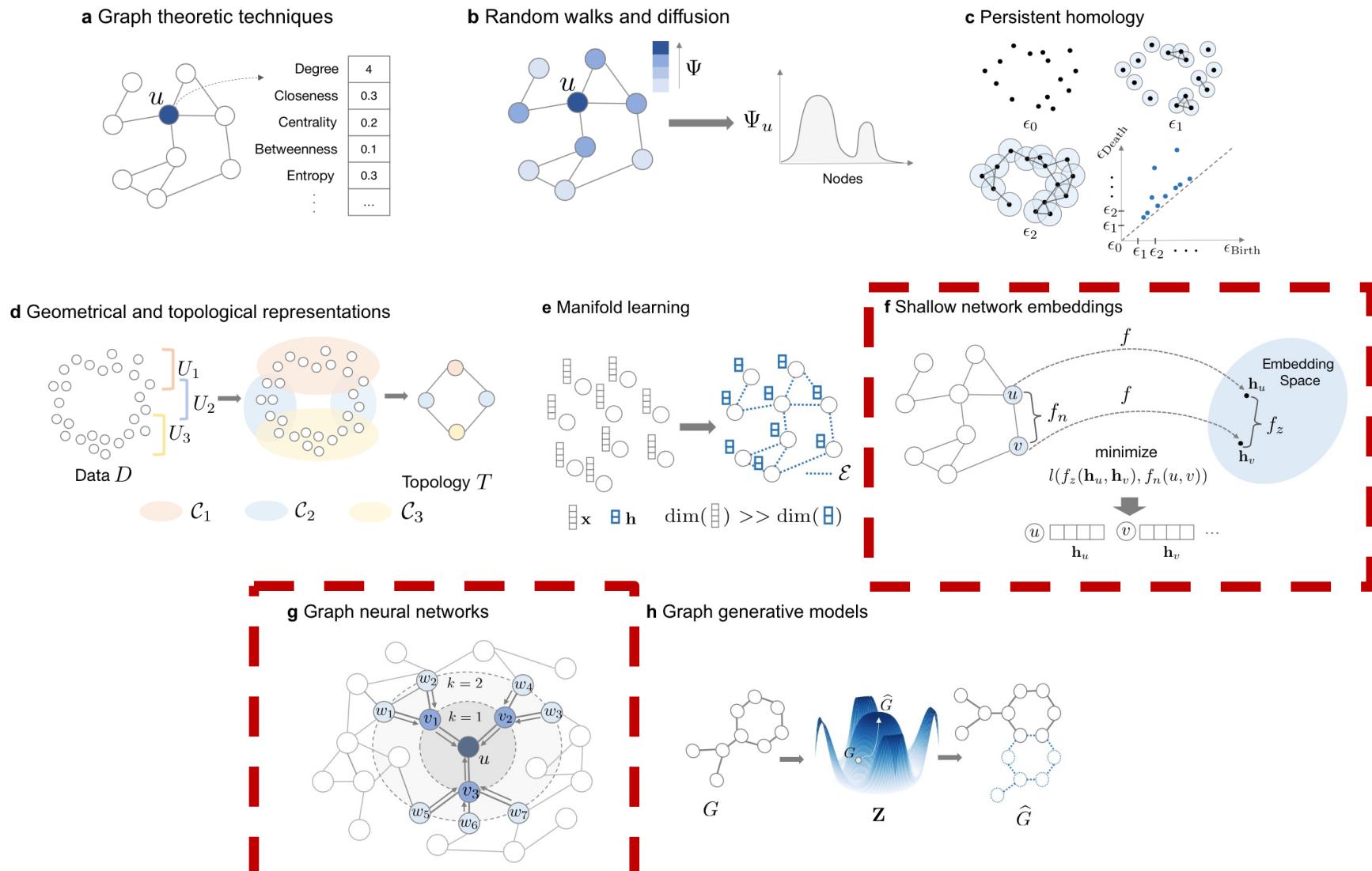
1. Similarity in the embedding space approximates similarity in the network
2. Requires feature engineering
3. Requires a network similarity metric
4. Requires an embedding similarity metric

# Poll Question

Which of the following are true about shallow network embeddings? *Select many*

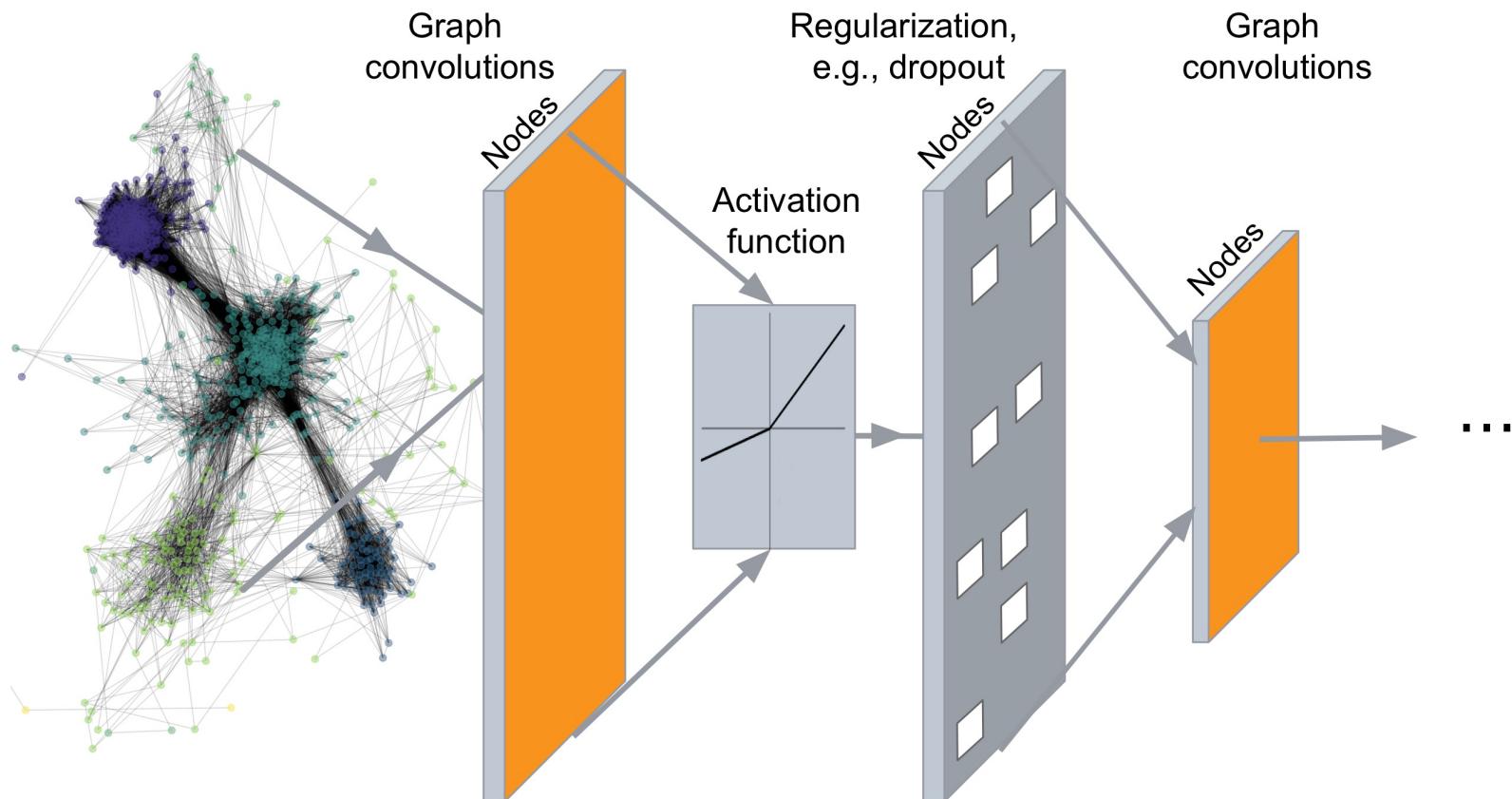
1. Similarity in the embedding space approximates similarity in the network
2. Requires feature engineering
3. Requires a network similarity metric
4. Requires an embedding similarity metric

