Graphs Counterfactual Explainability: A Comprehensive Landscape

Tutorial @ AAAI 2024

M.A. Prado-Romero - B. Prenkaj - G. Stilo

Centro Nazionale di Ricerca in HPC,
Big Data and Quantum Computing

aiim

Mario A. Prado-Romero
PhD. Candidate
Gran Sasso Science Institute

Bardh Prenkaj
PostDoc
Sapienza University of Rome

Giovanni Stilo
Associate Prof.
University of L'Aquila

We are with the *Artificial Intelligence & Information Mining* (**aiim** - pronounced as i'm /aɪm/, and aim /eɪm/) a collective of *Individuals* (/aɪm/) who share a common *Interest* (/eɪm/) in Artificial Intelligence, Data Mining, and Machine Learning

Roadmap

- **Part I: Graphs fundamentals (20 mins)** **[STILO]**
  - Wide-spread adoption of GNNs in graph prediction problems
  - How do GNNs work?
  - Applications of different types of GNNs

- **Part II: eXplainable AI (25 mins)** **[PRENKAJ]**
  - Issues of black-box models and the importance of interpretability
  - What is a factual explanation?
  - (Briefly) Revisiting GNNExplaner and GraphLIME

- **Part III: Counterfactual Explanations in Graphs (60 mins)** **[PRADO]**
  - What is a graph counterfactual explanation (GCE)?
  - GCE taxonomy description and method classification
    - Model-level explainers
    - Instance-level explainers
      - Search-based
      - Heuristic-based
      - Learning-based **[PRENKAJ]**
  - Benchmarking datasets and evaluation metrics (pro et contra) **[PRENKAJ]**

based on ACM survey

https://dl.acm.org/doi/abs/10.1145/3618105

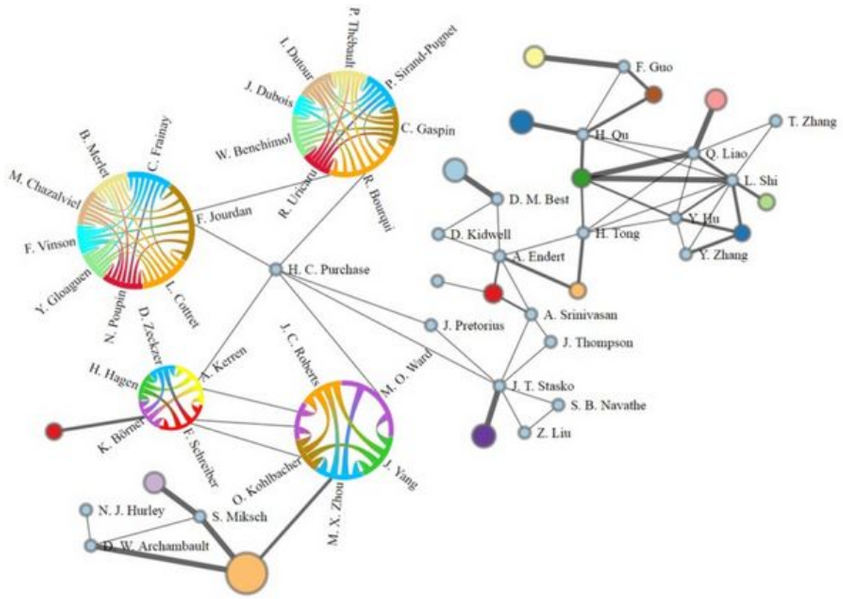
Part I

Graphs fundamentals and their Neural Network

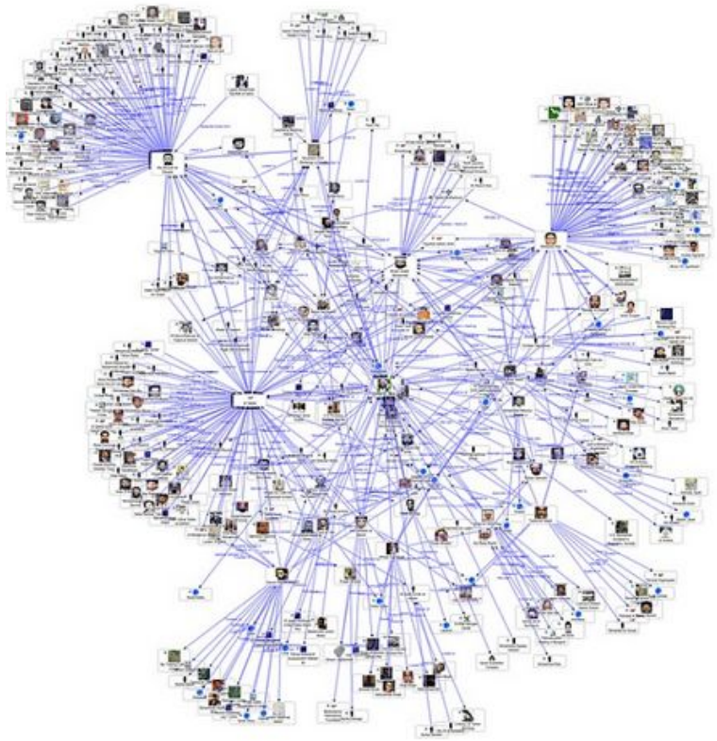
by Giovanni Stilo

images based on Understanding Deep Learning - book by Simon J.D. Prince

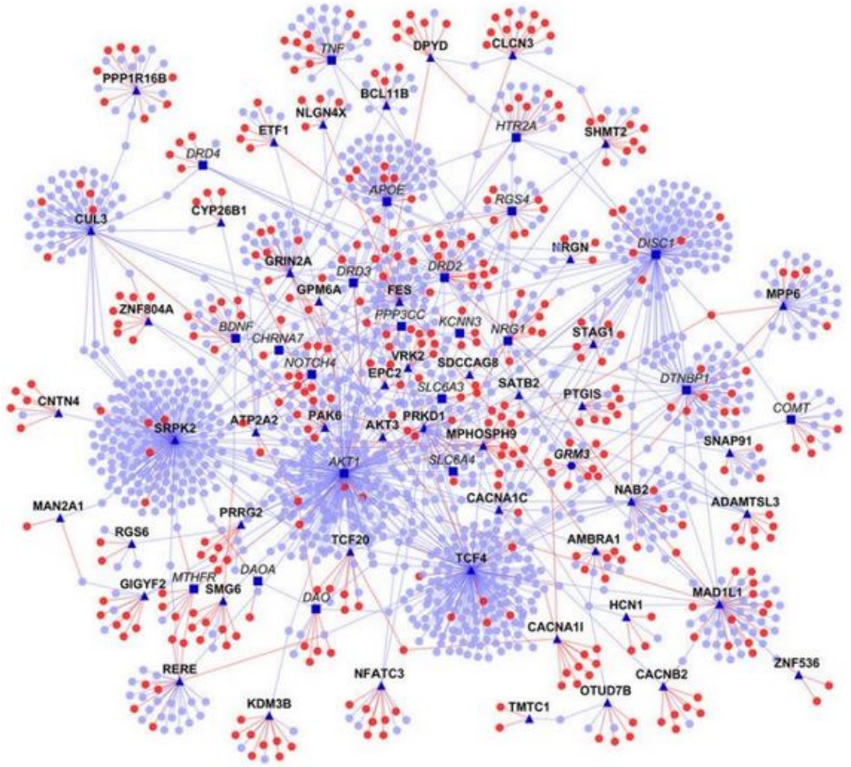
What is a graph?



Co-Autorship Networks



Social Networks



Protein interactions

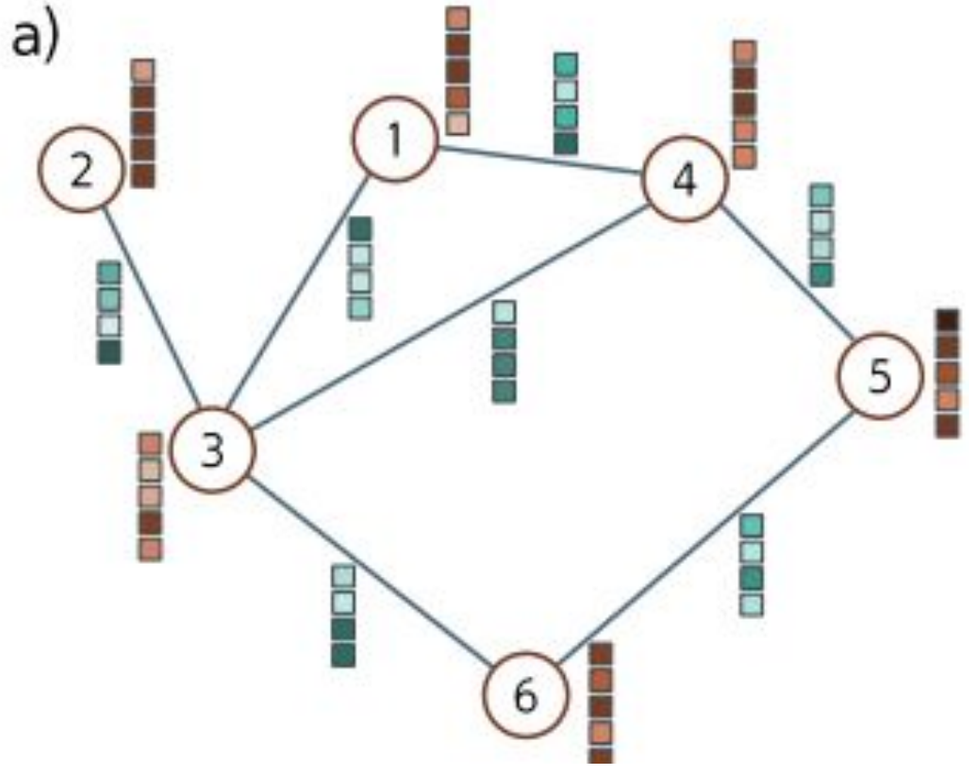
Challenges with graphs

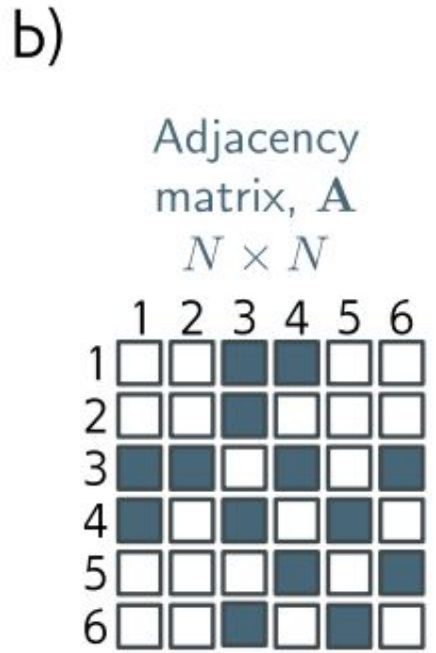
There are **three main challenges** associated with processing graphs:

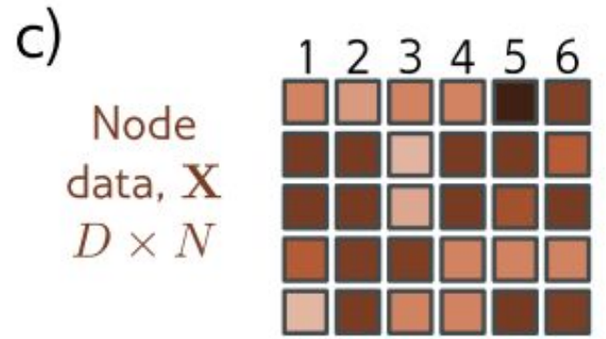
- **Variable topology:** hard to design an NN that is sufficiently expressive and can cope with this variation
- **Huge graphs:** we can have millions of nodes and billions of edges (see Twitter)
- **Single monolithic graph:** the usual protocol of training with many data examples and testing with new data is not always appropriate or possible

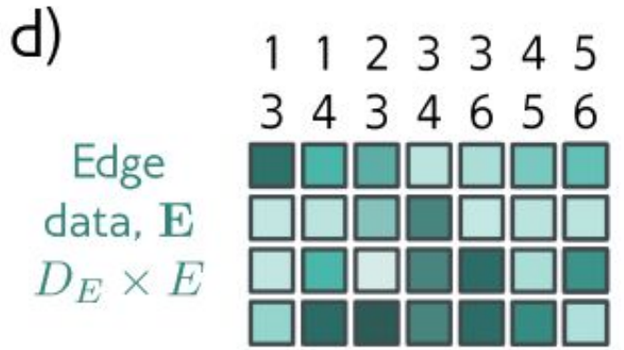
Graph Representation

adjacency matrix, A is $N \times N$;
node embeddings, X is $D \times N$;
edges embeddings, E is $D_E \times N$



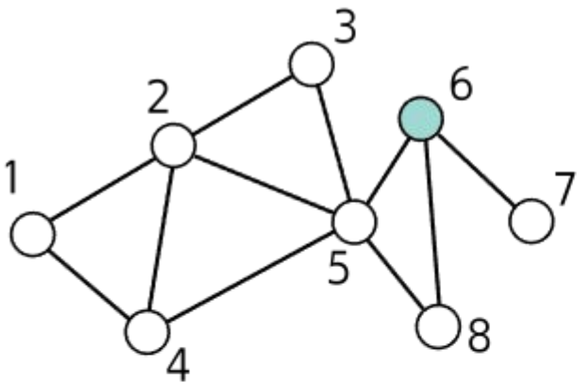






Adjacency Matrix Properties

a)



b)

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

c)

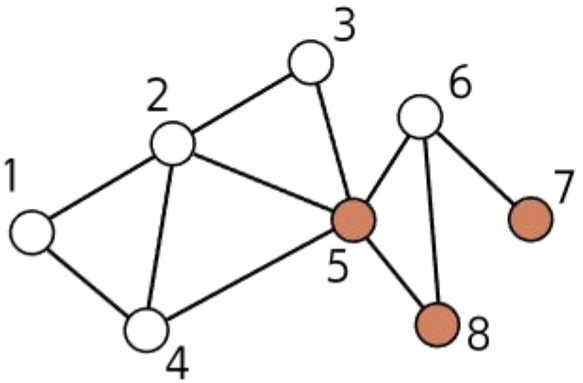
$$\mathbf{A}^2 = \begin{bmatrix} 2 & 1 & 1 & 1 & 2 & 0 & 0 & 0 \\ 1 & 4 & 1 & 2 & 2 & 1 & 0 & 1 \\ 1 & 1 & 2 & 2 & 1 & 1 & 0 & 1 \\ 1 & 2 & 2 & 3 & 1 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 5 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 3 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 2 \end{bmatrix}$$

d)

$$\mathbf{x} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

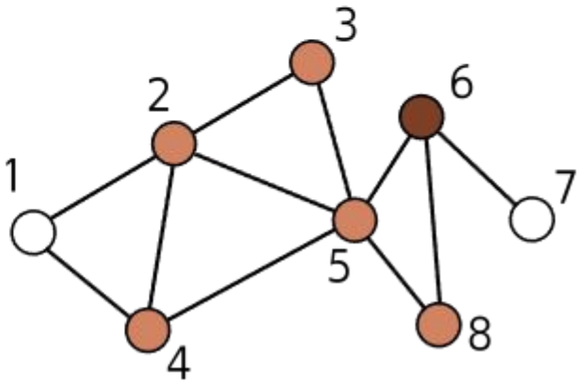
e)

$$\mathbf{Ax} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$



f)

$$\mathbf{A}^2\mathbf{x} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 3 \\ 0 \\ 1 \end{bmatrix}$$

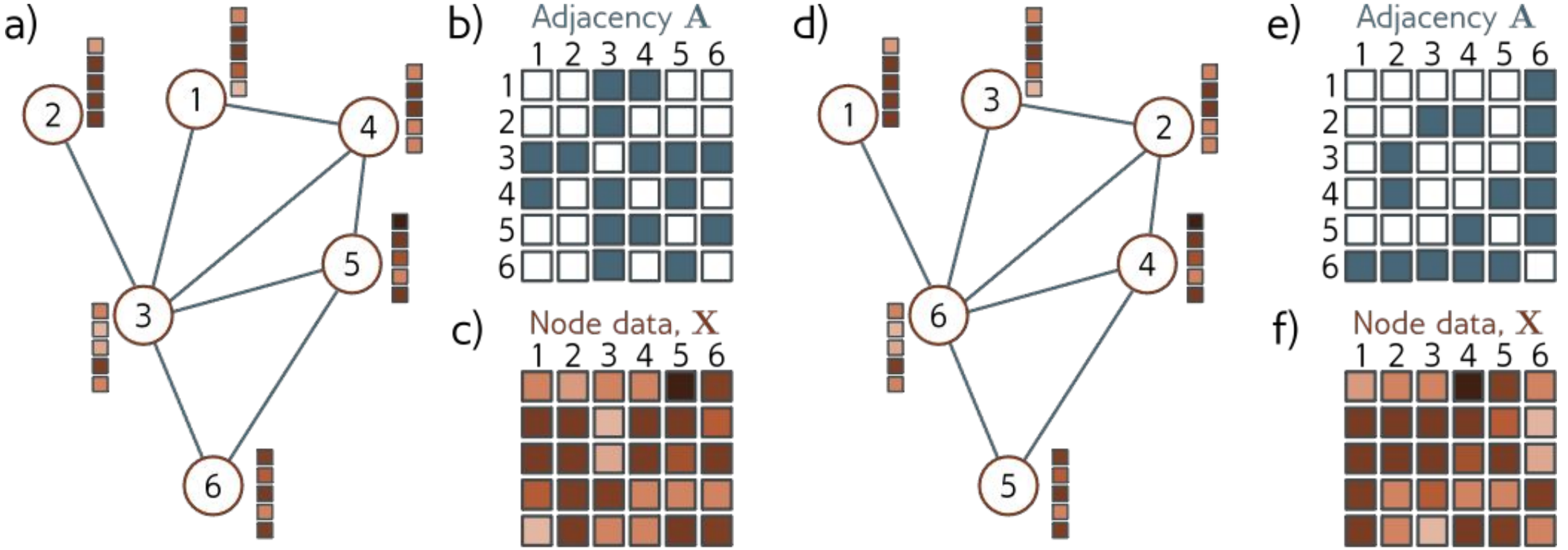


A^l contains the number of unique walks of length l from node m to node n

Nodes Permutation

Node indexing in graphs is arbitrary

Permuting the node **indices** results in a **permutation** of the **columns** of the node data matrix $\boldsymbol{X}$ and a permutation of both the **rows** and **columns** of the adjacency matrix $\boldsymbol{A}$.



$$\mathbf{X'=XP}, \quad \mathbf{A'=P^T AP}$$

Graph Learning

We want to learn a (dense) representation ***H*** of the graph usable for different downstream tasks

A **graph neural network** is a model that takes:

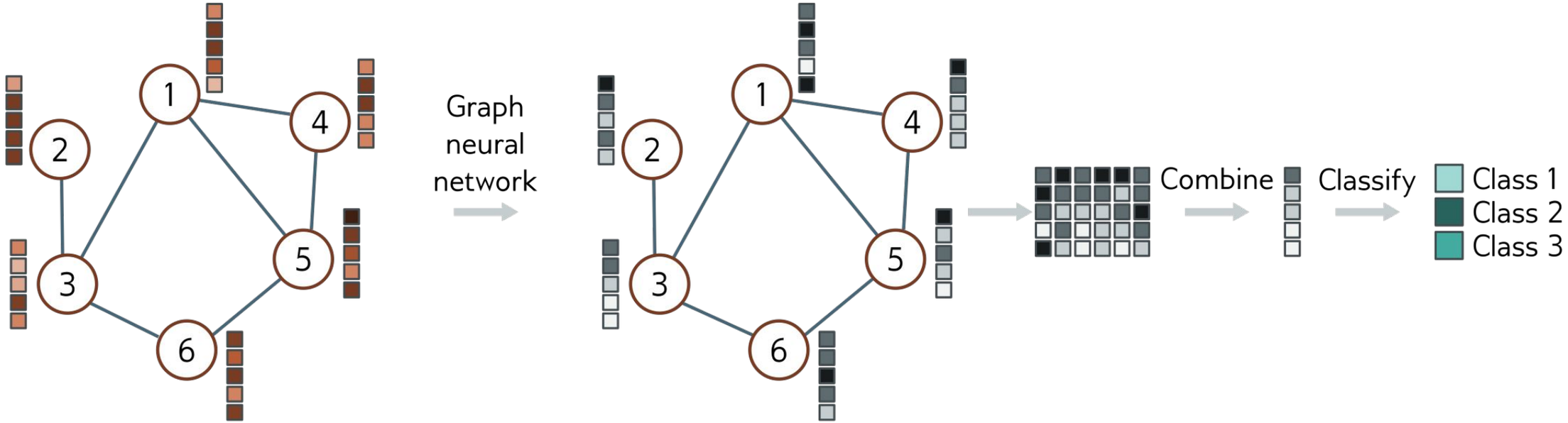
- the node embeddings ***X*** and the adjacency matrix ***A*** as inputs and passes them through a series of *k* layers.
- the node embeddings are updated at each layer to create intermediate "hidden" representations ***h*** before finally computing output embeddings h_K.

Graph Classification Tasks

For example, we might want to **predict:**

- the **temperature** at which a **molecule** becomes **liquid** (a regression task);
- whether a **molecule** is **poisonous** to human beings or not (a classification task).

