Graph Neural Network Causal Explanation via Neural Causal Models

Arman Behnam[1] and Binghui Wang[1]

Illinois Institute of Technology, Chicago IL 60616, USA
abehnam@hawk.iit.edu bwang70@iit.edu

Abstract. Graph neural network (GNN) explainers identify the important subgraph that ensures the prediction for a given graph. Until now, almost all GNN explainers are based on association, which is prone to spurious correlations. We propose CXGNN, a GNN causal explainer via causal inference. Our explainer is based on the observation that a graph often consists of a causal underlying subgraph. CXGNN includes three main steps: 1) It builds causal structure and the corresponding structural causal model (SCM) for a graph, which enables the cause-effect calculation among nodes. 2) Directly calculating the cause-effect in real-world graphs is computationally challenging. It is then enlightened by the recent neural causal model (NCM), a special type of SCM that is trainable, and design customized NCMs for GNNs. By training these GNN NCMs, the cause-effect can be easily calculated. 3) It uncovers the subgraph that causally explains the GNN predictions via the optimized GNN-NCMs. Evaluation results on multiple synthetic and real-world graphs validate that CXGNN significantly outperforms existing GNN explainers in exact groundtruth explanation identification[1].

Keywords: Graph neural network explanation · Neural causal model

1 Introduction

Graph is a pervasive data type that represents complex relationships among entities. Graph Neural Networks (GNNs) [6, 13, 15, 44], a mainstream learning paradigm for processing graph data, take a graph as input and learn to model the relation between nodes in the graph. GNNs have demonstrated state-of-the-art performance across various graph-related tasks such as node classification, link prediction, and graph classification, to name a few [42].

Explainable GNN provides a human-understandable way of the prediction outputted by a GNN. Given a graph and a label (correctly predicted by a GNN model), a GNN explainer aims to determine the important subgraph (called *explanatory subgraph*) that is able to predict the label. Various GNN explanation methods [3,5,8,9,14,18,21,26–28,30,33,39,40,46–50] have been proposed, wherein almost all of them are based on *associating* the prediction with a subgraph that

[1] Code is available at https://github.com/ArmanBehnam/CXGNN

has the maximum predictability (more details see Section 2). However, recent studies [7, 32, 41] show that association-based explanation methods are prone to biased subgraphs as the valid explanation due to *spurious correlations* in the training data. For instance, when the groundtruth explanatory subgraph is the *House*-motif, it often occurs with the *Tree* bases. Then a GNN may not learn the true relation between the label and the *House*-motif, but the *Tree* base, due to it being easier to learn. We argue a truly explainable GNN should uncover the intrinsic *causal relation* between the explanatory subgraph and the label, which we call the *causal explanation* [4, 10, 29]. Note that a few GNN explanation methods [19, 20, 32] are motivated by the causality concepts, e.g., Granger causality [11][2], but they are not causal explanations in essence.

Our GNN causal explainer: In this paper, we take the first step to propose a GNN explainer via causal inference [24], which focuses on understanding and quantifying cause-and-effect relations between variables in the task of interest. In the context of GNN causal explanation, we base on a common observation that a graph often consists of a causal subgraph and a non-causal counterpart [7, 19, 20, 32, 38, 41]. Then given a graph and its (predicted) label, we aim to identify the *causal explanatory subgraph* that causally yields such prediction. Our key idea is that the causal explainer should be able to identify the causal interactions among nodes/edges and interpret the label based on these interactions.

Specifically, we propose a GNN causal explainer, called CXGNN, that consists of three steps. 1) We first define causal structure (w.r.t. a reference node) for the graph, which admits structure causal models (we call GNN-SCM). Such GNN-SCM enables interventions to calculate cause and effects among nodes via do-calculus [24]. 2) In real-world graphs, however, it is computationally challenging to perform do-calculus computation due to a large number of nodes and edges. To address it, we are inspired by the recent neural causal model (NCM) [43], which is a special type of SCM that can be trainable. We prove that, for each GNN-SCM, we can build a family of the respective GNN-NCMs. We then construct a parameterized GNN-NCM such that when it is optimized, the cause-effects defined on the GNN-NCM are easily calculated. 3) We finally determine the causal explanatory subgraph. To do so, we first introduce the node expressivity that reflects how well the reference node is in the causal explanatory subgraph. Then we iterate all nodes in the input graph and identify the trained GNN-NCM leading to the highest node expressivity. The underlying causal structure of this GNN-NCM is then the causal explanatory subgraph.

