Graph Neural Networks in Computational Biology

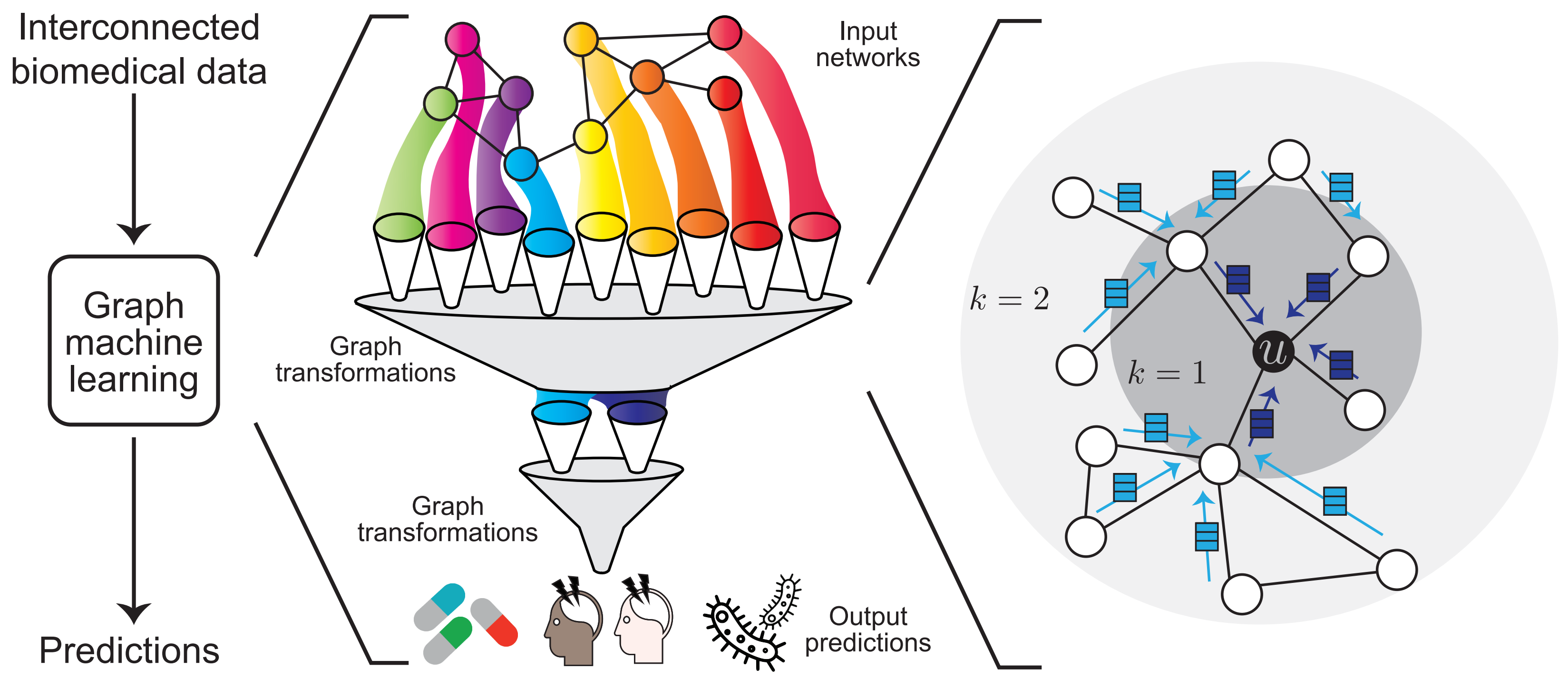
Marinka Zitnik

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

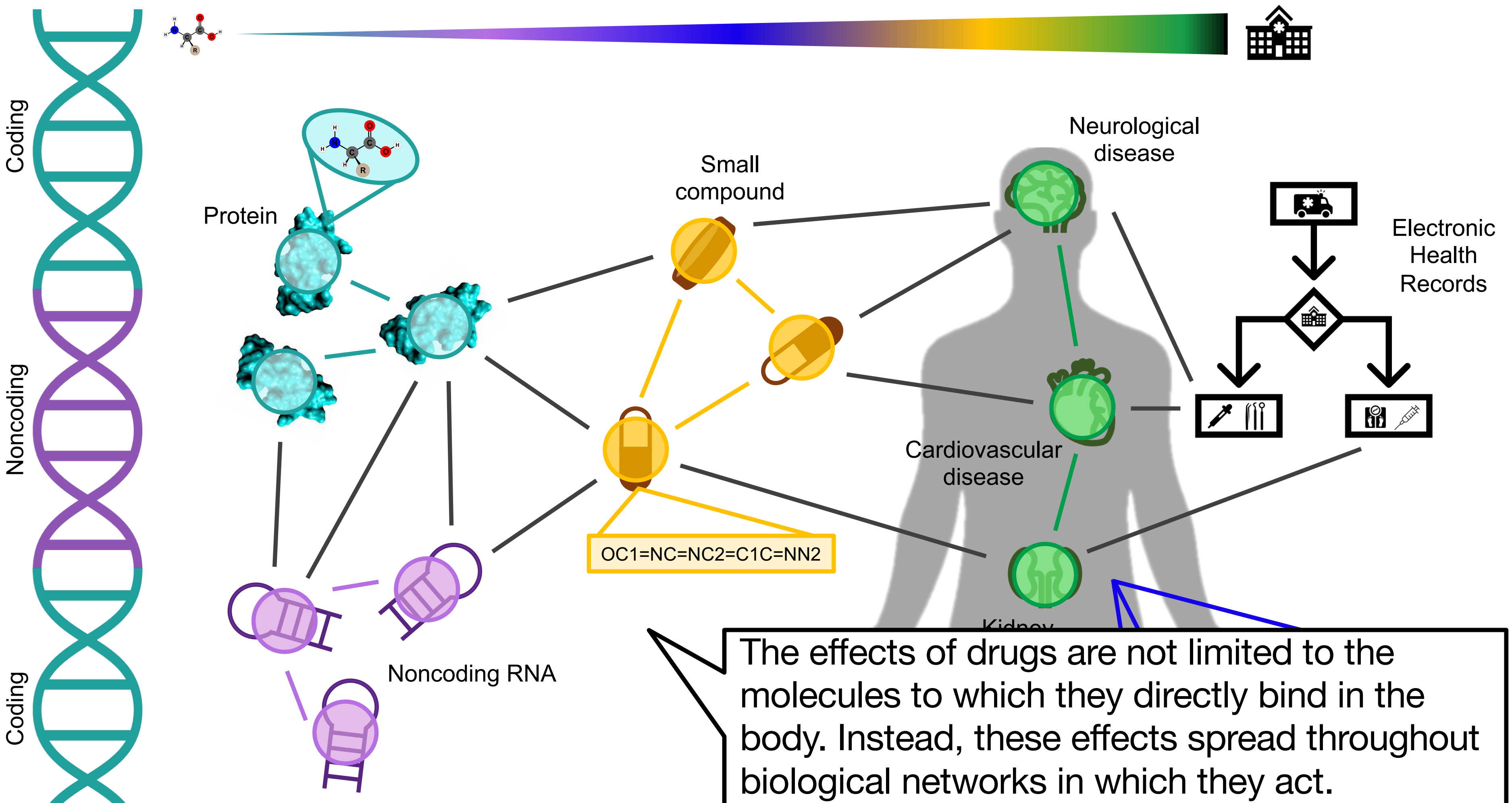
marinka@hms.harvard.edu
zitniklab.hms.harvard.edu

Graph ML for Computational Biology

- There has been a surge of interest in leveraging GNNs for learning meaningful representations of biology
- GNNs have been used to learn representations that enabled critical predictions in downstream applications



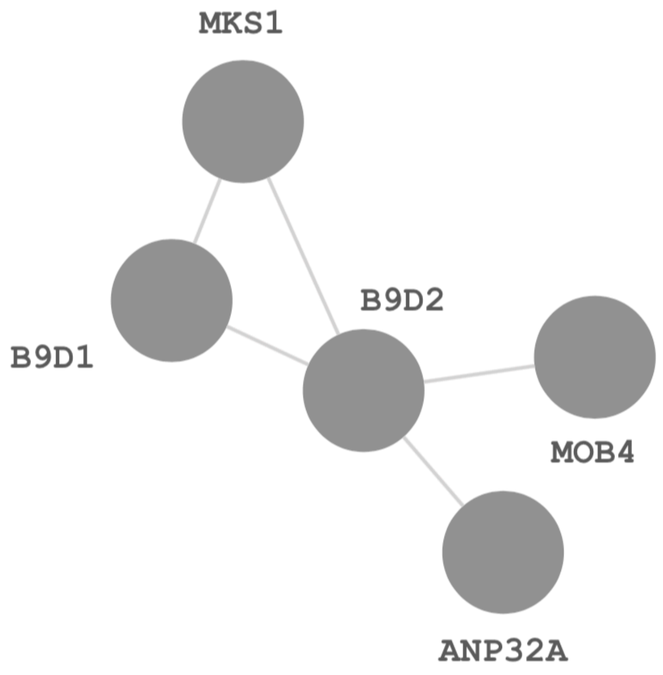
Biology is Interconnected!

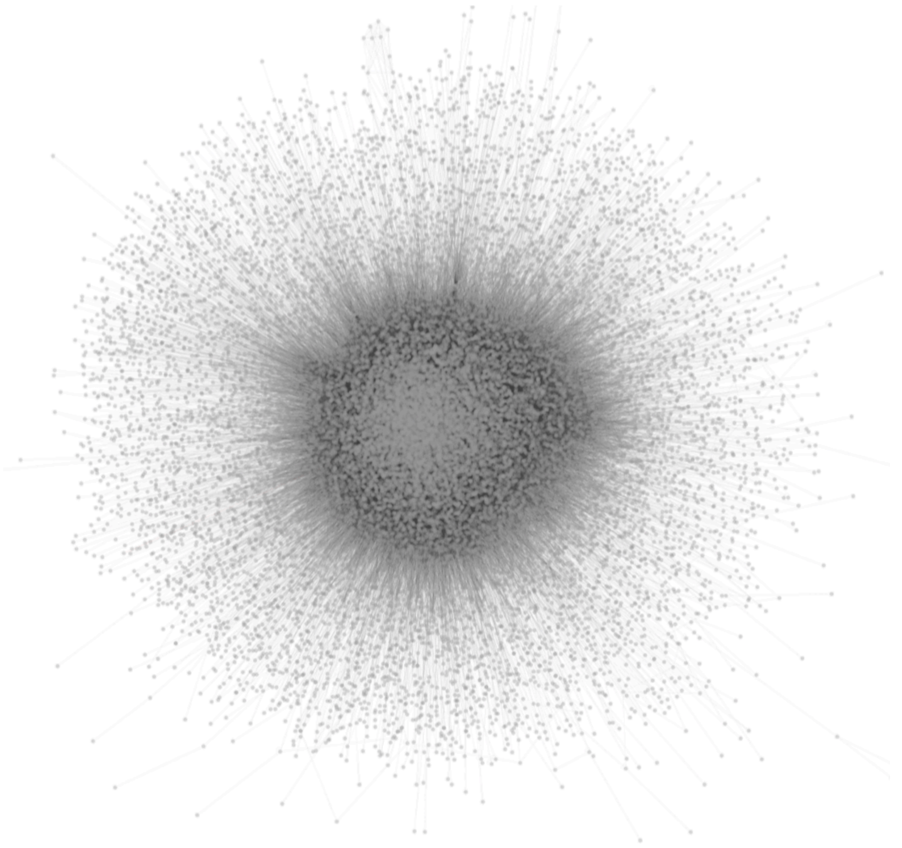


The effects of drugs are not limited to the molecules to which they directly bind in the body. Instead, these effects spread throughout biological networks in which they act. Therefore, the effect of a drug on a disease is inherently a network phenomenon

Why Networks in Biology?

Network of protein-protein interactions in human cells.

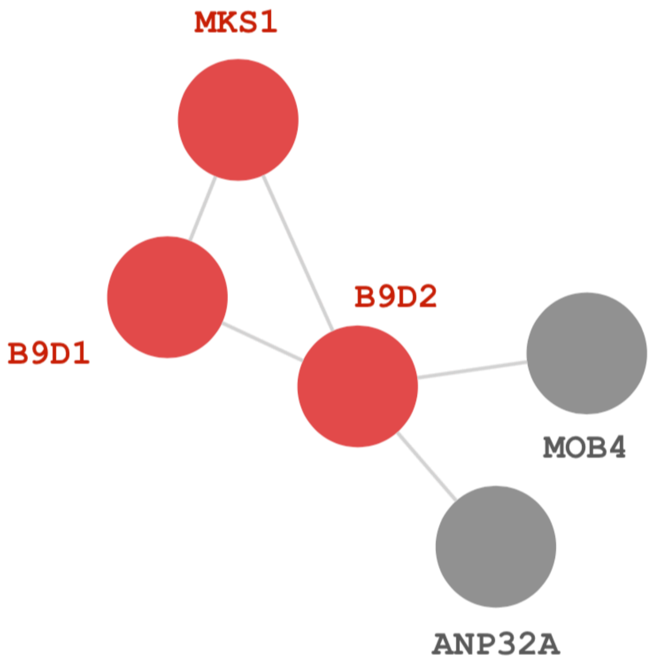




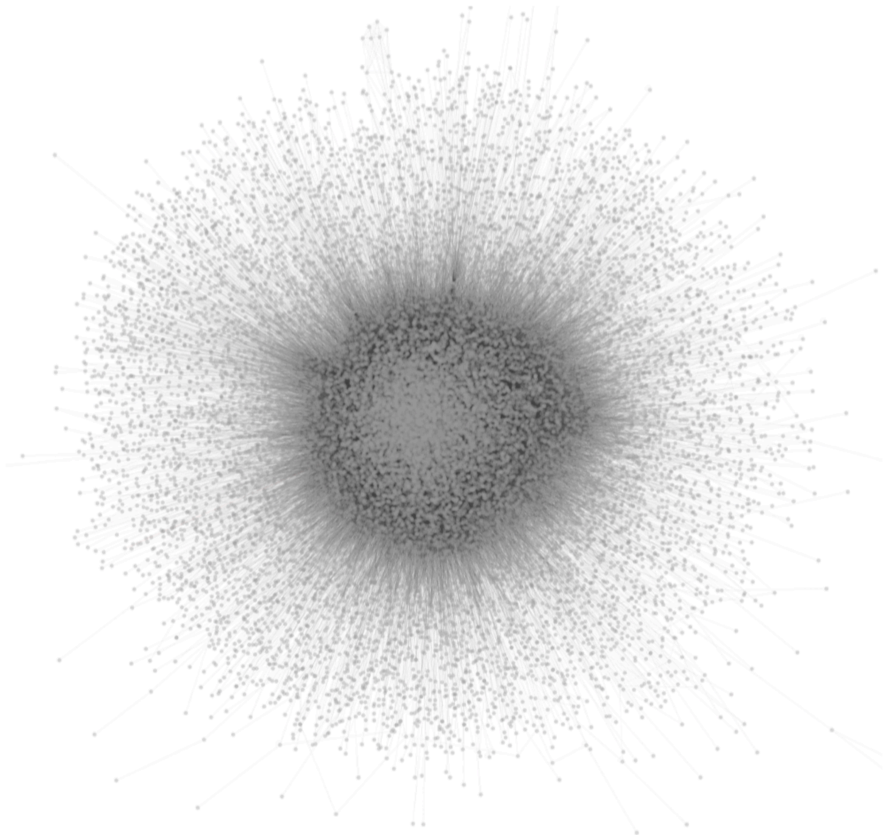
21,557 proteins
342,353 interactions

Why Networks in Biology?

Network of protein-protein interactions in human cells.

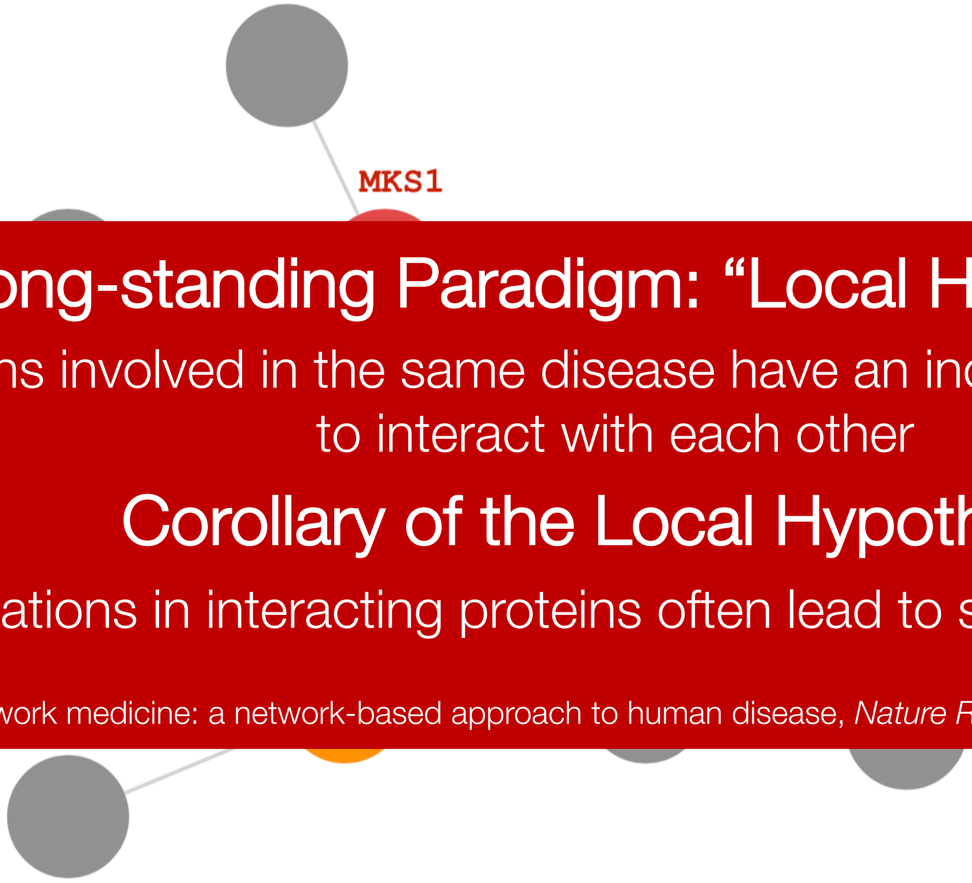


● Disease protein



21,557 proteins
342,353 interactions

Why Networks in Biology?