*For graph-level tasks, the output **node embeddings are combined** (e.g., by averaging), and the resulting vector is **mapped** via a linear transformation or neural network to a **fixed-size vector**.*

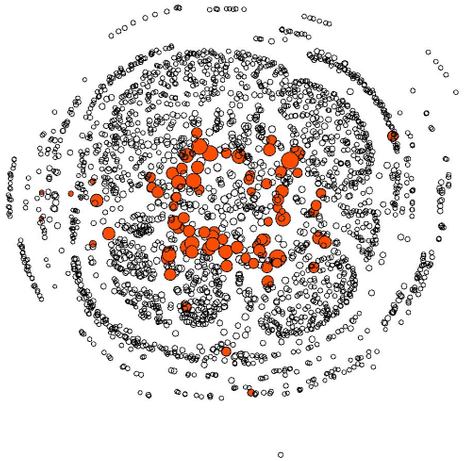


$$Pr(y = 1 \mid \boldsymbol{X}, \boldsymbol{A}) = sig\left(\beta_k + \boldsymbol{\omega}_k \boldsymbol{H}_k \frac{\mathbf{1}}{N}\right)$$

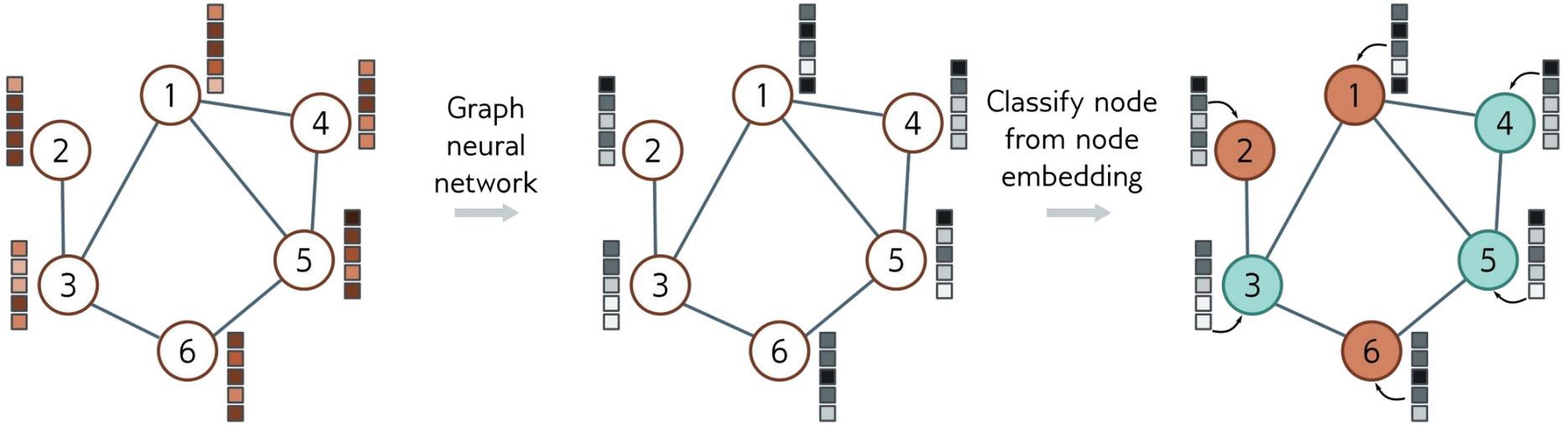
Node Classification Tasks

For example, in an *PPI* network we might want to **predict:**

- the probability that a **given node** might be **attacked/being part** of a certain **disease** (classification) as it is shown for COVID19 (**red**) - PPI on the right.



*The network assigns **one or more label** (classification) or values (regression) to **each node** of the graph, **using** both the **graph structure** and learned **node embeddings**.*



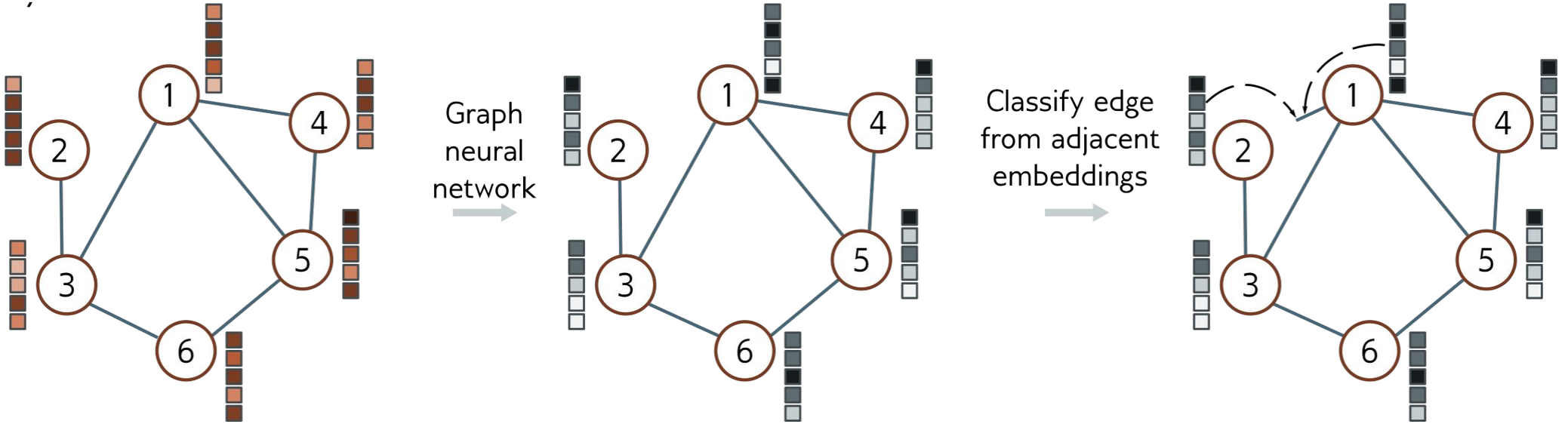
$$Pr\left(y^{(n)} = 1 \mid \boldsymbol{X}, \boldsymbol{A}\right) = sig\left(\beta_k^{(n)} + \boldsymbol{\omega}_k^{(n)} \boldsymbol{H}_k^{(n)}\right)$$

Edge Classification Tasks

For example, in the **social network** setting, the network might **predict** whether:

- two people know each other and suggest that they connect if that is the case.

*The network assigns **one or more label** (classification) or values (regression) to **each edges** of the graph, using both the graph **structure** and learned **node embeddings**.*



$$Pr\left(\boldsymbol{y}^{(mn)} = 1 \mid \boldsymbol{X}, \boldsymbol{A}\right) = sig\left(\boldsymbol{H}_k^{(m)T} \cdot \boldsymbol{H}_k^{(n)}\right)$$

GNNs

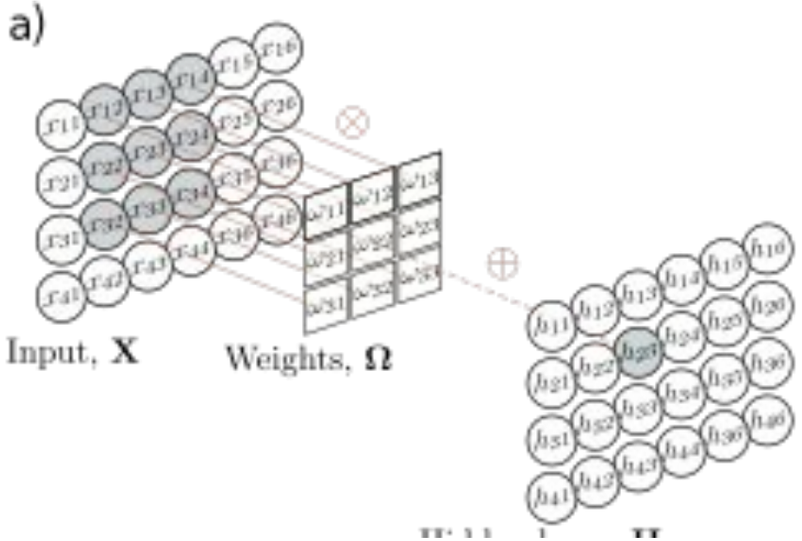
Spatial-based convolutional graph neural networks **(GCN)** are convolutional network that update each node embeddings by **aggregating information from nearby nodes** using the original g**raph structure**.

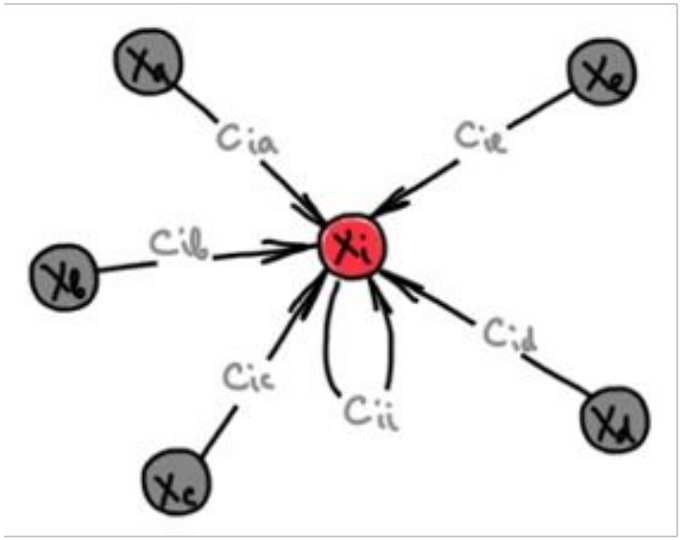
Each **layer** of the GCN is a function *F[·]* with parameters **Φ** that **takes** the node **embeddings** and **adjacency matrix** and outputs new node embeddings:

$$
\begin{aligned}
\boldsymbol{H}_1 &= \mathrm{F}[\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{\phi}_0] \\
\boldsymbol{H}_2 &= \mathrm{F}[\boldsymbol{H}_1, \boldsymbol{A}, \boldsymbol{\phi}_1] \\
\boldsymbol{H}_3 &= \mathrm{F}[\boldsymbol{H}_2, \boldsymbol{A}, \boldsymbol{\phi}_2] \\
\vdots &= \quad \vdots \\
\boldsymbol{H}_{\mathrm{K}} &= \mathrm{F}[\boldsymbol{H}_{\mathrm{K}-1}, \boldsymbol{A}, \boldsymbol{\phi}_{K-1}]
\end{aligned}
$$

Parameter Sharing

- Likewise CNN, we want the **same parameters** at every **node**: **reducing** the number of **parameters** and **sharing** what the network **learns** at each node **across** the **entire** graph.





  - **Each neighbor sends a message** to the variable of interest, which **aggregates** these messages to **form** the **update**.

In **images**, the neighbors were **pixels** from a **fixed-size square region** around the current position, so the **spatial relationships** at each position **are the same**.

  - In a **graph**, each node may have a **different number of neighbors**, and there are **no consistent relationships**.

GCN (1)

each node at layer *k*, we aggregate information from neighboring nodes by e.g. summing their node embeddings:

$$\boldsymbol{agg}[n,k] = \sum_{m \in \mathrm{ne}[n]} \boldsymbol{H}_k^{(m)}$$

linear transformation Ω to the embedding H of the current node and to his aggregated value, we add a bias term *β*, and pass the result through a nonlinear activation function *a[·]*, which is applied independently to every member of its vector argument:

$$\boldsymbol{H}_{k+1}^{(n)} = a\left[\boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \cdot \boldsymbol{H}_k^{(n)} + \boldsymbol{\Omega}_k \cdot \boldsymbol{agg}[n,k]\right]$$

the n^{th} column of ***A*** contains ones at the positions of neighbors. If **post-multiply** the **embeddings** by **A** the n^{th} column is **agg**[n,k]:

$$\boldsymbol{H}_{k+1} = a[\boldsymbol{\beta}_k \mathbf{1}^T + \boldsymbol{\Omega}_k \boldsymbol{H}_k + \boldsymbol{\Omega}_k \boldsymbol{H}_k \boldsymbol{A}] = a[\boldsymbol{\beta}_k \mathbf{1}^T + \boldsymbol{\Omega}_k \boldsymbol{H}_k (\boldsymbol{A} + \boldsymbol{I})]$$

GCN (2)

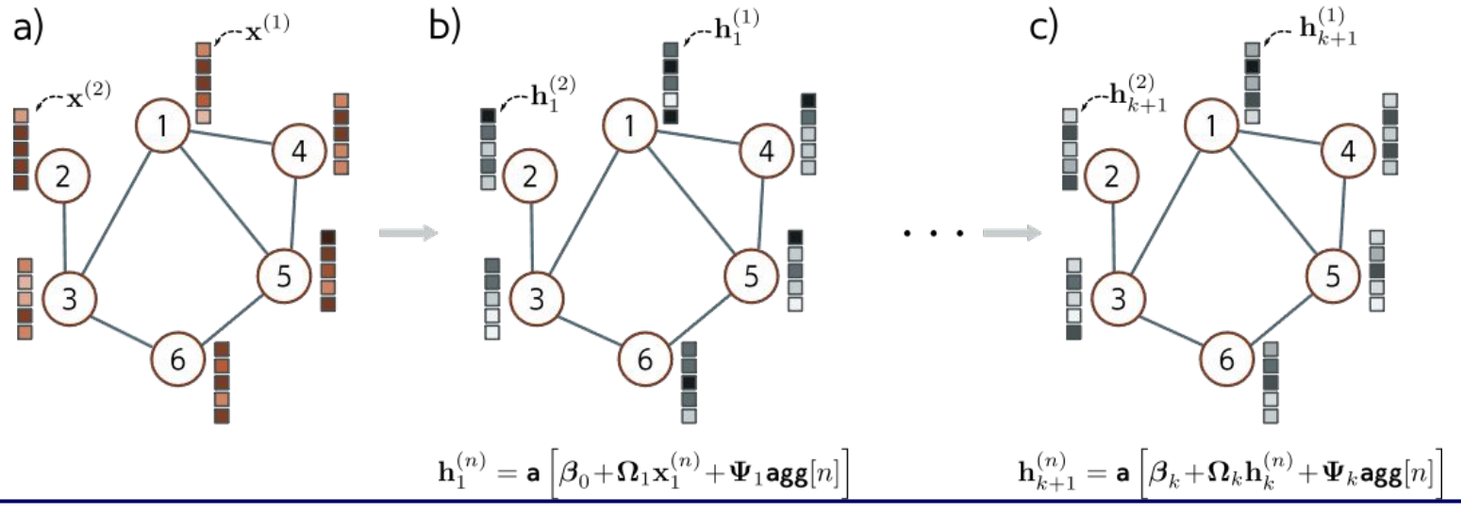
$$\boldsymbol{H}_{k+1} = a[\boldsymbol{\beta}_k \mathbf{1}^T + \boldsymbol{\Omega}_k \boldsymbol{H}_k (\boldsymbol{A} + \boldsymbol{I})]$$

This layer **satisfies** the **design** considerations:

- it **is equivariant** to **permutations** of the node **indices**

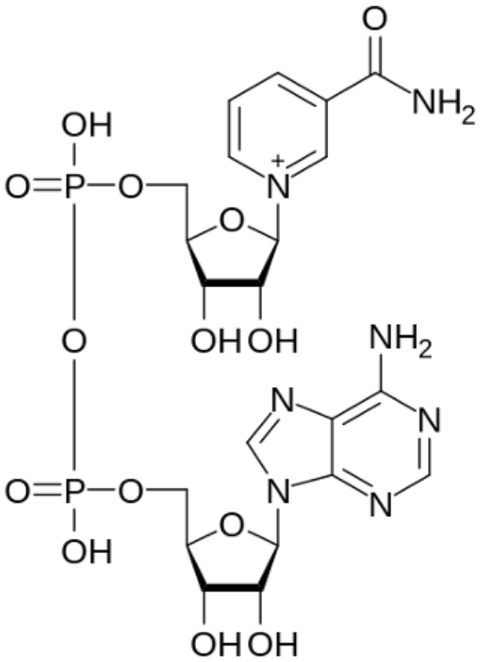
$$\boldsymbol{H}_{k+1}\boldsymbol{P} = \mathrm{F}[\boldsymbol{H}_k\boldsymbol{P}, \boldsymbol{P}^T\boldsymbol{A}\boldsymbol{P}, \boldsymbol{\phi}_k] = a[\boldsymbol{\beta}_k \mathbf{1}^T\boldsymbol{P} + \boldsymbol{\Omega}_k \boldsymbol{H}_k\boldsymbol{P}(\boldsymbol{P}^T\boldsymbol{A}\boldsymbol{P} + \boldsymbol{I})]$$

- can **cope** with **any** number of **neighbors** due to the **agg**[n,k] function;
- **exploits** the **graph structure** to provide a relational inductive bias,
- and **shares parameters** throughout the graph i.e. **Ω**.



Graph Classification (Rev.)

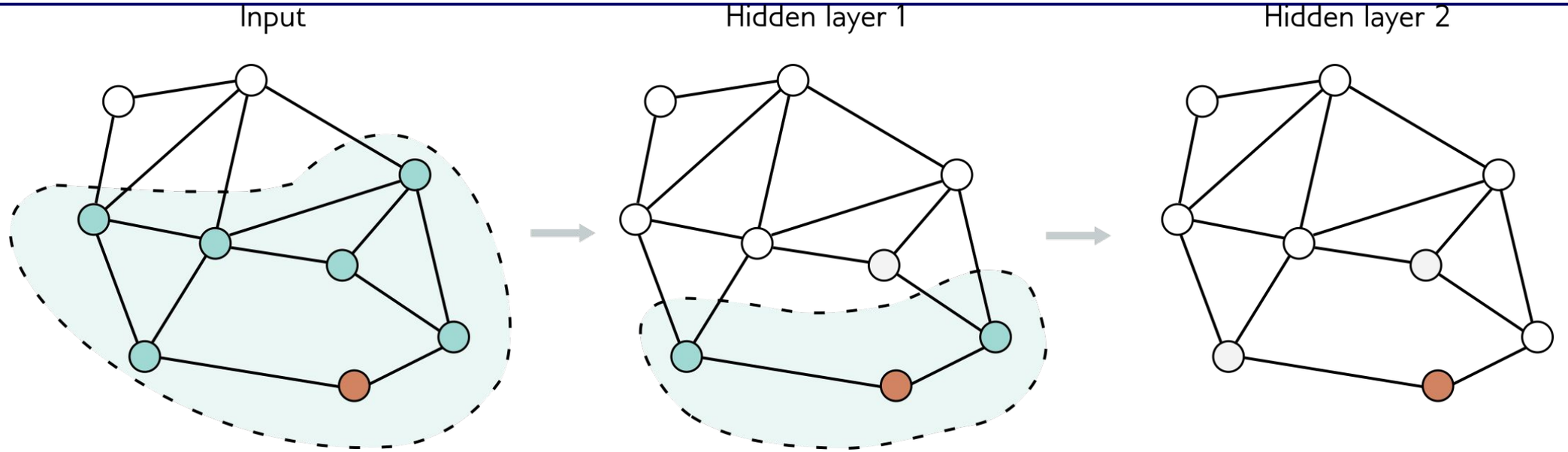
- We want a **neural network** f[**X**,**A**,Φ] that **classifies** (predicts) **molecules** as toxic or harmless.
- The **adjacency matrix** A ∈ R^(N×N) derives from the molecular structure.
- The columns of the **node embedding** matrix are **one-hot vectors** indicating which of the 118 **elements of the periodic** table are present.



$$\boldsymbol{H}_1 = a[\boldsymbol{\beta}_0 \mathbf{1}^T + \boldsymbol{\Omega}_0 \boldsymbol{X}\ (\boldsymbol{A} + \boldsymbol{I})]$$
$$\boldsymbol{H}_2 = a[\boldsymbol{\beta}_1 \mathbf{1}^T + \boldsymbol{\Omega}_1 \boldsymbol{H}_1 (\boldsymbol{A} + \boldsymbol{I})]$$
$$\vdots = \quad \vdots$$
$$\boldsymbol{H}_k = a[\boldsymbol{\beta}_{k-1} \mathbf{1}^T + \boldsymbol{\Omega}_{k-1} \boldsymbol{H}_{k-1} (\boldsymbol{A} + \boldsymbol{I})]$$
$$\mathrm{f}[\boldsymbol{X}, \boldsymbol{A}, \Phi] = sig\left[\beta_k + \boldsymbol{\omega}_k \boldsymbol{H}_k \frac{\mathbf{1}}{N}\right]$$

$$\boldsymbol{X}(\boldsymbol{A} + \boldsymbol{I})^k, \text{e.g } \boldsymbol{H}_3 \colon \boldsymbol{X}(\boldsymbol{A} + \boldsymbol{I})^3 = \boldsymbol{X}(\boldsymbol{A}^3 + 3\boldsymbol{A}^2 + 3\boldsymbol{A} + \boldsymbol{I}^3)$$

GNNs (briefly)



$$H^{\ell} \text{ depends on } X(A+I)^{\ell}$$

graph expansion problem

If there are **many layers** and the graph is **densely connected:**

every input node may be in the receptive field of every output.