We evaluate CXGNN on multiple synthetic and real-world graph datasets with groundtruth explanations, and compare them with state-of-the-art association-based and causality-inspired GNN explainers. Our results show CXGNN significantly outperforms the baselines in exactly finding the groundtruth explanations. Our contributions are summarized below:

– We propose the first GNN causal explainer CXGNN.

[2] Granger causality can only identify that one variable helps predict another, but it does not tell you which variable is the cause and which is the effect.

– We leverage the neural-causal connection, design the GNN neural causal models and train them to identify the causal explanatory subgraph.
– CXGNN shows superiority over the state-of-the-art GNN explainers.

2 Related Work

Association-based explainable GNN: Almost all existing GNN explainers are based on association. These methods can be roughly classified into five types. (i) *Decomposition-based methods [8, 27]* consider the prediction of a GNN model as a score and decompose it backward layer-by-layer until it reaches the input. The score of different parts of the input can be used to explain its importance to the prediction. (ii) *Gradient-based methods [3, 27]* take the gradient of the prediction with respect to the input to show the sensitivity of a prediction to the input. The sensitivity can be used to explain the input for that prediction. (iii) *Surrogate-based methods [5, 14, 26, 33, 50]* replace the GNN model with a simple and interpretable surrogate one. (iv) *Generation-based methods [18, 30, 40, 47]* use generative models or graph generators to generate explanations. (v) *Perturbation-based methods [9, 21, 28, 39, 46, 48, 49]* aim to find the important subgraphs as explanations by perturbing the input. State-of-the-art explainers from (iii)-(iv) show better performance than those from (i) and (ii).
Causality-inspired explainable GNN: Recent GNN explainers [7, 19, 20, 32, 38, 41] are motivated by causality. These methods are based on a common observation that a graph consists of the causal subgraph and its non-causal counterpart. For instance, OrphicX [20] uses information-theoretic measures of causal influence [2], and proposes to identify the (non)causal factors in the embedding space via information flow maximization. CAL [32] introduces edge and node attention modules to estimate the causal and non-causal graph features.
CXGNN vs. causality-inspired explainers. The key difference lies in CXGNN focuses on identifying the *causal explanatory subgraph* by *directly quantifying the cause-and-effect relations* among nodes/edges in the graph. Instead, causality-inspired explainers are inspired by causality concepts to infer the explanatory subgraph, but they inherently do not provide causal explanations.

3 Preliminaries

In this section, we provide the necessary background on GNNs and causality to understand this work. For brevity, we will consider GNNs for graph classification.
Notations: We denote a graph as $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the node set and edge set, respectively. $v \in \mathcal{V}(G)$ represents a node and $e_{u,v} \in \mathcal{E}(G)$ is an edge between u and v. Each graph G is associated with a label $y_G \in \mathcal{Y}$, with $\mathcal{Y}$ the label domain. An uppercase letter X and the corresponding lowercase one x indicate a random variable and its value, respectively; bold $\mathbf{X}$ and $\mathbf{x}$ denote a set of random variables and its corresponding values, respectively. We use $\mathrm{Dom}(X)$ to denote the domain of X, and $P(\mathbf{X})$ as a probability distribution over $\mathbf{X}$.

Graph neural network (GNN): A GNN is a multi-layer neural network that operates on the graph and iteratively learns node/graph representations via message passing. A GNN mainly uses two operations to compute node representations in each layer. Assume a node v's representations in the $(l-1)$-th layer is learnt and denoted as $h_v^{(l-1)}$. In the l-th layer, the message between two connected nodes u and v is defined as $m_{u,v}^l = \text{MSG}(h_u^{l-1}, h_v^{l-1}, e_{u,v})$. The aggregated message for node u is then defined as u's representation in the current layer l: $h_u^l = \text{AGG}(m_{u,v}^l : v \in \mathcal{N}_1(u))$. Assume L layers of computation, the final representation for v is $\boldsymbol{z}_v = \boldsymbol{h}_v^{(L)}$ and $\mathbf{Z} = \{\mathbf{z}_v\}_{v \in \mathcal{V}}$. GNN can add a predictor on top of $\mathbf{Z}$ to perform graph-relevant tasks. For instance, when graph classification is the task of interest, GNNs use $\mathbf{Z}$ to predict the label for a whole graph.