Long-standing Paradigm: “Local Hypothesis”

Proteins involved in the same disease have an increased tendency to interact with each other

Corollary of the Local Hypothesis

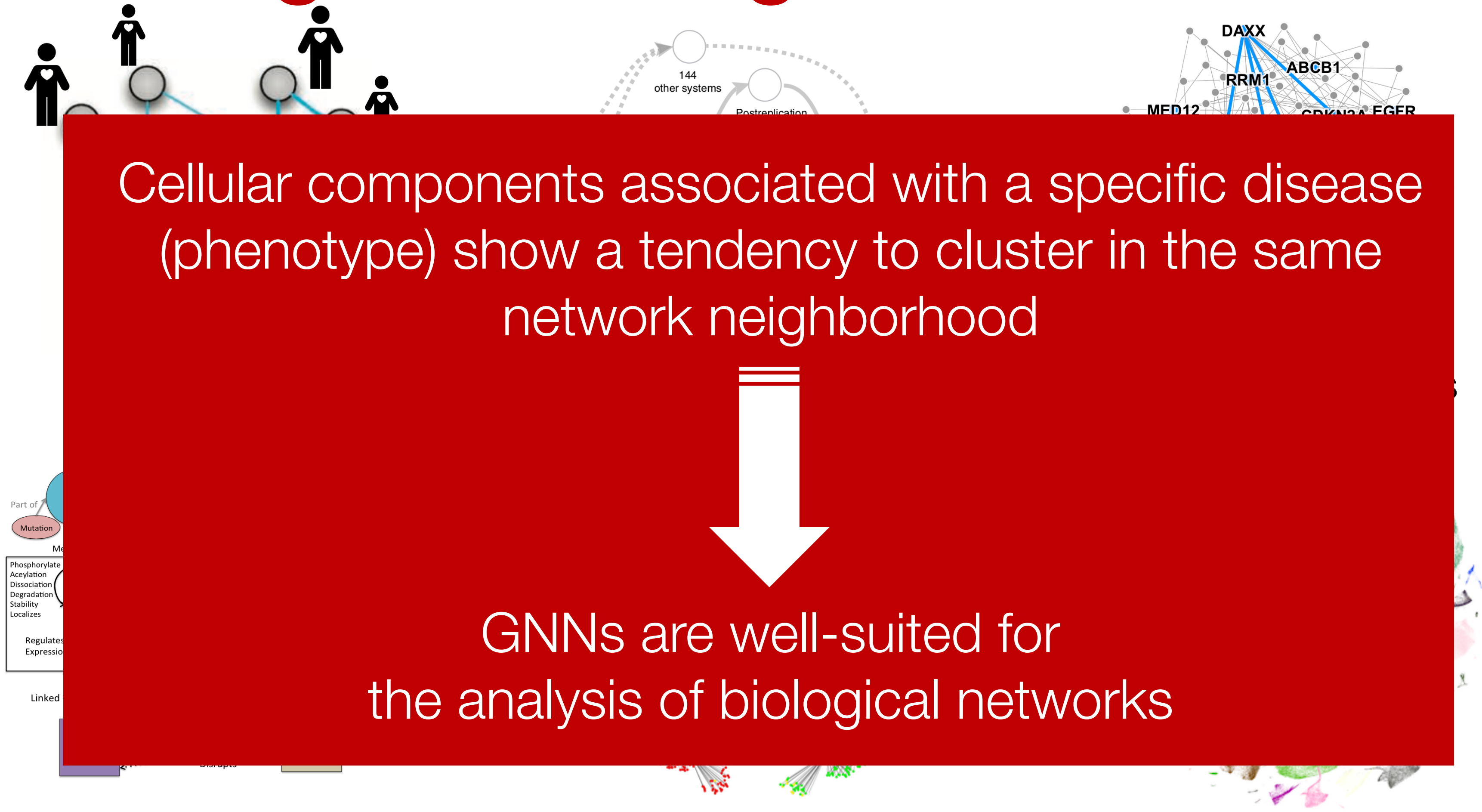
Mutations in interacting proteins often lead to similar diseases

Network medicine: a network-based approach to human disease, *Nature Reviews Genetics, 2011*

🟥 Known disease proteins

🟧 Predicted disease proteins

Similar findings apply to a broad range of biological networks



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

⇓

GNNs are well-suited for the analysis of biological networks

Biomedical knowledge graphs

Gene interaction networks

Cell-cell similarity networks

Machine learning for biomedical networks: Advancements, challenges, and opportunities, 2021 (to appear)

Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities, *Information Fusion* 2019

Why are Biological Networks Challenging?

1. Networks involve heterogeneous interactions that span from molecules to whole populations
   - The challenge is how to computationally operationalize these data and make them amenable to ML
2. Networks contain data from diverse sources, including experimental readouts, curated annotations, metadata
   - No single data type can capture all the factors necessary to understand a phenomenon such as a disease
3. Networks are noisy due to inherent natural variations and limitations of measurement platforms
   - Missing data, repeated measurements, and contradictory observations can plague the analysis

Plan for Today

- Safe drugs and drug combinations

  Methods: Multi-relational link prediction on KGs

- Patient outcomes & disease classification

  Methods: Subgraph embeddings

- Effective disease treatments

  Methods: Few-shot learning for graphs

Poly-Therapy

Patients take multiple drugs to treat complex or co-existing diseases

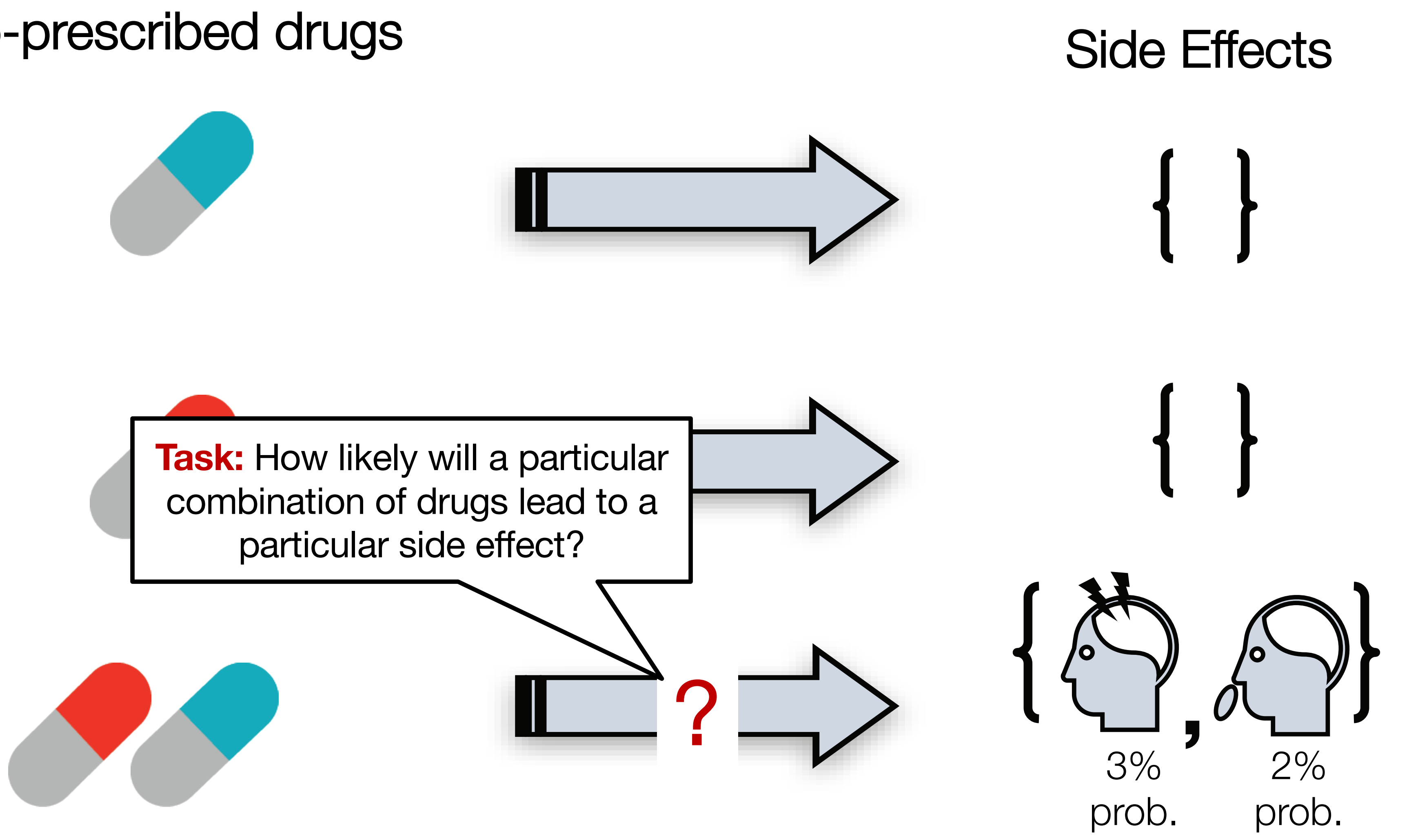
46% of people over 65 years take more than 5 drugs

Many take more than 20 drugs to treat heart diseases, depression or cancer

15% of the U.S. population affected by unwanted side effects

Annual costs in treating side effects exceed $177 billion in the U.S. alone

Unexpected Drug Interactions

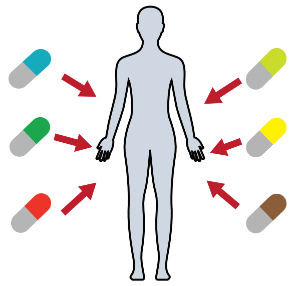


Task: How likely will a particular combination of drugs lead to a particular side effect?

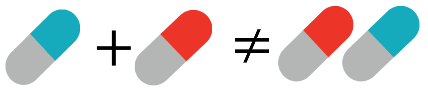
Why is modeling drug combinations chalenging?

Combinatorial explosion

- >13 million possible combinations of 2 drugs
- >20 billion possible combinations of 3 drugs



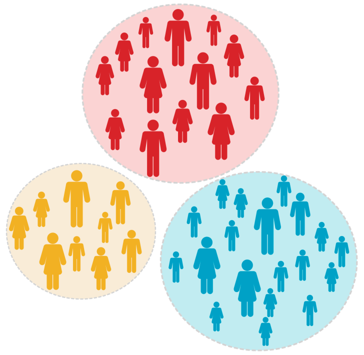
Non-linear & non-additive interactions



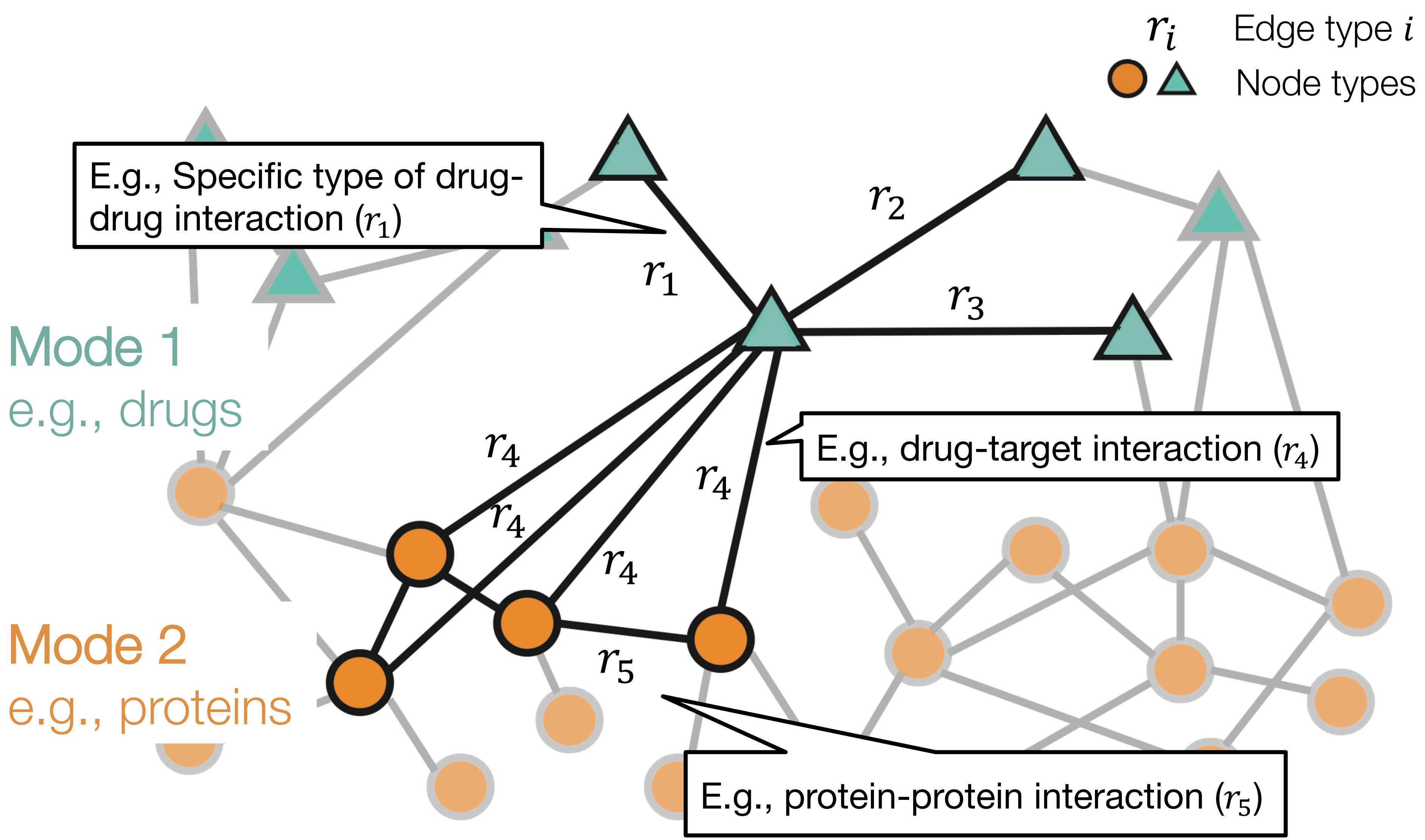
- Different effect than the additive effect of individual drugs

Small subsets of patients

- Side effects are interdependent
- No info on drug combinations not yet used in patients

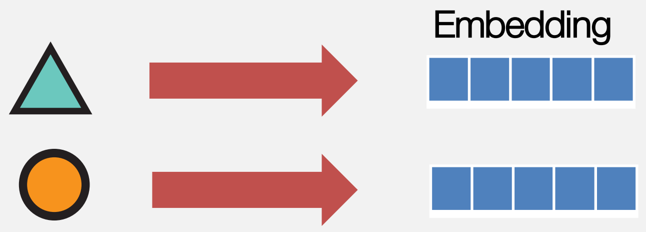


Polypharmacy Knowledge Graph

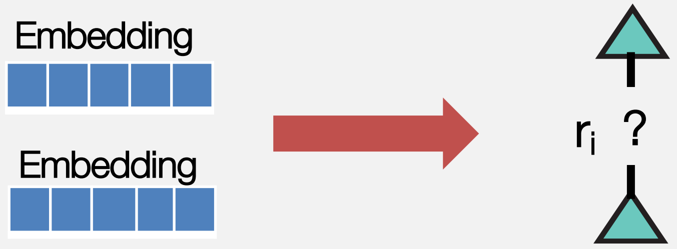

r_i Edge type i
● ▲ Node types
E.g., Specific type of drug-drug interaction (r_1)
r_1
r_2
r_3
Mode 1
e.g., drugs
r_4
r_4
r_4
r_4
E.g., drug-target interaction (r_4)
Mode 2
e.g., proteins
r_5
E.g., protein-protein interaction (r_5)

Approach: Decagon

1. **Encoder:** Take a multimodal network and learn an *embedding* for every node



2. **Decoder:** Use the learned embeddings to predict labeled edges between nodes



Training the model: Feed embeddings into any loss function and run stochastic gradient descent to train weight parameters:

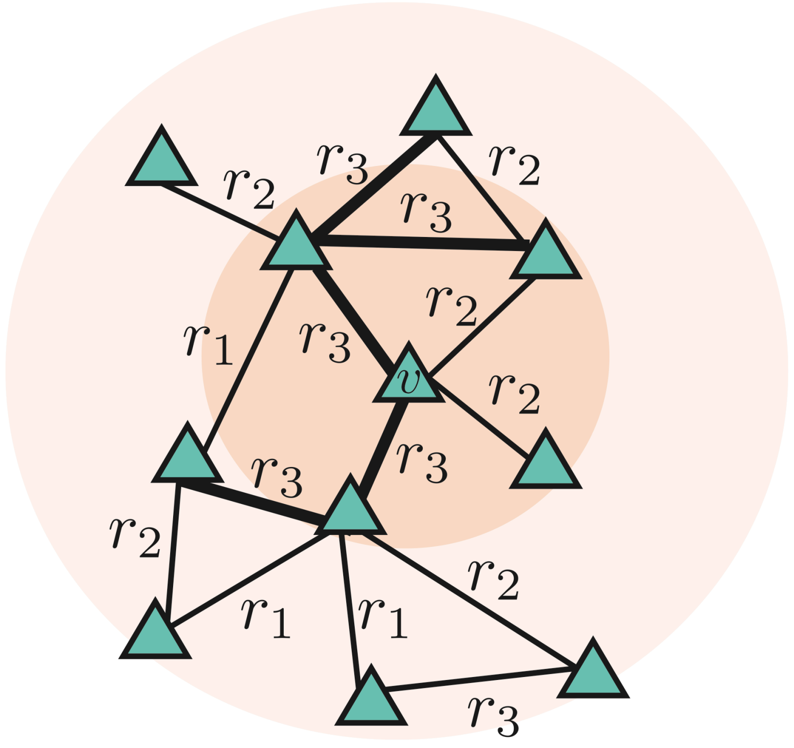
- Use a loss based on e.g., random walks, node proximity in the graph
- Directly train the model for a supervised task (e.g., node classification)

Key Idea: Aggregate Neighbors

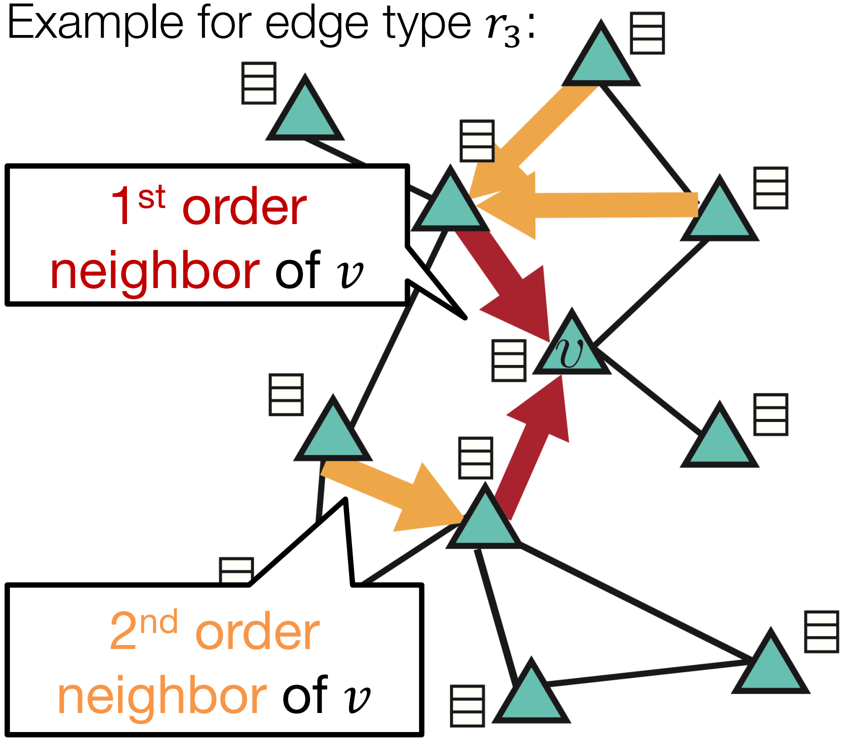
Generate embeddings based on local network neighborhoods separated by edge type

1) Determine a node's computation graph for each edge type

2) Learn how to transform and propagate information across computation graph



Example for edge type r_3:



Multirelational Graph Encoder

Key element: Each node's computation graph defines a neural network with a different architecture

- Initial 0-th layer embeddings are equal to node features:

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

Aggregate neighbor's previous-layer embeddings, separated by edge type

Ability to integrate side information about nodes

- Per-layer update of n e embeddings:

Previous-layer embedding of v

$$\mathbf{h}_v^{(k)} = \phi\left(\sum_r \sum_{u \in N_v^r} c_r^{uv} \mathbf{W}_r^{(k-1)} \mathbf{h}_u^{(k-1)} + c_r^v \mathbf{h}_v^{(k-1)}\right) \qquad k = 1, \ldots, K$$