In general we want that k << diam(G)

Part II

eXplainable Artificial Intelligence

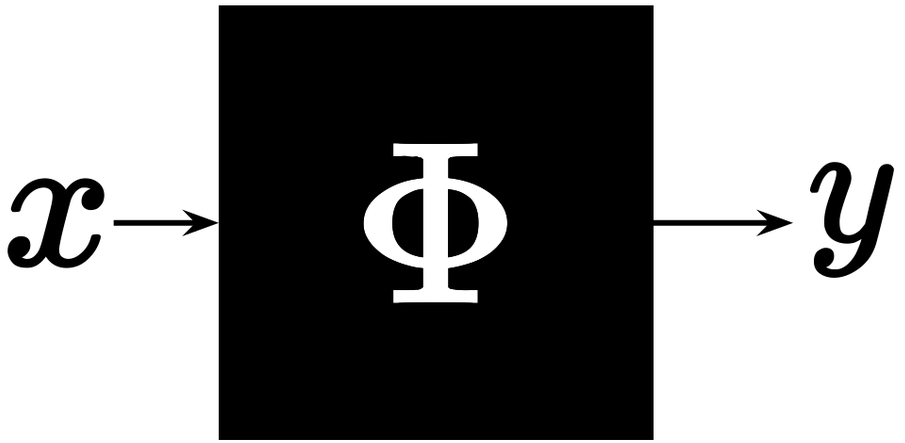
by Bardh Prenkaj

first part of the slides based on: CSEP 590B: Explainable AI from University of Washington

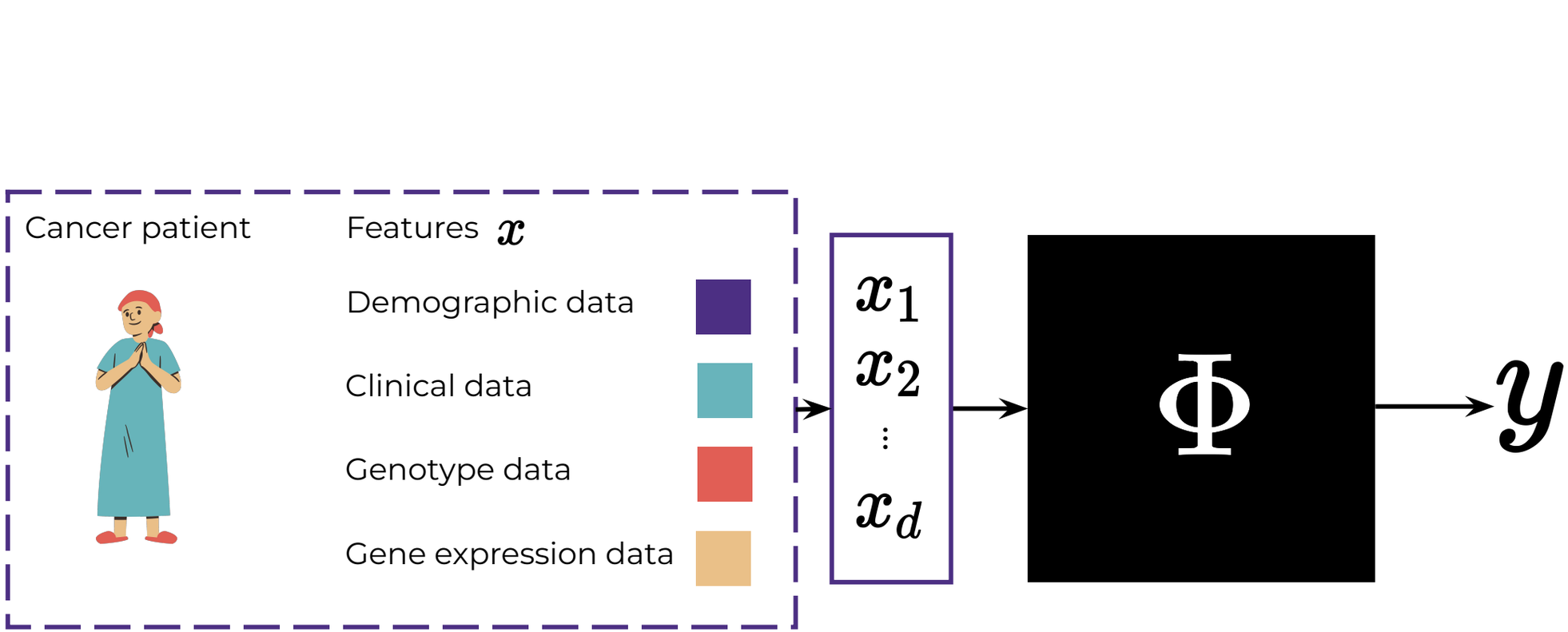
What's going on today in ML?

Lack of transparency

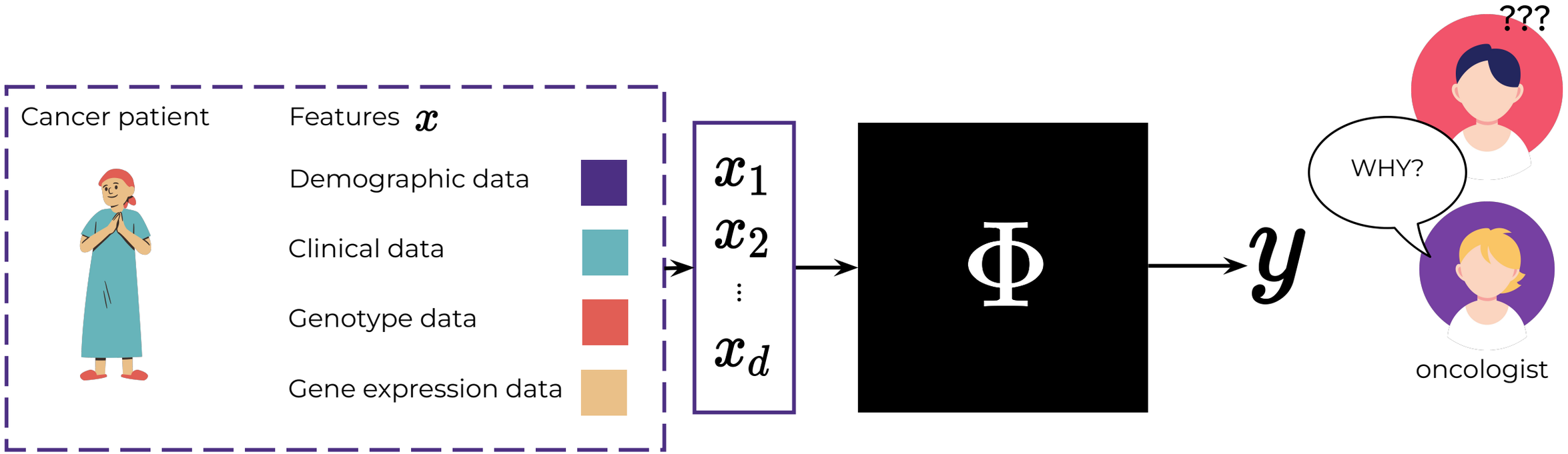
- Identify key factors in underlying processes
- Generate scientific hypotheses



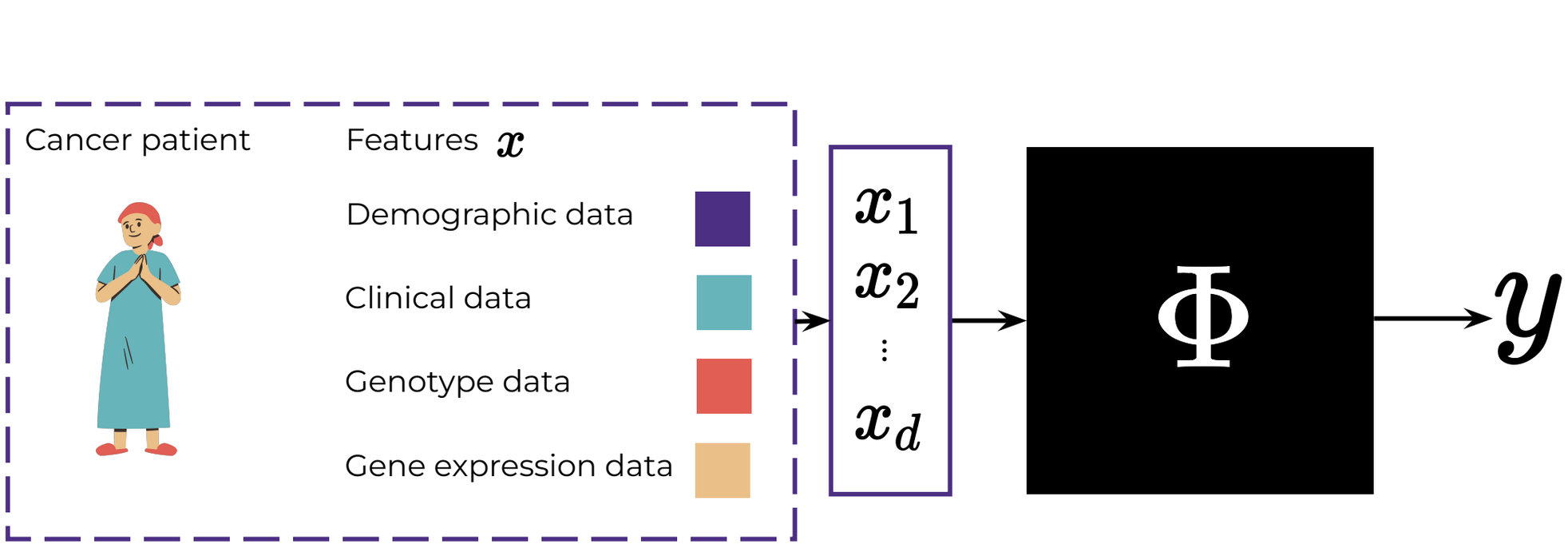
WHY ACCURATE PREDICTIONS ARE IMPORTANT?



WHY ACCURATE PREDICTIONS ARE IMPORTANT?



WHY ACCURATE PREDICTIONS ARE IMPORTANT?



Explainability

- Which features contributed to a certain prediction and how?
- How to learn or select features that are most interpretable or informative?
- How to make biological or clinical sense of a black-box model?

Types of Explainability

Feature importance explanations

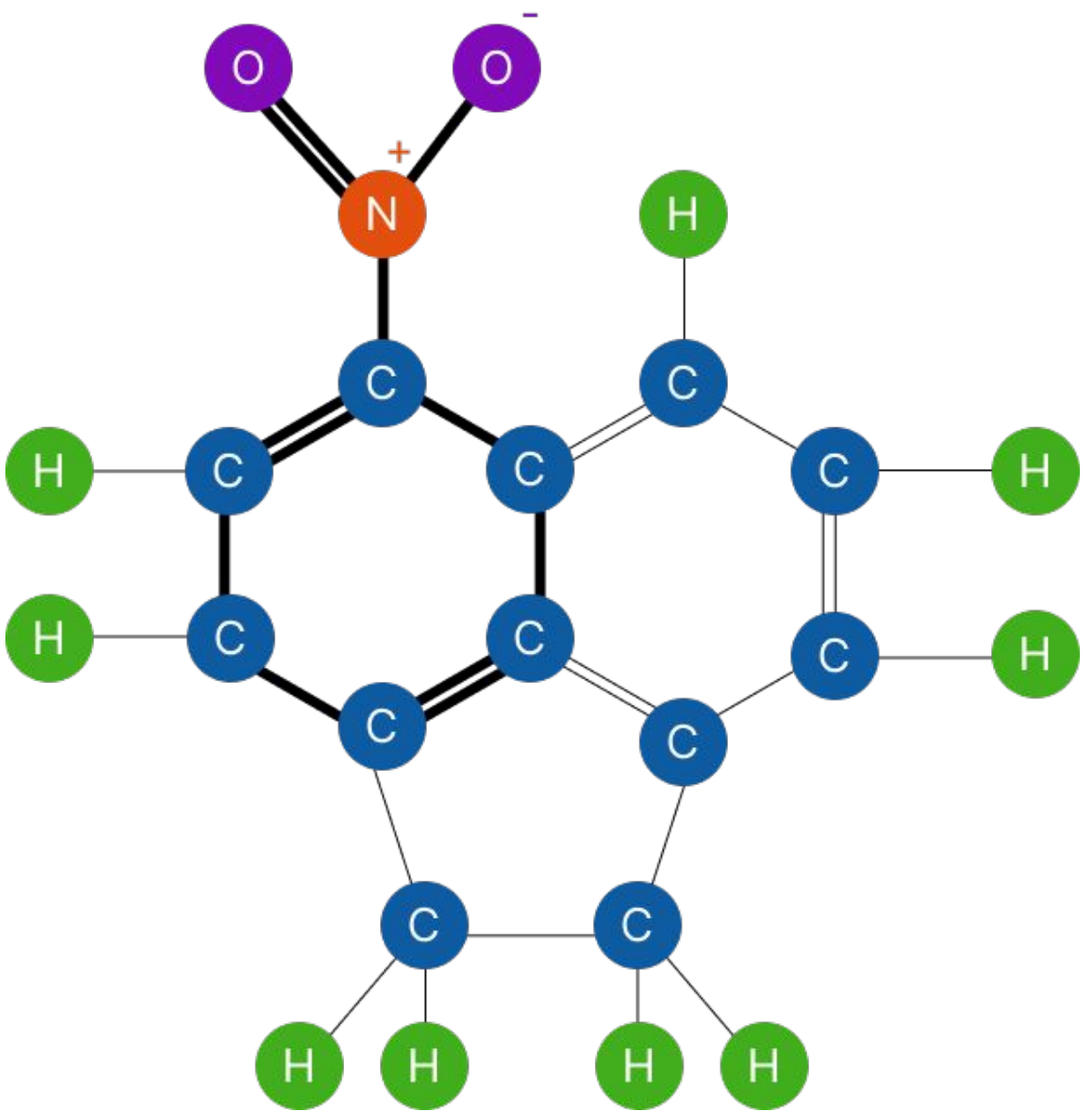
- Removal-based explanations
- Shapley values
- Propagation-based explanations

Some sort of factual explanations

Inherently interpretable models

Counterfactual explanations

Factual explanations on Graphs



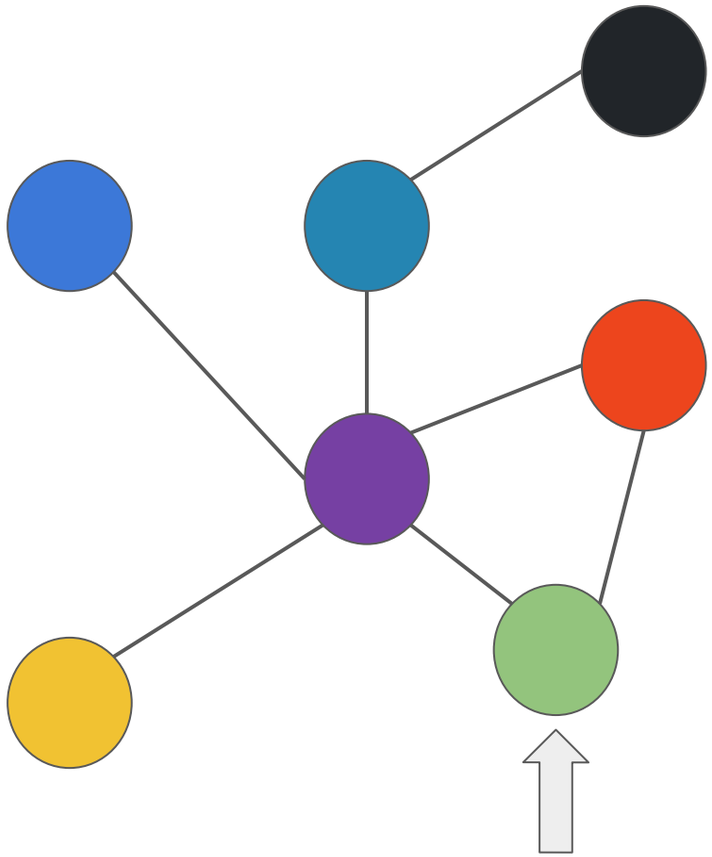
Factual explanations on Graphs

- Find those edges whose subgraph induced on them has the same label as the whole graph (**desideratum #1**)
- This subgraph should be minimal (**desideratum #2**)
- When you remove this subgraph, the remainder should have the opposite class (**corollary #1**) – *this gives sprout to factual-based counterfactual explainers*

Revisiting GNNExplainer[1] and GraphLIME[2]

based on (1) Ying et al. “GNNExplainer: Generating Explanations for Graph Neural Networks”, NeurIPS 2019
& (2) Huang et al. “GraphLIME:Local Interpretable Model Explanations for Graph Neural Networks”, TKDE 2023

GNNEXPLAINER (INTRO)



node classification

GNNExplainer: Generating Explanations for Graph Neural Networks

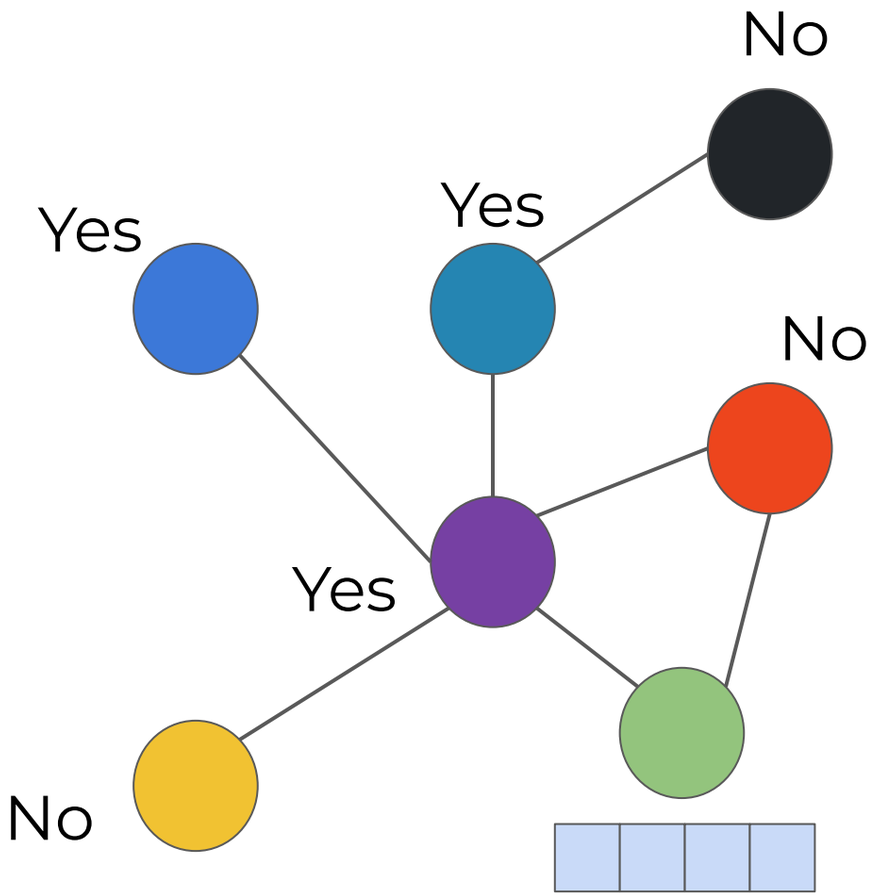
Rex Ying[†] Dylan Bourgeois[†,‡] Jiaxuan You[†] Marinka Zitnik[†] Jure Leskovec[†]

[†]Department of Computer Science, Stanford University
[‡]Robust.AI
{rexying, dtsbourg, jiaxuan, marinka, jure}@cs.stanford.edu

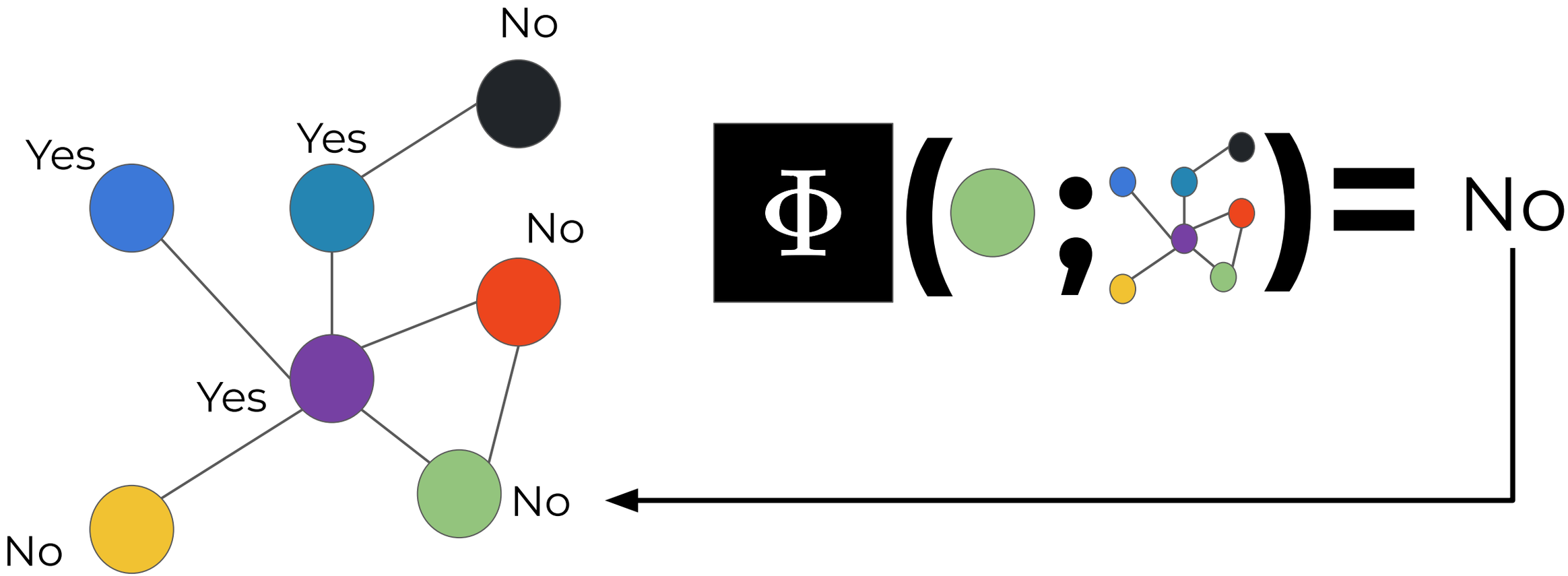
Abstract

Graph Neural Networks (GNNs) are a powerful tool for machine learning on graphs. GNNs combine node feature information with the graph structure by recursively passing neural messages along edges of the input graph. However, incorporating both graph structure and feature information leads to complex models and explaining predictions made by GNNs remains unsolved. Here we propose GNNEXPLAINER, the first general, model-agnostic approach for providing interpretable explanations for predictions of any GNN-based model on any graph-based machine learning task. Given an instance, GNNEXPLAINER identifies a compact subgraph structure and a small subset of node features that have a crucial role in GNN's prediction. Further, GNNEXPLAINER can generate consistent and concise explanations for an entire class of instances. We formulate GNNEXPLAINER as an optimization task that maximizes the mutual information between a GNN's prediction and distribution of possible subgraph structures. Experiments on synthetic and real-world graphs show that our approach can identify important graph structures as well as node features, and outperforms alternative baseline approaches by up to 43.0% in explanation accuracy. GNNEXPLAINER provides a variety of benefits, from the ability to visualize semantically relevant structures to interpretability, to giving insights into errors of faulty GNNs.