Structural Causal Model (SCM): SCMs [23] provide a rigorous definition of cause-effect relations between random variables. An SCM $\mathcal{M}$ is a four-tuple $\mathcal{M} \equiv (\mathbf{U}, \mathbf{V}, \mathcal{F}, P(\mathbf{U}))$, where $\mathbf{U}$ is a set of exogenous (or latent) variables determined by factors outside the model and they are the only source of randomness in an SCM; V is a set $\{V_1, V_2, \ldots, V_n\}$ of n endogenous (or observable) variables of interest determined by other variables within the model, i.e., in $\mathbf{U} \cup \mathbf{V}$; $\mathcal{F}$ is set of functions (define causal mechanisms) $\{f_{V_1}, f_{V_2}, \ldots, f_{V_n}\}$ such that each f_{V_i} is a mapping function from $\mathbf{U}_{V_i} \bigcup \mathbf{Pa}_{V_i}$ to V_i, where $\mathbf{U}_{V_i} \subseteq \mathbf{U}$ and $\mathbf{Pa}_{V_i} \subseteq \mathbf{V} \setminus V_i$ is the parent of V_i. That is, $v_i \leftarrow f_{V_i}(\mathbf{pa}_{V_i}, \mathbf{u}_{V_i})$ and $\mathcal{F}$ forms a mapping from $\mathbf{U}$ to $\mathbf{V}$. $P(\mathbf{U})$ is the probability function over the domain of $\mathbf{U}$. With SCM, one can perform interventions to find causes and effects and design a model that has the capability of predicting the effect of interventions.

Definition 1 (Intervention and Causal Effects). *Interventions are changes made to a system to study the causal effect of a particular variable or treatment on an outcome of interest. An SCM $\mathcal{M}$ induces a set of interventional distributions over $\mathbf{V}$, one for each intervention $do(\mathbf{X} = \mathbf{x})$ (short for $do(\mathbf{x})$), which forces the value of variable $\mathbf{X} \subseteq \mathbf{V}$ to be $\mathbf{x}$. Then for each $\mathbf{Y} \subseteq \mathbf{V}$:*

$$p^{\mathcal{M}}(\mathbf{y}|do(\mathbf{x})) = \sum_{\{\mathbf{u}|\mathbf{Y}_{\mathbf{x}}(\mathbf{u})=\mathbf{y}\}} P(\mathbf{u}). \tag{1}$$

In words, an intervention forcing a set of variables $\mathbf{X}$ to take values $\mathbf{x}$ is modeled by replacing the original mechanism f_X for each $X \in \mathbf{X}$ with its corresponding value in $\mathbf{x}$. The impact of the intervention $\mathbf{x}$ on an outcome variable $\mathbf{Y}$ is called potential response $\mathbf{Y}_{\mathbf{x}}(\mathbf{u})$, which expresses *causal effects* and is the solution for $\mathbf{Y}$ after evaluating: $\mathcal{F}_{\mathbf{x}} := \{f_{V_i} : V_i \in \mathbf{V} \setminus \mathbf{X}\} \cup \{f_X \leftarrow x : X \in \mathbf{X}\}$.

One possible strategy to estimate the underlying SCM of a task is using its observational inputs and outputs. However, a critical issue is that causal properties are *provably impossible* to recover solely from the joint distribution over the input graphs and labels [24]. In this paper, we are inspired by the emerging *Neural Causal Model (NCM)* [43], which is a special type of SCM that is amenable to gradient descent-based optimization.

Neural Causal Model (NCM): A NCM [43] $\widehat{\mathcal{M}}(\theta)$ over variables $\mathbf{V}$ with parameters $\theta = \{\theta_{V_i}; V_i \in V\}$ is an SCM estimation as: $\widehat{\mathcal{M}}(\theta) \equiv (\widehat{\mathbf{U}}, \mathbf{V}, \widehat{\mathcal{F}}, P(\widehat{\mathbf{U}}))$,