# Predominant graph learning paradigms



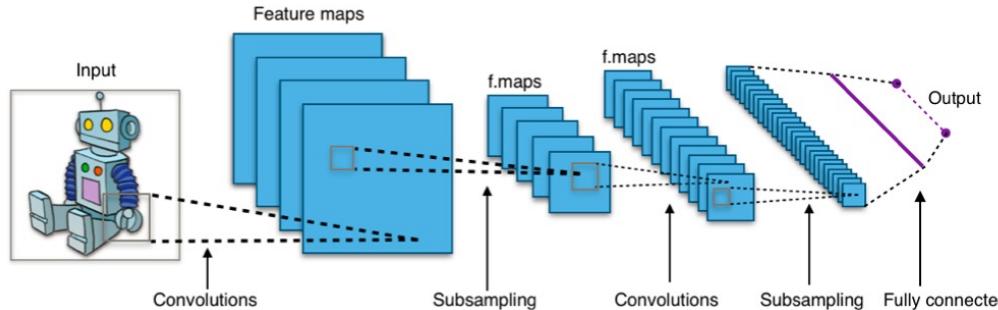
# Graph neural networks

- Encoder: Multiple layers of nonlinear transformation of graph structure

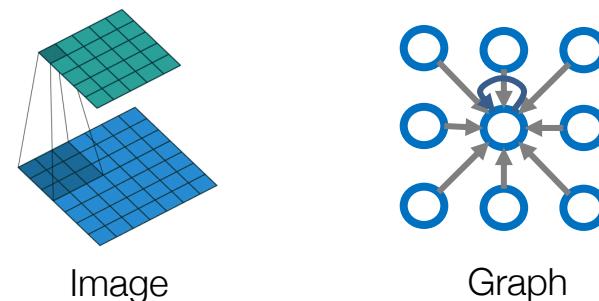


# Convolutional networks

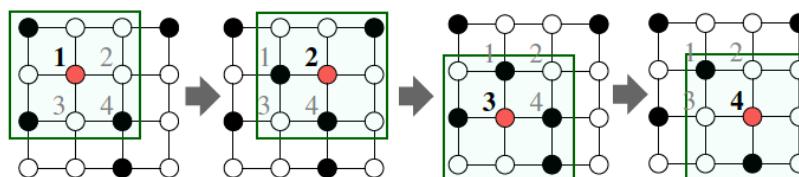
- Let's start with convolutional networks on an image:



- Single convolutional network with a 3x3 filter:

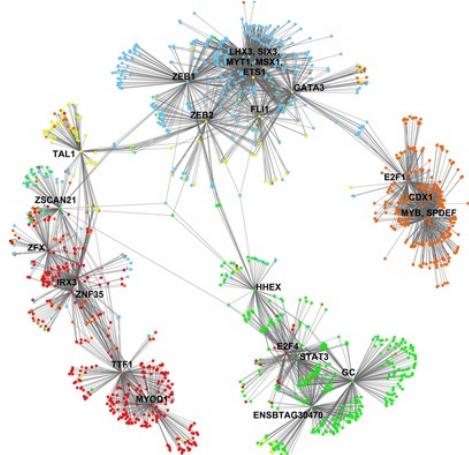


- Transform information (or messages) from the neighbors and combine them:  $\sum_i W_i h_i$

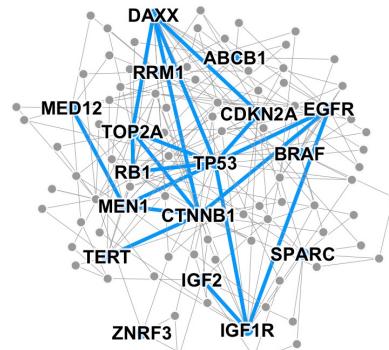


# Real world graphs

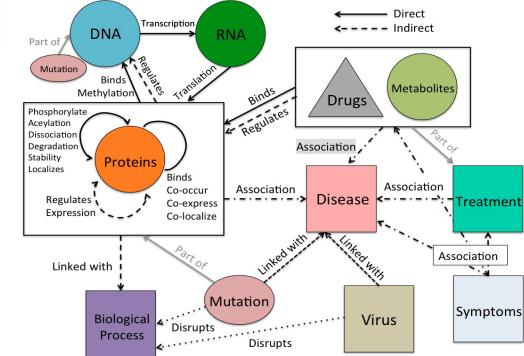
- But what if your graphs look like this?



Gene interaction network



Disease pathways



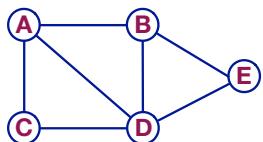
Biomedical knowledge graphs

- Examples:

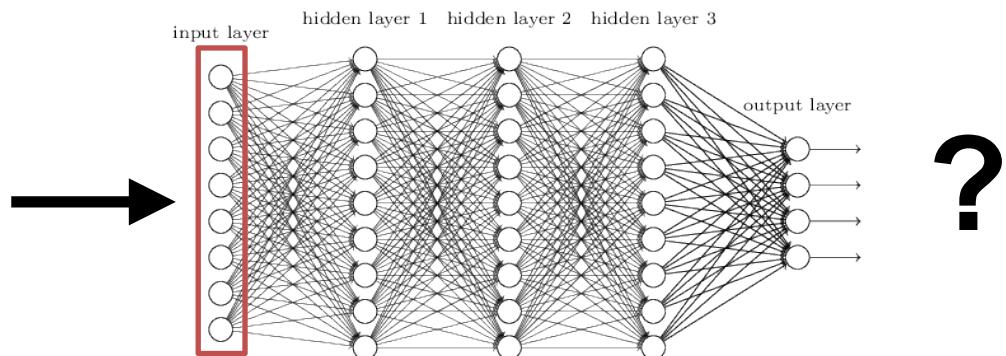
- Biological or medical networks
- Social networks
- Information networks
- Knowledge graphs
- Communication networks
- Web graphs
- ...

