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
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.

- MKS1
- B9D1
- B9D2
- MOB4
- ANP32A

21,557 proteins
342,353 interactions
Why Networks in Biology?

Network of protein-protein interactions in human cells.

- **MKS1**
- **B9D1**
- **B9D2**
- **MOB4**
- **ANP32A**

**Disease protein**

21,557 proteins
342,353 interactions
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

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 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

Co-prescribed drugs

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

Side Effects

\{ \}

\{ \}

\{ \}

3% prob.

2% prob.

Modeling Polypharmacy Side Effects with Graph Convolutional Networks, Bioinformatics, 2018
Why is modeling drug combinations challenging?

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

Modeling Polypharmacy Side Effects with Graph Convolutional Networks, *Bioinformatics*, 2018
Polypharmacy Knowledge Graph

Mode 1
E.g., Specific type of drug-drug interaction ($r_1$)

Mode 2
E.g., drug-target interaction ($r_4$)
E.g., protein-protein interaction ($r_5$)

Node types
Edge type $i$

E.g., drugs
E.g., proteins

Marinka Zitnik - https://zitniklab.hms.harvard.edu - March 11, 2021
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 $r_3$:

- $1^{st}$ order neighbor of $v$
- $2^{nd}$ order neighbor of $v$
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:

  \[ h_v^{(0)} = x_v \]

- Per-layer update of node embeddings:

  \[ h_v^{(k)} = \phi \left( \sum_{r} \sum_{u \in N^r_v} c_{ru} W_r^{(k-1)} h_u^{(k-1)} + c_r h_v^{(k-1)} \right) \quad k = 1, \ldots, K \]

- Embeddings after \( K \) layers of neighborhood aggregation:

  \[ z_v = h_v^{(K)} \]
Heterogeneous Edge Decoder

**Input:** Embeddings of two nodes, $C$ and $S$

**Output:** Predicted edges, new discovered relationships

\[
p(C, r_1, S) = \sigma(z_c^T D_{r_1} R D_{r_1} z_s)\]

\[
p(C, r_2, S) = \sigma(z_c^T D_{r_2} R D_{r_2} z_s)\]

\[
p(C, r_3, S) = \sigma(z_c^T D_{r_3} R D_{r_3} z_s)\]

\[
p(C, r_4, S) = \sigma(z_c^T D_{r_4} R D_{r_4} z_s)\]

\[
p(C, r_n, S) = \sigma(z_c^T D_{r_n} R D_{r_n} z_s)\]

Tensor factorized model captures dependences between different edge types

$R$, $D_{r_i}$ Parameter weight matrices

Modeling Polypharmacy Side Effects with Graph Convolutional Networks, *Bioinformatics*, 2018
We need Polypharmacy Dataset

Objective:
Capture molecular, drug, and patient data for all drugs prescribed in the U.S.

We build a unique dataset:
- 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

Modeling Polypharmacy Side Effects with Graph Convolutional Networks, *Bioinformatics*, 2018

Marinka Zitnik - https://zitniklab.hms.harvard.edu - March 11, 2021
We apply Decagon to the polypharmacy network

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

Modeling Polypharmacy Side Effects with Graph Convolutional Networks, *Bioinformatics*, 2018
Results: Side Effect Prediction

![Graph showing AUROC and AP@50 scores for different methods]

- **AUROC**
  - 0.834 (Our method (Decagon))
  - 0.693 (RESCAL Tensor Factorization [Nickel et al., ICML'11])
  - 0.705 (Multi-relational Factorization [Perros, Papalexakis et al., KDD'17])
  - 0.725 (Shallow Network Embedding [Zong et al., Bioinformatics'17])

- **AP@50**
  - 0.731 (Our method (Decagon))
  - 0.476 (RESCAL Tensor Factorization [Nickel et al., ICML'11])
  - 0.567 (Multi-relational Factorization [Perros, Papalexakis et al., KDD'17])
  - 0.643 (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?