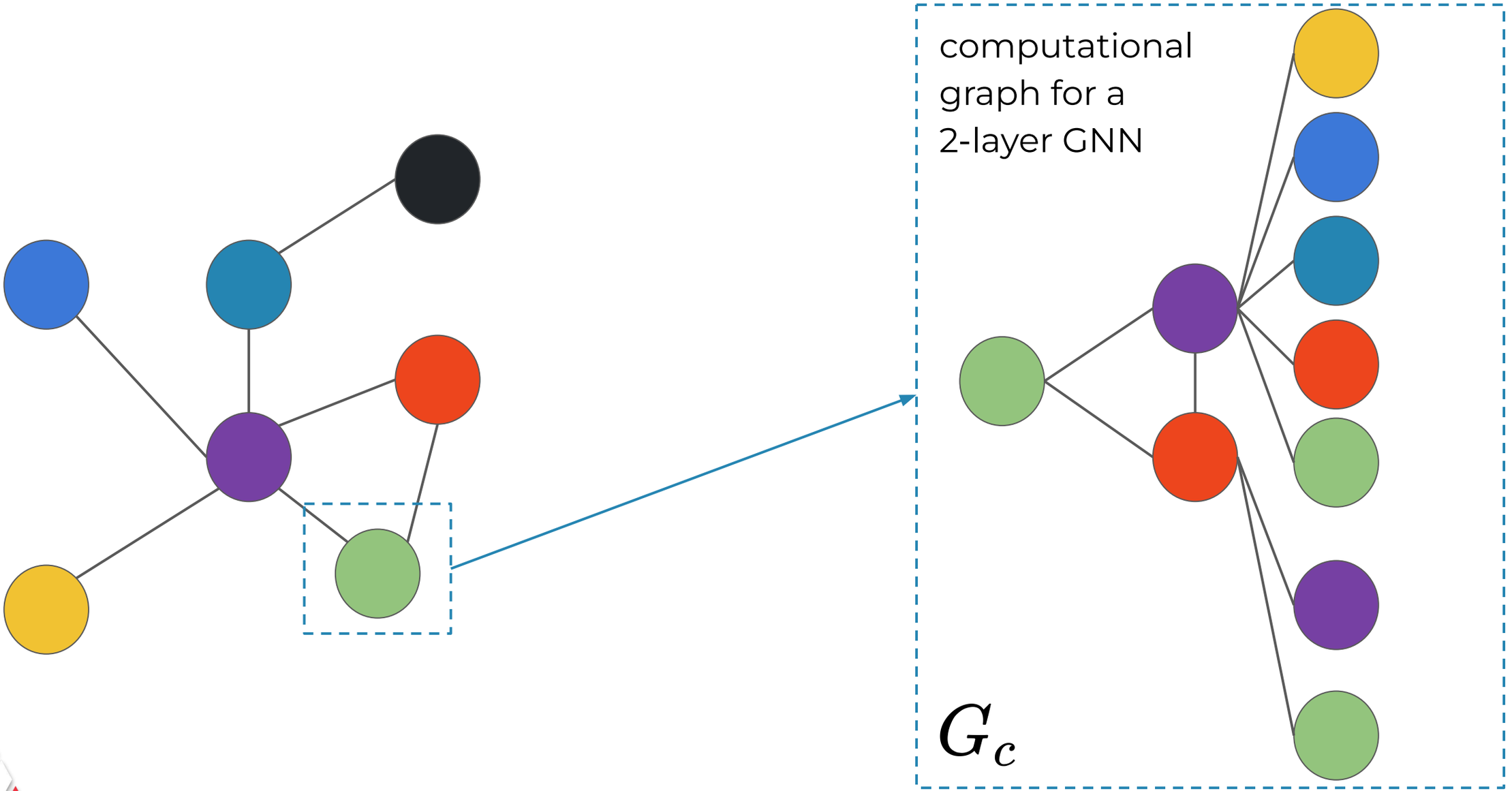
GNNExplainer (Intro)



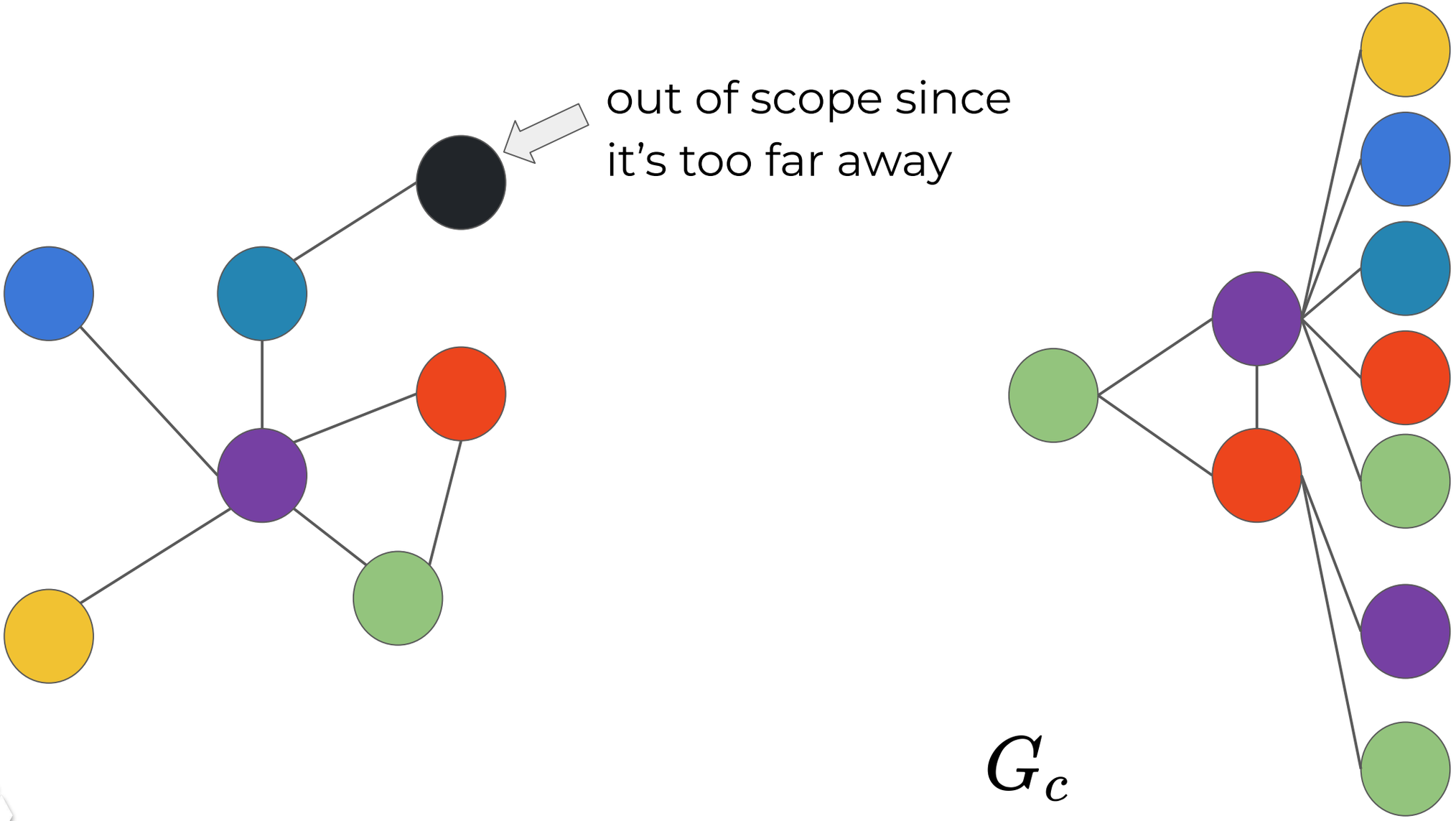
GNNExplainer (Intro)



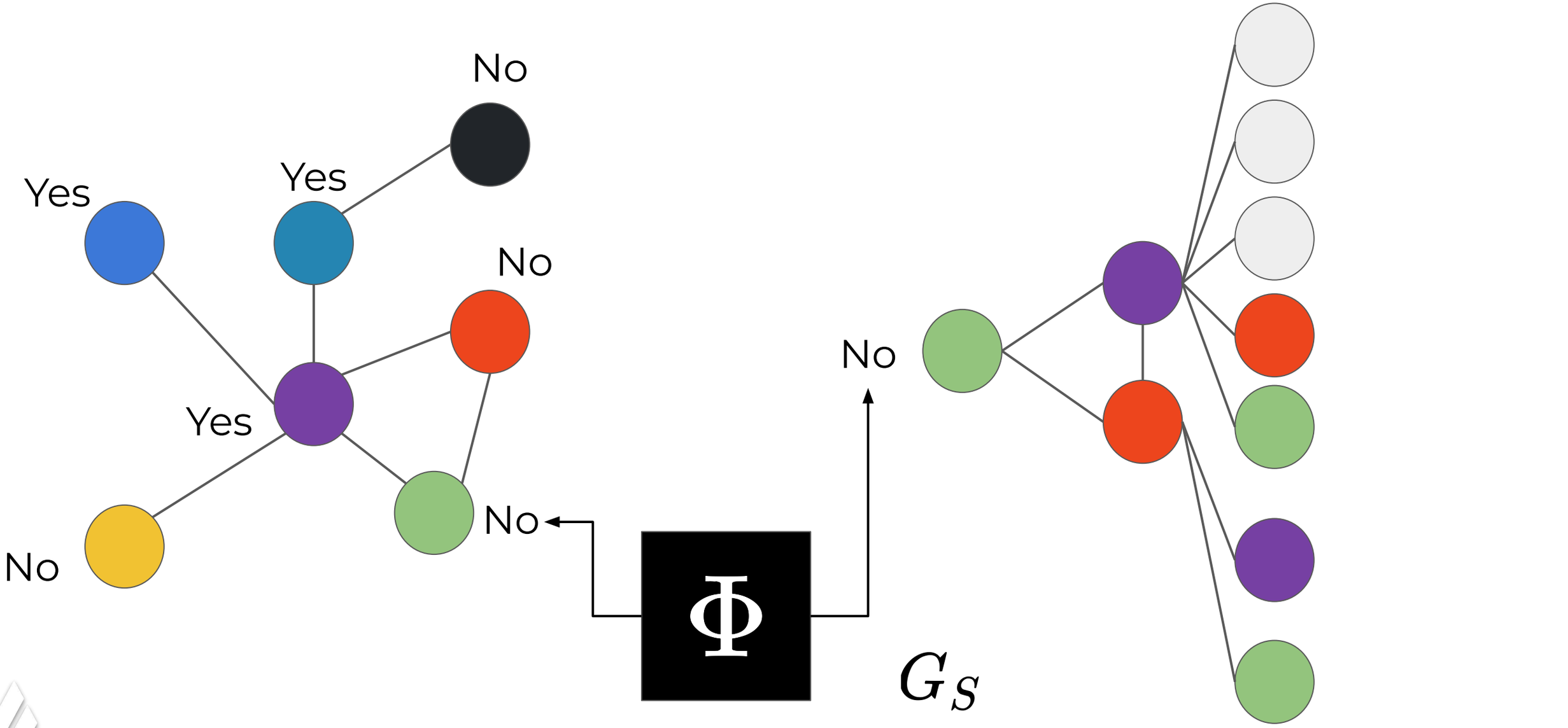
GNNExplainer (Comp. Graph)



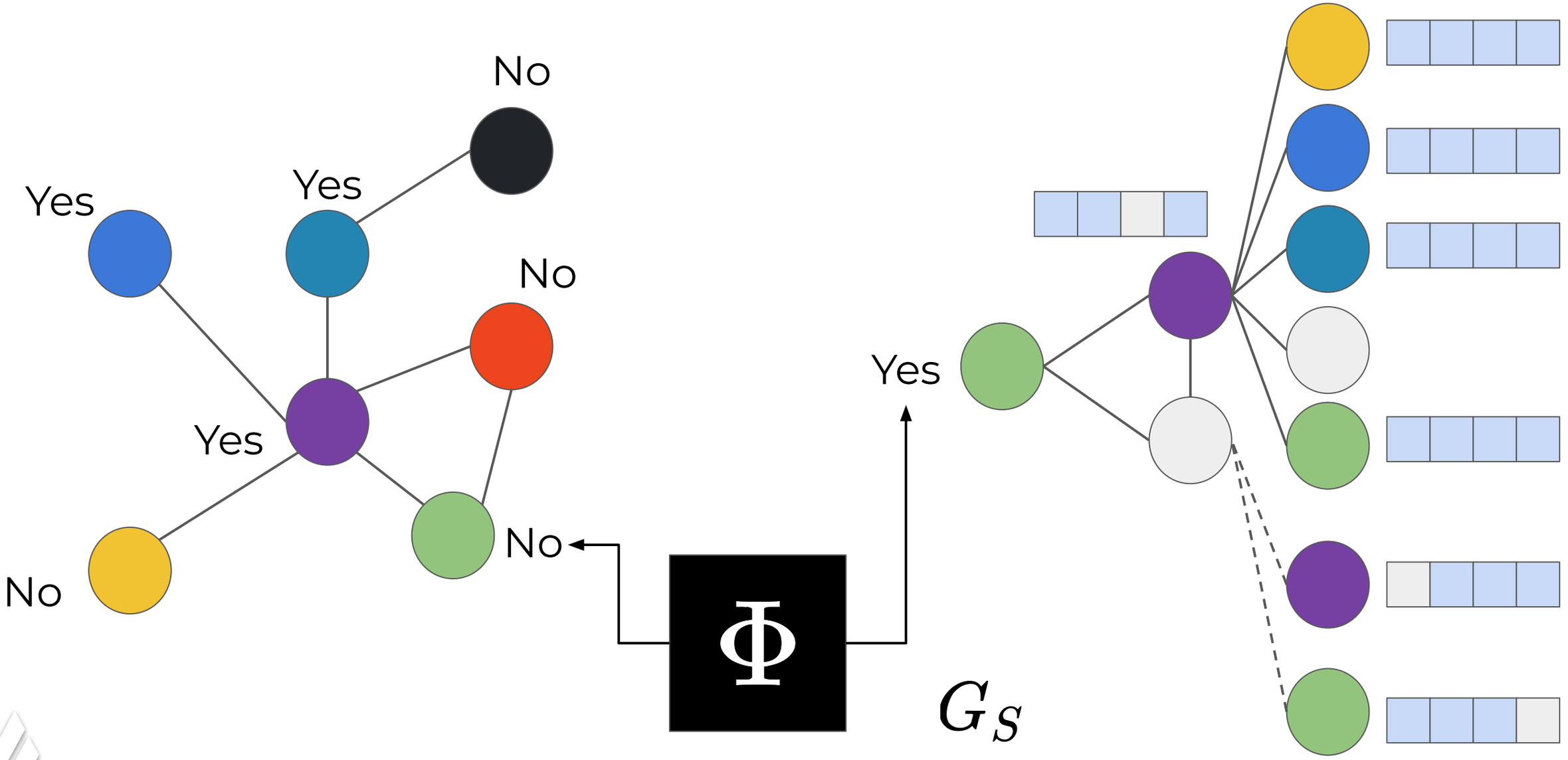
GNNExplainer (Comp. Graph)



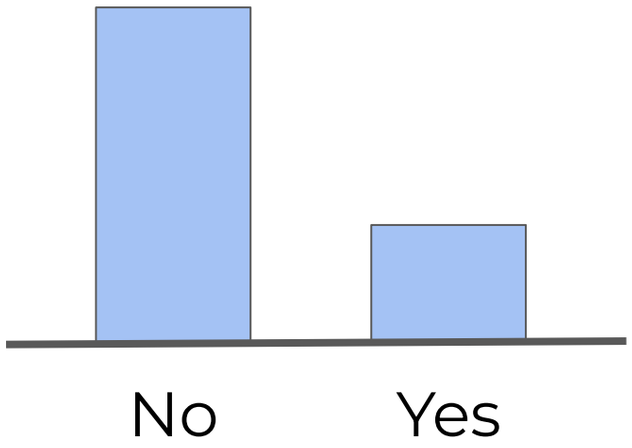
GNNExplainer (Remove nodes)

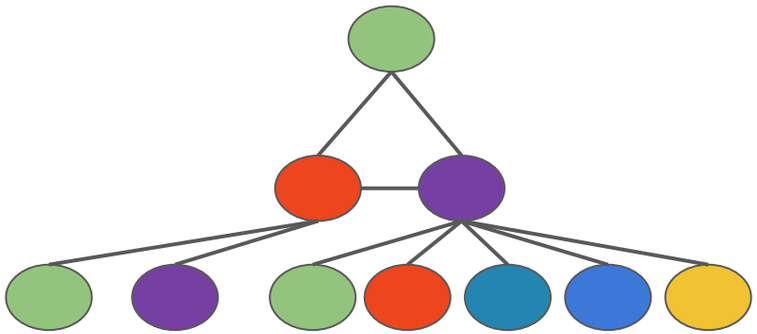


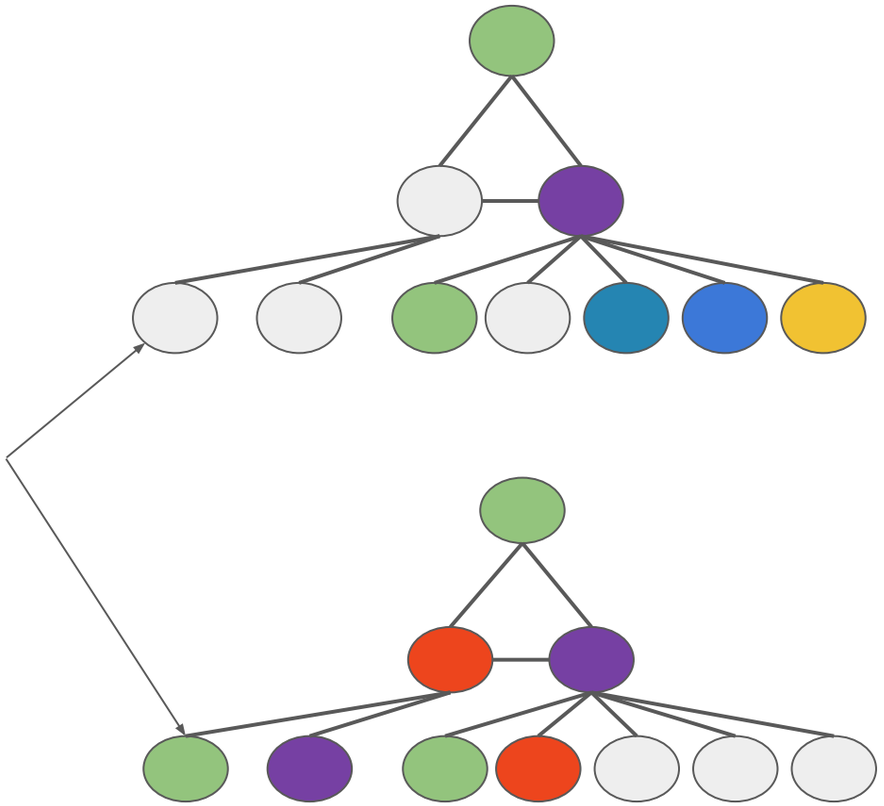
GNNExplainer (Remove nodes)

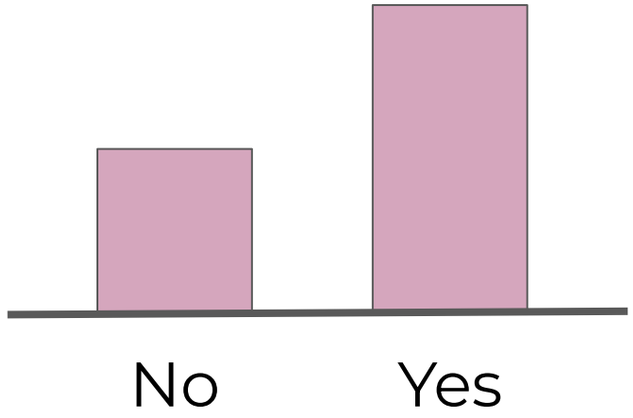


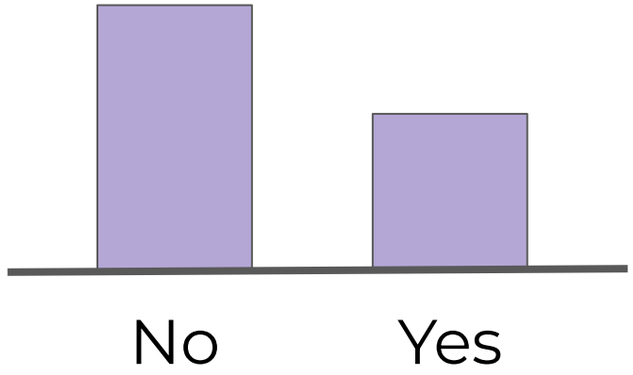
GNNExplainer (Probability distributions)











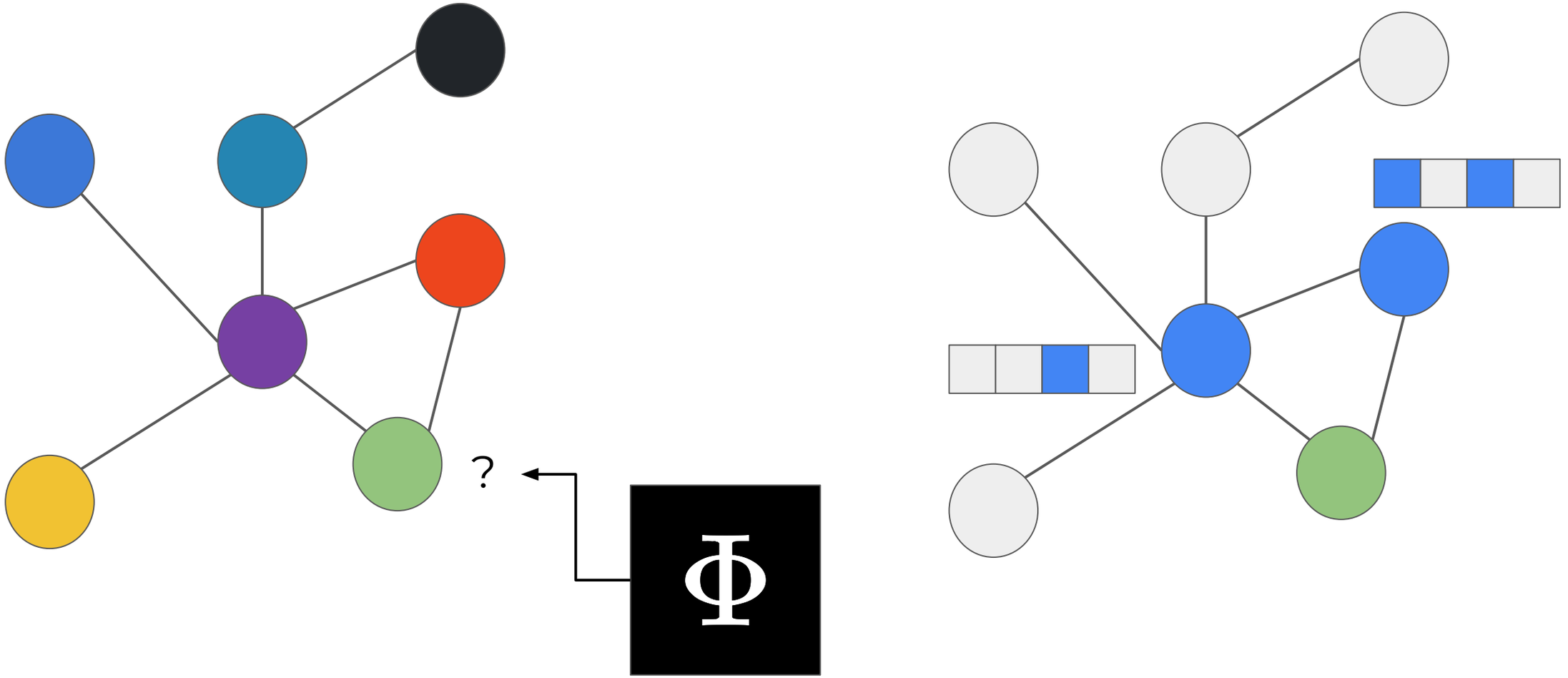
GNNExplainer (Formal)

$$\max_{G_S} MI(Y, (G_S, X_S)) = H(Y) - H(Y \mid G = G_S, X = X_S)$$

Human-interpretable intuition

Find a subgraph that maintains as much information as possible compared to the full graph

GNNExplainer (Explanation Example)



GraphLIME in a Nutshell

- GraphLIME builds **local interpretable models** for specific nodes in the graph
- It uses the **Hilbert-Schmidt Independence Criterion (HSIC) Lasso** for feature selection.
- GraphLIME operates **locally within the subgraph** of the node being explained.