# Naïve approach

- Join adjacency matrix and features
- Feed them into a deep neural network:



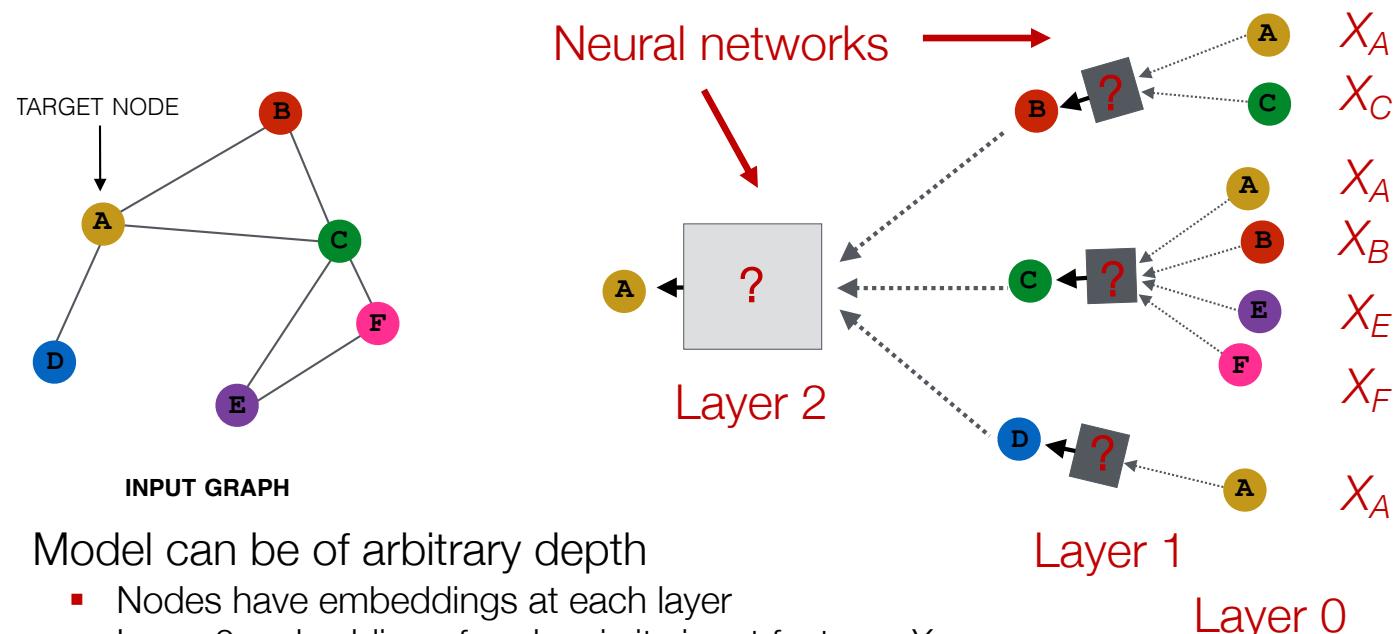
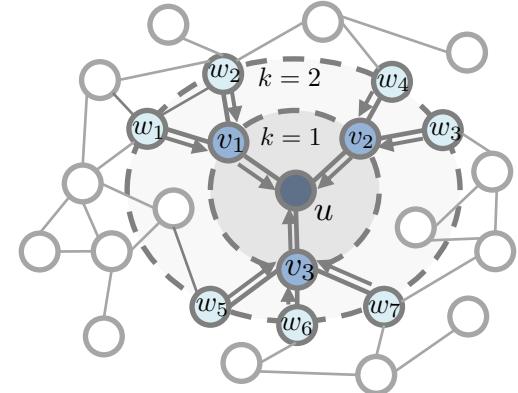
	A	B	C	D	E	Feat	
A	0	1	1	1	0	1	0
B	1	0	0	1	1	0	0
C	1	0	0	1	0	0	1
D	1	1	1	0	1	1	1
E	0	1	0	1	0	1	0



- Issues with this idea:
  - $O(N)$  parameters
  - Not applicable to graphs of different sizes
  - Not invariant to node ordering

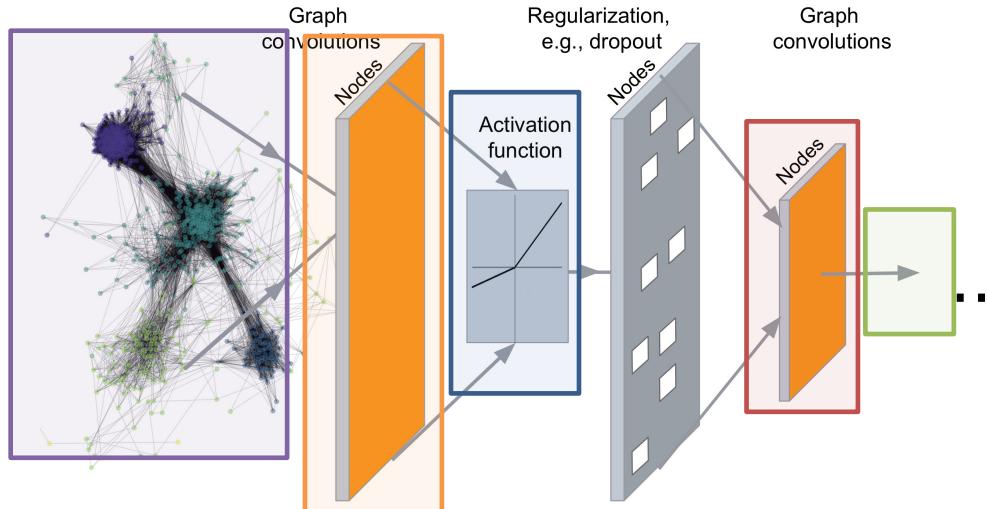
# Graph neural networks

- Intuition:
  - Each node's neighborhood defines a computational graph
  - Generate node embeddings based on local network neighborhoods
- Neighborhood aggregation:



- Model can be of arbitrary depth
  - Nodes have embeddings at each layer
  - Layer 0 embedding of node  $u$  is its input features  $X_u$
- **Basic neighborhood aggregation approach (i.e. □):** Average information from neighbors and apply a neural network

# Basic approach



$$\mathbf{h}_v^0 = \mathbf{x}_v$$

Initial 0-th layer embeddings are equal to node features

$$\mathbf{h}_v^k = \sigma \left( \mathbf{W}_k \left( \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right) \right), \quad \forall k \in \{1, \dots, K\}$$