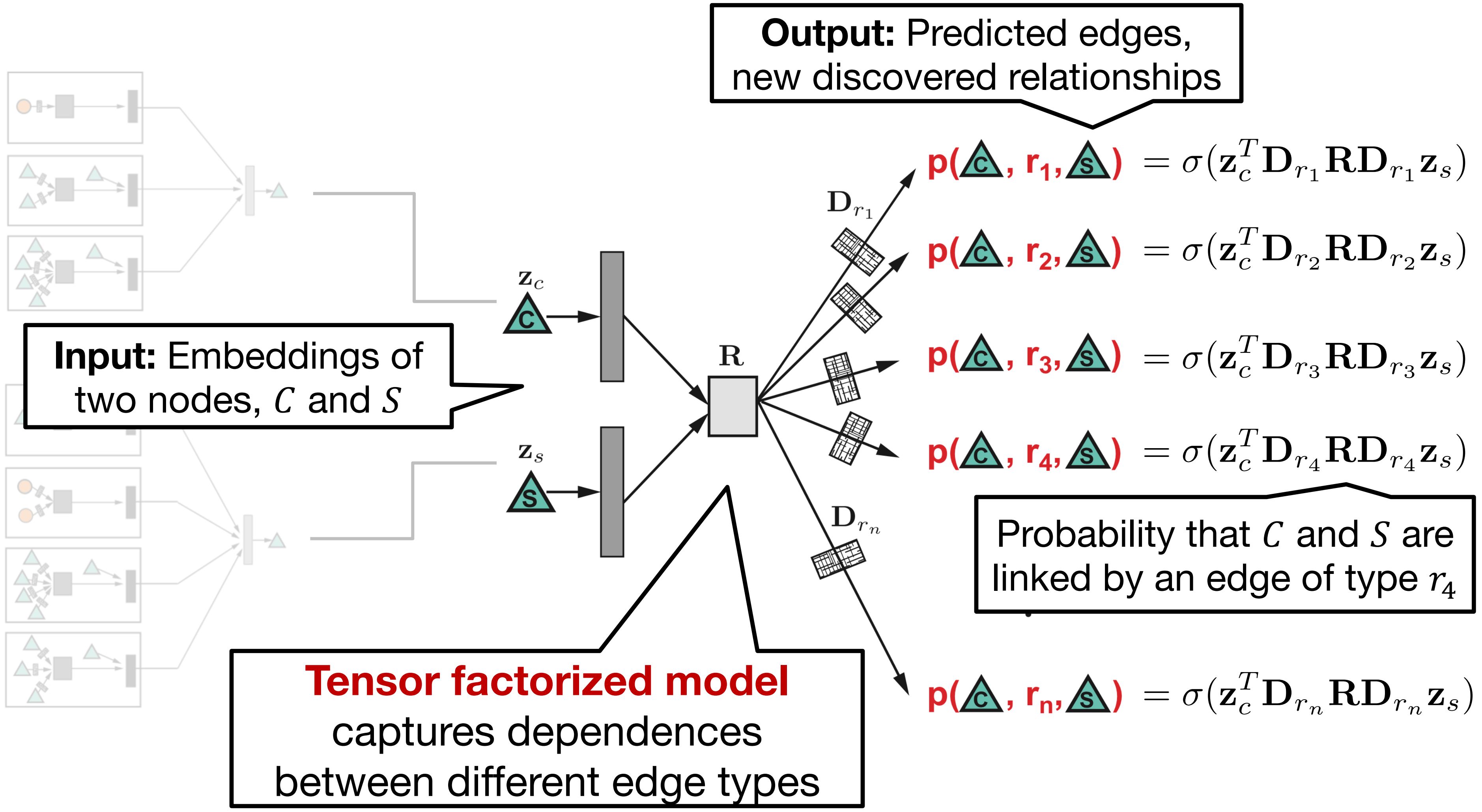
- Embeddings after K layers of neighborhoo ggregation:

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

Normalization constant, fixed *e.g.,* $1/|N_v^r|$, or learned

$\mathbf{W}_r^{(k)}$ Par

Heterogeneous Edge Decoder



Output: Predicted edges, new discovered relationships

Input: Embeddings of two nodes, C and S

$p(C, r_1, S) = \sigma(\mathbf{z}_c^T \mathbf{D}_{r_1} \mathbf{R} \mathbf{D}_{r_1} \mathbf{z}_s)$

$p(C, r_2, S) = \sigma(\mathbf{z}_c^T \mathbf{D}_{r_2} \mathbf{R} \mathbf{D}_{r_2} \mathbf{z}_s)$

$p(C, r_3, S) = \sigma(\mathbf{z}_c^T \mathbf{D}_{r_3} \mathbf{R} \mathbf{D}_{r_3} \mathbf{z}_s)$

$p(C, r_4, S) = \sigma(\mathbf{z}_c^T \mathbf{D}_{r_4} \mathbf{R} \mathbf{D}_{r_4} \mathbf{z}_s)$

Probability that C and S are linked by an edge of type r_4

$p(C, r_n, S) = \sigma(\mathbf{z}_c^T \mathbf{D}_{r_n} \mathbf{R} \mathbf{D}_{r_n} \mathbf{z}_s)$

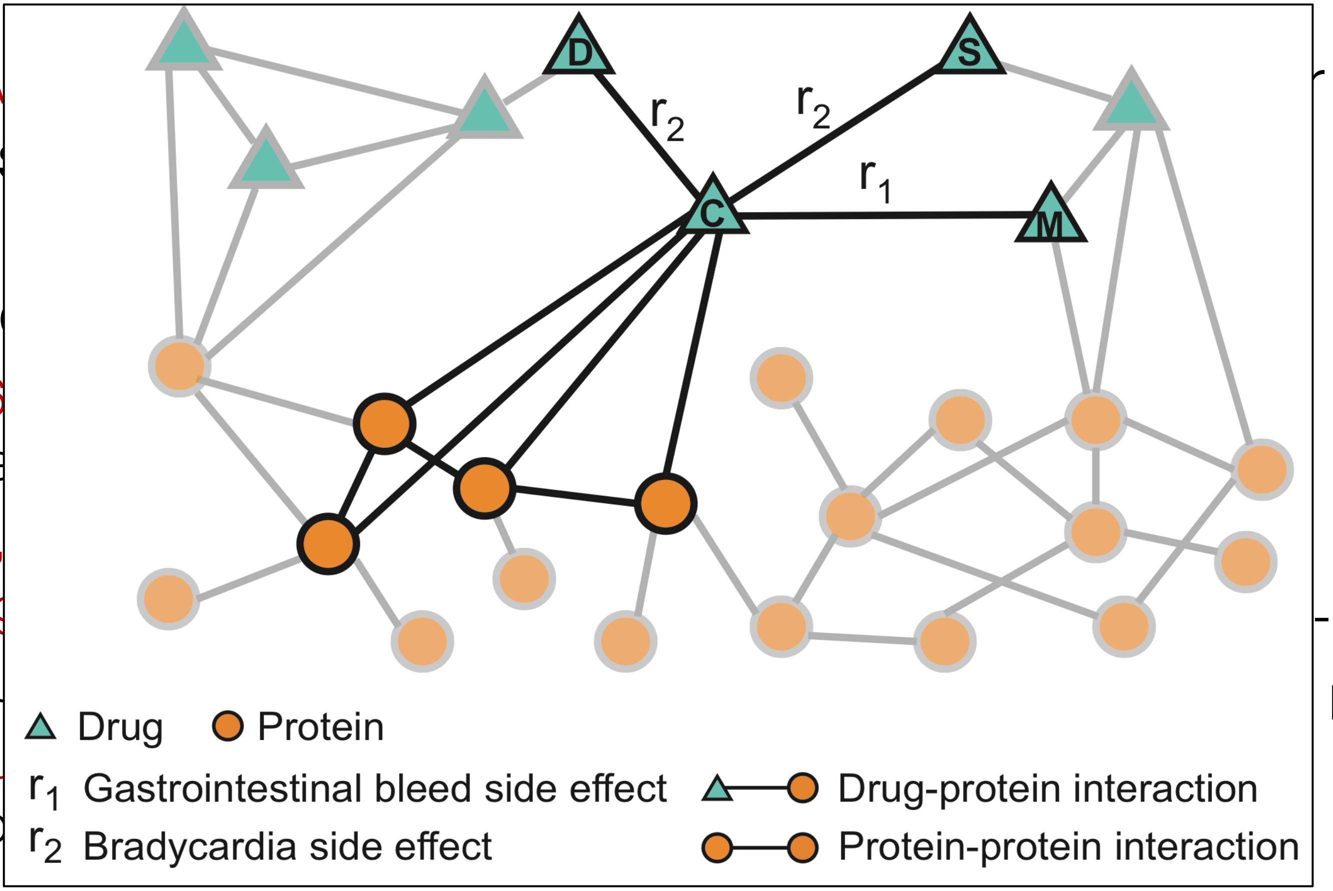
Tensor factorized model captures dependences between different edge types

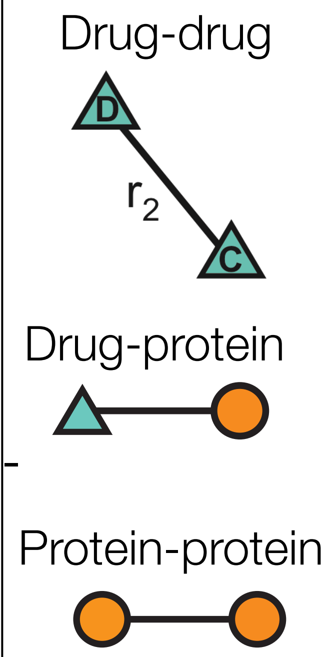
$\mathbf{R}, \mathbf{D}_{r_i}$ Parameter weight matrices

We need Polypharmacy Dataset

Objectiv r
all drugs

We buil
- 4,6
  eve
- 18,
- 71
  cou -
- Dru
  pro

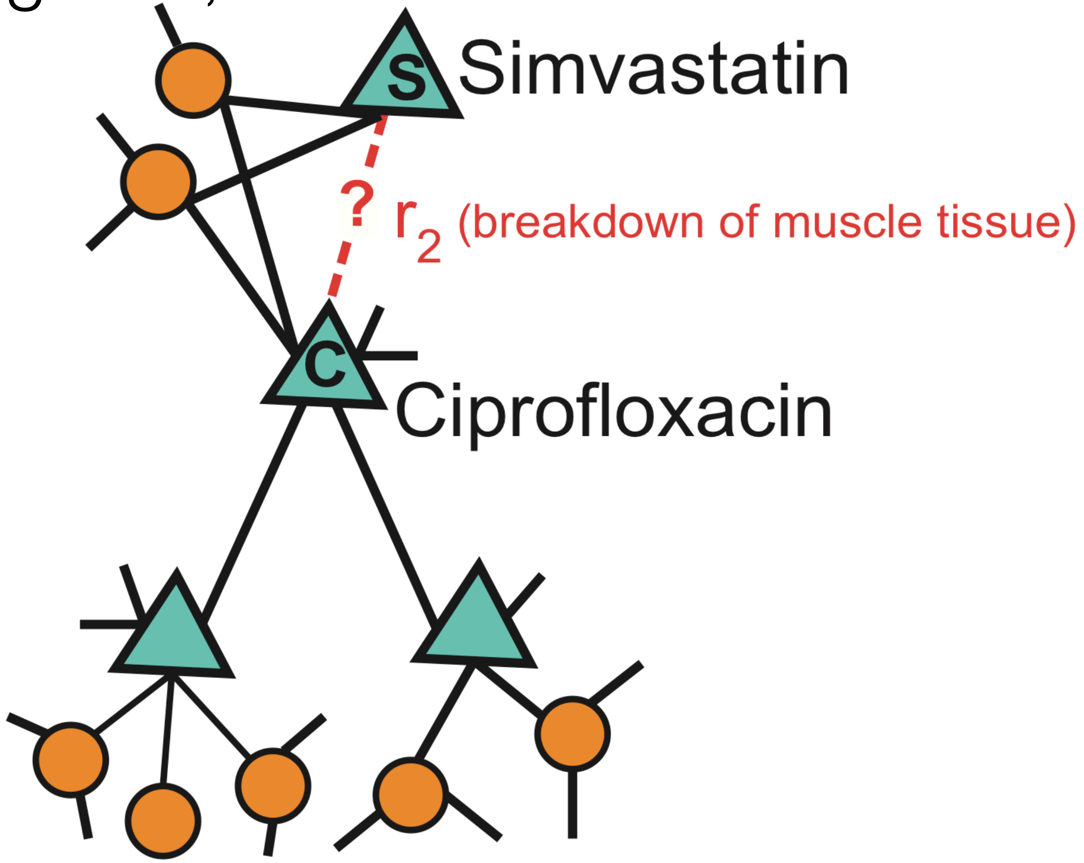

D
S
C
M
r_2
r_2
r_1
Drug
Protein
r_1 Gastrointestinal bleed side effect
r_2 Bradycardia side effect
Drug-protein interaction
Protein-protein interaction



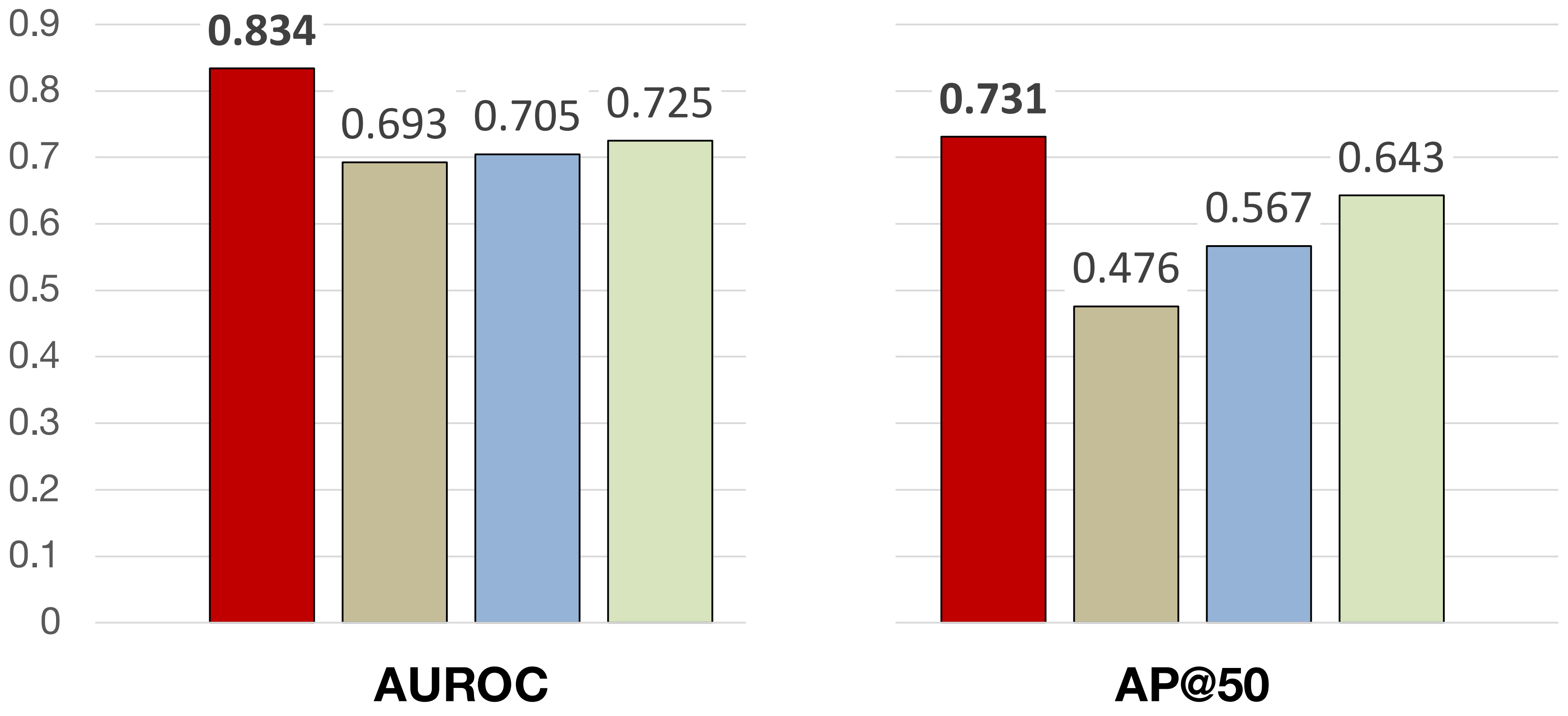
Gives multimodal network with over 5 million edges separated into 1,000 different edge types

We apply Decagon to the polypharmacy network

E.g.: How likely will Simvastatin and Ciprofloxacin, when taken together, break down muscle tissue?



Results: Side Effect Prediction



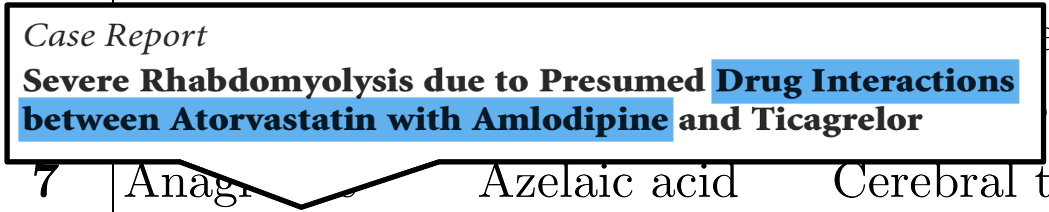
🟥 Our method (Decagon)
🟫 RESCAL Tensor Factorization [Nickel et al., ICML'11]
🟦 Multi-relational Factorization [Perros, Papalexakis et al., KDD'17]
🟩 Shallow Network Embedding [Zong et al., Bioinformatics'17]

New Predictions

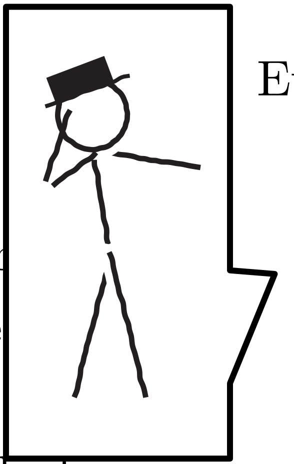
Approach:

1) Train deep model on data generated **prior to 2012**
2) How many **predictions** have been **confirmed after 2012**?