HSIC: M. Yamada et al., “Ultra high-dimensional nonlinear feature selection for big biological data,” IEEE Trans. Knowl. Data Eng., vol. 30, no. 7, pp. 1352–1365, Jul. 2018.

Selecting the Node's Neighborhood

- For a target node, GraphLIME selects its **N-hop neighborhood** (where N is the number of GNN layers)

- This neighborhood captures relevant **structural information** around the node

Building the Local Model

- GraphLIME learns a **nonlinear interpretable model** within the selected subgraph
- This model explains the node's **prediction behavior**
- GraphLIME computes the **K most representative features** using HSIC Lasso

Feature Importance with HSIC Lasso

- HSIC Lasso is a **nonlinear method** for selecting important features
- It balances the trade-off between **model complexity and interpretability**
- HSIC Lasso identifies the most influential features for the node's prediction

Part III

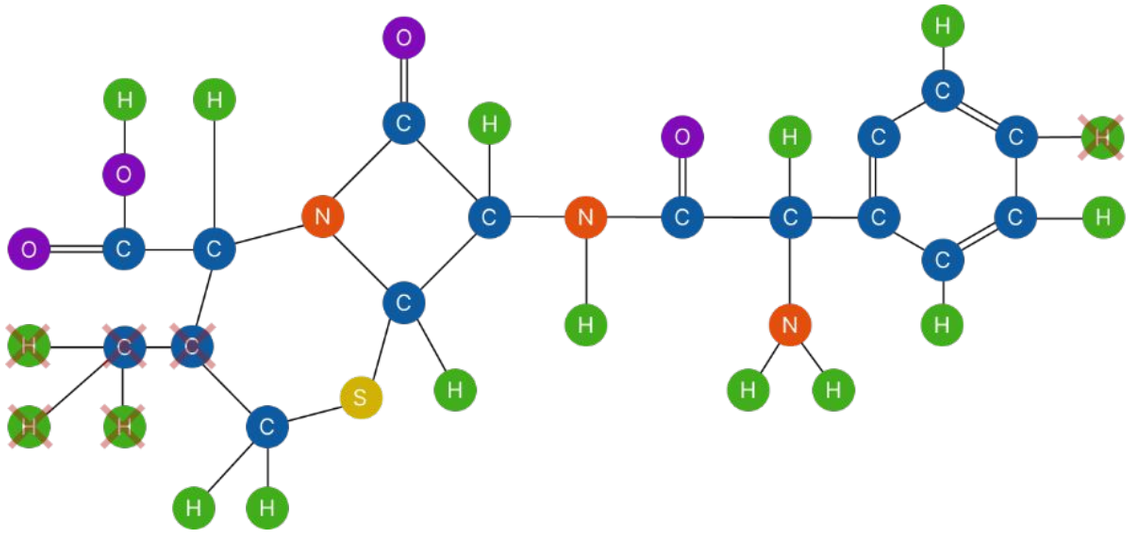
COUNTERFACTUAL EXPLANATIONS IN GRAPHS

by Mario A. Prado-Romero

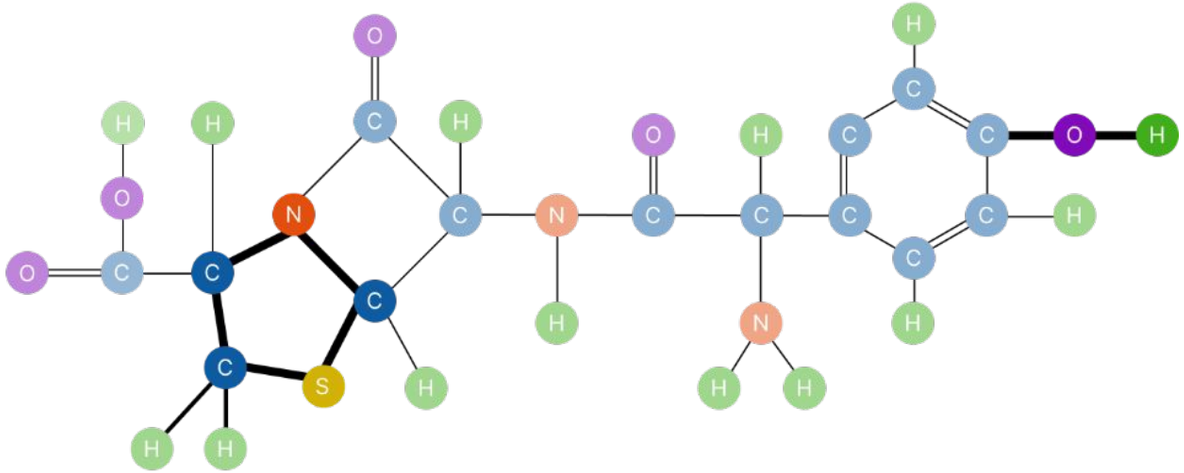
Based on:

Prado-Romero et al. “A survey on graph counterfactual explanations: definitions, methods, evaluation”, ACM CSUR 2023

Graph Counterfactuals?

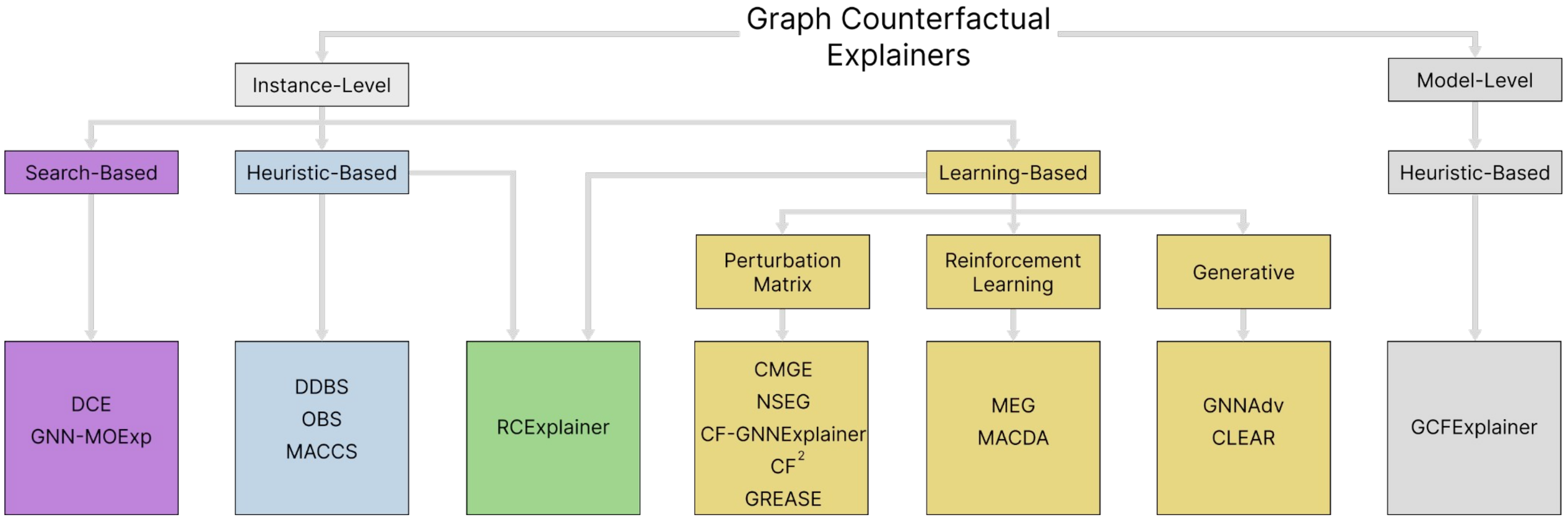


Cephalexin



Amoxicillin

GCE Methods


Graph Counterfactual Explainers
Instance-Level
Model-Level
Search-Based
Heuristic-Based
Learning-Based
Heuristic-Based
Perturbation Matrix
Reinforcement Learning
Generative
DCE
GNN-MOExp
DDBS
OBS
MACCS
RCExplainer
CMGE
NSEG
CF-GNNExplainer
CF^2
GREASE
MEG
MACDA
GNNAdv
CLEAR
GCFExplainer

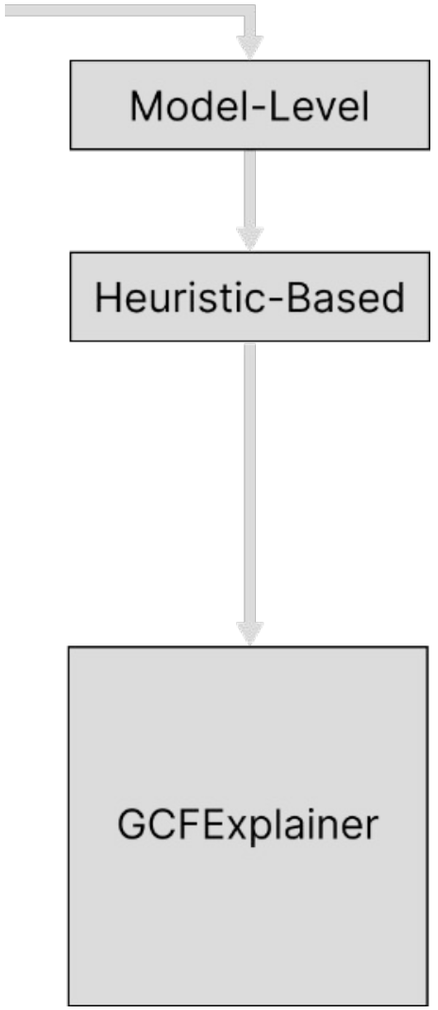
Global Counterfactual Explainers

Zexi Huang, Mert Kosan, Sourav Medya, Sayan Ranu, and Ambuj Singh. 2023. Global Counterfactual Explainer for Graph Neural Networks. In Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23). Association for Computing Machinery, New York, NY, USA, 141–149. https://doi.org/10.1145/3539597.3570376

GCFExplainer

- The **input** is a set of instances to explain $\mathbb{G}$
- The **output** is a set of counterfactual instances $\mathbb{C}$ that are considered an explanation of the input set
- **Coverage:** the proportion of graphs in $\mathbb{G}$ that have a close counterfactual in $\mathbb{C}$ under a given distance threshold θ:

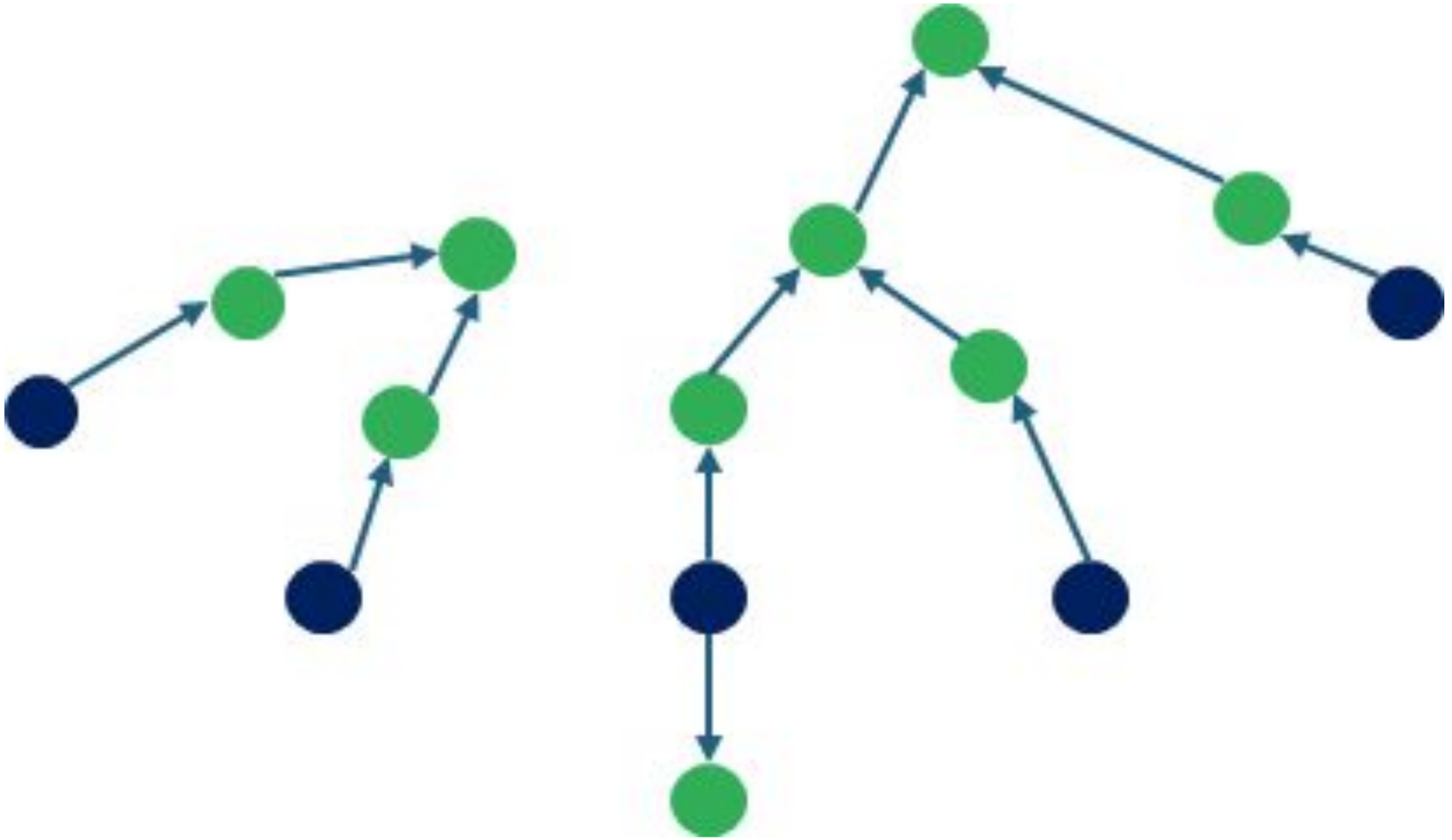
$$coverage(\mathbb{C}) = |\{G \in \mathbb{G} \mid \min_{\{C \in \mathbb{C}\}} \{d(G, C)\} \leq \theta\}|/|G|$$



Search Space

- The search space of counterfactual graphs = **graphs in the same domain as the input within a distance of θ**
- The number of potential graphs within θ **increases exponentially** since the space of graph edits is combinatorial
- Use an edit map to organize these graphs as a meta-graph $\mathfrak{G}$

Search Space (Illustrative)

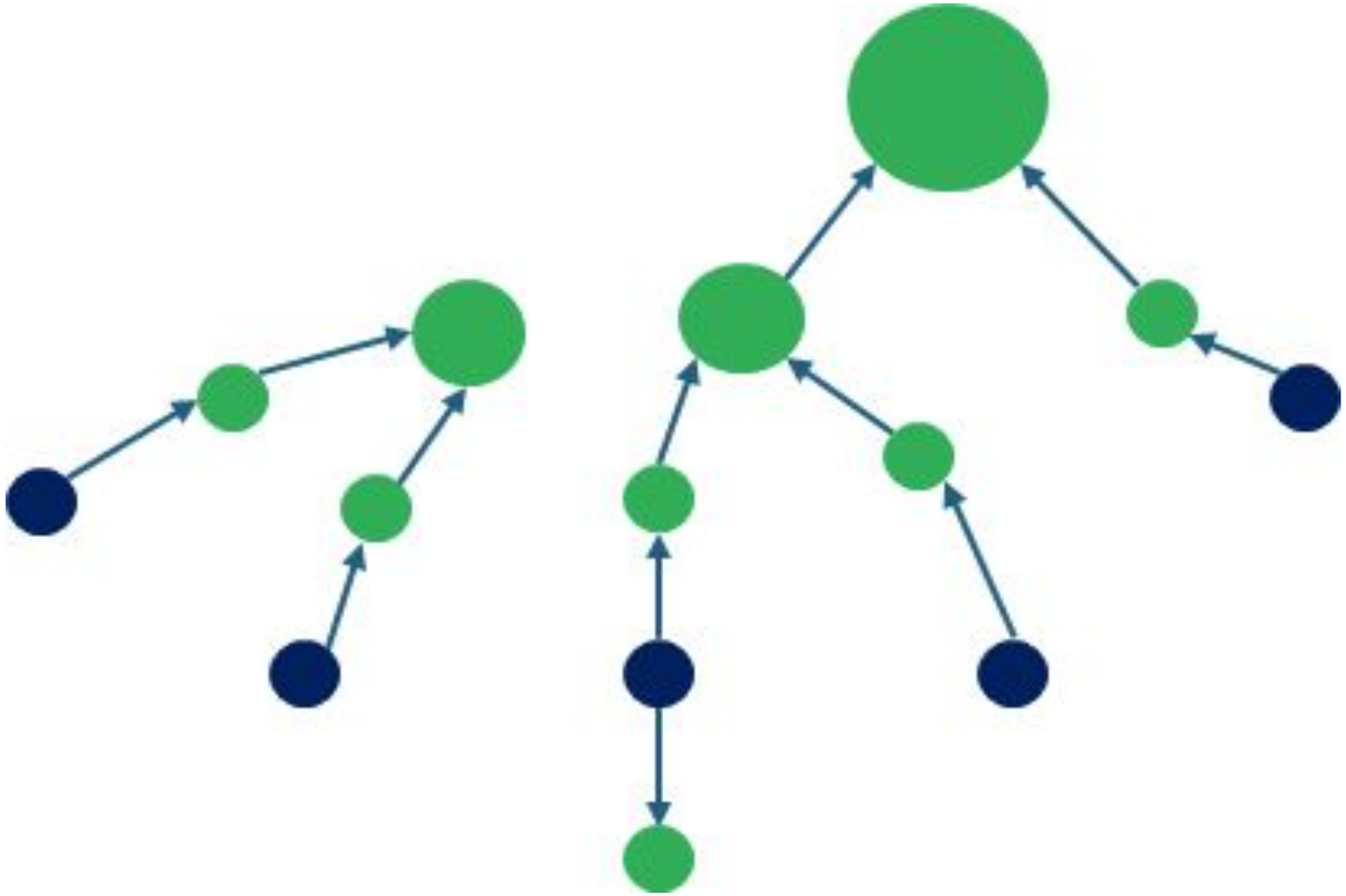


Vertex-Reinforced Random Walk

- Leverage vertex-reinforced random walks (VRRW) on the edit map $\mathfrak{G}$

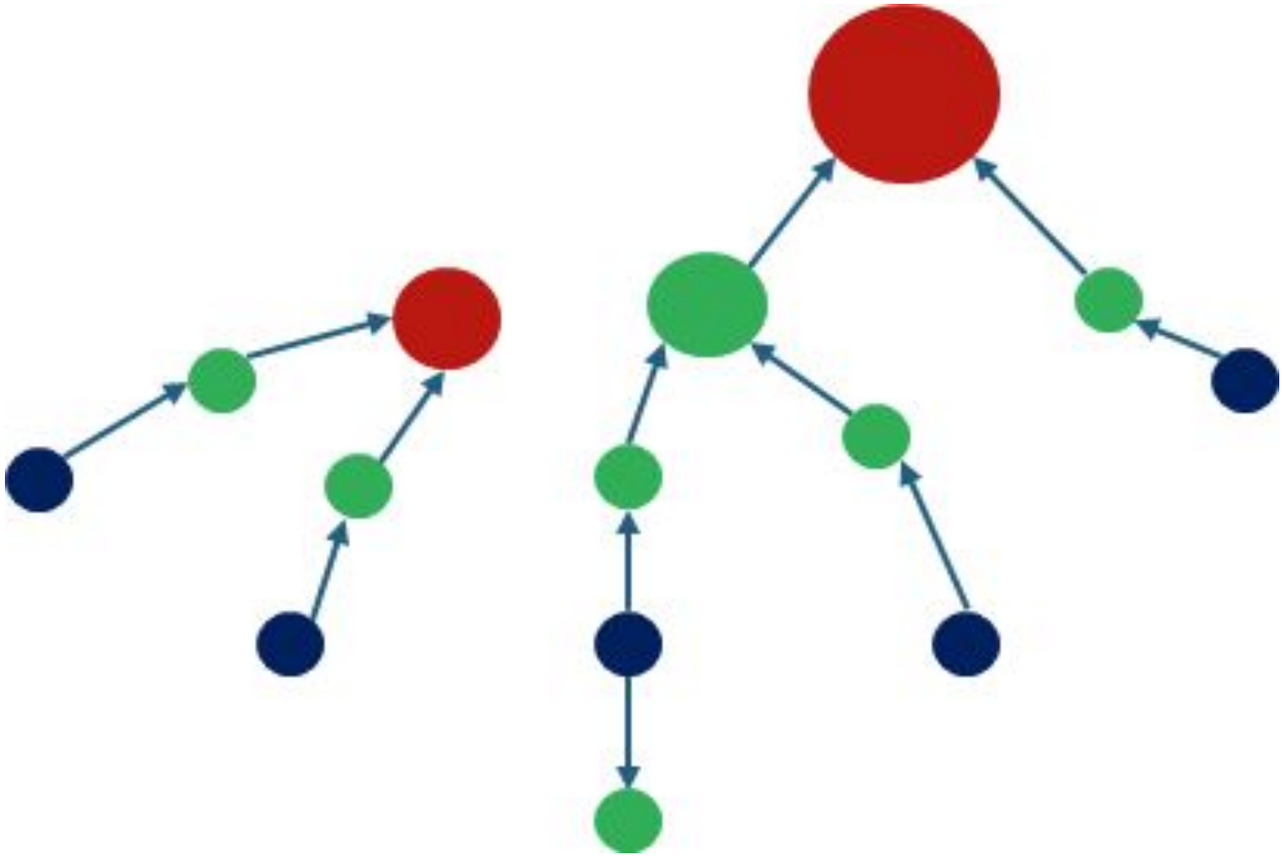
- VRRW converges to a set of nodes that are **important** and **diverse**, which will form a small set of counterfactual candidates for further processing