Previous layer embedding of  $v$

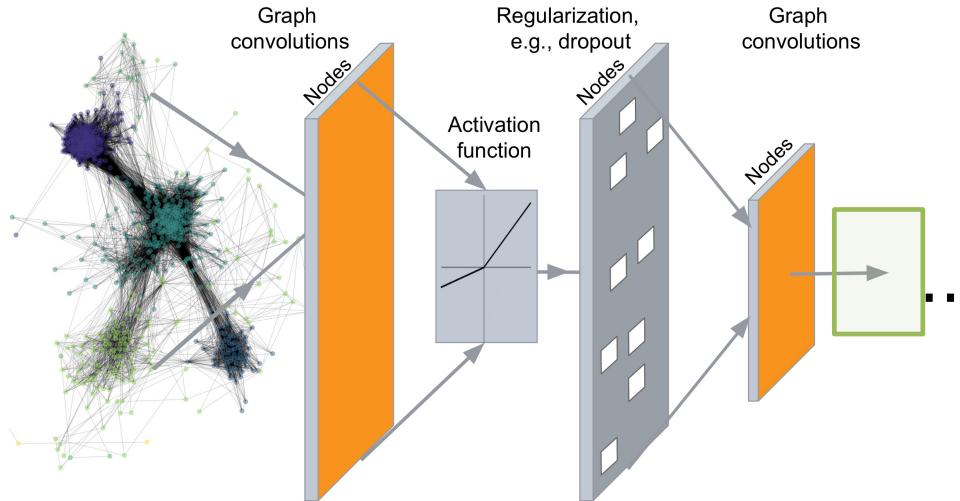
Average of neighbor's previous layer embeddings

$$\mathbf{z}_v = \mathbf{h}_v^K$$

Embedding after  $K$  layers of neighborhood aggregation

Non-linearity (e.g., ReLU)

# Basic approach



trainable weight matrices

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

(i.e., what we learn)

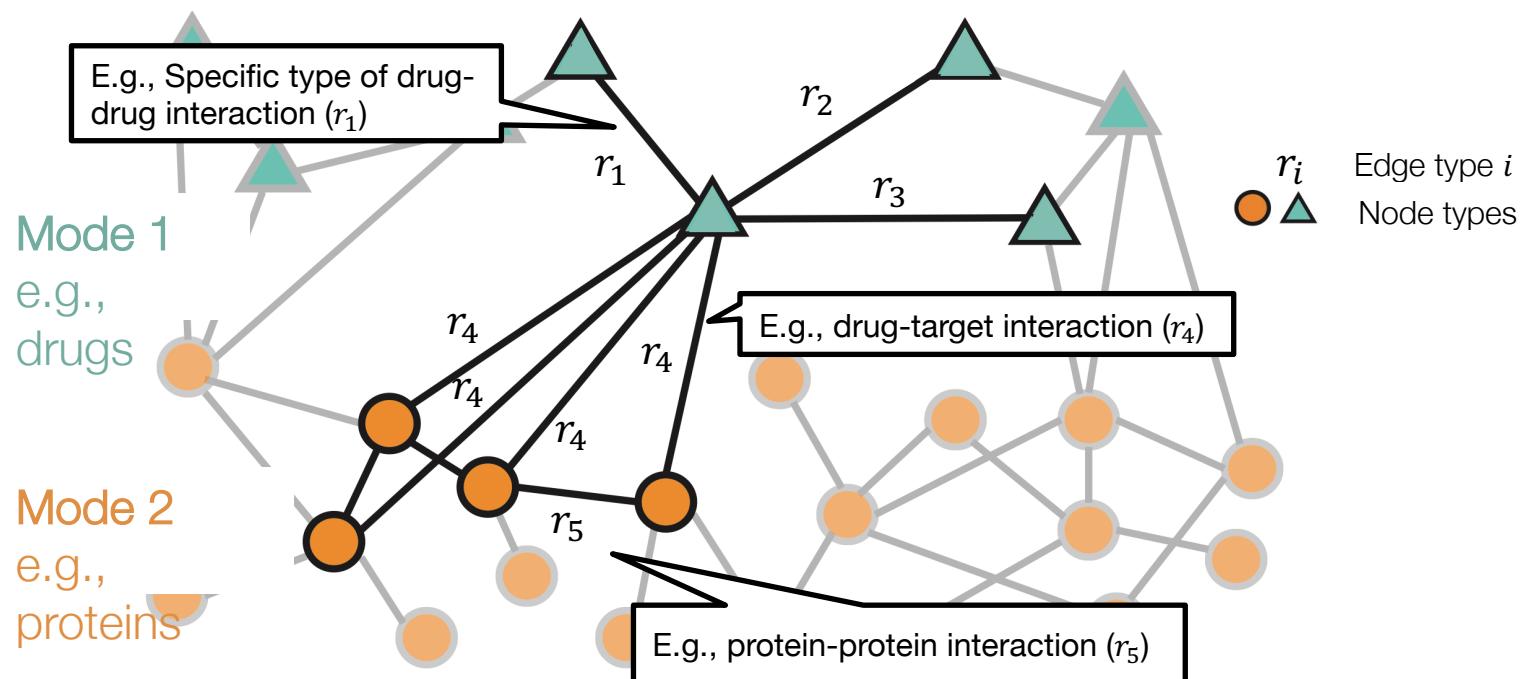
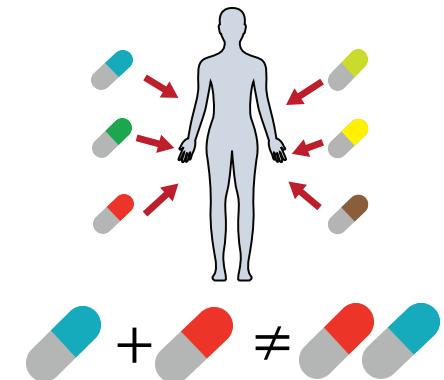
$$\mathbf{h}_v^k = \sigma \left( \mathbf{W}_k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right), \quad \forall k \in \{1, \dots, K\}$$

$$\boxed{\mathbf{z}_v} = \mathbf{h}_v^K$$

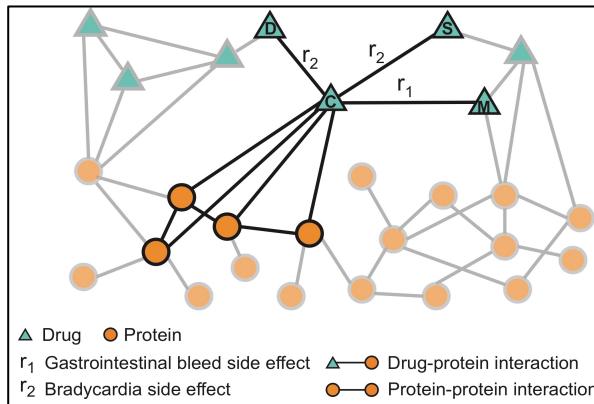
We can feed these into any loss function and run stochastic gradient descent to train the weight parameters

# Application: Prioritize drug combos

- Combinatorial explosion
  - >13 million possible combinations of 2 drugs
  - >20 billion possible combinations of 3 drugs
- Non-linear & non-additive interactions
  - Different effect than the additive effect of individual drugs
- Small subsets of patients
  - Side effects are interdependent
  - No info on drug combinations not yet used in patients