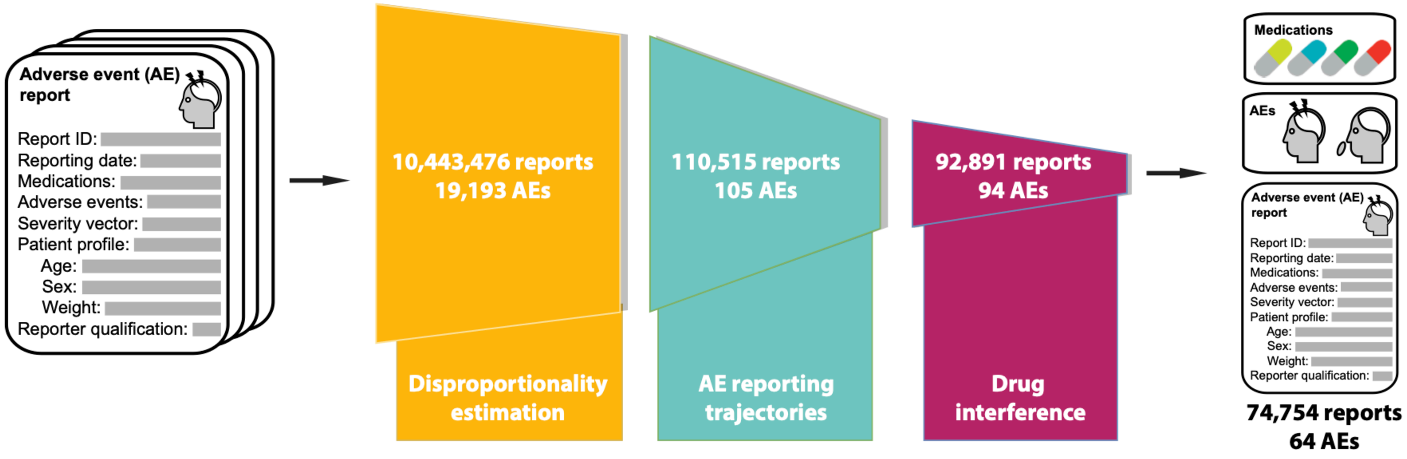
Rank	Drug	Drug	Side effect	Evidence found
1	Pyrimethamine	Aliskiren	Sarcoma	
2	Tigecycline	Bimatoprost	Autonomic n	
3	Telangiectases	Omeprazole	Dacarbazine	
4	Tolcapone	Pyrimethamine	Blood brain	
			eadache	
			ular acidosis	
7	Anag	Azelaic acid	Cerebral thrombosis	
8	Atorvastatin	Amlodipine	Muscle inflammation	
9	Aliskiren	Tioconazole	Breast inflammation	
10	Estradiol	Nadolol	Endometriosis	

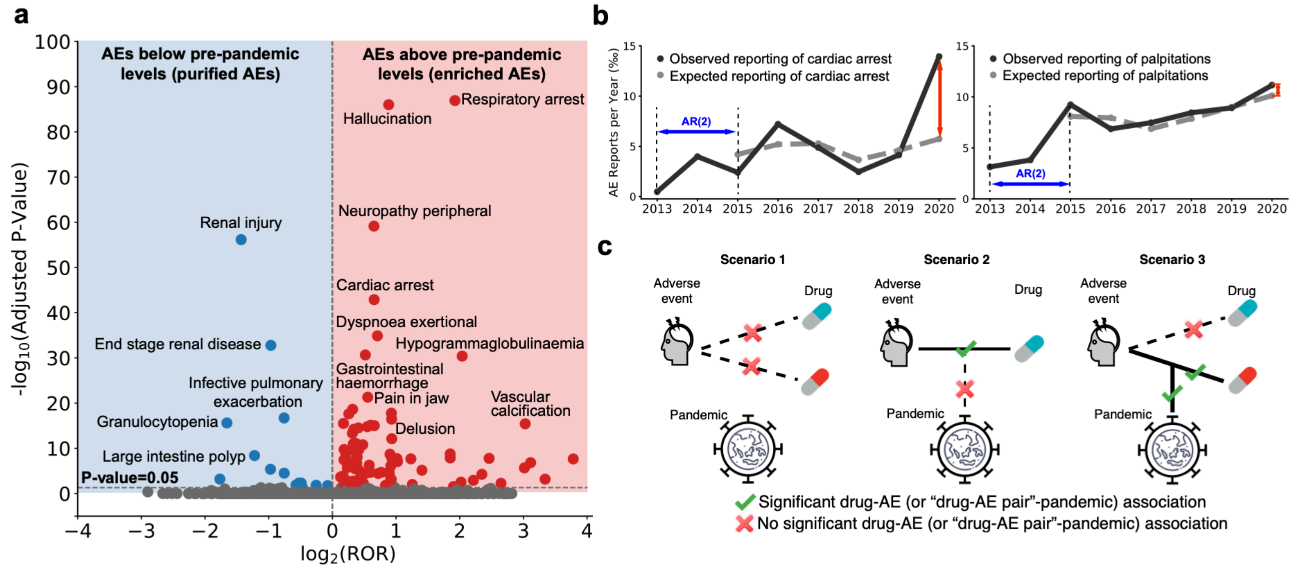

Case Report

Severe Rhabdomyolysis due to Presumed Drug Interactions between Atorvastatin with Amlodipine and Ticagrelor



Follow-Up: Adverse Events for Patient Groups





Population-scale patient safety data reveal inequalities in adverse events before and during COVID-19 pandemic, *medRxiv: 2021.01.17.21249988*

Plan for Today

- ✓ Safe drugs and drug combinations

  Methods: Multi-relational link prediction on KGs

- Patient outcomes & disease classification

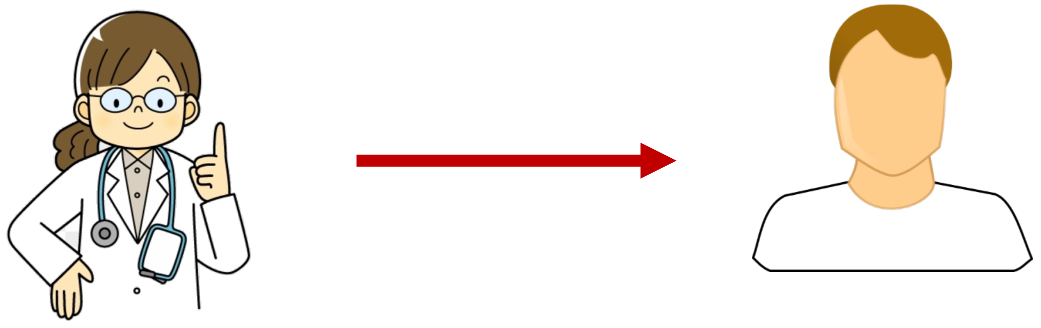
  Methods: Subgraph embeddings

- Effective disease treatments

  Methods: Few-shot learning for graphs

Disease Diagnosis

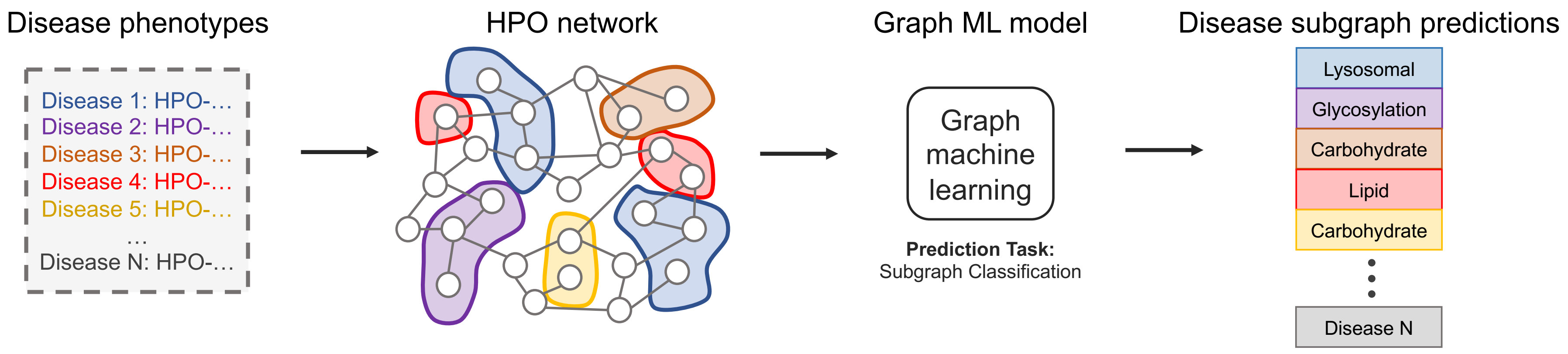
- Phenotypes are observable characteristics resulting from interactions between genotypes, as well as environment
  - Physicians utilize standardized vocabulary of phenotypes to describe human diseases.
  - By modeling diseases as collections of associated phenotypes, we can diagnose patients based on their presenting symptoms



Medical History:
Has asthma?
Other chronic issues?
……
Symptoms:
Severe Cough
Wheezing
…..

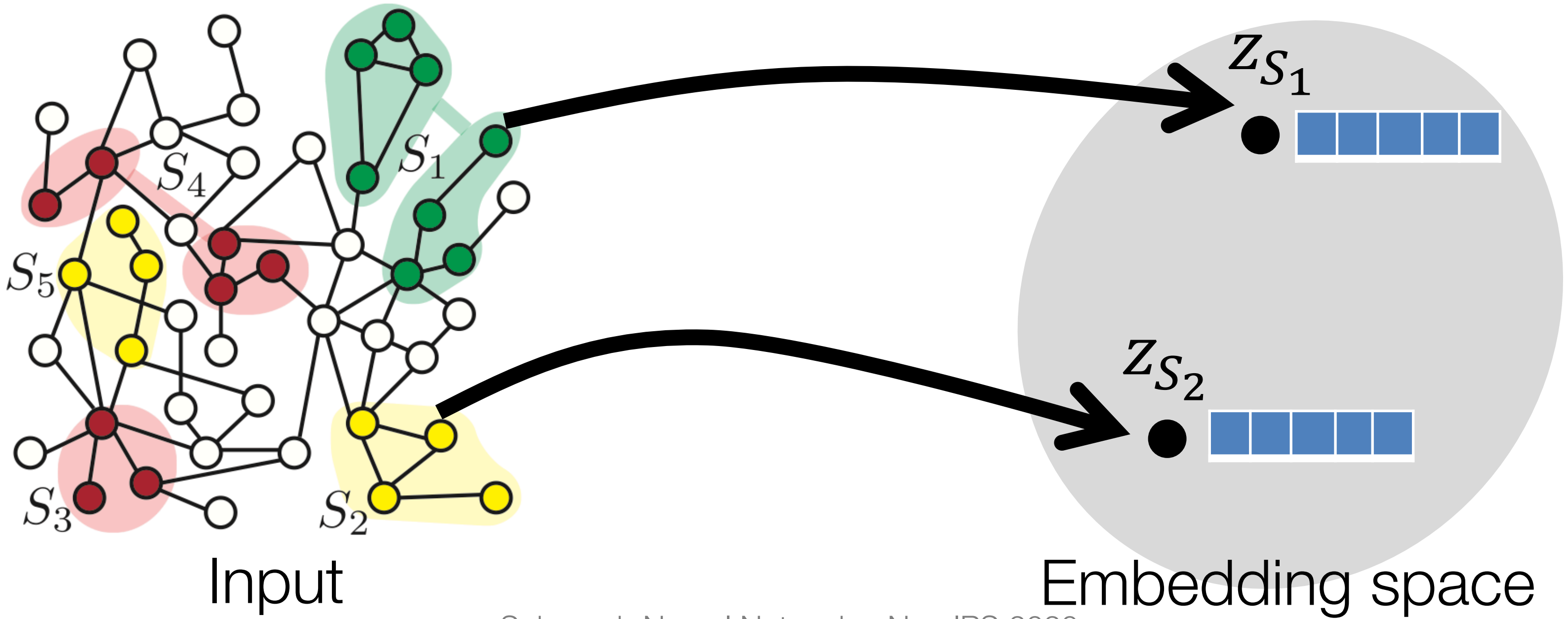
Diagnosis Task

- **Graph:** Consider a graph G built from the standardized vocabulary of phenotypes:
  - Nodes: phenotypes; edges: relationships between phenotypes
  - Patient is a set of phenotypes, a subgraph S in G
- **Learning Task:** Predict the disease (label) most consistent with the phenotype subgraph S



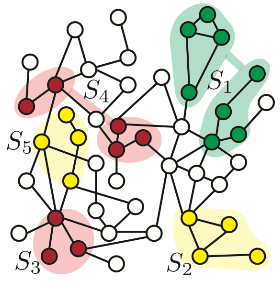
Problem Formulation

- **Goal:** Learn subgraph embeddings such that the likelihood of preserving subgraph topology is maximized in the embedding space
  - S_i and S_j with similar subgraph topology should be embedded close together in the embedding space

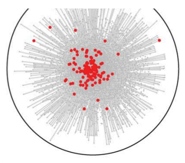
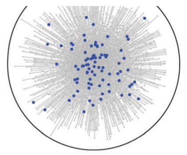


Why are subgraphs challenging?

- Need to predict over structures of varying size:
  - How to represent subgraphs that are not k-hop neighborhoods?

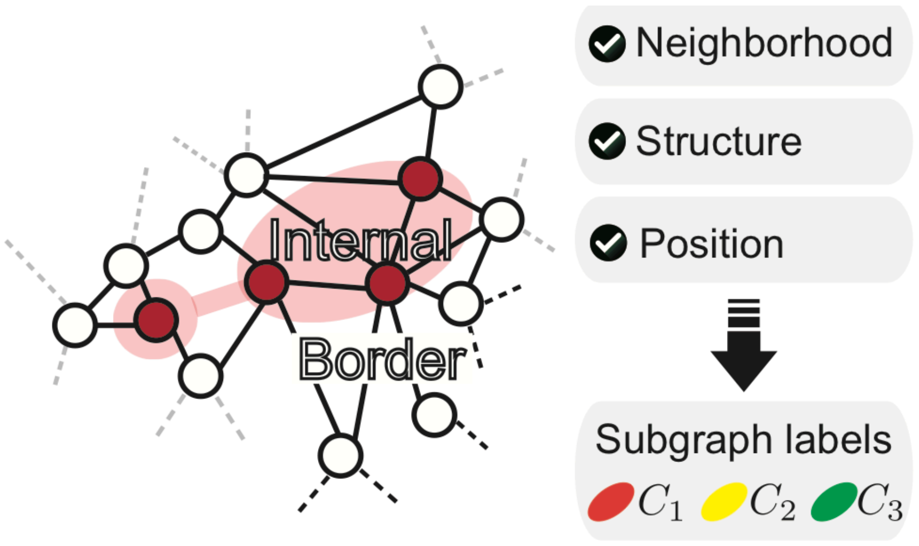


- Rich connectivity patterns, both internally a externally through interactions with the rest of G:
  - How to inject this information into a GNN?
- Subgraphs can be:
  - Localized and reside in our region of the graph
  - Distributed across multiple local neighborhoods

Subgraph Neural Networks

Problem (Subgraph Representations and Property Prediction). *Given subgraphs* $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$, SUBGNN *specifies a neural message passing architecture* $E_\mathcal{S}$ *that generates a* d_s*-dimensional subgraph representation* $\mathbf{z}_S \in \mathbb{R}^{d_s}$ *for every subgraph* $S \in \mathcal{S}$. SUBGNN *uses the representations to learn a subgraph classifier* $f : \mathcal{S} \rightarrow \{1, 2, \ldots, C\}$ *for subgraph labels* $f(S) = \hat{y}_S$.



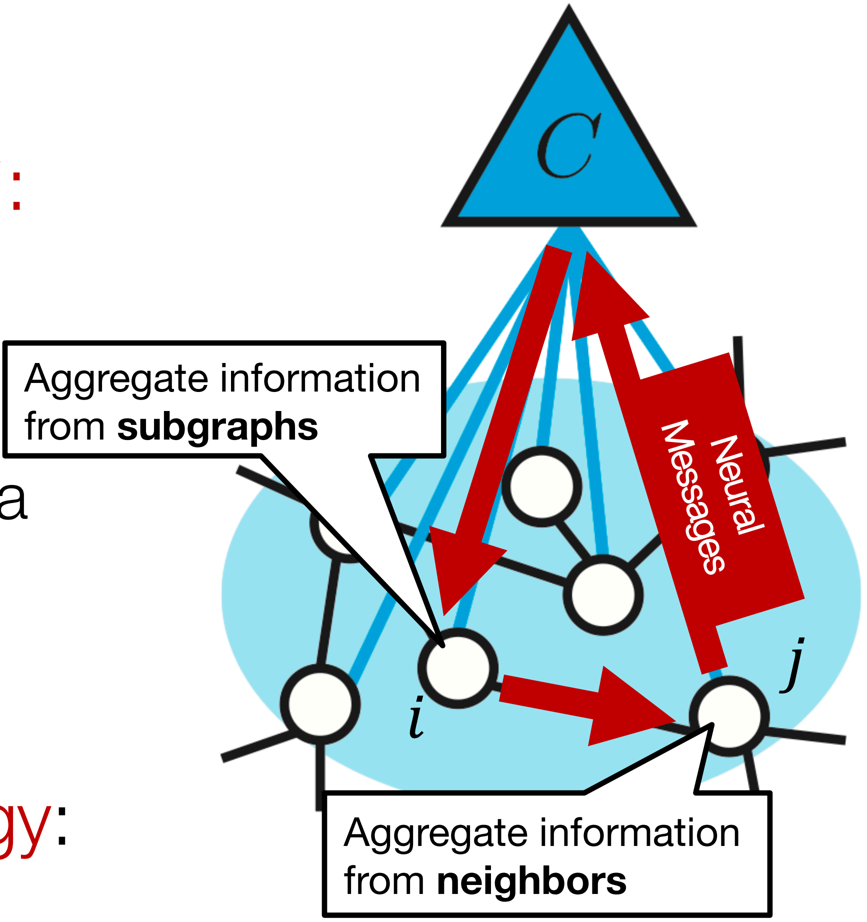
SUB-GNN Channel	SUB-GNN Subchannel: Internal (I)	SUB-GNN Subchannel: Border (B)
Position (P)	Distance between S_i's components	Distance between S_i and rest of G
Neighborhood (N)	Identity of S_i's internal nodes	Identity of S_i's border nodes
Structure (S)	Internal connectivity of S_i	Border connectivity of S_i

A Note on Problem Formulation

- SubGNN puts forward a definition of a subgraph prediction learning task
- It is different from other canonical tasks on graphs:
  - Node prediction: Predict property of <u>a node</u>
  - Link prediction: Predict property of <u>a node pair</u>
  - Graph prediction: Predict property of <u>an entire graph</u>

SubGNN: Overview

- Part 1: Hierarchically propagate messages in G:
  - Propagate messages from anchor patches to subgraphs
  - Aggregate messages into a final subgraph embedding
- Part 2: Route messages through 3 channels to capture subgraph topology: position, neighborhood, structure


C
Aggregate information from **subgraphs**
Neural Messages
i
j
Aggregate information from **neighbors**

#1: Subgraph Message Passing

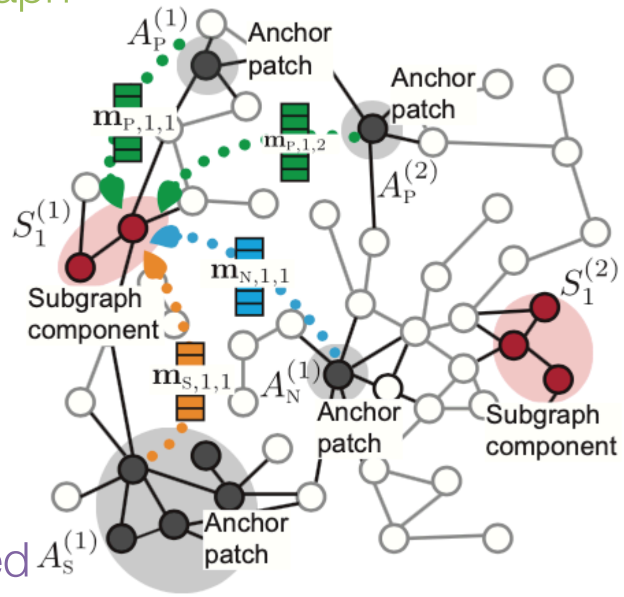
- Property x-specific messages m_x are propagated from anchor patches to subgraph components
- Anchor patches are helper subgraphs randomly sampled from G; patches A_P, A_N, and A_S for position, neighborhood and structure

similarity function between subgraph component and an anchor patch

$$\mathrm{MSG}_{\mathrm{X}}^{A\to S} = \boxed{\gamma_{\mathrm{X}}}(S^{(c)}, A_{\mathrm{X}}) \cdot \mathbf{a}_{\mathrm{X}}$$

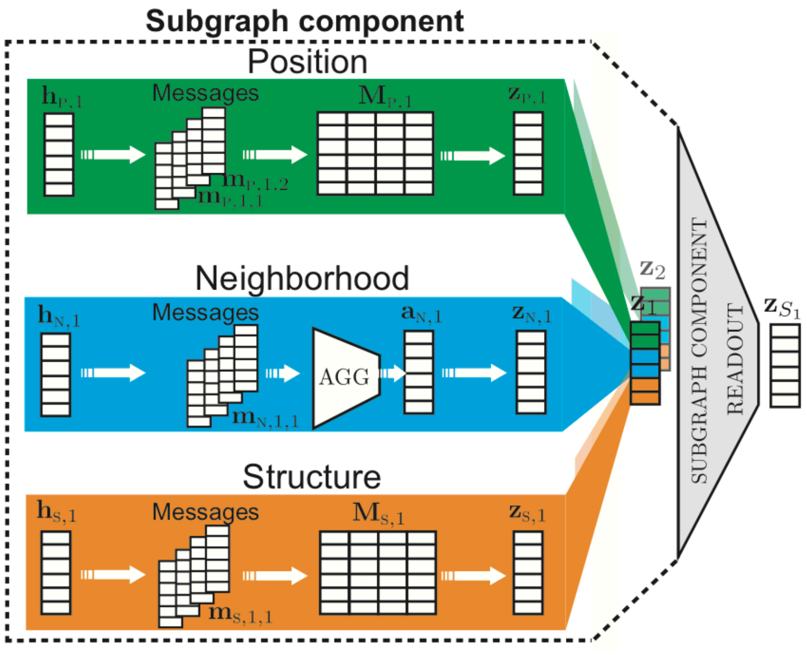
$$\mathbf{g}_{\mathrm{X},c} = \mathrm{AGG}_M(\{\mathrm{MSG}_{\mathrm{X}}^{A_{\mathrm{X}}\to S^{(c)}}\ \forall A_{\mathrm{X}} \in \mathcal{A}_{\mathrm{X}}\}),$$
$$\boxed{\mathbf{h}_{\mathrm{X},c}} \leftarrow \sigma(\mathbf{W}_{\mathrm{X}} \cdot [\mathbf{g}_{\mathrm{X},c}; \mathbf{h}_{\mathrm{X},c}]),$$

property-specific representation of subgraph component at the previous layer that gets updated