where 1) $\widehat{\mathbf{U}} \subseteq \{\widehat{U}_{\mathbf{C}}; \mathbf{C} \subseteq \mathbf{V}\}$, with each $\widehat{U}$ associated with some subset of variables $\mathbf{C} \subseteq \mathbf{V}$, and $\text{Dom}(\widehat{\mathbf{U}}) = [0,1]$; 2) $\widehat{\mathcal{F}} = \{\widehat{f}_{V_i} : V_i \in \mathbf{V}\}$, with each $\widehat{f}_{V_i}$ a neural network parameterized by θ_{V_i} that maps $\widehat{\mathbf{U}}_{V_i} \cup \mathbf{Pa}_{V_i}$ to V_i, and $\widehat{\mathbf{U}}_{V_i} = \{\widehat{U}_{\mathbf{C}} : V_i \in \mathbf{C}\}$; 3) $P(\widehat{\mathbf{U}}) : \widehat{U} \sim \text{Unif}(0,1), \forall \widehat{U} \in \widehat{\mathbf{U}}$.

[43] shows that *NCM is proved to be as expressive as SCM*, and hence *all NCMs are SCMs*. However, expressiveness does not mean the learned NCM model has the same empirical observations as the SCM model. To ensure equivalence, there should be a necessary structural assumption on NCMs, called *causal structure consistency*. More details are referred to [43] and Appendix B.

4 GNN Causal Explanation via NCMs

In this section, we propose our GNN causal explainer, CXGNN, for explaining graph classification. Our explainer also utilizes the common observation that a graph consists of a causal subgraph and a non-causal counterpart [7,19,20,32,38, 41]. The overview of CXGNN is shown in Figure 10 in Appendix and all proofs are deferred to Appendix C.

4.1 Overview

Given a graph $G = (\mathcal{V}, \mathcal{E})$ and a ground truth or predicted label by a GNN model, our causal explainer bases on causal learning and identifies the *causal explanatory subgraph* (denoted as Γ) that intrinsically yields the label.

Our CXGNN consists of three key steps: 1) define the causal structure $\mathcal{G}$ for the graph G and the respective SCM $\widehat{\mathcal{M}}(\mathcal{G})$ (we call GNN-SCM) to enable causal effect calculation via interventions; 2) However, directly calculating the causal effect in real graphs is computationally challenging. We then construct and train a family of parameterized GNN neural causal model $\widehat{\mathcal{M}}(\mathcal{G}, \theta))$ (we call GNN-NCM), a special type of GNN-SCM that is trainable. 3) We uncover the causal explanatory subgraph (denoted as Γ) based on the trained GNN-NCM that best yields the graph label. Next, we will illustrate step-by-step in detail.

4.2 Causal Structure and Induced SCM on a Graph

In the context of causality, the problem of GNN explanation can be solved by cause and effect identification among nodes and their connections in a graph. Interventions enable us to interpret the causal relation between nodes. To perform interventions on a graph, one often needs to first define the causal structure for this graph, which involves the observable and latent variables.

Observable and latent variables in a graph: Given a $G = (\mathcal{V}, \mathcal{E})$. For each node $v \in \mathcal{V}$, there are both known and unknown effects from other nodes and edges on v, which we call observable variables (denoted as $\mathbf{V}_v$) and latent variables (denoted as $\mathbf{U}_v$), respectively. With it, we define a congruent causal structure for enabling the graphs to admit SCMs.

Definition 2 (Causal structure of a graph). *Consider a graph $G = (\mathcal{V}, \mathcal{E})$, we define the causal structure $\mathcal{G}$ of G as a subgraph that centers on a reference node v and accepts the SCM structure:*

$$\mathcal{G}(G) = \Big\{\mathbf{V}_v = \{y_v\} \cup \{y_{v_i} : v_i \in \mathcal{N}_{\leq k}(v)\}, \mathbf{U}_v = \{\mathbf{U}_{v_i} : v_i \in \mathcal{N}_{\leq k}(v)\} \cup \{\mathbf{U}_{v,v_i} : e_{v,v_i} \in \mathcal{E}\}\Big\}, \tag{2}$$

where v is to be learnt (see Section 4.4), y_{v_i} is the node v_i's label, $\mathcal{N}_{\leq k}(v)$ means nodes within the k-hop neighbors of v, $\mathbf{U}_{v_i}$ is v_i's latent variable, called node effect*; and $\mathbf{U}_{v,v_i}$ the edge e_{v,v_i} latent variable, called* edge effect*. In practice, we can specify $\mathbf{U}_{v_i}$ and $\mathbf{U}_{v,v_i}$ as random variable, e.g., from a Gaussian distribution.*

With a causal structure for a graph, we can build the corresponding SCM in the following theorem:

Theorem 1 (GNN-SCM). *For a GNN operating on a graph G, there exists an SCM $\mathcal{M}(\mathcal{G})$ w.r.t. the causal structure $\mathcal{G}$ of the graph G.*

Appendix A shows an example on how to compute the causal effects on a toy graph via a SCM truth table.