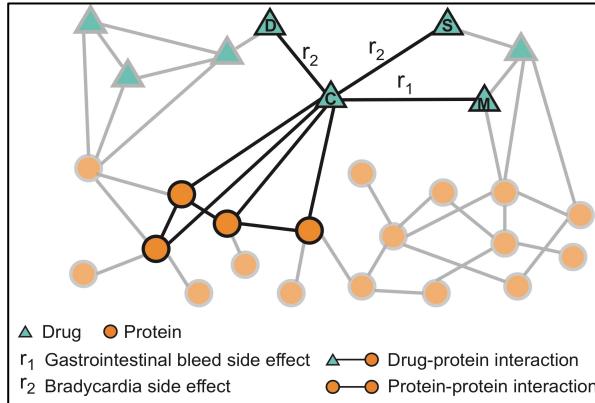


# Polypharmacy dataset

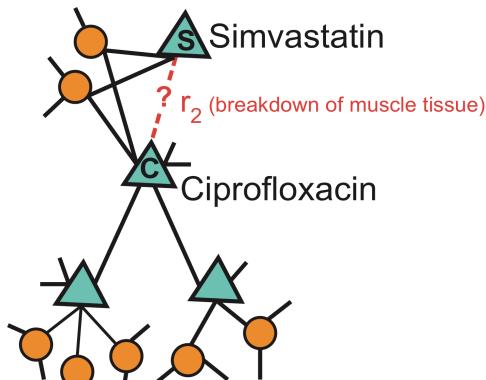


- Molecular, drug, and patient data for all drugs prescribed in US
  - **4,651,131 drug-drug edges:** Patient data from adverse event system, tested for confounders [FDA]
  - **18,596 drug-protein edges**
  - **719,402 protein-protein edges:** Physical, metabolic enzyme-coupled, and signaling interactions
  - **Drug and protein features:** drugs' chemical structure, proteins' membership in pathways
- Gives multimodal network with over 5 million edges separated into 1,000 different edge types

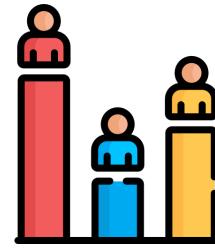
# Experimental setup



- Two main stages:
  1. Learn an **embedding** for every node in polypharmacy network
  2. Predict a score for **every drug-drug, drug-protein, protein-protein pair** in the test set based on the embeddings



**Example:** How likely will Simvastatin and Ciprofloxacin, when taken together, break down muscle tissue?



Time for a poll question about...

# GRAPH NEURAL NETWORKS

1. Which of the following are false about graph neural networks? *Select many*

# Poll Question

Which of the following are false about graph neural networks? *Select many*

1. Requires an encoder
2. Not invariant to node ordering
3. Can be of arbitrary depth
4. Not applicable to graphs of different sizes

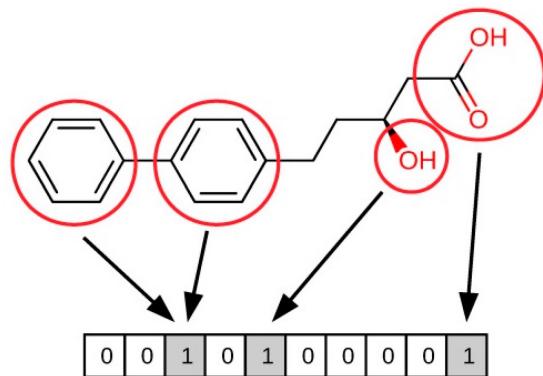
# Poll Question

Which of the following are false about graph neural networks? *Select many*

1. Requires an encoder
2. Not invariant to node ordering
3. Can be of arbitrary depth
4. Not applicable to graphs of different sizes

Molecular graphs and structures (xyz coordinates) are fundamental features in molecules

Often these are converted into **molecular descriptors** or some other representations



Why is that? Why can we not work with the data directly?

# Illustrative example (1/2)

- Let's say we have a butane molecule and would like to predict its potential energy from its position. We could train a linear model  $\hat{E}$  that predicts energy:

$$\hat{E} = XW + b$$

where  $X$  is 14 (atoms)  $\times$  3 (xyz coordinates) matrix containing positions and  $W$ ,  $b$  are trainable parameters

- Now what if we translate all the coordinates by  $-10$ :

$$(X - 10)W + b = X + b - 10|W|$$

- We know the **energy should not change if we translate all the coordinates equally** – the molecule **is not changing conformations**
- However, our linear regression will change by  $-10|W|$
- We have accidentally made our model sensitive to the origin of our coordinate system, which is not physical. **This is translational variance** – the model changes when we translate the coordinates

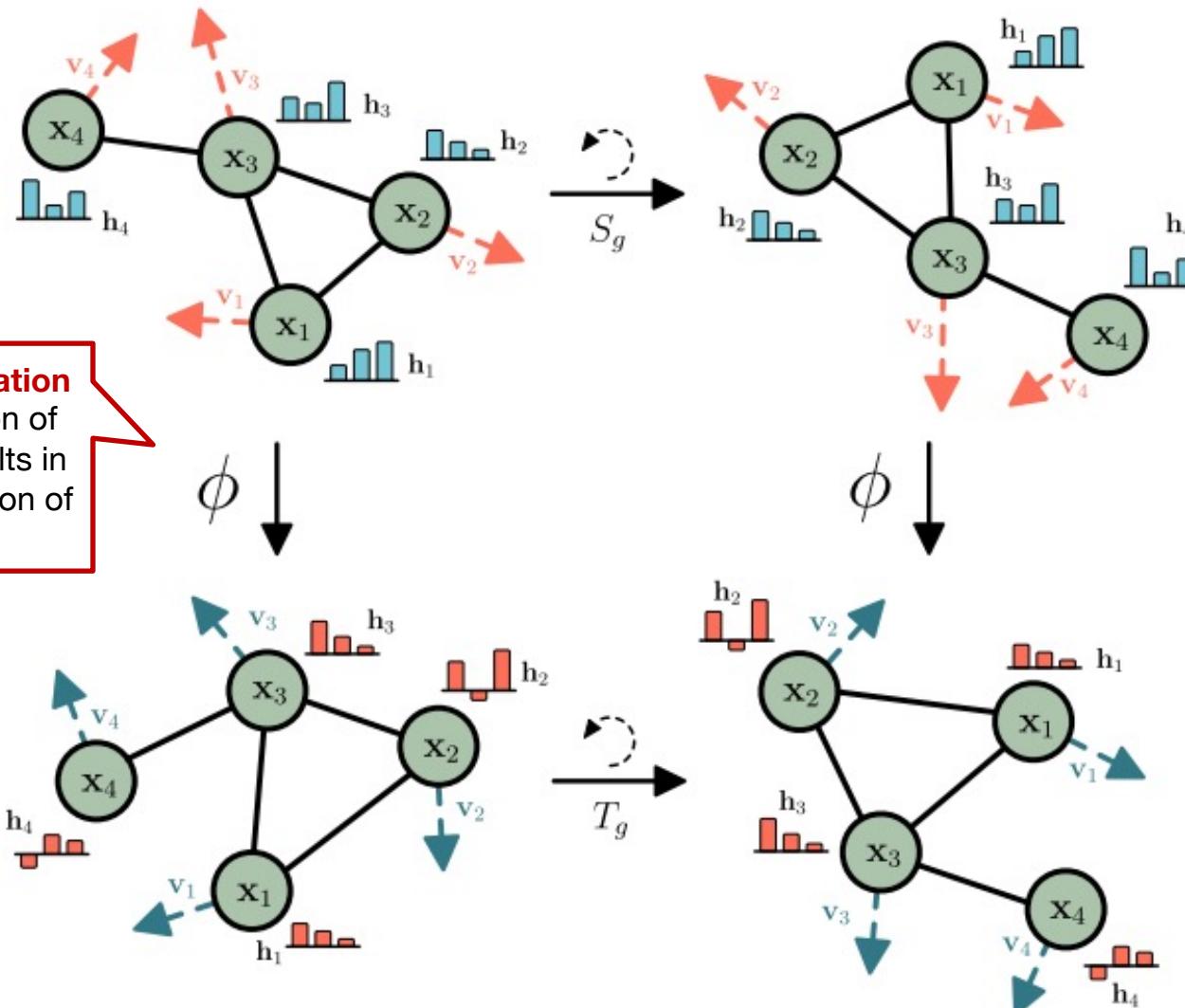
# Illustrative example (2/2)