<table>
<thead>
<tr>
<th>Rank</th>
<th>Drug</th>
<th>Drug</th>
<th>Side effect</th>
<th>Evidence found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pyrimethamine</td>
<td>Aliskiren</td>
<td>Sarcoma</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Tigecycline</td>
<td>Bimatoprost</td>
<td>Autonomic reactions</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Telangiectases</td>
<td>Omeprazole</td>
<td>Dacarbazine</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Tolcapone</td>
<td>Pyrimethamine</td>
<td>Blood brain</td>
<td></td>
</tr>
</tbody>
</table>

Case Report

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

7 Anagrelide Azelaic acid Cerebral thrombosis
8 Atorvastatin Amlodipine Muscle inflammation
9 Aliskiren Tioconazole Breast inflammation
10 Estradiol Nadolol Endometriosis
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 $S$

Disease phenotypes

```
Disease 1: HPO-...
Disease 2: HPO-...
Disease 3: HPO-...
Disease 4: HPO-...
Disease 5: HPO-...
... 
Disease N: HPO-...
```

HPO network

Graph ML model

Disease subgraph predictions

- Lysosomal
- Glycosylation
- Carbohydrate
- Lipid
- Carbohydrate
- Disease N
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 and 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 $S = \{S_1, S_2, \ldots, S_n\}$, SubGNN specifies a neural message passing architecture $E_S$ that generates a $d_s$-dimensional subgraph representation $z_S \in \mathbb{R}^{d_s}$ for every subgraph $S \in S$. SubGNN uses the representations to learn a subgraph classifier $f : S \rightarrow \{1, 2, \ldots, C\}$ for subgraph labels $f(S) = \hat{y}_S$. 

![Diagram of Subgraph Neural Networks]

<table>
<thead>
<tr>
<th>SUB-GNN Channel</th>
<th>SUB-GNN Subchannel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal (I)</td>
<td>Border (B)</td>
</tr>
<tr>
<td>Position (P)</td>
<td>Distance between $S_i$’s components</td>
</tr>
<tr>
<td>Neighborhood (N)</td>
<td>Identity of $S_i$’s internal nodes</td>
</tr>
<tr>
<td>Structure (S)</td>
<td>Internal connectivity of $S_i$</td>
</tr>
<tr>
<td></td>
<td>Distance between $S_i$ and rest of $G$</td>
</tr>
<tr>
<td></td>
<td>Identity of $S_i$’s border nodes</td>
</tr>
<tr>
<td></td>
<td>Border connectivity of $S_i$</td>
</tr>
</tbody>
</table>
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 a node
  - **Link prediction**: Predict property of a node pair
  - **Graph prediction**: Predict property of an entire graph
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
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.

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

$$\text{Msg}_x^{A \rightarrow S} = \gamma_x(S^{(c)}, A_x) \cdot a_x$$

$$g_{x,c} = \text{AGG}_M(\{\text{Msg}_x^{A \rightarrow S^{(c)}} \quad \forall A_x \in A_x\}),$$

$$h_{x,c} \leftarrow \sigma(W_x \cdot [g_{x,c}; h_{x,c}]),$$
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 $z_x$ are concatenated to produce a final subgraph representation $z_S$.

Aggregate information from subgraphs

Aggregate information from neighbors

Subgraph Neural Networks, NeurIPS 2020
Setup: Subgraph Datasets

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

<table>
<thead>
<tr>
<th>Method</th>
<th>DENSITY</th>
<th>CUT RATIO</th>
<th>CORENESS</th>
<th>COMPONENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUB-GNN (Ours)</td>
<td>0.919 ± 0.016</td>
<td>0.629 ± 0.039</td>
<td>0.659 ± 0.092</td>
<td>0.958 ± 0.098</td>
</tr>
<tr>
<td>Node Averaging</td>
<td>0.429 ± 0.041</td>
<td>0.358 ± 0.055</td>
<td>0.530 ± 0.050</td>
<td>0.516 ± &lt;0.001</td>
</tr>
<tr>
<td>Meta Node (GIN)</td>
<td>0.442 ± 0.052</td>
<td>0.423 ± 0.057</td>
<td>0.611 ± 0.050</td>
<td>0.784 ± 0.046</td>
</tr>
<tr>
<td>Meta Node (GAT)</td>
<td>0.690 ± 0.021</td>
<td>0.284 ± 0.052</td>
<td>0.519 ± 0.076</td>
<td>0.935 ± &lt;0.001</td>
</tr>
<tr>
<td>Sub2W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sub2W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sub2W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graph classification</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**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

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

- 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.