4.3 GNN Neural Causal Model

In reality, it is computationally challenging to build a truth table for variables in GNN-SCM and perform do-calculus computation due to the large number of nodes/edges in real-world graphs. Such a challenge impedes the calculation of causal effects. To address it, we are motivated by estimating the causal effect via NCM (see Section 3). Specifically, Definition 6 in Appendix shows: to ensure the equivalence between NCM and SCM, NCM is required to be $\mathcal{G}$-constrained. However, the general $\mathcal{G}$-constrained NCM cannot be directly applied in our setting. To this end, we first define a customized $\mathcal{G}$-constrained GNN-NCM as below:

Definition 3 ($\mathcal{G}$-Constrained GNN-NCM (constructive)). *Let GNN-SCM $\mathcal{M}(\mathcal{G}, \theta)$ be induced from the causal structure $\mathcal{G}(G)$ on a graph G. Then GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ will be constructed based on the causal structure $\mathcal{G}(G)$.*

This construction ensures that any inferences made by $\widehat{\mathcal{M}}_{NCM}(\mathcal{G}, \theta)$ respect the causal dependencies as captured by $\mathcal{G}(G)$. Note that $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ represents a family of GNN-NCMs since the parameters θ of the neural networks are not specified by the construction. Next, we propose a construction of a $\mathcal{G}$-constrained GNN-NCM, following Definition 3.

GNN Neural Causal Model Construction One should consider the sound and complete structure of GNN-NCMs that are consistent with Definition 2. Here, we define the general GNN-NCM structure as shown in below Equation 3, which is an instantiation of Theorem 2.

$$\widehat{\mathcal{M}}(\mathcal{G}, \theta) = \begin{cases} \mathbf{V} := \mathcal{V}(\mathcal{G}) \\ \widehat{\mathbf{U}} := \{\widehat{\mathbf{U}}_{v_i}, v_i \in \mathcal{V}(\mathcal{G}), P(\widehat{\mathbf{U}}) := \{\widehat{\mathbf{U}}_{v_i} \sim \text{Unif}(0,1)\} \cup \{T_{k,v_i} \sim \mathcal{N}(0,1) : k \in \{0,1\}\} \\ \widehat{f}_{v_i}(\widehat{\mathbf{u}}_{v_i}, \widehat{\mathbf{u}}_{v_i,v_j}) := \arg\max_{k \in \{0,1\}} T_{k,v_i} + \begin{cases} \log \sigma(ff_{v_i}(\widehat{\mathbf{u}}_{v_i}, \widehat{\mathbf{u}}_{v_i,v_j}; \theta_{v_i})) \text{ if } k = 1 \\ \log(1 - \sigma(ff_{v_i}(\widehat{\mathbf{u}}_{v_i}, \widehat{\mathbf{u}}_{v_i,v_j}; \theta_{v_i}))) \text{ if } k = 0, \end{cases} \\ \widehat{\mathcal{F}} := \{\widehat{f}_{v_i}(\widehat{\mathbf{u}}_{v_i}, \widehat{\mathbf{u}}_{v_i,v_j})\} \end{cases} \tag{3}$$

Algorithm 1 GNN Neural Causal Model Training

Input: The causal structure $\mathcal{G}$ (including a reference node v, its within k-hop neighbors $\mathcal{N}_{\leq k}(v)$, and set of latent variables $\mathbf{U}_v$), node label y_v

Output: An optimized GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta^*)$ for the causal structure $\mathcal{G}$ centered at v

1: Build the GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ based on $\mathcal{G}$ and Eqn. 3
2: **for** each node $v_i \in \mathcal{N}_{\leq k}(v)$ **do**
3: Calculate $p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v \mid do(v_i))$ via Eqn. 4
4: **end for**
5: Calculate $p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v)$ via Eqn. 5
6: Calculate the loss $\mathcal{L}(\widehat{\mathcal{M}}(\mathcal{G}, \theta); v)$ via Eqn. 6
7: Minimize the loss to reach the GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta^*)$