- Consider another example from our butane molecule. What if we swapped the order of the atoms in our model? The **Take-away message:** We want models predicting potential energy of molecules from their position to be insensitive to the translation and rotation of molecular fragments — namely, they should be **permutation invariant and translationally invariant.** This means if we swap the knowledge of atom types, the ordering of atoms does not matter. We have accidentally made the model sensitive to the ordering of the atoms

# Methods so far presented in Tutorial

- **Method presented so far:** Neural networks that process input data in a way that is **invariant to node and edge permutations**
- **Next:** Neural networks that process input data in a way that is sensitive to some type of transformations
  - **Equivariant neural networks!**
  - Permutation equivariant: If you rearrange the order of atoms, the output changes in the same way
  - Translationally equivariant: If you translate the input to a neural network the output will be translated in the same way

# Example of rotational equivariance



Example of rotation equivariance on a graph with a graph neural network  $\phi$

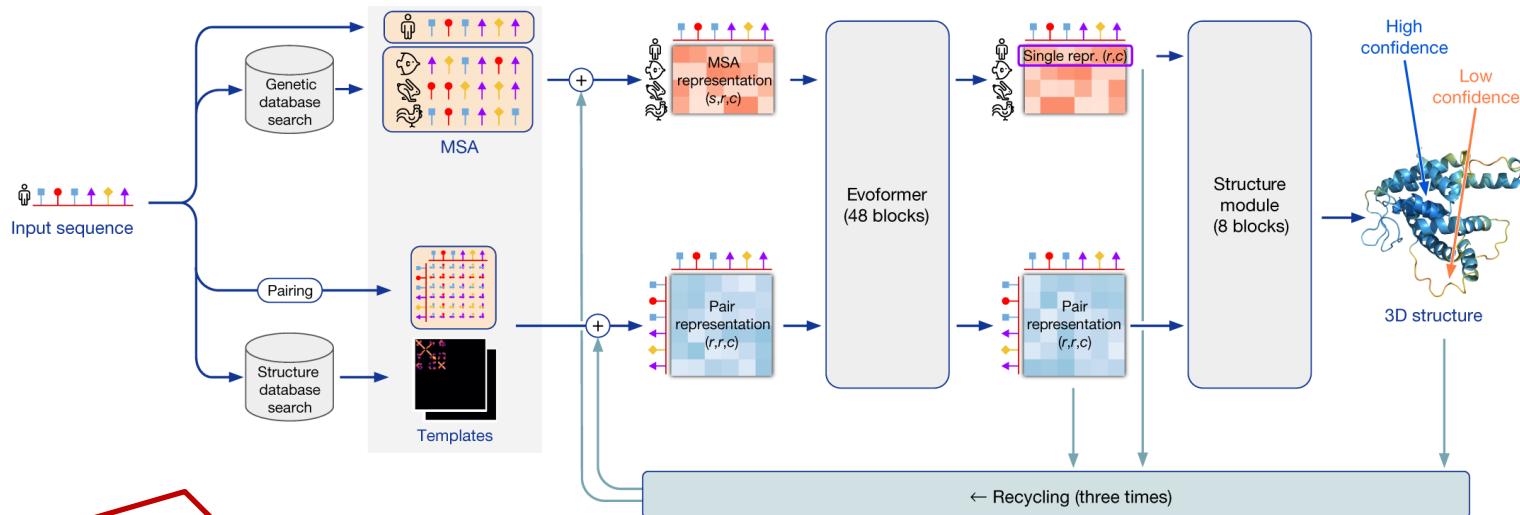
# Equivariant neural networks

- Equivariant neural networks are part of a broader topic of geometric deep learning, **which is learning with data that has some underlying geometric relationships**
  - Geometric deep learning is a broad-topic and includes the “5Gs”: grids, groups, graphs, geodesics, and gauges
  - Typical nomenclature in current literature:
    - Point clouds (gauges) → equivariant neural networks
    - Graphs → graph neural networks
    - Grids → convolutional neural networks

Highly-active research area, especially for predicting energies, forces, protein docking, and structures of molecules

# Why should I care about equivariant neural networks?

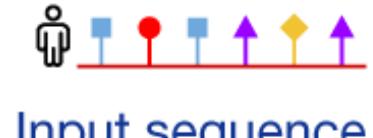
**Equivariant neural networks** underly several breakthroughs, including AlphaFold 2 for protein structure prediction



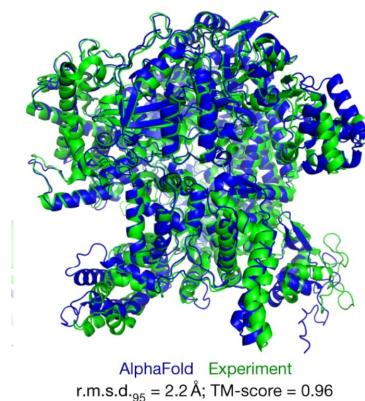
Equivariant neural networks are receiving increasing interest from the natural and medical sciences as they represent a new tool for analyzing molecules and their properties

# Motivation: AlphaFold network (1/2)

- What drives accurate protein structure prediction?
  - Novel neural architecture based on the **evolutionary**, **physical** and **geometric** constraints of protein structures
- Input:
  - Primary AA sequence of a given protein
  - Aligned sequences of homologues
- Output:
  - Predicted 3D coordinates of all heavy atoms in the protein