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.

Drug discovery 3–6 years
Preclinical testing 3 years
Phase I 3 years
Phase II 2 years
Phase III 2 years
FDA approval 1–2 years

12–16 years, ~$1 billion to $2 billion

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.

Drug repositioning
~6 years, ~$300 million
What drug treats what disease?

Drugs

Diseases

Goal: Predict what diseases a new molecule might treat

“Treats” relationship

Unknown drug-disease relationship
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

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

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 $p(Y|G)$. The goal is to adapt to an unseen label set $Y_* \sim p(Y|G)$ by learning from tasks with other label sets $Y_i \sim p(Y|G)$, where $Y_i \cap Y_* = \emptyset$ for every label set $Y_i$. 

Graph Meta Learning via Local Subgraphs, NeurIPS 2020
G-Meta: Overview

Meta-Training

Training task 1
Support set
K=2
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
N=3
Query set
Node a
Node b
Node c
Label set 1

Training task 2
Support set
Node 7
Node 8
Node 9
Node 10
Node 11
Node 12

Meta-Testing

Test task 1
Support set
Node 13
Node 14
Node 15
Node 16
Node 17
Node 18

Query set
Node d
Node e
Node f
Label set 2

Label set 3
Node g
Node h
Node i
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
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 \( D_{GM}^t \) be a geometric mean of node degrees occurring on path \( t \). Let \( D_{GM}^{t*} = \min_t \{ D_{GM}^t \} \) and \( h_* = d(u, v) \). Consider the node influence \( I_{u,v} \) from \( v \) to \( u \). Then, \( I_{u,v} \leq C / (D_{GM}^{t*})^{h_*} \).

**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: \( v = \arg\max_w \{ I_{u,w} \mid w \in V \setminus V^u \} \). Let \( \bar{t} \) be a path between \( u \) and \( v \) and let \( D_{GM}^{\bar{t}} \) be a geometric mean of node degrees occurring on path \( \bar{t} \). Let \( D_{GM}^{\bar{t}*} = \min_{\bar{t}} \{ D_{GM}^{\bar{t}} \} \). The following holds: \( R_h(u) \leq C / (D_{GM}^{\bar{t}*})^{h+1} \).

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
COVID-19 Repurposing Dataset

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
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)

<table>
<thead>
<tr>
<th>CRank</th>
<th>Drug Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ritonavir</td>
</tr>
<tr>
<td>2</td>
<td>Isoniazid</td>
</tr>
<tr>
<td>3</td>
<td>Troleandomycin</td>
</tr>
<tr>
<td>4</td>
<td>Cilostazol</td>
</tr>
<tr>
<td>5</td>
<td>Chloroquine</td>
</tr>
<tr>
<td>6</td>
<td>Rifabutin</td>
</tr>
<tr>
<td>7</td>
<td>Flutamide</td>
</tr>
<tr>
<td>8</td>
<td>Dexamethasone</td>
</tr>
<tr>
<td>9</td>
<td>Rifaximin</td>
</tr>
<tr>
<td>10</td>
<td>Azelastine</td>
</tr>
<tr>
<td>11</td>
<td>Crizotinib</td>
</tr>
</tbody>
</table>

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

Ranked lists of drugs

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

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

58/77 drugs with positive experimental outcome are among top 750 ranked drugs.
Motivation: How can we leverage PPI networks of model organisms to complete human PPI network?
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.
Few-Shot Learning across Graphs

Brief vignette into cross-graph learning

Meta-Training

Training task 1
Support set

K=2
N=3

Query set

Label set 1

Meta-Testing

Training task 2
Support set

Test task 1
Support set

Query set

Label set 1

~ Graph 1

~ Graph 2

~ Graph 3
G-Meta: Results

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

- **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