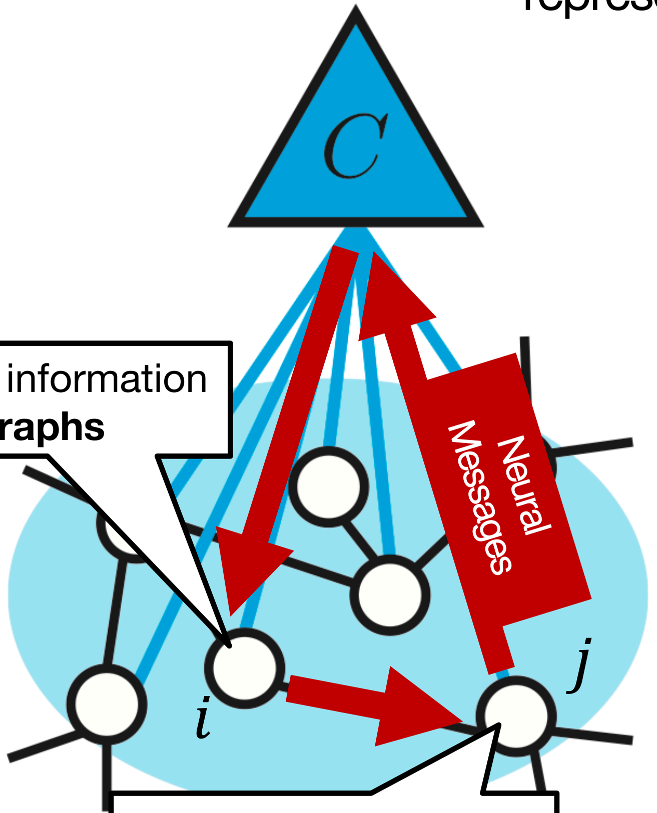
#2: Property-aware Routing

- SubGNN specifies three channels for position, neighborhood, and structure
- Each channel x has three key elements:
  - Similarity function $\gamma_x: (S^{(c)}, A_x) \rightarrow [0,1]$ to weigh messages exchanged between patches and subgraph components
  - Anchor patch sampling function $\varphi_x: (G, S^{(c)}) \rightarrow A_x$ to sample patches from underlying graph
  - Anchor patch encoder $\psi_x: A_x \rightarrow a_x$ to encode patches into embeddings a_x
- These functions can be learned or pre-defined



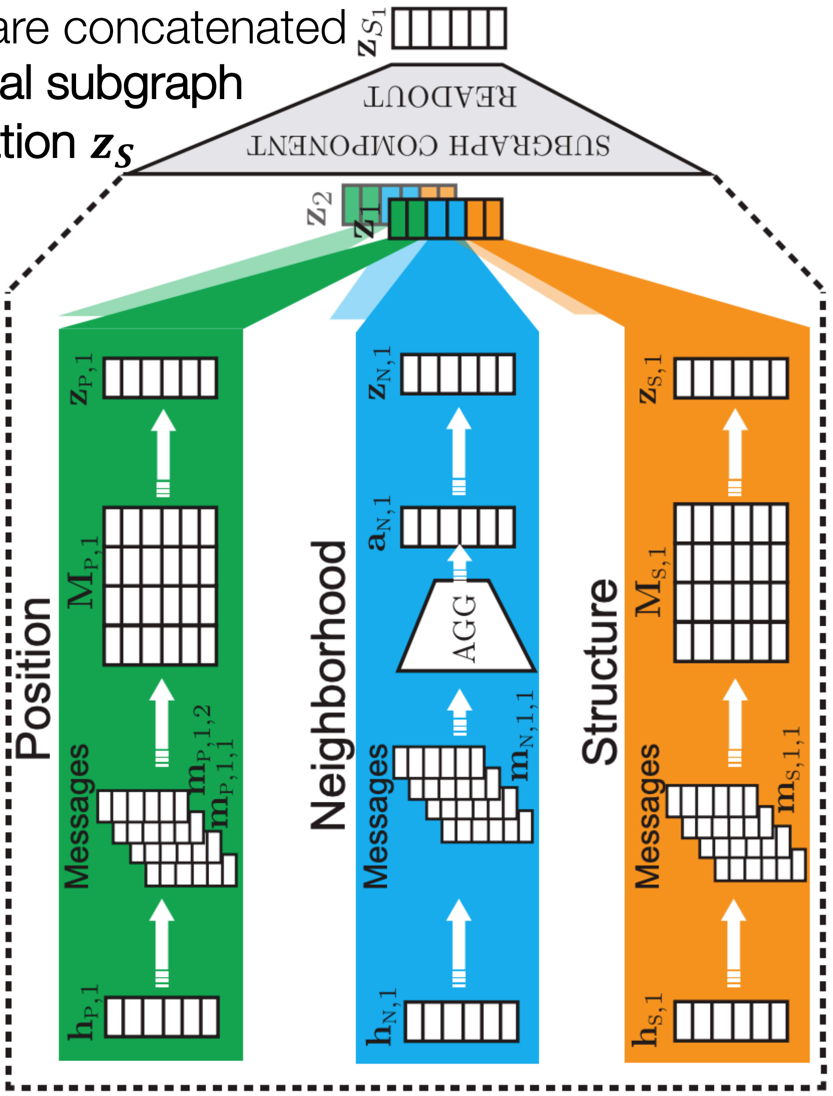
SubGNN: Recap

Channel outputs $\boldsymbol{z}_x$ are concatenated to produce **a final subgraph representation $\boldsymbol{z}_S$**

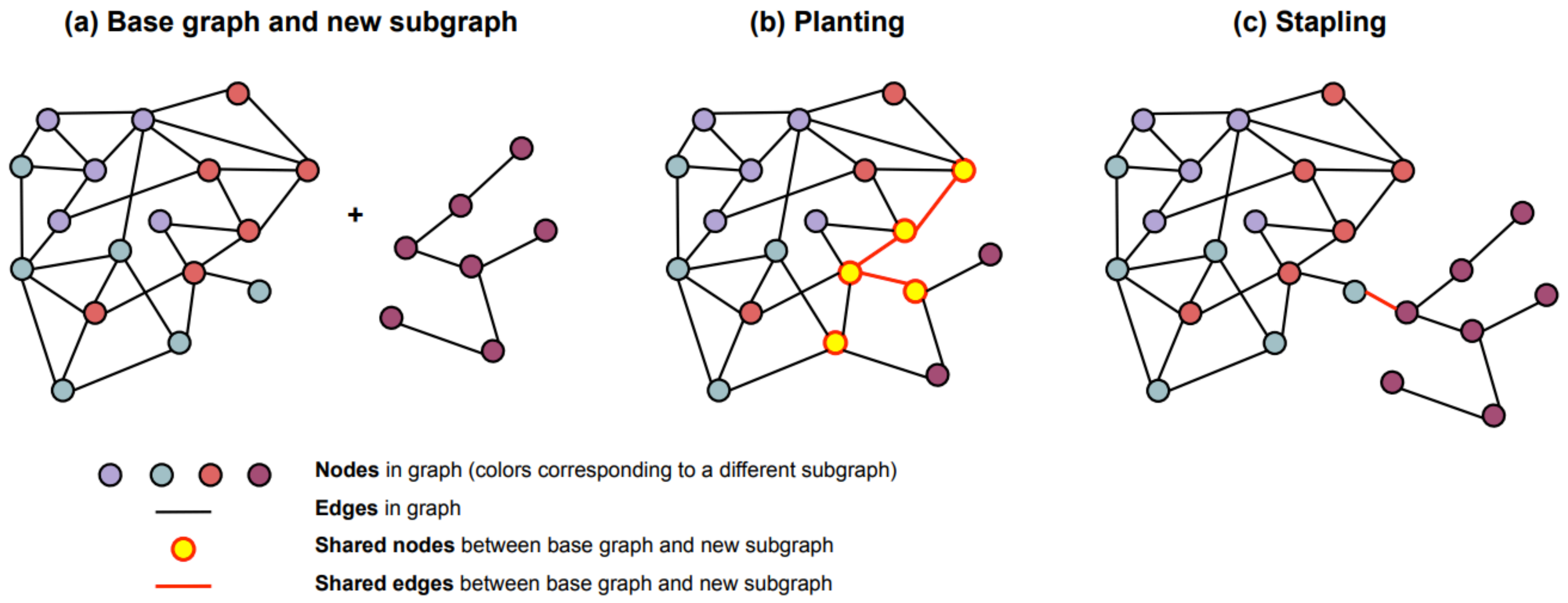


Aggregate information from **subgraphs**

Aggregate information from **neighbors**



Setup: Subgraph Datasets


(a) Base graph and new subgraph
+
(b) Planting
(c) Stapling
Nodes in graph (colors corresponding to a different subgraph)
Edges in graph
Shared nodes between base graph and new subgraph
Shared edges between base graph and new subgraph

Subgraph labels: Binned values of a metric act as subgraph labels

Metrics:

- DENSITY tests if a method can capture the internal structure of subgraphs
- CUT RATIO tests if a method can capture border structure
- CORENESS tests if a method can capture border structure and position
- COMPONENT tests if a method can capture internal and external position

Results: Synthetic Data

Method	DENSITY	CUT RATIO	CORENESS	COMPONENT
SUB-GNN (Ours)	**0.919**±**0.016**	**0.629**±**0.039**	**0.659**±**0.092**	**0.958**±**0.098**
Node Averaging	0.429±0.041	0.358±0.055	0.530±0.050	0.516±<0.001
Meta Node (GIN)	0.442±0.052	0.423±0.057	0.611±0.050	0.784±0.046
Meta Node (GAT)	0.690±0.021	0.284±0.052	0.519±0.076	0.935±<0.001
Sub2V				0.039
Sub2V				0.013
Sub2V				0.021
Graph				0.081

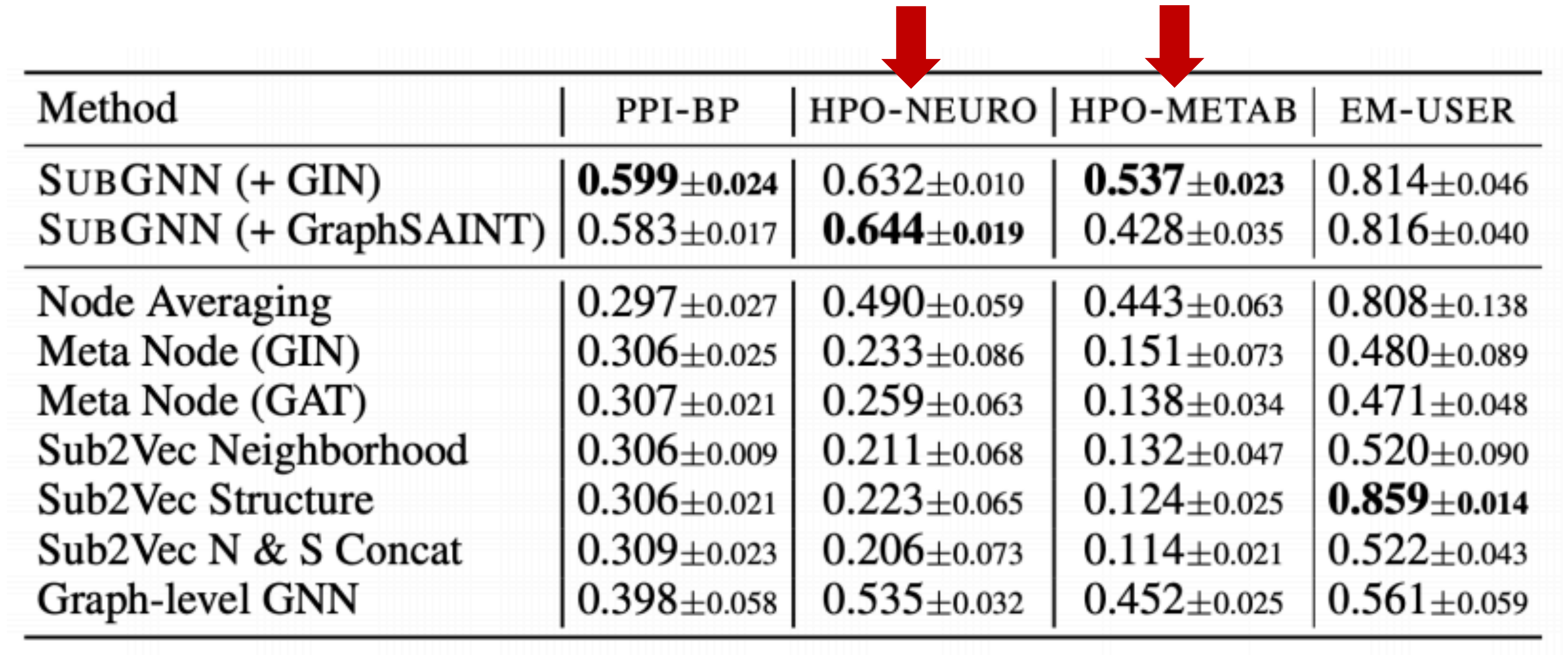
Conclusion: SubGNN can capture well different aspects of subgraph topology (position, neighborhood, structure)

- Shown are Micro-F1 scores + std across 100 runs
- SubGNN outperforms baselines by 75.4%; the strongest baseline by 17%
- Graph classification (GC) methods:
  - perform quite well on DENSITY (internal structure), as expected
  - perform poorly on datasets requiring a notion of position or border connectivity
- Meta-node methods:
  - perform well on COMPONENT dataset

Real-World Datasets

- Four real world datasets
- Each consists of a base graph and subgraphs with associated labels
  - HPO-METAB and HPO-NEURO are clinical diagnostic tasks
  - They ask the following: What is the subcategory of metabolic/neurological disease consistent with the phenotypes (i.e., phenotype subgraph)?

Results: Real-World Datasets



Method	PPI-BP	HPO-NEURO	HPO-METAB	EM-USER
SUBGNN (+ GIN)	**0.599**$\pm$**0.024**	0.632$\pm$0.010	**0.537**$\pm$**0.023**	0.814$\pm$0.046
SUBGNN (+ GraphSAINT)	0.583$\pm$0.017	**0.644**$\pm$**0.019**	0.428$\pm$0.035	0.816$\pm$0.040
Node Averaging	0.297$\pm$0.027	0.490$\pm$0.059	0.443$\pm$0.063	0.808$\pm$0.138
Meta Node (GIN)	0.306$\pm$0.025	0.233$\pm$0.086	0.151$\pm$0.073	0.480$\pm$0.089
Meta Node (GAT)	0.307$\pm$0.021	0.259$\pm$0.063	0.138$\pm$0.034	0.471$\pm$0.048
Sub2Vec Neighborhood	0.306$\pm$0.009	0.211$\pm$0.068	0.132$\pm$0.047	0.520$\pm$0.090
Sub2Vec Structure	0.306$\pm$0.021	0.223$\pm$0.065	0.124$\pm$0.025	**0.859**$\pm$**0.014**
Sub2Vec N & S Concat	0.309$\pm$0.023	0.206$\pm$0.073	0.114$\pm$0.021	0.522$\pm$0.043
Graph-level GNN	0.398$\pm$0.058	0.535$\pm$0.032	0.452$\pm$0.025	0.561$\pm$0.059

- SubGNN outperforms baselines by an average of 77% on synthetic and 125% on real-world datasets
- SubGNN channels encode their intended properties

Standard deviations from runs with 10 random seeds

Plan for Today

- ✓ Safe drugs and drug combinations

  Methods: Multi-relational link prediction on KGs

- ✓ Patient outcomes & disease classification

  Methods: Subgraph embeddings

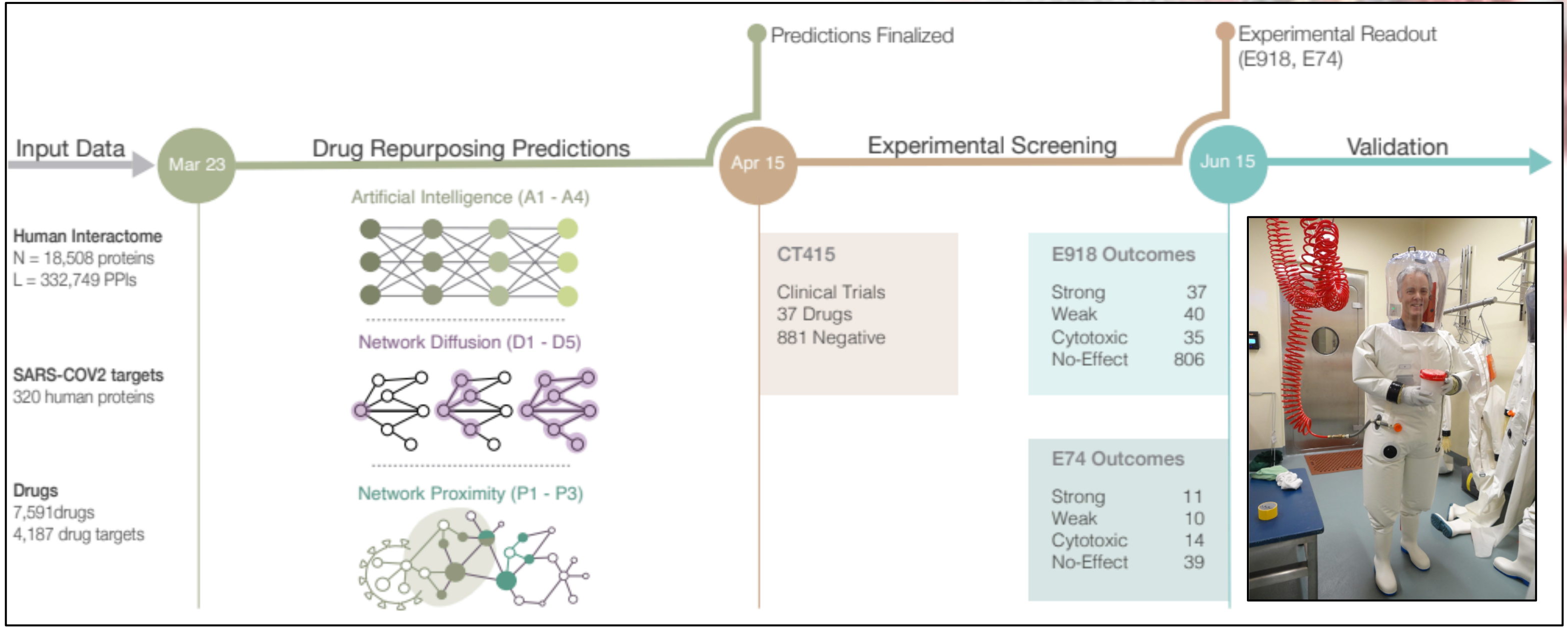
- Effective disease treatments

  Methods: Few-shot learning for graphs

Finding Cures for Emerging Diseases

The traditional approach of iterative development, experimental testing, clinical validation, and approval of new drugs are not feasible.