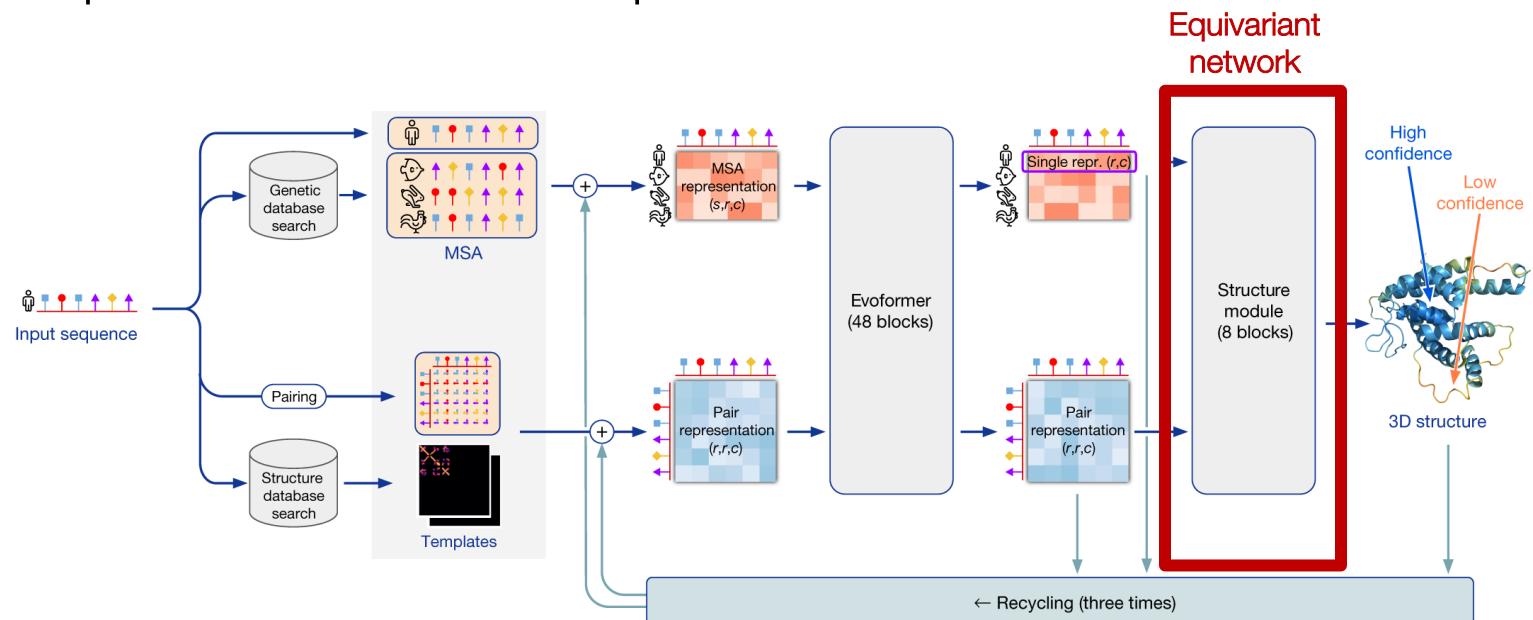


Input sequence



# Motivation: AlphaFold network (2/2)

- **Structure module** in the AlphaFold network:
  - It introduces an explicit 3D structure in the form of a rotation and translation for each residue of the protein
  - Representations are initialized in a trivial state with all rotations set to the identity and all positions set to the origin
  - Representations rapidly develop and refine a highly accurate protein structure with precise atomic details



# Problem definition

- **Setup:**
  - Graph  $G = (V, E)$  with nodes  $v_i \in V$  and edges  $e_{ij} \in E$
  - In addition to node embeddings  $h_i \in R^d$  we also consider a  $n$ -dimensional coordinate  $x_i \in R^n$  associated with each node
- **Goal:**
  - Model that will **preserve equivariance to rotations and translations** on these set of coordinates  $x_i$
  - Model that will preserve equivariance to permutations on the set of nodes  $V$  in the same fashion as GNNs

# Approach: Equivariant GNN (EGNN)

- EGNN model has multiple neural layers
- Layer  $l$  is an **equivariant graph convolutional layer**:
  - **Input:**
    - Set of node embeddings  $h^l = \{h_0^l, \dots, h_{M-1}^l\}$
    - Coordinate embeddings  $x^l = \{x_0^l, \dots, x_{M-1}^l\}$
    - Edge information  $E = (e_{ij})$
  - **Output:**
    - Updated node embeddings  $h^{l+1} = \{h_0^{l+1}, \dots, h_{M-1}^{l+1}\}$
    - Updated coordinate embeddings  $x^{l+1} = \{x_0^{l+1}, \dots, x_{M-1}^{l+1}\}$
- Concisely:  $h^{l+1}, x^{l+1} = \text{EGCL}(h^l, x^l, E)$

# Approach: EGG layer

Main difference with the original GNN

**Update coordinate embeddings:**  
Update the position of each particle  $x_i$  as a vector field in a radial direction

**Aggregate coordinate info:** We choose to aggregate coordinates according to the relative squared distance between two coordinates

$$\mathbf{m}_{ij} = \phi_x(\mathbf{h}_i^l, \mathbf{h}_j^l, \|\mathbf{x}_i^l - \mathbf{x}_j^l\|^2, a_{ij})$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + C \sum_{j \neq i} (\mathbf{x}_i^l - \mathbf{x}_j^l) \phi_x(\mathbf{m}_{ij})$$

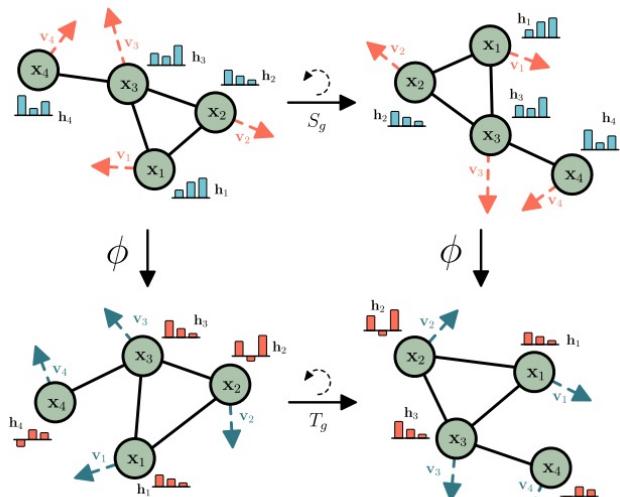
$$\mathbf{m}_i = \sum_{j \neq i} \mathbf{m}_{ij}$$

**Aggregate node info:** We choose to aggregate messages from all other nodes  $j \neq i$ . Alternatively, we could limit the message exchange to a given neighborhood  $j \in N(i)$