Vertex-Reinforced Random Walk



Iterative Summary Computation

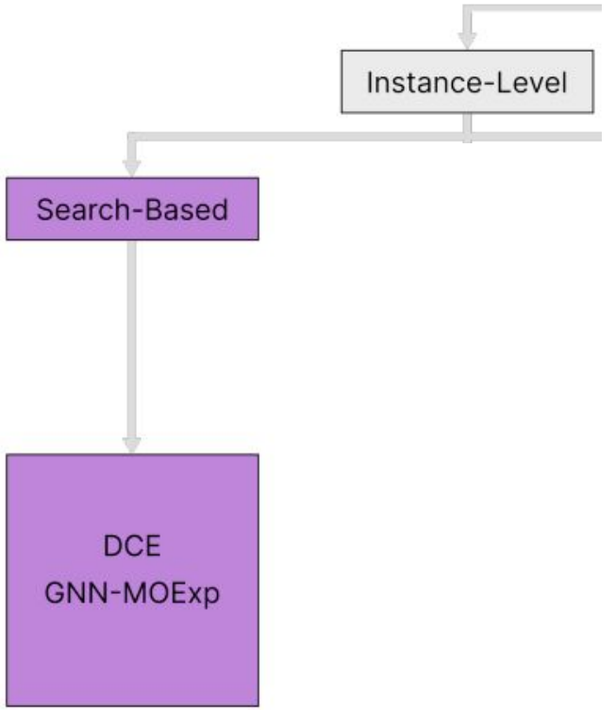
Create a summary of counterfactual graphs as the recourse representation by iteratively adding the best candidate based on the **maximal gain of the coverage**



BASELINE GRAPH COUNTERFACTUAL EXPLAINER

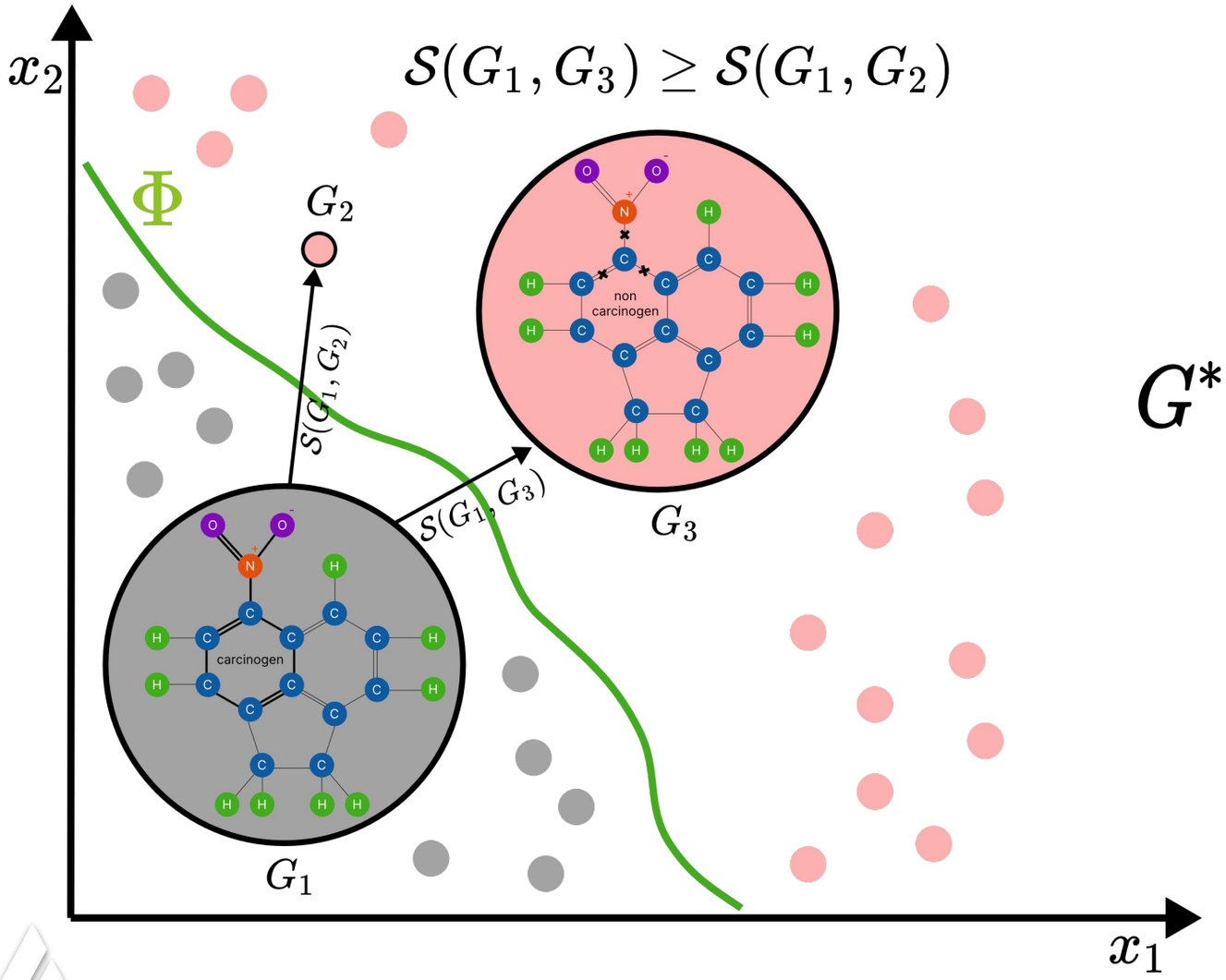
L. Faber, A. K. Moghaddam, and R. Wattenhofer. 2020. Contrastive Graph Neural Network Explanation. In Proc. of the 37th Graph Repr. Learning and Beyond Workshop at ICML 2020. Int. Conf. on Machine Learning, 28

Search-Based GCE Methods



- Find a counterfactual **within the data**
- For a graph $\boldsymbol{G} \in \mathcal{G}$ find a $\boldsymbol{G'} \in \mathcal{G}$ s.t. $\boldsymbol{\Phi}(\boldsymbol{G}) \neq \boldsymbol{\Phi}(\boldsymbol{G'})$
- These methods **fail** to produce a counterfactual if the explainer **cannot access** the original **dataset**

DCE (Distribution Complaint Explanations)



$$G^* = \arg \min_{G' \in \mathcal{G}, \Phi(G) \neq \Phi(G')} d(G, G')$$

Questions on DCE

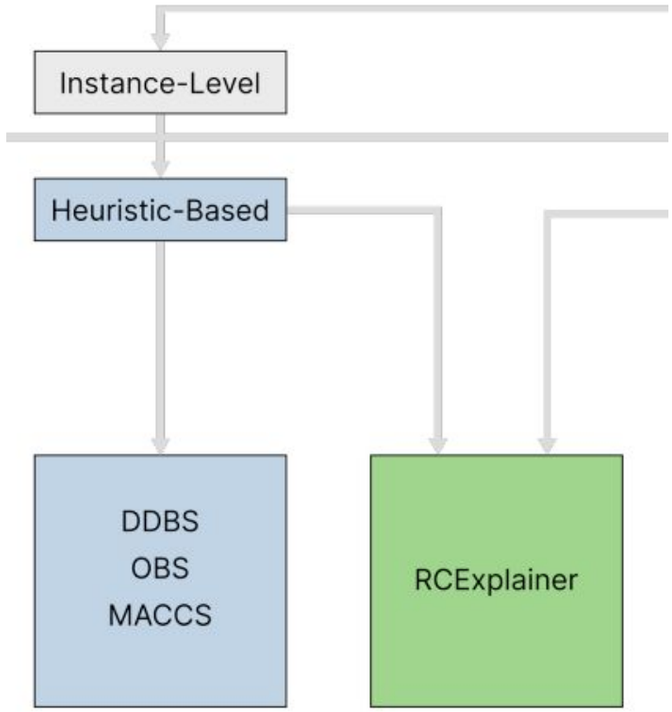
- Does it guarantee to always produce valid counterfactuals?
- Does it guarantee to produce the closest counterfactual to the input we want to explain?
- Are counterfactuals within the data manifold?

HEURISTIC-BASED EXPLAINERS

Abrate C, Bonchi F. Counterfactual graphs for explainable classification of brain networks. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining 2021 Aug 14 (pp. 2495-2504).

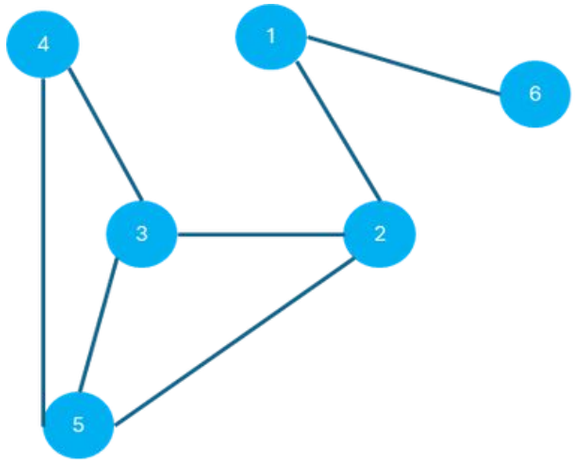
Wellawatte GP, Gandhi HA, Seshadri A, White AD. A Perspective on Explanations of Molecular Prediction Models. Journal of Chemical Theory and Computation. 2023 Mar 27;19(8):2149-60.

Heuristic-Based GCE Methods

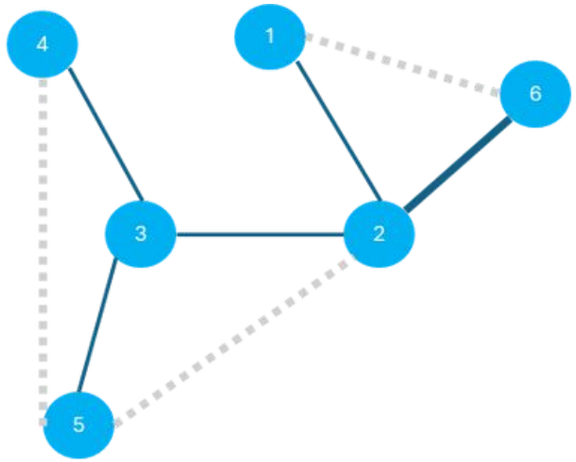


- **Perturb** the original graph such that $\Phi(G) \neq \Phi(G')$ **without** accessing the original dataset
- **Requires** to define the perturbation **rules** after a **careful examination** of the data

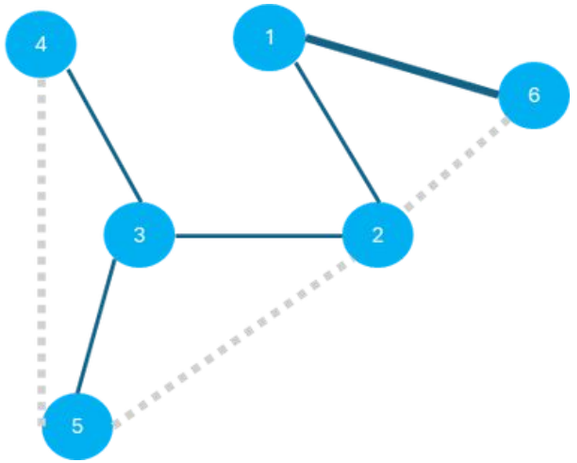
Oblivious Bidirectional Search (OBS)



Cyclic Graph



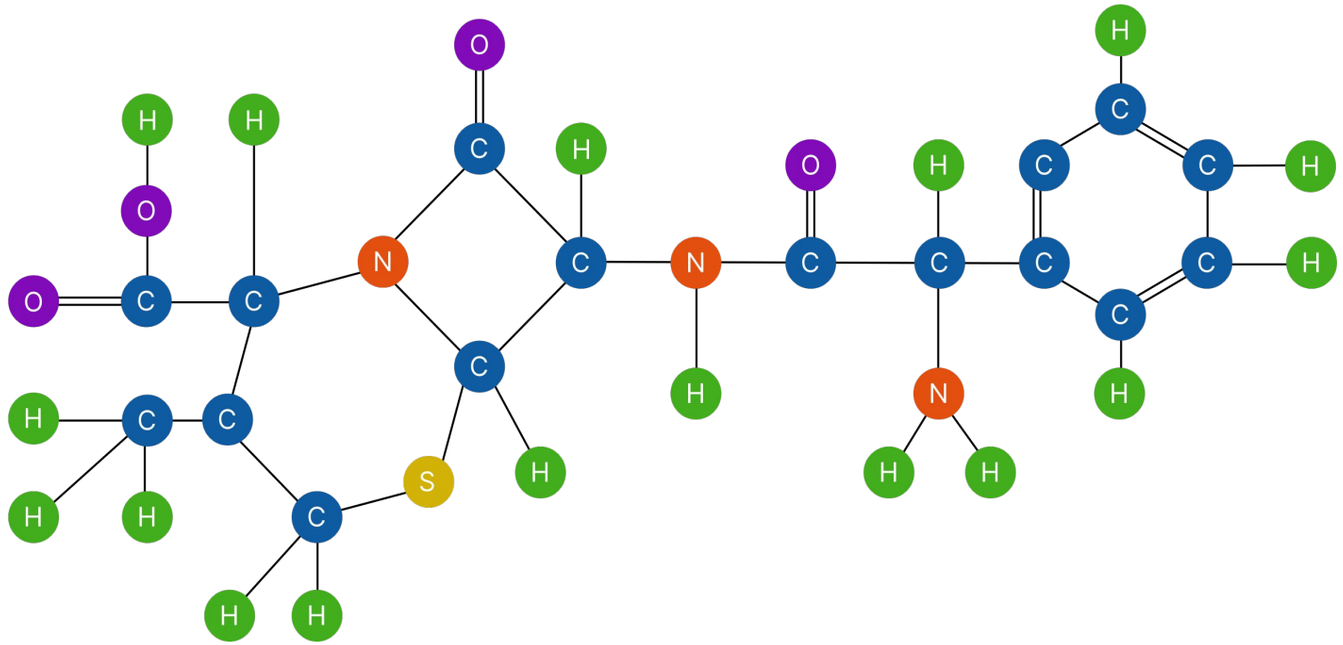
Step 1: Find a Counterfactual



Step 2: Reduce distance between original graph and counterfactual

MACCS

Uses SMILES representations of molecules



CC1=C(N2C(C(C2=O)NC(=O)C(C3=CC=CC=C3)N)SC1)C(=O)O

LEARNING-BASED EXPLAINERS

D. Numeroso and D. Bacciu. 2021. Meg: Generating molecular counterfactual explanations for deep graph networks. In 2021 Int. Joint Conf. on Neural Networks. IEEE, 1–8

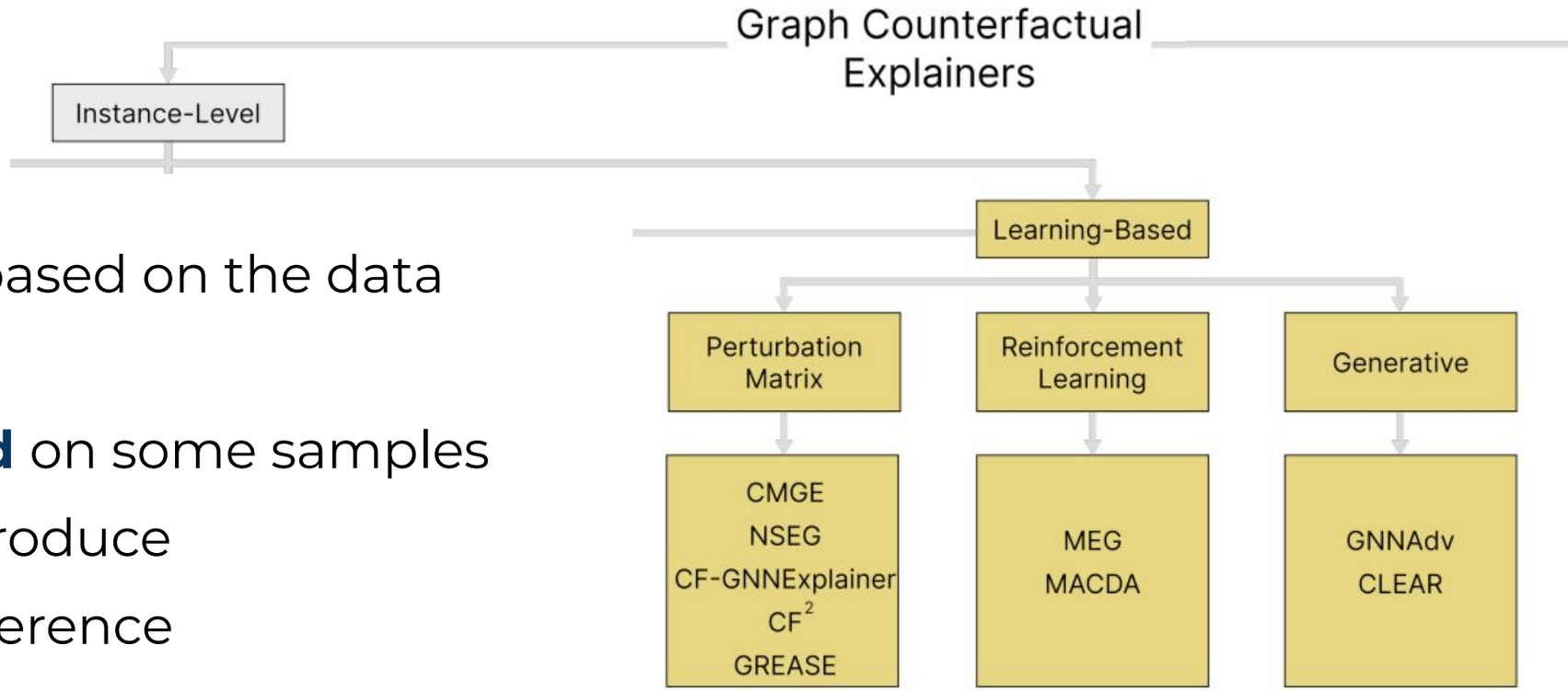
Wellawatte GP, Gandhi HA, Seshadri A, White AD. A Perspective on Explanations of Molecular Prediction Models. Journal of Chemical Theory and Computation. 2023 Mar 27;19(8):2149-60.

Tan, J., Geng, S., Fu, Z., Ge, Y., Xu, S., Li, Y. and Zhang, Y., 2022, April. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In Proceedings of the ACM Web Conference 2022 (pp. 1018-1027).

Ma, J., Guo, R., Mishra, S., Zhang, A. and Li, J., 2022. Clear: Generative counterfactual explanations on graphs. Advances in Neural Information Processing Systems, 35, pp.25895-25907.

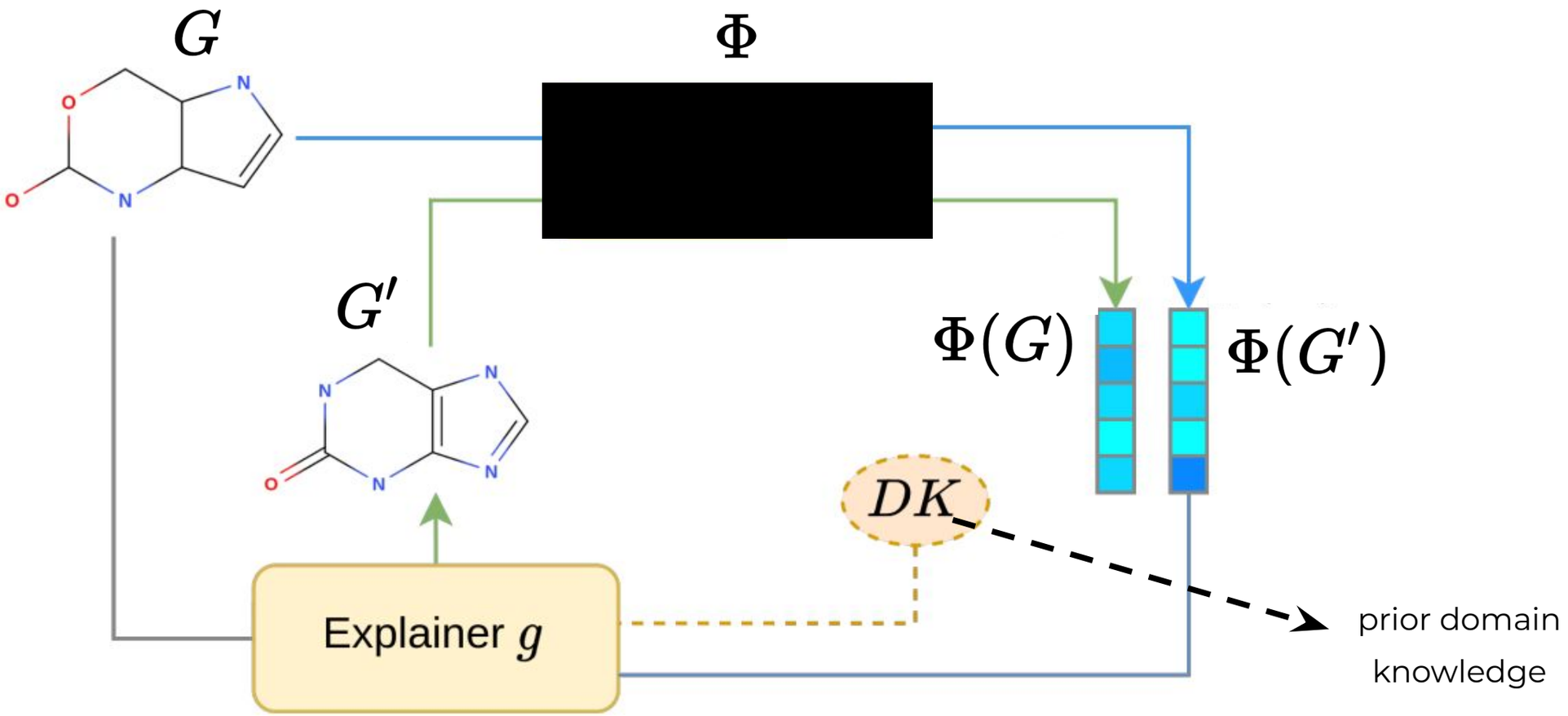
Prado-Romero MA, Prenkaj B, Stilo G. Robust Stochastic Graph Generator for Counterfactual Explanations. arXiv preprint arXiv:2312.11747. 2023 Dec 18.