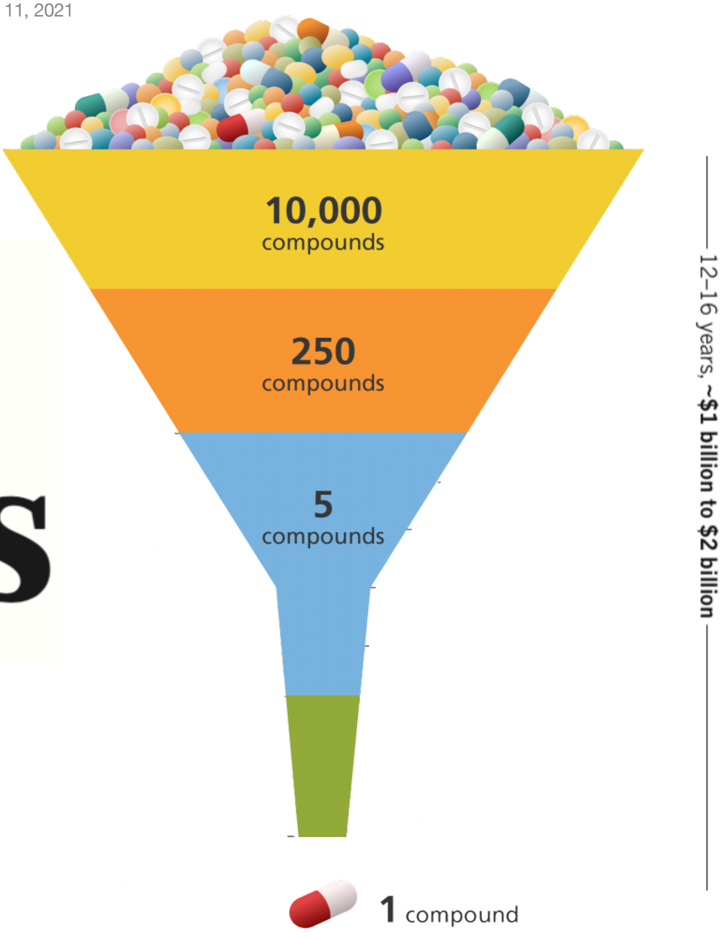
A more realistic strategy relies on drug repurposing, requiring us to identify clinically approved drugs that have a therapeutic effect in COVID-19 patients.

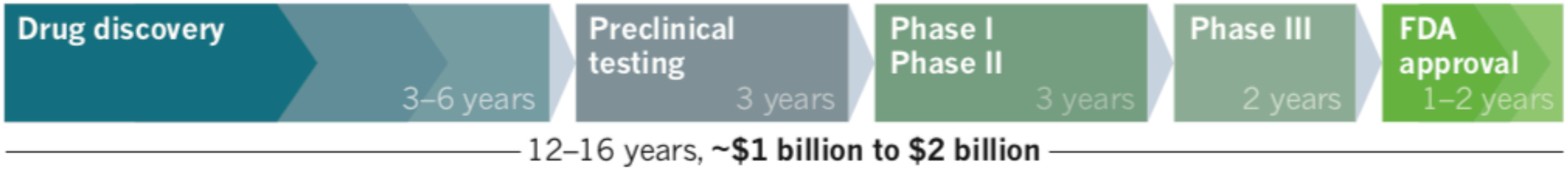


Network Medicine Framework for Identifying Drug Repurposing Opportunities for Covid-19, *arXiv:2004.07229*

New tricks for old drugs

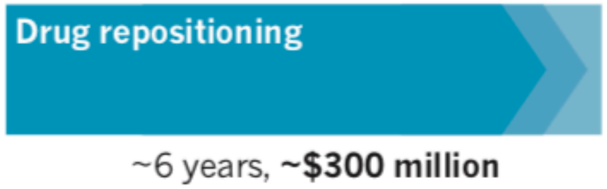
Faced with skyrocketing costs for developing new drugs, researchers are looking at ways to repurpose older ones — and even some that failed in initial trials.



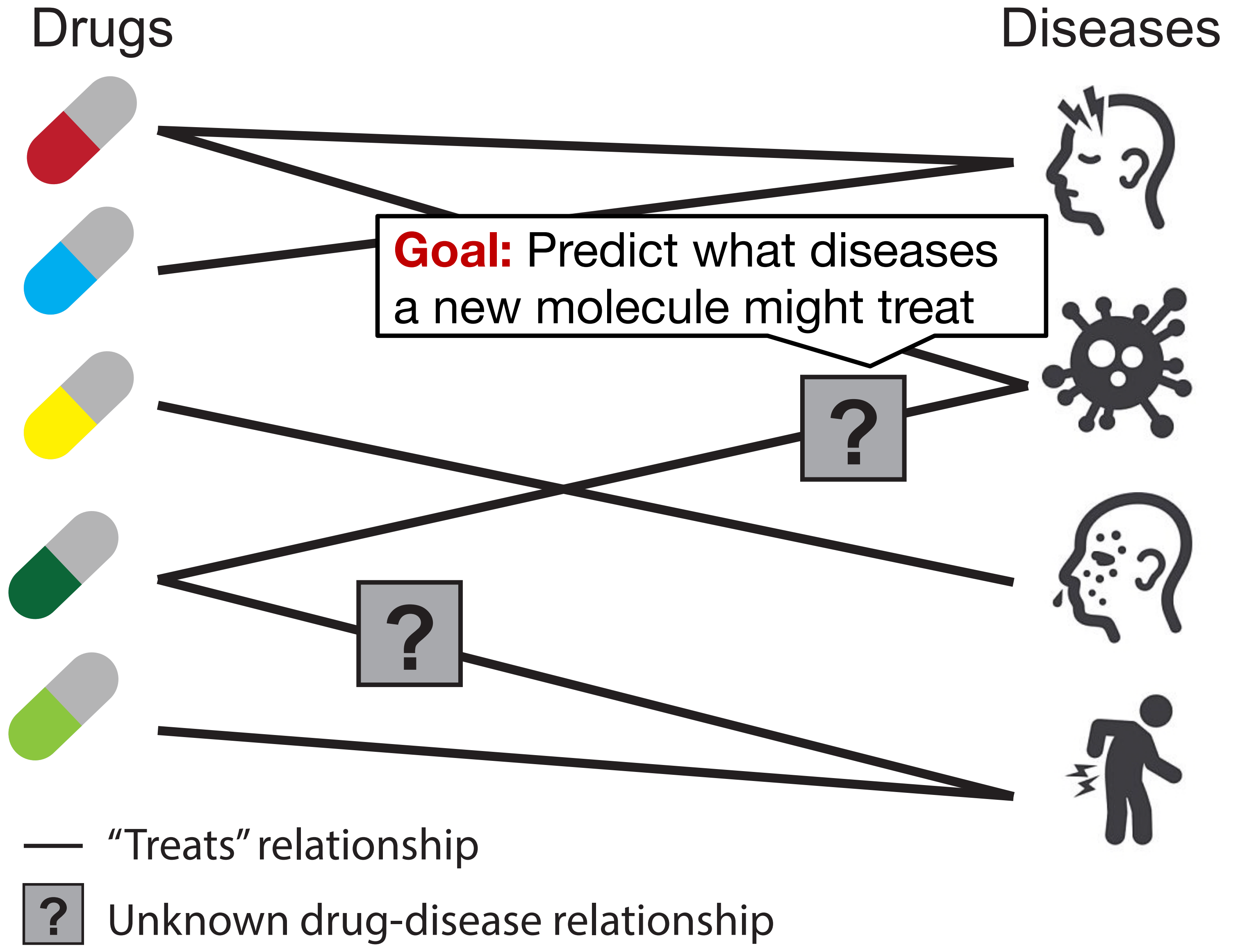


A SHORTER TIMESCALE

Because most repositioned drugs have already passed the early phases of development and clinical testing, they can potentially win approval in less than half the time and at one-quarter of the cost.



What drug treats what disease?



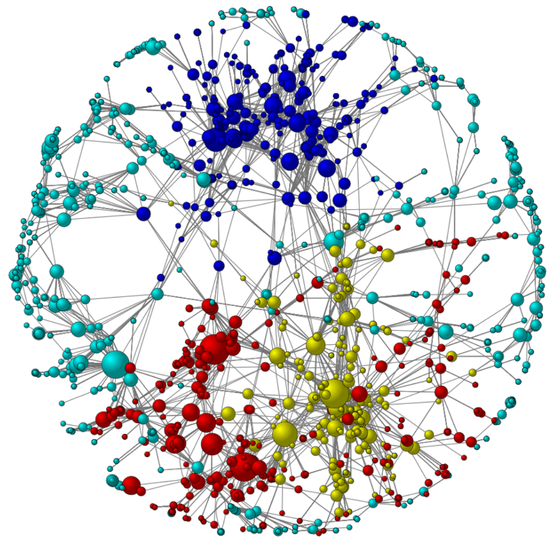
Goal: Predict what diseases a new molecule might treat

Why is finding treatments for a new disease challenging?

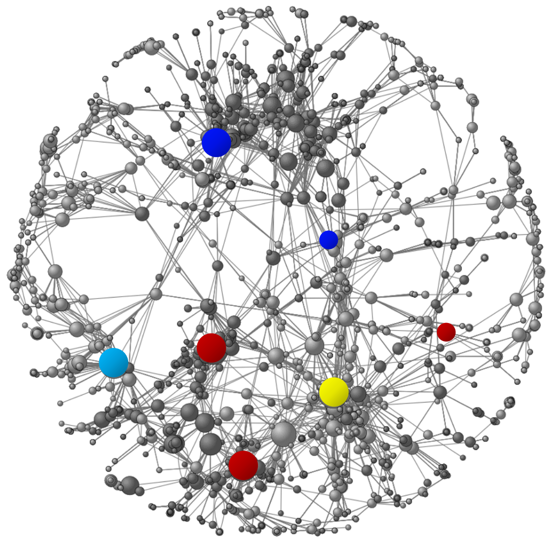
Generalizing to new phenomena is hard:

- Prevailing methods require abundant label information
- However, labeled examples are scarce
- Examples: Novel drugs in development, emerging diseases, rare diseases, hard-to-diagnose patients

What prevailing methods assume



What happens in real world



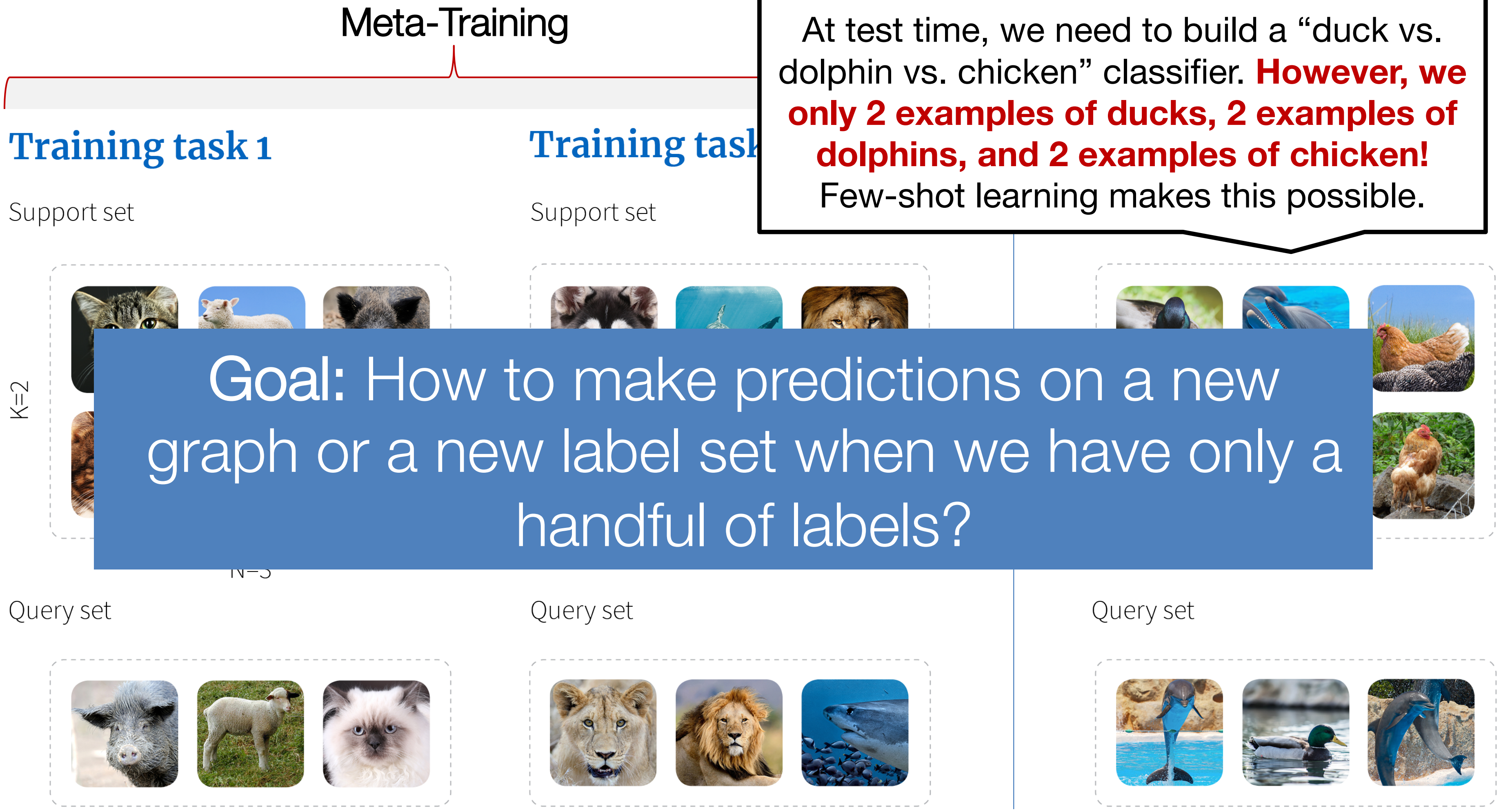
Background: Meta Learning

- Meta-learning model
  - Trained over a variety of learning tasks
  - Optimized for best performance on a distribution of tasks, including potentially unseen tasks
- Each task is associated with a dataset D, containing both feature vectors and true labels
- The optimal model parameters are:

$$\theta^* = \arg\min_\theta \mathbb{E}_{D\sim p(D)}[\mathcal{L}_\theta(D)]$$

- It looks very similar to a normal learning task, but one dataset is considered as one data sample

Background: Few-Shot Learning



At test time, we need to build a "duck vs. dolphin vs. chicken" classifier. **However, we only 2 examples of ducks, 2 examples of dolphins, and 2 examples of chicken!** Few-shot learning makes this possible.

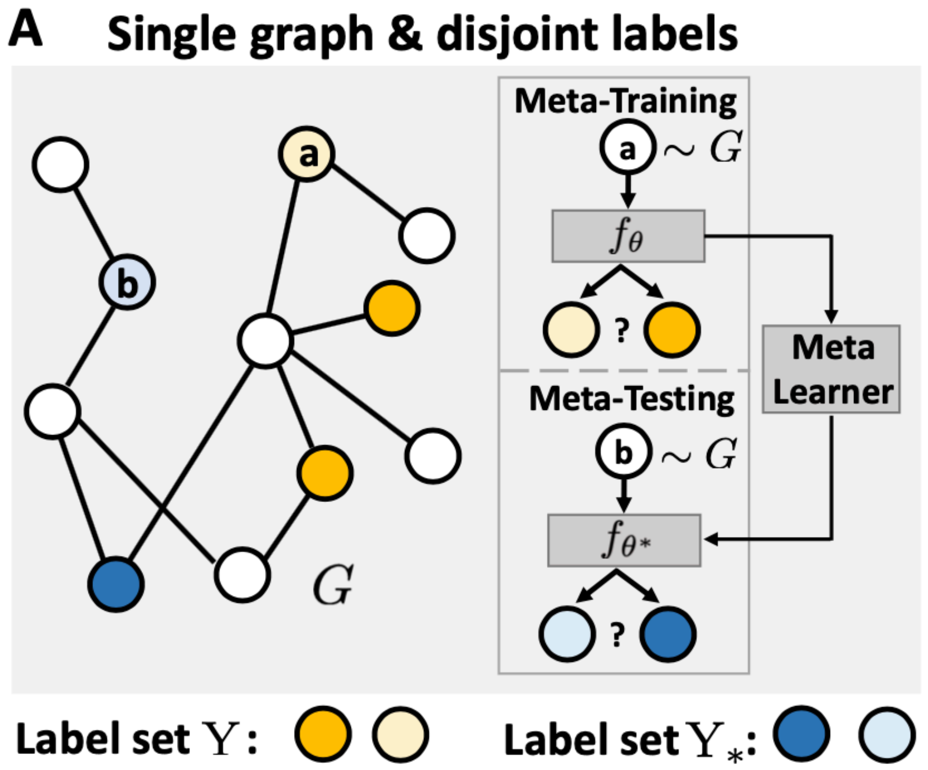
Goal: How to make predictions on a new graph or a new label set when we have only a handful of labels?

An example of 2-shot 3-way image classification

Few-shot learning: Instantiation of meta learning in the field of supervised learning

K-shot N-class classification: K labeled examples for each of N classes

Problem Formulation: G-Meta

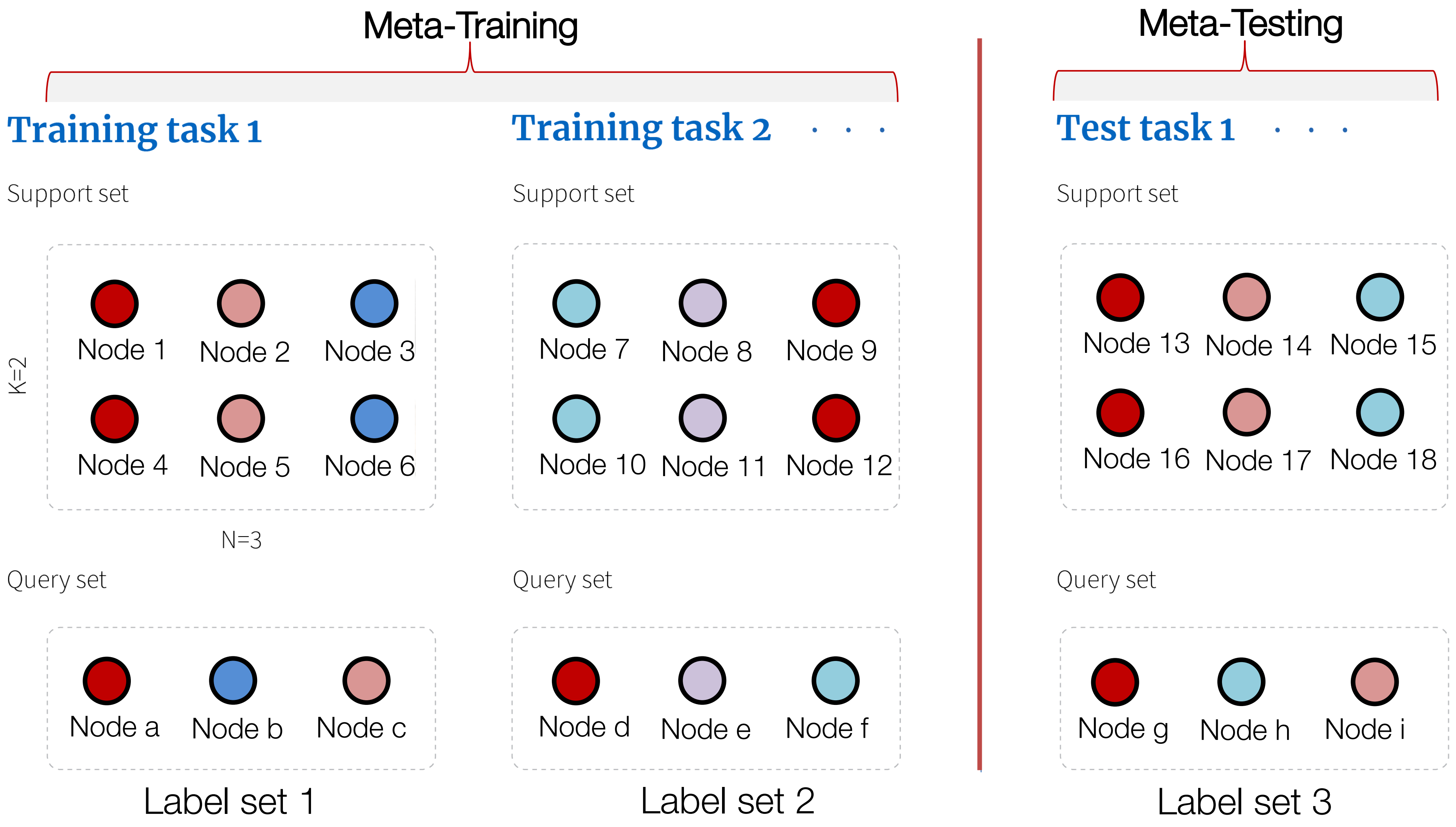


Meta-learner needs to classify an unseen label set by observing other label sets in the same graph

Each task is a batch of **a few** nodes/edges from a **different** label set in the **same** graph

Graph meta-learning problem 1: Single Graph and Disjoint Labels. We have a graph G with a distribution of label set $p(\mathrm{Y}|G)$. The goal is to adapt to an unseen label set $\mathrm{Y}_* \sim p(\mathrm{Y}|G)$ by learning from tasks with other label sets $\mathrm{Y}_i \sim p(\mathrm{Y}|G)$, where $\mathrm{Y}_i \cap \mathrm{Y}_* = \emptyset$ for every label set Y_i.