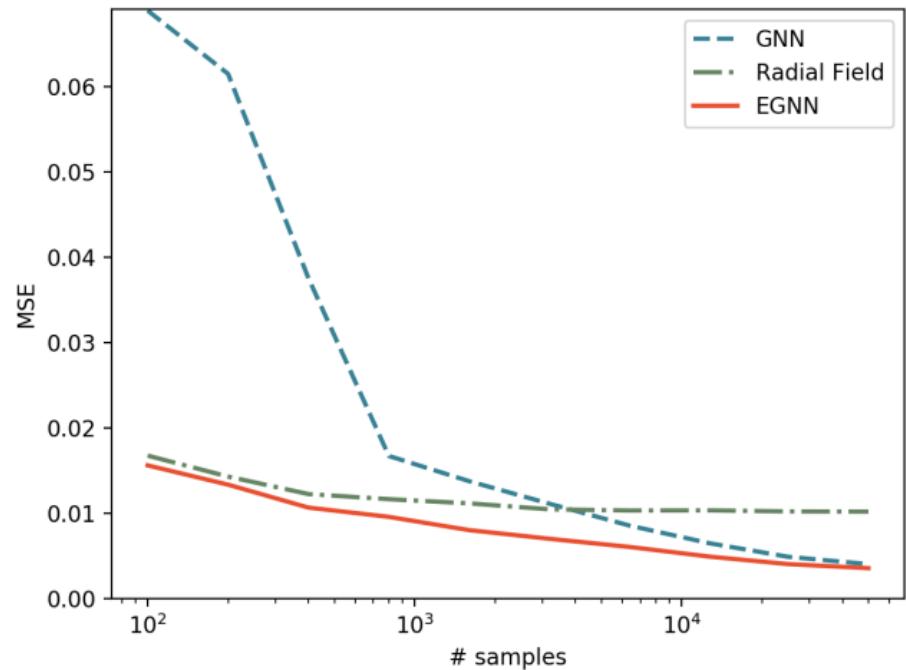
$$\mathbf{h}_i^{l+1} = \phi_h(\mathbf{h}_i^l, \mathbf{m}_i)$$

**Update node embeddings:** It takes as input the aggregated message, the node embedding from layer  $l$  and outputs the updated node embedding at layer  $l + 1$

# Recap: Equivariant neural network

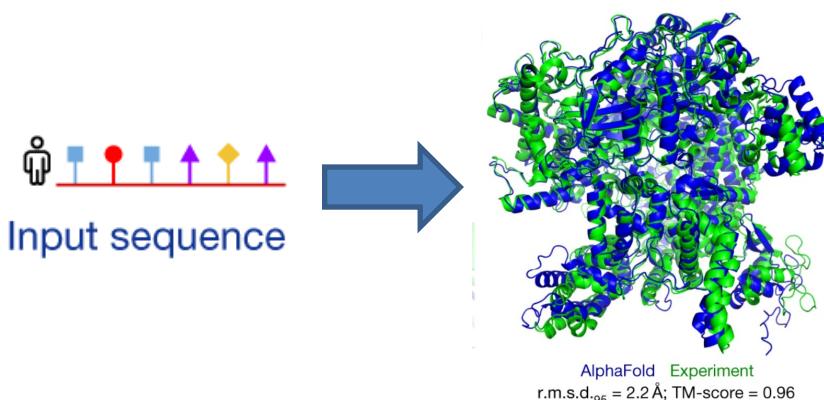


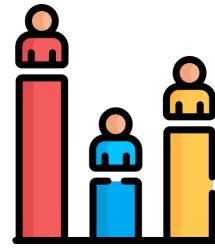
Example of rotation equivariance on a graph with a graph neural network  $\phi$



Mean Squared Error in the N-body experiment for the Radial Field, GNN and **EGNN** methods when sweeping over different amounts of training data

E(n) Equivariant Graph Neural Networks, ICML, 2021





Time for a poll question about...

# EQUIVARIANT NEURAL NETWORKS

1. Which of the following are applicable to equivariant neural networks? *Select many*

# Poll Question

Which of the following are applicable to equivariant neural networks? *Select many*

Equivariant neural networks are

1. ...permutation invariant
2. ...sensitive to rotation
3. ...translationally invariant
4. ...sensitive to changes from the origin

# Poll Question

Which of the following are applicable to equivariant neural networks? *Select many*

Equivariant neural networks are

1. ...permutation invariant
2. ...sensitive to rotation
3. ...translationally invariant
4. ...sensitive to changes from the origin

# This Tutorial

- ✓ 1. Methods: Network diffusion, shallow network embeddings, graph neural networks, equivariant neural networks
- 2. Applications: Fundamental biological discoveries and precision medicine
- 3. Hands-on exercises: Demos, implementation details, tools, and tips

# Resources

- Books & survey papers
  - William Hamilton, *Graph Representation Learning* ([morganclaypool.com/doi/abs/10.2200/S01045ED1V01Y202009AIM046](http://morganclaypool.com/doi/abs/10.2200/S01045ED1V01Y202009AIM046))
  - Li et al., Graph Representation Learning for Biomedicine ([arxiv.org/abs/2104.04883](https://arxiv.org/abs/2104.04883))
- Keynotes & seminars
  - Michael Bronstein, “Geometric Deep Learning: The Erlangen Programme of ML” (ICLR 2021 keynote) ([youtube.com/watch?v=w6Pw4MOzMu0](https://youtube.com/watch?v=w6Pw4MOzMu0))
  - Broad Institute Models, Inference & Algorithms: Actionable machine learning for drug discovery; Primer on graph representation learning ([youtube.com/watch?v=9YpTYdru0Rg](https://youtube.com/watch?v=9YpTYdru0Rg))
  - Stanford University (CS224W Lecture): Graph neural networks in computational biology ([youtube.com/watch?v=\\_hy9AgZXhbQ](https://youtube.com/watch?v=_hy9AgZXhbQ))
  - AI Cures Drug Discovery Conference ([youtube.com/watch?v=wNXSkISMTw8](https://youtube.com/watch?v=wNXSkISMTw8))
- Conferences & summer schools
  - London Geometry and Machine Learning Summer School ([logml.ai](https://logml.ai))
  - Learning on Graphs Conference ([logconference.github.io](https://logconference.github.io))

# Resources

- Software & packages
  - PyTorch Geometric
  - NetworkX
  - Stanford Network Analysis Platform (SNAP)
- Tutorials & code bases
  - Pytorch Geometric Colab Notebooks ([pytorch-geometric.readthedocs.io/en/latest/notes/colabs.html](https://pytorch-geometric.readthedocs.io/en/latest/notes/colabs.html))
  - Zitnik Lab Graph ML Tutorials ([github.com/mims-harvard/graphml-tutorials](https://github.com/mims-harvard/graphml-tutorials))
  - Stanford University's CS224 ([web.stanford.edu/class/cs224w](https://web.stanford.edu/class/cs224w))
- Datasets
  - Precision Medicine Oriented Knowledge Graph (PrimeKG) ([zitniklab.hms.harvard.edu/projects/PrimeKG](https://zitniklab.hms.harvard.edu/projects/PrimeKG))
  - Therapeutic Data Commons (TDC) ([tdcommons.ai](https://tdcommons.ai))
  - BioSNAP ([snap.stanford.edu/biodata/](https://snap.stanford.edu/biodata/))
  - Open Graph Benchmark (OGB) ([ogb.stanford.edu](https://ogb.stanford.edu))