Learning-Based GCE Methods



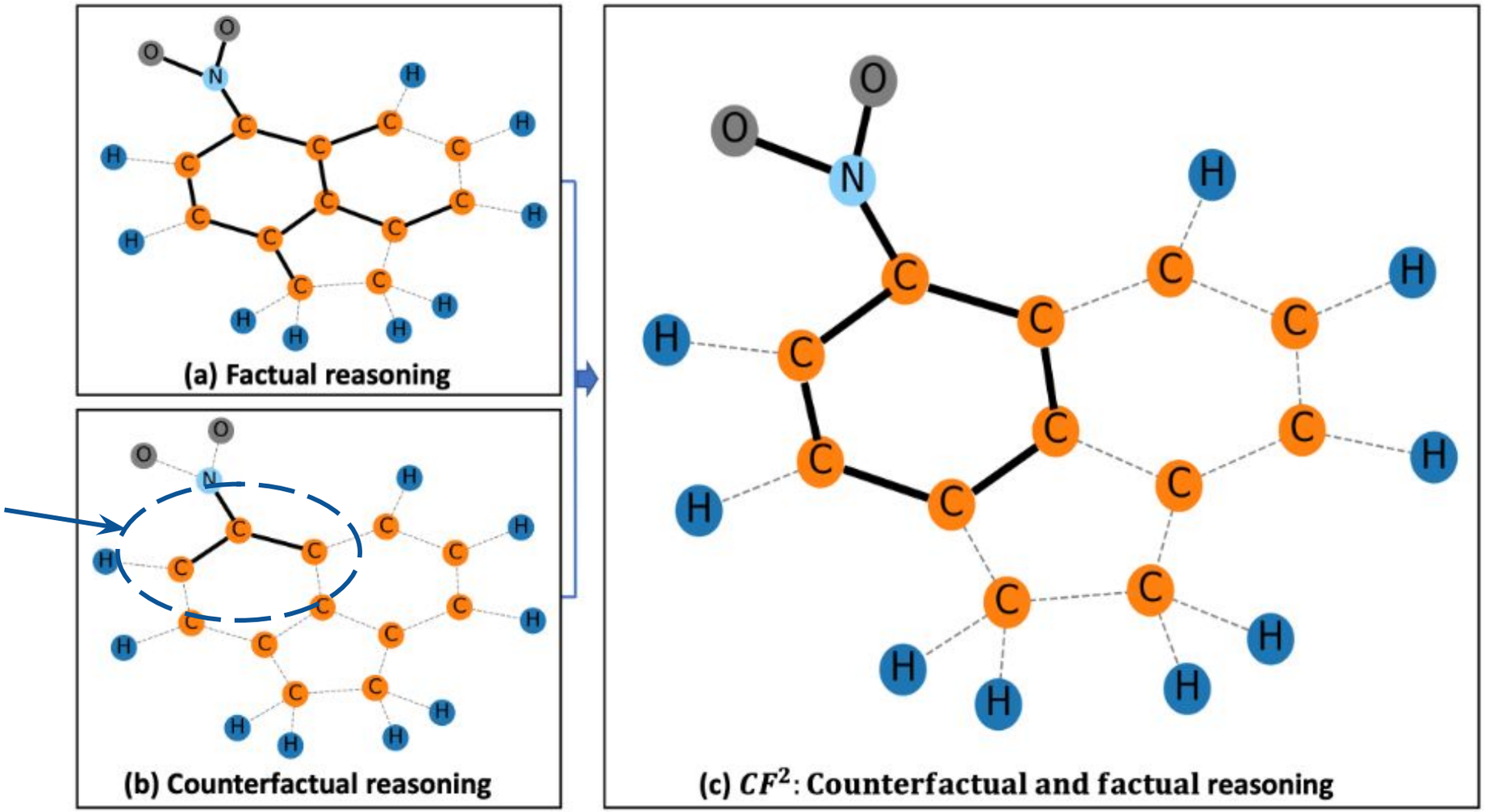
- **Learn the heuristic** based on the data
- Explainers are **trained** on some samples and can be used to produce counterfactuals at inference

MEG (ARCHITECTURE)



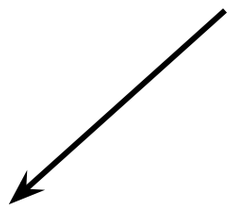
Counterfactual and Factual (CF^2)

these edges, **if removed**, create the **counterfactual**



CF² (Necessity & Sufficiency)

An explanation needs to be **necessary** and **sufficient**



Counterfactual Reasoning

Factual Reasoning

CF² (Sufficiency)

Factual Reasoning

$$\Phi(A * M, X * F) = y$$

Masked adjacency matrix

Masked node feature vectors

Both masks are learned by the explainer

Counterfactual Reasoning

$$\Phi(A - A*M, X - X*F) \neq y$$

Why do we subtract the masked adjacency matrix?

CF² (Relaxed Optimization)

$$L_f = \text{ReLU}(\gamma + P(\Phi(A * M, X * F) = \neg y) - S_f(M, F))$$

$$L_c = \text{ReLU}(\gamma - P(\Phi(A - A * M, X - X * F) = \neg y) - S_c(M, F))$$

$$\text{minimize } ||M||_1 + ||F||_1 + \lambda(\alpha L_f + (1 - \alpha)L_c)$$

Complexity

CLEAR

First work to treat **causality** when producing counterfactuals

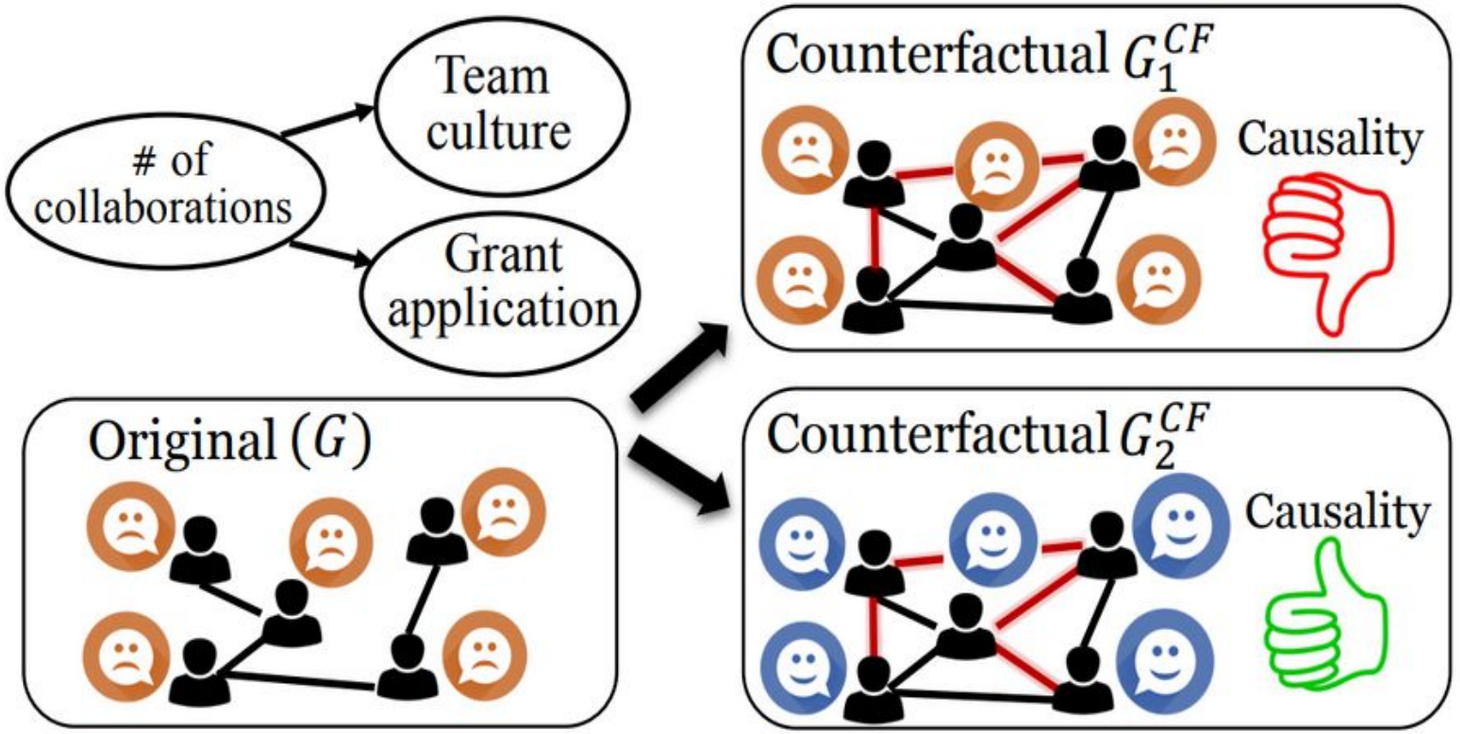
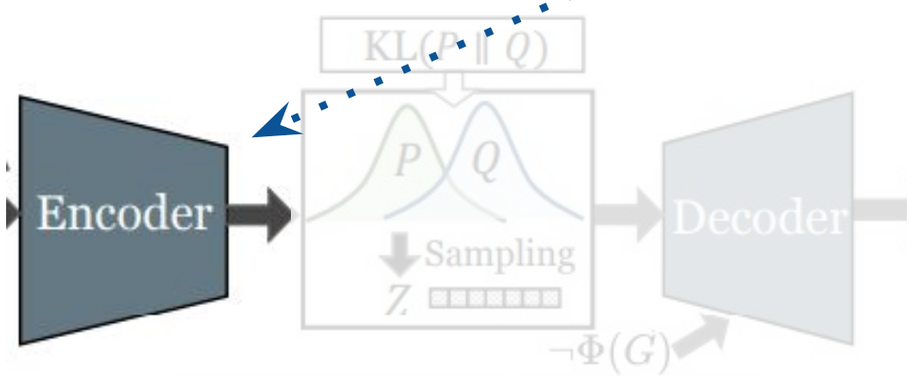


Image taken from: Ma J, Guo R, Mishra S, Zhang A, Li J. Clear: Generative counterfactual explanations on graphs. Advances in Neural Information Processing Systems. 2022 Dec 6;35:25895-907.

CLEAR

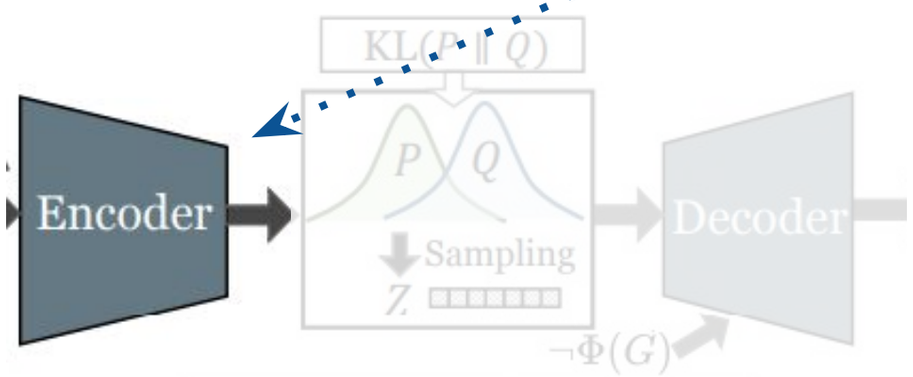
$$\mathcal{L} = \mathbb{E}_Q \left[d(G, G') + \alpha \ell\big(\Phi(G'), \neg\Phi(G)\big) \right] + \mathbf{KL}\Big(Q(Z \mid G, \neg\Phi(G)) \,||\, P(Z \mid G, \neg\Phi(G)) \Big)$$

Q is learned by the encoder



CLEAR

$$\mathcal{L} = \mathbb{E}_Q \left[d(G, G') + \alpha \ell \left(\Phi(G'), \neg\Phi(G) \right) \right] + \mathbf{KL} \left(Q(Z \mid G, \neg\Phi(G)) \,||\, P(Z \mid G, \neg\Phi(G)) \right)$$

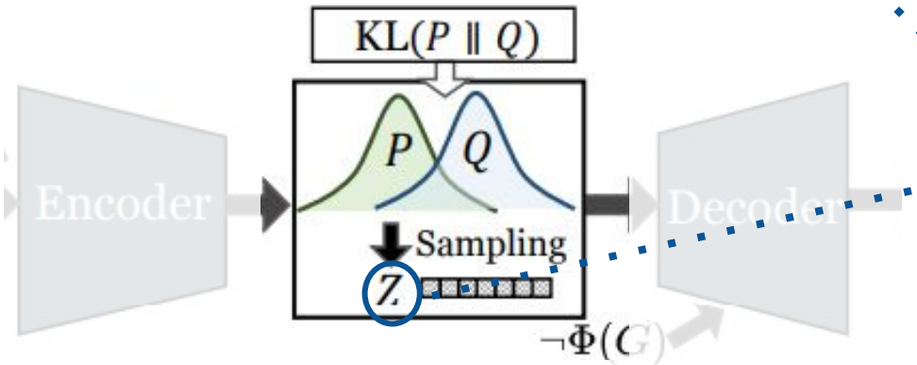


$$P(Z \mid G, \neg\Phi(G)) = \mathcal{N} \left(\mu_z \left(\neg\Phi(G) \right), \mathrm{diag}(\sigma_z^2 \left(\neg\Phi(G) \right)) \right)$$

The Gaussian distribution as prior to enforce the learned distribution Q to be close to the prior by minimizing their KL divergence

CLEAR

$$\mathcal{L} = \mathbb{E}_Q \left[d(G, G') + \alpha \ell \big(\Phi(G'), \neg\Phi(G) \big) \right] + \mathbf{KL} \Big(Q(Z | G, \neg\Phi(G)) \,||\, P(Z \mid G, \neg\Phi(G)) \Big)$$

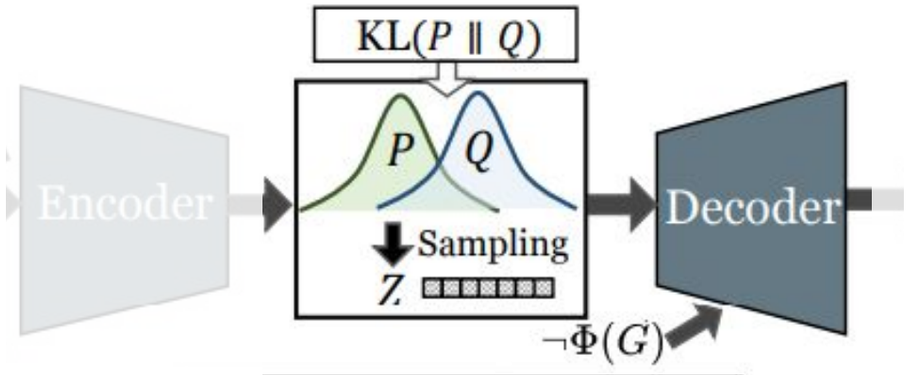


Z is sampled via the reparameterization trick

Kingma DP, Welling M. *Auto-encoding variational bayes*. arXiv preprint arXiv:1312.6114. 2013 Dec 20.

CLEAR

- One can generate **multiple counterfactuals** from sampling multiple Z
- The decoder produces a probabilistic graph where edges have weights $\mathbb{R}_0^1$
- Binarize the graph according to the Bernoulli distribution



CLEAR

$$d(G, G') = d_A(A, \text{Bernoulli}(A')) + d_X(X, X')$$

distance between the original adjacency matrix and the generated one

distance between the original node features and the generated ones

RSGG-CE

- Cornerstone paper in the debate "*Are generative counterfactual explanation approaches worth it?*"
- **Besides CLEAR, all other explainers are discriminative**
- Uses Residual GANs to learn how to generate counterfactuals

RSGG-CE (INTUITION)

- Train the generator on graphs of class c (**the class to explain**)
- Train the discriminator on graphs different from class c and the synthetic data generated
- Because the generator needs to fool the discriminator, it'll learn to produce graphs of class **not c**
- Use the oracle to guide the generator in crossing the boundary

RSGG-CE @ AAAI2024

Saturday **24th** as **Poster**

on Hall A on the Exhibition Level

Sunday **25th** as **Oral Paper** in Room **217** at **9:30**

CODE

Robust Stochastic Graph Generator for Counterfactual Explanations

Mario Alfonso Prado-Romero[*1], **Bardh Prenkaj**[*2], **Giovanni Stilo**[3]

[1] Gran Sasso Science Institute [2] Sapienza University of Rome [3] University of L'Aquila
marioalfonso.prado@gssi.it, prenkaj@di.uniroma1.it, giovanni.stilo@univaq.it
https://github.com/MarioTheOne/GRETEL

Abstract

Counterfactual Explanation (CE) techniques have garnered attention as a means to provide insights to the users engaging with AI systems. While extensively researched in domains such as medical imaging and autonomous vehicles, Graph Counterfactual Explanation (GCE) methods have been comparatively under-explored. GCEs generate a new graph similar to the original one, with a different outcome grounded on the underlying predictive model. Among these GCE techniques, those rooted in generative mechanisms have received relatively limited investigation despite demonstrating impressive accomplishments in other domains, such as artistic styles and natural language modelling. The preference for generative explainers stems from their capacity to generate counterfactual instances during inference, leveraging autonomously acquired perturbations of the input graph.

Prenkaj et al. 2021, 2020, 2023a; Verma, Mandal, and Gupta 2022; Wang, Yu, and Miao 2017).

Recently, deep learning (relying on GNNs (Scarselli et al. 2008)) has been beneficial in solving graph-based prediction tasks, such as community detection (Wu et al. 2022), link prediction (Wei et al. 2022), and session-based recommendations (Wu et al. 2019; Xu, Xi, and Wang 2021). Despite their remarkable performance, GNNs are black boxes, making them unsuitable for high-impact and high-risk scenarios. The literature has proposed several post-hoc explainability methods to understand *what is happening under the hood* of the prediction models. Specifically, counterfactual explainability is useful to understand how modifications in the input lead to different outcomes. Similarly, a recent field in Graph Counterfactual Explainability (GCE) has emerged