G-Meta: Overview

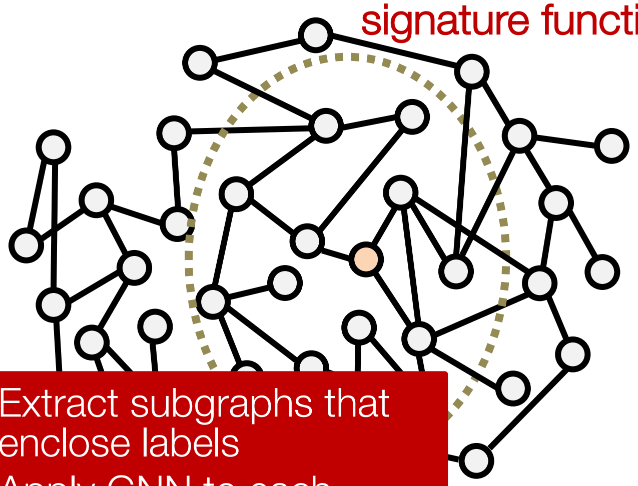


○ ~ single graph

Key Idea: Local Subgraphs

- Neural routing across subgraphs (not entire graphs!)
  - Subgraph signature functions learn how to map the structure of a sampled subgraphs to an effective initialization for a GNN
- We consider a distribution over subgraphs as the distribution over tasks from which a global set of parameters are learnt
- Deploy this strategy to train GNNs few-shot link prediction

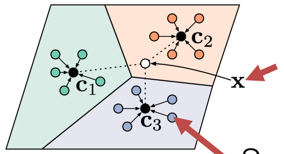
G-Meta: Subgraph signature functions



Extract subgraphs that enclose labels

Apply GNN to each subgraph **individually**

What is the value of subgraphs?



- Two sources of GNN power:
  - Label propagation: Nodes with the same label are nearby in the graph
  - Structure similarity: Nodes with the same label have similar network shapes in their local neighborhoods
- When labels are scarce:
  - **Label propagation** is not sufficient
    - When only a handful of nodes are labeled, it is challenging to efficiently propagate labels through the entire graph
    - Graph-level embeddings cannot capture structure of large graphs
  - Need to better leverage **structural equivalence**
    - Local subgraphs capture structural information
    - G-Meta learns a metric to classify query subgraph using the closest point from the support set [It compares query subgraph embedding to the support subgraph embedding]

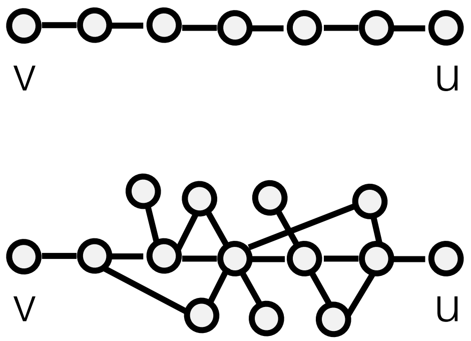
Theoretical Motivation for G-Meta

Theorem 1 (Decaying Property of Node Influence). *Let t be a path between node u and node v and let D^{t}_{GM} be a geometric mean of node degrees occurring on path t. Let $D^{t_*}_{\mathrm{GM}} = \min_t\{D^{t}_{\mathrm{GM}}\}$ and $h_* = d(u, v)$. Consider the node influence $I_{u,v}$ from v to u. Then, $I_{u,v} \le C/(D^{t_*}_{\mathrm{GM}})^{h_*}$.*

Theorem 2 (Local Subgraph Preservation Property). *Let S_u be a local subgraph for node u with neighborhood size h. Let node v be defined as: $v = \operatorname{argmax}_w(\{I_{u,w} | w \in \mathcal{V} \setminus \mathcal{V}^u\})$. Let $\bar{t}$ be a path between u and v and let $D^{\bar{t}}_{\mathrm{GM}}$ be a geometric mean of node degrees occurring on path $\bar{t}$. Let $D^{\bar{t}_*}_{\mathrm{GM}} = \min_{\bar{t}}\{D^{\bar{t}}_{\mathrm{GM}}\}$. The following holds: $R_h(u) \le C/(D^{\bar{t}_*}_{\mathrm{GM}})^{h+1}$.*

The influence of a node on the target node decays exponentially as we go further away from the target

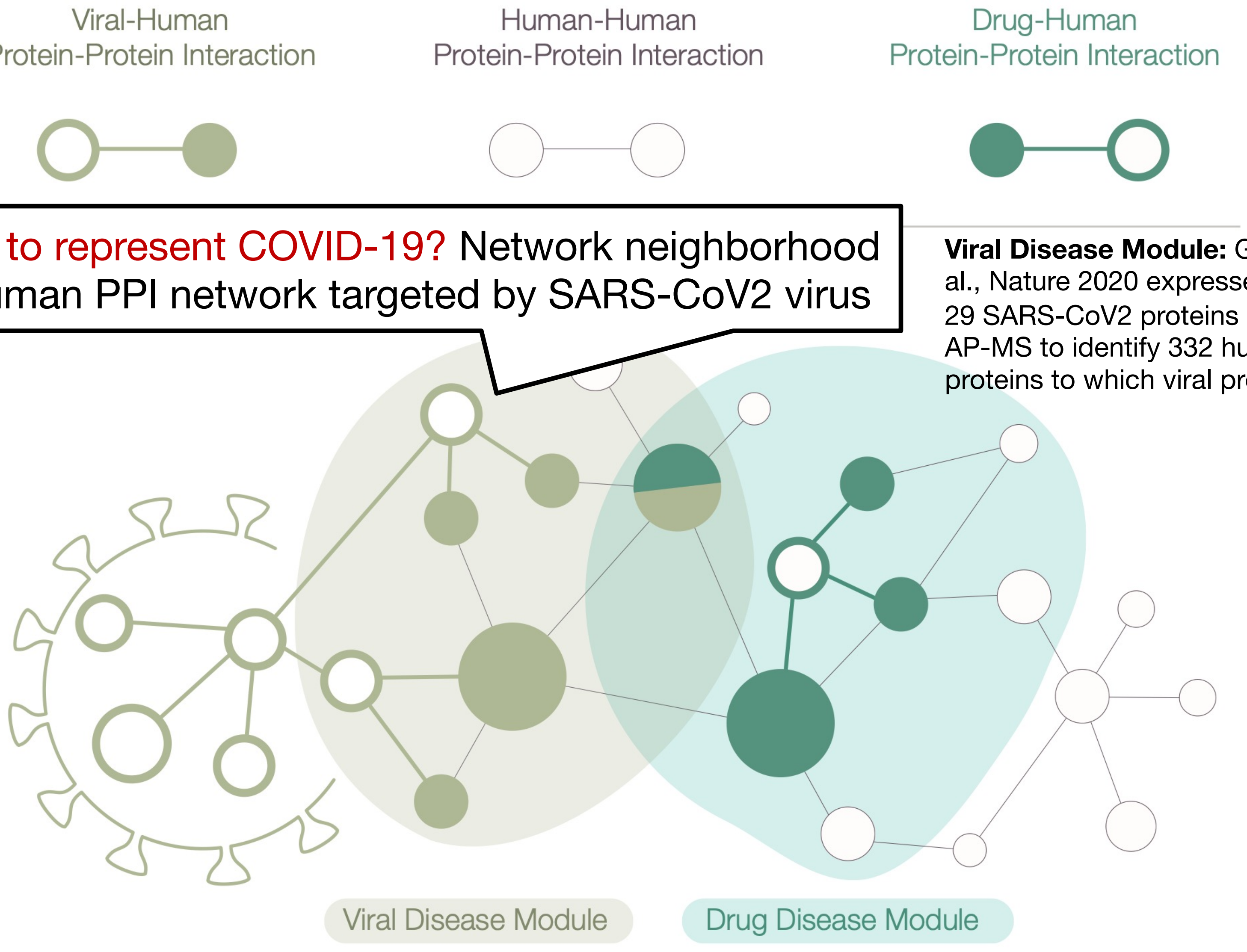
TL;DR:
- Local subgraphs around target nodes contain all the relevant information
- Local subgraphs preserve near the same feature information as the entire graph



High Influence

Low Influence

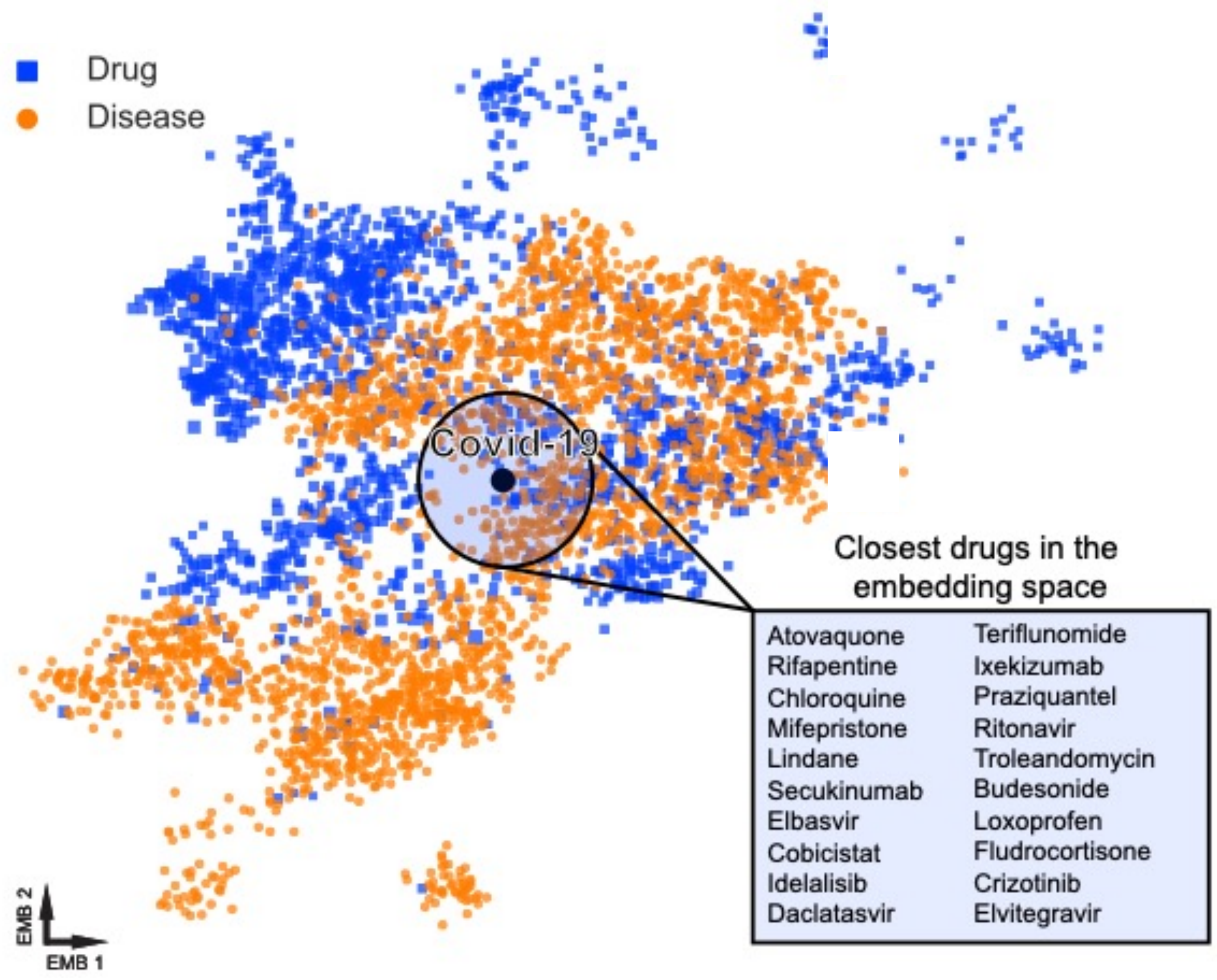
COVID-19 Repurposing Dataset


Viral-Human
Protein-Protein Interaction
Human-Human
Protein-Protein Interaction
Drug-Human
Protein-Protein Interaction
Viral Disease Module
Drug Disease Module

How to represent COVID-19? Network neighborhood of human PPI network targeted by SARS-CoV2 virus

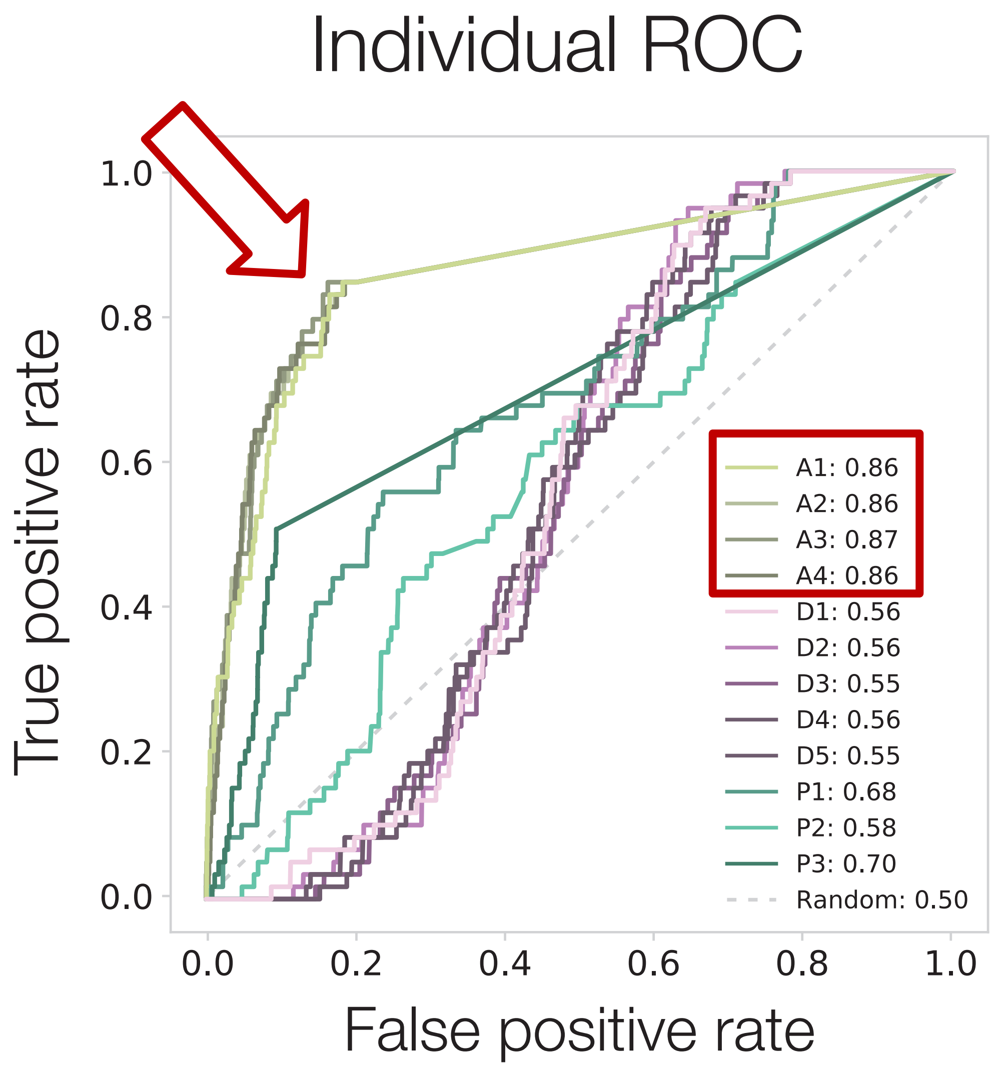
Viral Disease Module: Gordon et al., Nature 2020 expressed 26 of the 29 SARS-CoV2 proteins and used AP-MS to identify 332 human proteins to which viral proteins bind

Results: Embedding Space



Atovaquone	Teriflunomide
Rifapentine	Ixekizumab
Chloroquine	Praziquantel
Mifepristone	Ritonavir
Lindane	Troleandomycin
Secukinumab	Budesonide
Elbasvir	Loxoprofen
Cobicistat	Fludrocortisone
Idelalisib	Crizotinib
Daclatasvir	Elvitegravir

Results: COVID-19 Repurposing



We test each pipeline's ability to recover drugs currently in clinical trials for COVID-19 (67 drugs from ClinicalTrials.gov).

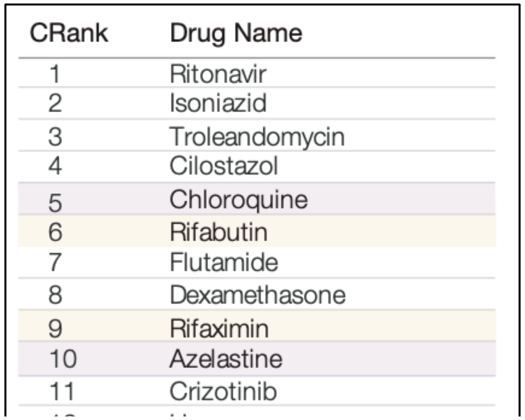
The best individual ROC curves are obtained by the AI-based methods.

The second-best performance is provided by the proximity P3. Close behind is P1 with AUC = 0.68 and AUC = 0.58.

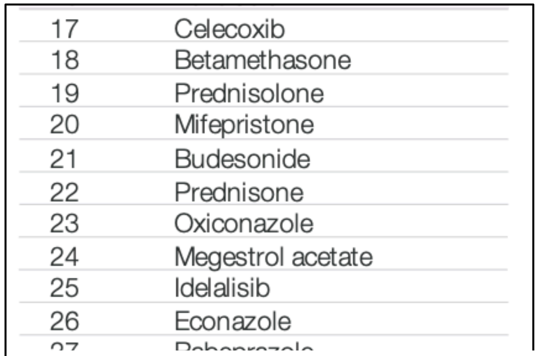
Diffusion methods offer ROC between 0.55-0.56.

Results: Experimental Validation of Predictions

National Emerging Infectious Diseases Laboratories (NEIDL)



CRank	Drug Name
1	Ritonavir
2	Isoniazid
3	Troleandomycin
4	Cilostazol
5	Chloroquine
6	Rifabutin
7	Flutamide
8	Dexamethasone
9	Rifaximin
10	Azelastine
11	Crizotinib



17	Celecoxib
18	Betamethasone
19	Prednisolone
20	Mifepristone
21	Budesonide
22	Prednisone
23	Oxiconazole
24	Megestrol acetate
25	Idelalisib
26	Econazole

Ranked lists of drugs

New algorithms:
Prioritizing Network Communities, *Nature Communications* 2018
Subgraph Neural Networks, *NeurIPS* 2020
Graph Meta Learning via Local Subgraphs, *NeurIPS* 2020

Results: 918 compounds screened for their efficacy against SARS-CoV-2 in VeroE6 cells:

- **77 showed strong/weak effect** being active over a broad range of concentrations
- An order of **magnitude higher hit rate** among top 100 drugs than **prior work**

Results: Network Drugs

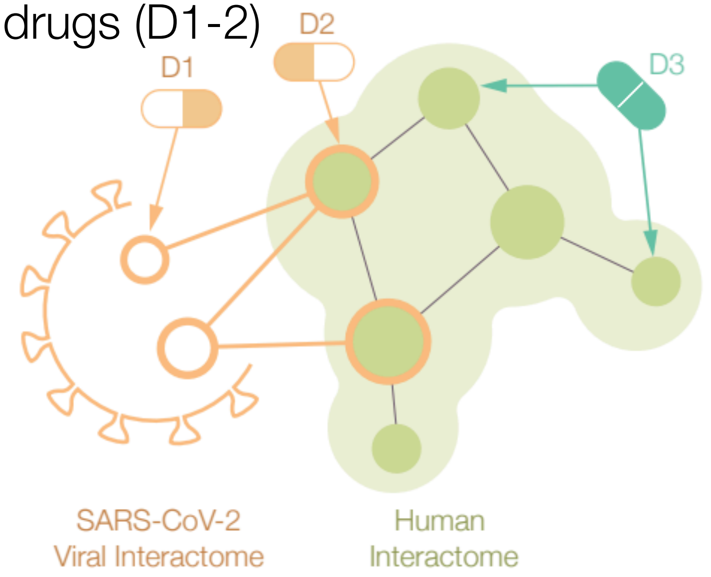
- 76/77 drugs that successfully reduced viral infection <u>do not bind</u> proteins targeted by SARS-CoV-2:
  - These drugs rely on **network-based actions** that cannot be identified by docking-based strategies

▨ Strong
□ Weak

CRank	Drug Name	CRank	Drug Name	CRank	Drug Name
5	Chloroquine	423	Pitavastatin	742	Mianserin
6	Rifabutin	431	Tenoxicam	755	Clofazimine
9	Rifaximin	438	Quinidine	767	Chlorpromazine
10	Azelastine	456	Sertraline	772	Imipramine
16	Folic acid	460	Ingenol mebutate	830	Promazine
32	Methotrexate	463	Norelgestromin	900	L-Alanine
33	Digoxin	493	Sildenafil	917	Moxifloxacin
44	Hydroxychloroquine	499	Eliglustat	933	Tasimelteon
50	Omeprazole	518	Ulipristal	995	Vandetanib
113	Clobetasol propionate	553	Cinacalcet	1000	Azilsartan medoxomil
118	Auranofin	556	Perphenazine	1020	Frovatriptan
120	Vinblastine	558	Idarubicin	1034	Zolmitriptan
199	Fluvastatin	564	Perhexiline	1035	Procarbazine
210	Clomifene	569	Amiodarone	1093	Asenapine
233	Ibuprofen	577	Duloxetine	1107	Dyclonine
235	Ivermectin	585	Toremifene	1140.5	Clemastine
243	Atorvastatin	586	Afatinib	1194	Prochlorperazine
253	Pralatrexate	601	Amitriptyline	1222	Miglustat
263	Cobimetinib	626	Meclizine	1224	Prenylamine
269	Hydralazine	635	Valsartan	1276	Dalfampridine
297	Propranolol	651	Eletriptan	1314	Cinchocaine
317	Osimertinib	673	Sotalol	1355	Methotrimeprazine
348	Vincristine	678	Thioridazine	1396	Methylthioninium
367	Doxazosin	695	Chlorcyclizine	1403	Metixene
397	Rosiglitazone	707	Omacetaxine mepesuccinate	1443	Trifluoperazine
398	Aminolevulinic acid	721	Candesartan		

58/77 drugs with positive experimental outcome are among top 750 ranked drugs

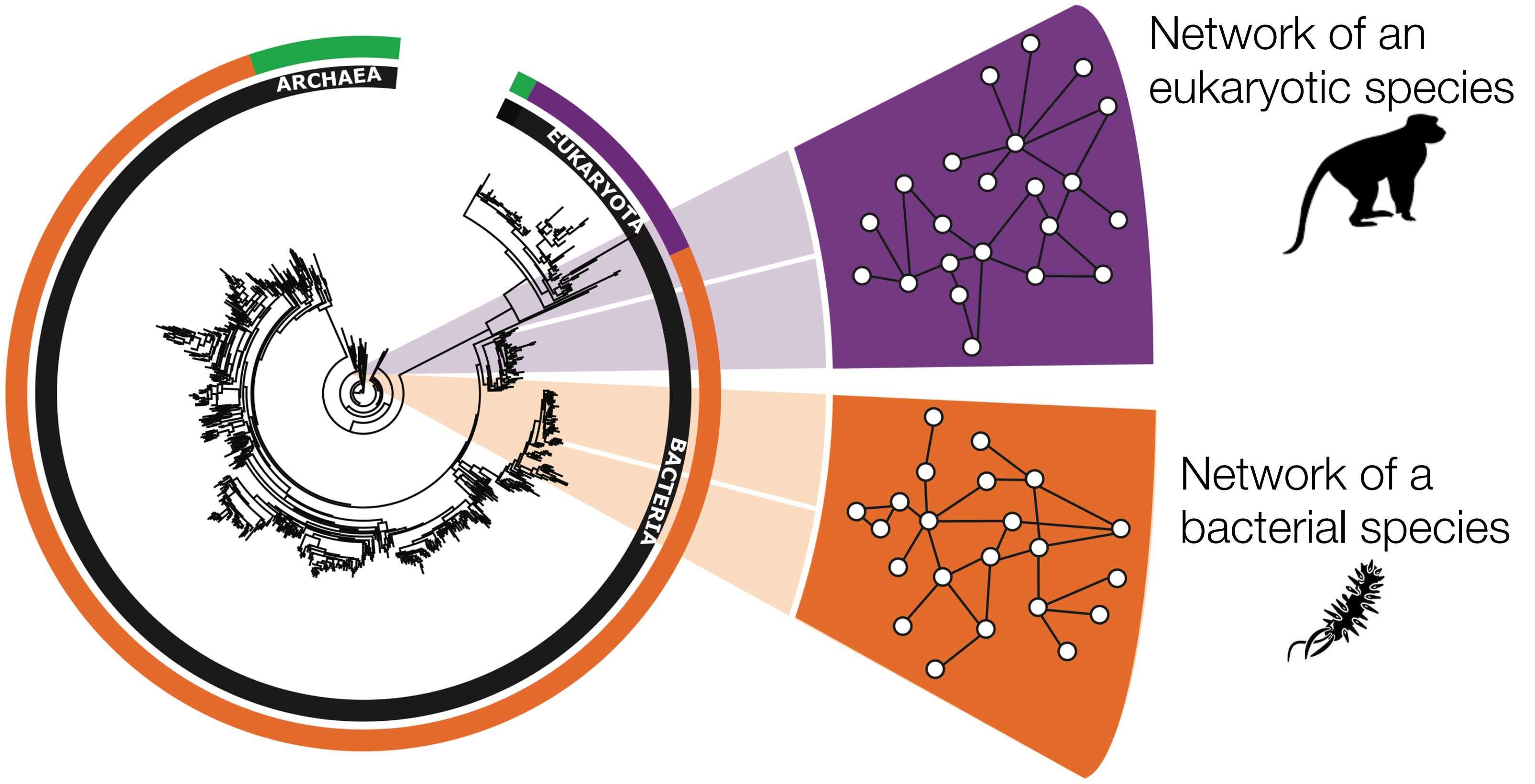
Direct target drugs (D1-2)



Network drugs (D3)

Marinka Zitnik - https://zitniklab.hms.harvard.edu - March 11, 2021

Transfer Learning Across Graphs: Tree-of-Life Dataset

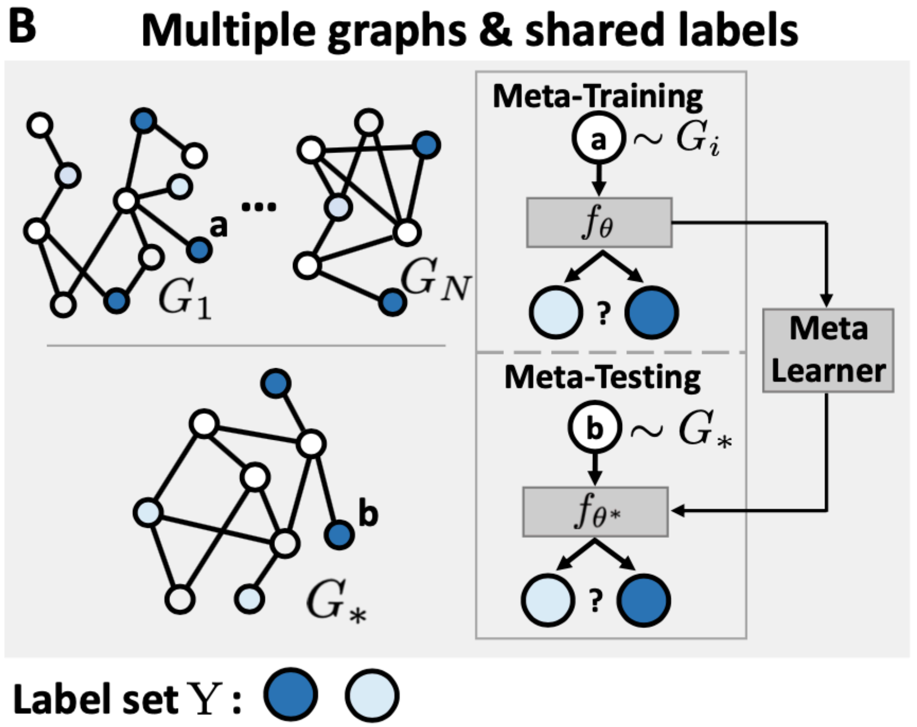

ARCHAEA
EUKARYOTA
BACTERIA
Network of an eukaryotic species
Network of a bacterial species

Motivation: How can we leverage PPI networks of model organisms to complete human PPI network?

Zitnik, Marinka, Marcus W. Feldman, and Jure Leskovec. "Evolution of resilience in protein interactomes across the tree of life." PNAS (2019): 4426-4433.

Marinka Zitnik - https://zitniklab.hms.harvard.edu - March 11, 2021

Problem Formulation: G-Meta



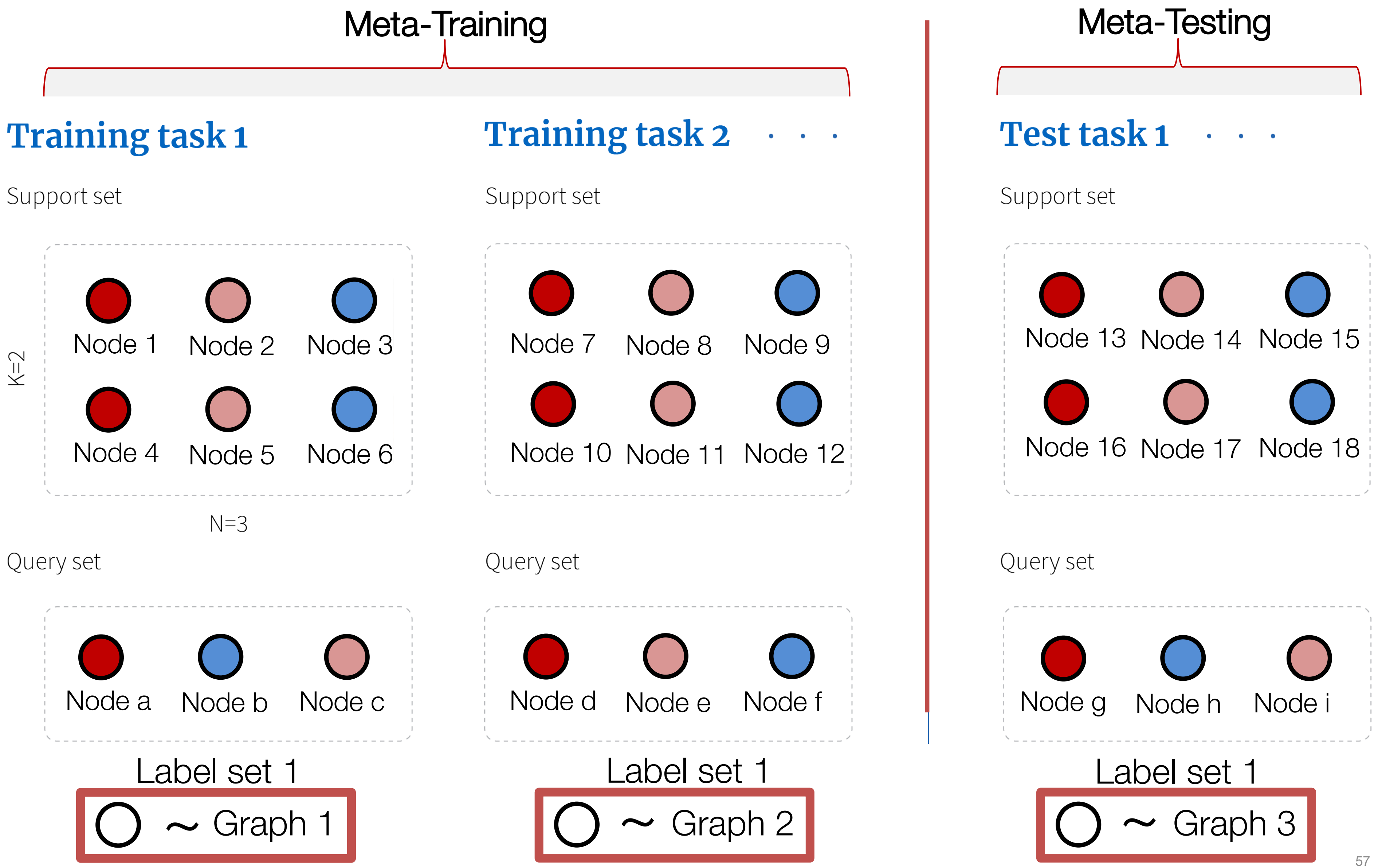
Meta-learner needs to make predictions on a new graph by learning from other graphs with the same label set

Each task is a batch of **a few** nodes/edges from the **same** label set but from a **different** graph

Graph meta-learning problem 2: Multiple Graphs and Shared Labels. We have a distribution of graphs $p(G)$ and one label set Y. The goal is to learn from graph $G_j \sim p(G)$ and quickly adapt to an unseen graph $G_* \sim p(G)$, where G_j and G_* are disjoint. All tasks share the same labels.

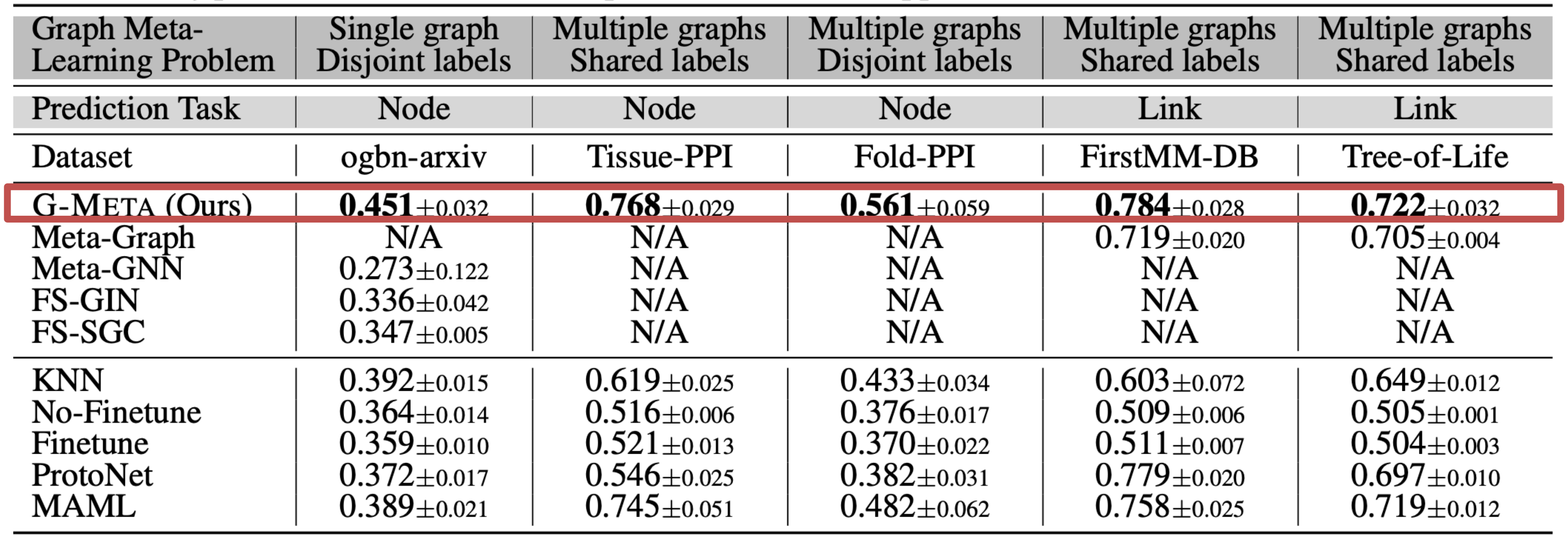
Marinka Zitnik - https://zitniklab.hms.harvard.edu - March 11, 2021

Few-Shot Learning across Graphs



Marinka Zitnik - https://zitniklab.hms.harvard.edu - March 11, 2021

G-Meta: Results



Graph Meta-Learning Problem	Single graph Disjoint labels	Multiple graphs Shared labels	Multiple graphs Disjoint labels	Multiple graphs Shared labels	Multiple graphs Shared labels
Prediction Task	Node	Node	Node	Link	Link
Dataset	ogbn-arxiv	Tissue-PPI	Fold-PPI	FirstMM-DB	Tree-of-Life
G-META (Ours)	**0.451**±0.032	**0.768**±0.029	**0.561**±0.059	**0.784**±0.028	**0.722**±0.032
Meta-Graph	N/A	N/A	N/A	0.719±0.020	0.705±0.004
Meta-GNN	0.273±0.122	N/A	N/A	N/A	N/A
FS-GIN	0.336±0.042	N/A	N/A	N/A	N/A
FS-SGC	0.347±0.005	N/A	N/A	N/A	N/A
KNN	0.392±0.015	0.619±0.025	0.433±0.034	0.603±0.072	0.649±0.012
No-Finetune	0.364±0.014	0.516±0.006	0.376±0.017	0.509±0.006	0.505±0.001
Finetune	0.359±0.010	0.521±0.013	0.370±0.022	0.511±0.007	0.504±0.003
ProtoNet	0.372±0.017	0.546±0.025	0.382±0.031	0.779±0.020	0.697±0.010
MAML	0.389±0.021	0.745±0.051	0.482±0.062	0.758±0.025	0.719±0.012

- G-Meta can successfully learn in challenging, few-shot learning settings: up to 29.9 % over previous works and 16.3 % over other meta learning methods
- G-Meta scales to large graphs: on our new Tree-of-Life dataset comprising of 1,840 graphs, 100x increase in graph size relative to prior work

Reported is multi-class classification accuracy (five-fold average) and standard deviation. N/A means the method does not apply.

Plan for Today

- ✓ Safe drugs and drug combinations

  Methods: Multi-relational link prediction on KGs

- ✓ Patient outcomes & disease classification

  Methods: Subgraph embeddings

- ✓ Effective disease treatments

  Methods: Few-shot learning for graphs