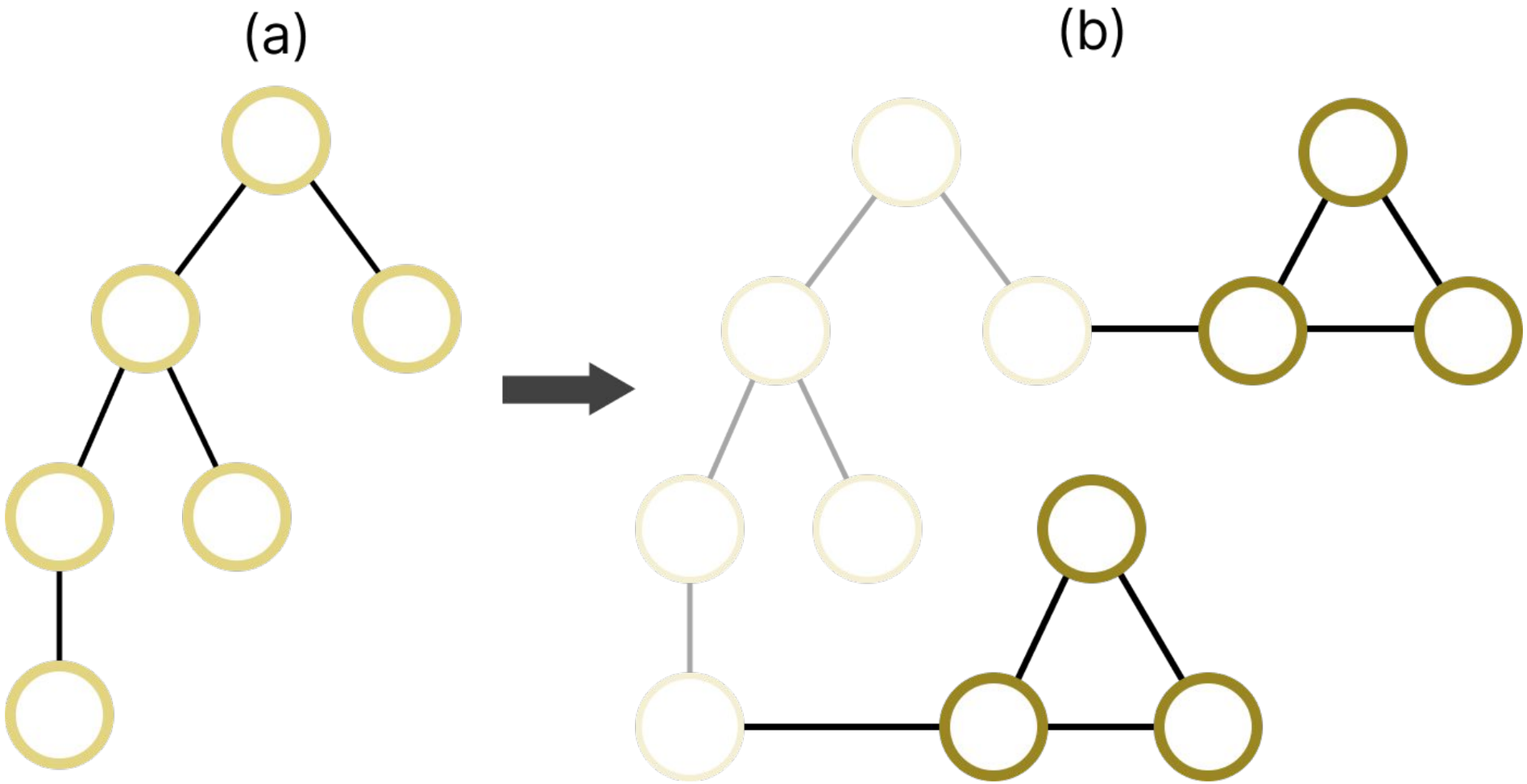
DATASETS AND EVALUATION METRICS

Graph Classification Datasets in the wild

Dataset	Domain	Publicly Available Repository (Data or Code)	Used by
Tree-Cycles [98]	synthetic	https://github.com/RexYing/gnn-model-explainer	[6, 15, 43, 81]
Tree-Grid [98]	synthetic	https://github.com/RexYing/gnn-model-explainer	[6, 15, 43]
Tree-Infinity	synthetic	https://github.com/MarioTheOne/GRETEL	[68]
BA-Shapes [98]	synthetic	https://github.com/RexYing/gnn-model-explainer	[6, 15, 43, 81]
BA-Community [98]	synthetic	https://github.com/RexYing/gnn-model-explainer	[6]
BA-2motifs [44]	synthetic	https://github.com/flyingdoog/PGExplainer	[6, 81]
ADHD [13]	-omics	https://github.com/MarioTheOne/GRETEL/tree/main/data/datasets/adhd	[1]
ASD [19, 37]	-omics	https://github.com/MarioTheOne/GRETEL/tree/main/data/datasets/autism/asd	[1]
BBBP [50]	molecular	https://www.kaggle.com/datasets/mmelahi/cheminformatics?select=bbbp.zip	[91]
HIV [20, 24, 70]	molecular	https://www.kaggle.com/datasets/mmelahi/cheminformatics?select=hiv.zip	[32, 91]
Ogbg-molhiv [31]	molecular	https://huggingface.co/datasets/OGB/ogbg-molhiv	[45]
Mutagenicity [34]	molecular	https://ls11-www.cs.tu-dortmund.de/people/morris/graphkerneldatasets/Mutagenicity.zip	[6, 32, 81]
NCI1 [87]	molecular	https://ls11-www.cs.tu-dortmund.de/people/morris/graphkerneldatasets/NCI1.zip	[6, 32, 81]
TOX21 [35]	molecular	https://tripod.nih.gov/tox21/challenge/data.jsp	[58]
ESOL [95]	molecular	https://github.com/deepchem/deepchem	[58, 81]
Proteins [11]	molecular	https://chrsmrrs.github.io/datasets/docs/datasets/	[32]
Davis [21]	molecular	http://staff.cs.utu.fi/~aatapa/data/DrugTarget/	[55]
PDBBind [89]	molecular	http://www.pdbbind.org.cn/	[55]
CiteSeer [26]	social	https://linqs.org/datasets/	[40, 81]
IMDB-M [96]	social	https://virginia.app.box.com/s/941v9pwh83lfw5vnwfbgcertlsoivg5j	[45]
CORA [51]	social	https://relational.fit.cvut.cz/dataset/CORA	[40]
Musae-Facebook [71]	social	https://www.kaggle.com/datasets/rozemberczki/musae-facebook-pagepage-network	[40]
LastFM [72]	social	https://github.com/gusye1234/LightGCN-PyTorch/tree/master/data/lastfm	[16]
Yelp [90]	social	https://github.com/gusye1234/LightGCN-PyTorch/tree/master/data/yelp2018/	[16]

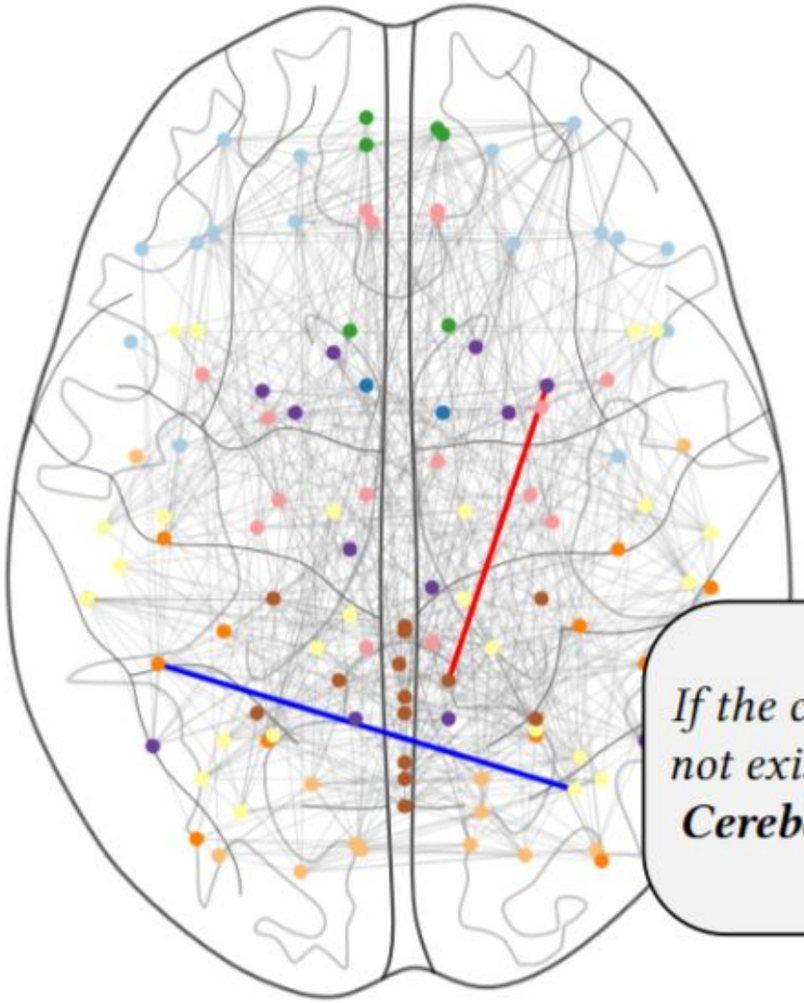
Prado-Romero MA, Prenkaj B, Stilo G, Giannotti F. A survey on graph counterfactual explanations: definitions, methods, evaluation. arXiv preprint arXiv:2210.12089. 2022 Oct 21.

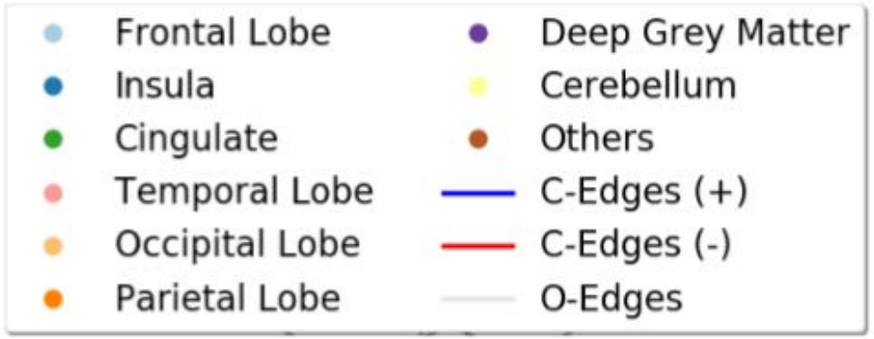
TreeCycles



Ying Z, Bourgeois D, You J, Zitnik M, Leskovec J. Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems. 2019;32.

Autism Spectrum Disorder (ASD)





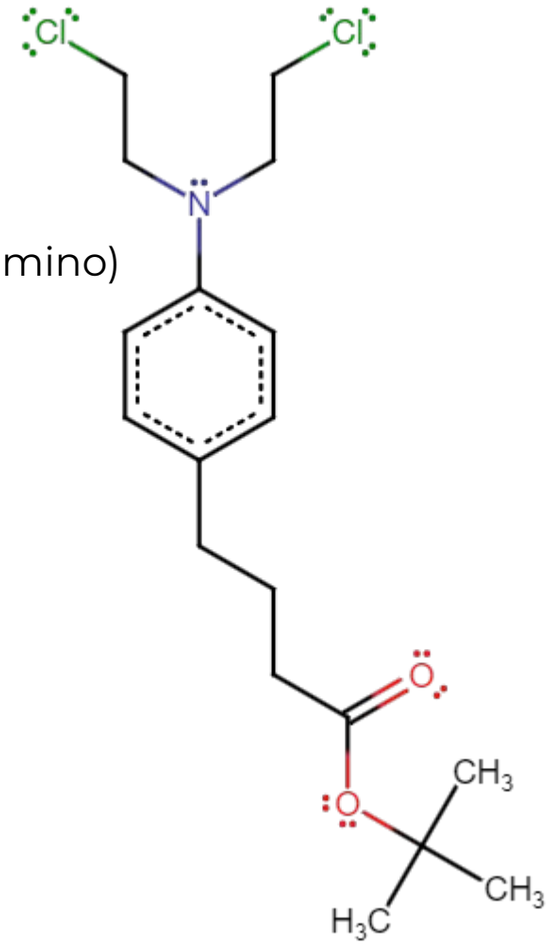
*Patient USM_0050453 is classified as ASD (class 1). If the connection between **Putamen_R** and **Cerebelum_9_R** did not exist and instead the connection between **Parietal_Inf_L** and **Cerebelum_Crus2_R** existed, then the patient would have been classified as TD (class 0).*

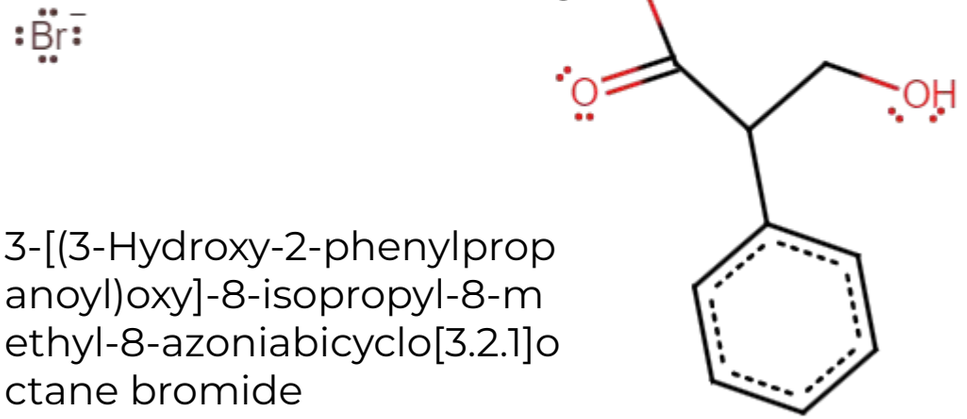
Abrate C, Bonchi F. Counterfactual graphs for explainable classification of brain networks. InProceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining 2021 Aug 14 (pp. 2495-2504).

Blood Barrier Penetration Prediction (BBBP)

1,1-dimethylethyl
4-(bis(2-chloroethyl)amino)
benzenebutanoate

$C_{18}H_{27}Cl_2NO_2$





3-[(3-Hydroxy-2-phenylprop
anoyl)oxy]-8-isopropyl-8-m
ethyl-8-azoniabicyclo[3.2.1]o
ctane bromide

$C_{20}H_{30}BrNO_3$

Martins IF, Teixeira AL, Pinheiro L, Falcao AO. A Bayesian approach to in silico blood-brain barrier penetration modeling. Journal of chemical information and modeling. 2012 Jun 25;52(6):1686-97.

Evaluation Metrics

Correctness (Validity) ⬆️

$$\Omega(G, G') = \mathbf{1}[\Phi(G) \neq \Phi(G')]$$

We're answering the question: *"Can the explainer produce an instance with a **different classification** from the original instance?"*

Evaluation Metrics

Graph Edit Distance (GED) ⬇️

$$\mathrm{GED}(G, G') = \min_{\{p_1,\ldots,p_n\}\in\mathcal{P}(G,G')} \sum_{i=1}^{n} \omega(p_i)$$

We're answering the question: *"How **similar** is the produced counterfactual candidate w.r.t. original input?"*

GED Trick

$$\mathrm{GED}(G, G') = \min_{\{p_1,\ldots,p_n\} \in \mathcal{P}(G,G')} \sum_{i=1}^{n} \omega(p_i)$$

$$\sum_i \sum_j \left| A_{i,j} - A'_{i,j} \right|$$

Just sum the absolute element-wise difference to get the edges that were changed in the counterfactual w.r.t. the original input

(Works only if we have binary adjacency matrices, and their dimensionalities remain the same)

Evaluation Metrics

Fidelity ⬆️

$$\Psi(G, G') = \mathbf{1}[\Phi(G) = y_G] - \mathbf{1}[\Phi(G') = y_G]$$

We're answering the question: "*How **faithful** are the explanations to the oracle considering their correctness?*"

Let's De-Reconstruct Fidelity

Fidelity ⬆️

$$\Psi(G, G') = \mathbf{1}[\Phi(G) = y_G] - \mathbf{1}[\Phi(G') = y_G]$$

We want the oracle to be correct in predicting the ground truth of the original instance (**aka accuracy**)

Let's De-Reconstruct Fidelity

Fidelity ⬆️

$$\Psi(G, G') = \mathbf{1}[\Phi(G) = y_G] - \mathbf{1}[\Phi(G') = y_G]$$

We don't want the oracle to predict the same class as the ground truth for the counterfactual candidate

Let's De-Reconstruct Fidelity

Fidelity can have 3 values

- **+1** → both the explainer and oracle are working correctly
- **0 & -1** → something is wrong with the explainer or the oracle

When the oracle has perfect accuracy, fidelity is equal to correctness

Takeaway Lessons

SOA PERFORMANCE

Accuracy = 100%

Dataset	Method	GED ↓	Correctness ↑	Fidelity ↑
Tree-Cycles	RAND@5	92.18 ± 5.44	0.55 ± 0.50	0.55 ± 0.50
	RAND@10	123.74 ± 7.43	0.51 ± 0.50	0.51 ± 0.50
	RAND@15	147.93 ± 8.26	0.58 ± 0.50	0.58 ± 0.50
	DCE	50.36 ± 0.00	**1.00 ± 0.00**	**1.00 ± 0.00**
	OBS	57.31 ± 0.03	0.96 ± 0.01	0.96 ± 0.01
	DDBS	71.79 ± 0.24	0.59 ± 0.01	0.59 ± 0.01
	MACCS	–	–	–
	CLEAR	79.76 ± 3.60	0.53 ± 0.10	0.53 ± 0.10
	CF^2	**31.54 ± 0.12**	0.47 ± 0.10	0.47 ± 0.10
	MEG	159.70 ± 1.34	0.53 ± 0.00	0.53 ± 0.00

Naïve baseline is better than intensive and complicated generative approach

Chance level correctness!

SOA PERFORMANCE

Accuracy = 79%

Dataset	Method	GED ↓	Correctness ↑	Fidelity ↑
ASD	RAND@5	618.06 ± 8.27	0.00 ± 0.00	0.00 ± 0.00
	RAND@10	1152.93 ± 20.19	0.00 ± 0.00	0.00 ± 0.00
	RAND@15	1600.78 ± 18.22	0.00 ± 0.00	0.00 ± 0.00
	DCE	1011.69 ± 0.00	**1.00 ± 0.00**	**0.54 ± 0.00**
	OBS	9.89 ± 0.11	1.00 ± 0.00	0.54 ± 0.00
	DDBS	11.79 ± 0.29	**1.00 ± 0.00**	**0.54 ± 0.00**
	MACCS	–	–	–
	CLEAR	1739.60 ± 131.16	0.47 ± 0.13	0.25 ± 0.18
	CF^2	655.49 ± 2.87	0.46 ± 0.09	0.37 ± 0.15
	MEG	×	×	×

Generative approaches don't even compare to the baselines

SOA PERFORMANCE

Accuracy = 86%

Dataset	Method	GED ↓	Correctness ↑	Fidelity ↑
BBBP	RAND@5	30.98 ± 33.27	0.85 ± 0.35	0.62 ± 0.69
	RAND@10	52.98 ± 58.96	0.86 ± 0.35	0.65 ± 0.66
	RAND@15	82.97 ± 137.37	0.85 ± 0.36	0.61 ± 0.69
	DCE	27.92 ± 0.12	**1.00 ± 0.00**	**0.72 ± 0.00**
	OBS	0.00 ± 0.00	0.00 ± 0.00	0.61 ± 0.00
	DDBS	×	×	×
	MACCS	**11.23 ± 0.08**	0.40 ± 0.00	0.23 ± 0.00
	CLEAR@1	27056.29 ± 9.69	0.87 ± 0.02	0.64 ± 0.03
	CLEAR@5	26711.57 ± 112.67	0.85 ± 0.02	0.62 ± 0.03
	CLEAR@15	25986.66 ± 170.41	0.85 ± 0.01	0.62 ± 0.06
	CF^2	25.72 ± 0.63	0.85 ± 0.02	0.63 ± 0.03
	MEG	269.35 ± 0.39	0.51 ± 0.04	0.32 ± 0.04

Removing the factual graph can get you good results...

Good correctness but too far away from the input

Our Critiques (Some)

- Most datasets are toy-like, which don't allow for GNNs (oracles) to be correctly trained
  - *See Abrate & Bonchi's paper: customly written oracle which looks like a linear separator with fixed slope and y-intercept*

- To date, only RSGG-CE can compare against search-based explainers. *Why aren't the other works comparing to them as baselines!?*

- Let's define a standard evaluation benchmark: *correctness, GED, and fidelity must be included in all future proposals*

What's Next?

Open Research Questions

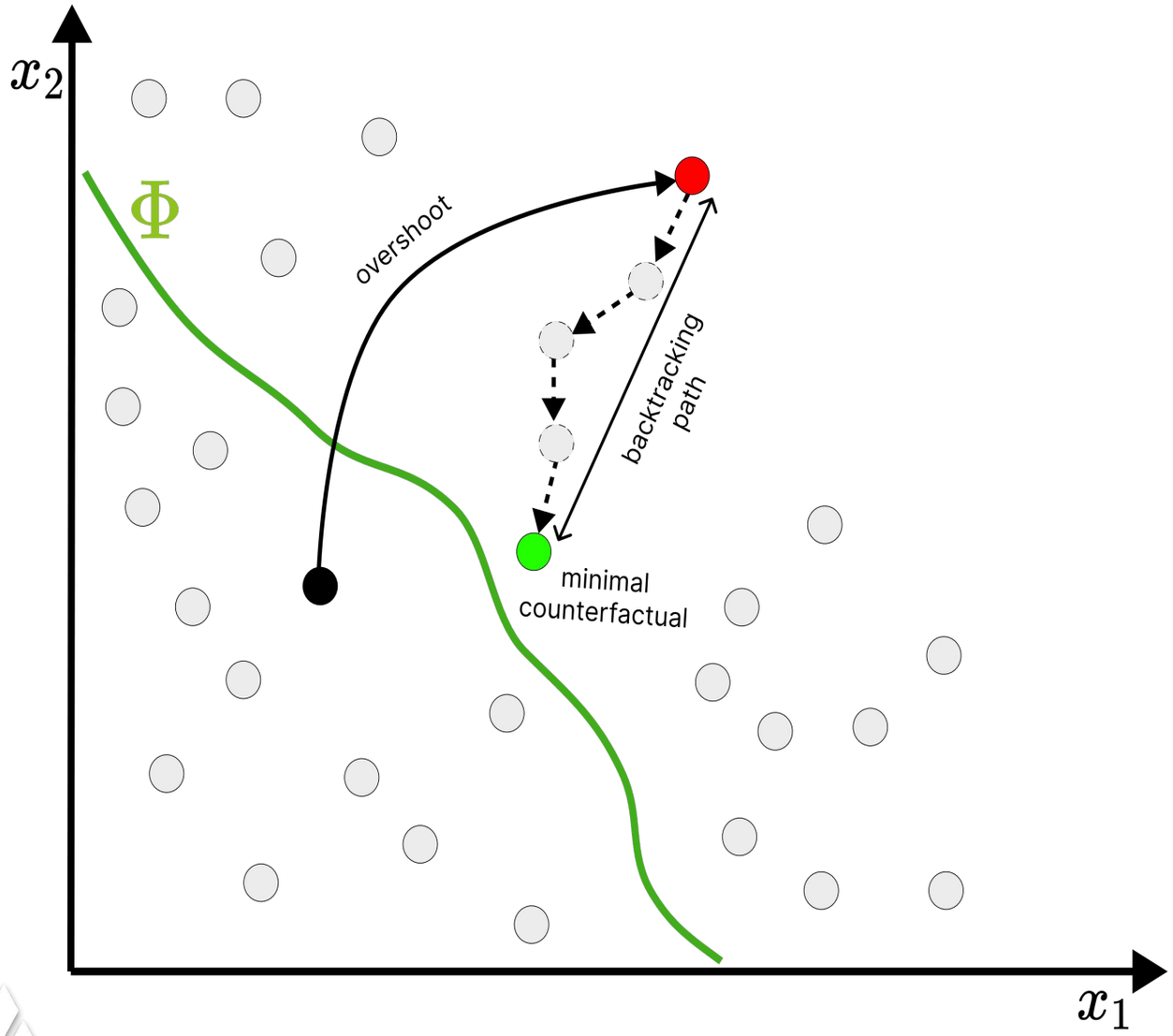
- **Are generative counterfactual explainers worth it?**
- What's the difference between **counterfactual explanations** and **adversarial attacks**?
- How sure are we about the generated counterfactuals? Can we incorporate **uncertainty** in them?

Open Research Questions

- Are the produced **counterfactuals actionable?** How to incorporate **domain knowledge** into the explanation methods?

- How to ensure our explanations methods are **stable** and **robust**, producing **similar explanations for similar instances**?

- How can we **backtrack near the decision boundary** once we overshoot on the other side **to produce minimal counterfactuals**? (see next slide)



Can we learn the backtracking steps one-by-one?

Looks like an RL problem now...

Care to help us? Work with us then

Thanks for your attention!

based on ACM survey

FOR QUESTIONS AND DISCUSSION

WE MEET AT 15:30 IN FRONT OF 121.

RSVP TO WHOVA

https://dl.acm.org/doi/abs/10.1145/3618105

Mario A. Prado-Romero
marioalfonso.prado@gssi.it

Bardh Prenkaj
prenkaj@di.uniroma1.it

Giovanni Stilo
giovanni.stilo@univaq.it