Theorem 2 (GNN-NCM). *Given causal structure $\mathcal{G}$ of a graph G and the underlying GNN-SCM $\mathcal{M}(\mathcal{G})$, there exists a $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ that enables any inferences consistent with $\mathcal{M}(\mathcal{G})$.*

In Equation 3, $\mathbf{V}$ are the nodes in the causal structure $\mathcal{G}(G)$; each T_{v_i} is a standard Gaussian random variable; each ff_{v_i} is a feed-forward neural network on v_i parameterized by θ_{v_i} (note one requirement of ff_{v_i} is it could approximate any continuous function), and σ is sigmoid activation function. The parameters $\{\theta_{v_i}\}$ are not yet specified and must be learned through training the NCM.

Training Neural Networks for GNN-NCMs We now compute the causal effects on a target node v. Based on Definition 1 and the constructed GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ in Equation 3, the causal effect on v of an intervention $do(v_i)$ ($v_i \in \mathcal{N}_1(v)$) is $p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v|do(v_i))$. This do-calculus then can be calculated as the expected value of nodes and edges affects values for v shown below:

$$p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v \mid do(v_i)) = \mathbb{E}_{p(\hat{\mathbf{u}}_v)}\Big[\prod_{(v,v_j)\in\mathcal{E}(\mathcal{G})} \widehat{f}_{v_i}(\hat{\mathbf{u}}_{v_j}, \hat{\mathbf{u}}_{v,v_j})\Big]$$
$$\approx \frac{1}{|\mathcal{N}_{\leq k}(v)|} \sum_{v_i \in \mathcal{N}_{\leq k}(v)} \prod_{(v,v_j)\in\mathcal{E}(\mathcal{G})} \widehat{f}_{v_i}(\hat{\mathbf{u}}_{v_j}, \hat{\mathbf{u}}_{v,v_j}). \quad (4)$$

Then one can calculate the probability of the target node label y_v as the expected value of all the effects from the neighbor nodes on v:

$$p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v) = \mathbb{E}_{p(\hat{\mathbf{u}}_v)}\left[\widehat{f}_v\right] \approx \frac{1}{|\mathcal{N}_1(v)|}\frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}}\sum_{v_i\in\mathcal{N}_1(v)} p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v = y \mid do(v_i)) \quad (5)$$

The true GNN-SCM induces a causal structure that encodes constraints over the interventional distributions. We now first investigate the feasibility of causal inferences in the class of $\mathcal{G}$-constrained GNN-NCMs. These models approximate the likelihood of the observed data based on the graph's latent variables. The cross-entropy loss measures the discrepancy between the target node's label prediction and its true label. Inspired by [43], we define the GNN-NCM loss as:

$$\mathcal{L}(\widehat{\mathcal{M}}(\mathcal{G}, \theta); v) = -\sum_{y_v\in\mathcal{Y}} y_v \log(p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v)) \quad (6)$$

Algorithm 2 CXGNN: GNN Causal Explainer

Input: Graph G with label, and expressivity threshold δ
Output: Explanatory subgraph Γ

1: **for** each node $v \in \mathcal{V}(G)$ **do**
2: Build $\mathcal{G}_v$ based on the reference node v
3: Train the GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}_v, \theta_v^*)$ via Alg. 1 and calculate the node expressivity $\exp_v(\widehat{\mathcal{M}}(\mathcal{G}_v, \theta_v^*))$
4: **end for**
5: Find $v^* = \text{argmax}_{v \in \mathcal{V}(G)} \exp_v(\widehat{\mathcal{M}}(\mathcal{G}_v, \theta_v^*))$;
6: Return the explanatory subgraph Γ induced by $\mathcal{G}_{v^*}$

To train neural networks for GNN-NCMs, one should generate samples from the GNN-SCM. If provided, it is the specific realization of the interventions. Specifically, GNN-NCMs are trained on node effects $\hat{\mathbf{u}}_{v_i}$ and edge effects $\hat{\mathbf{u}}_{v_i,v_j}$ on the target node, as shown in Equation 3, and should specify $\widehat{f}_{v_i}(\hat{\mathbf{u}}_{v_i}, \hat{\mathbf{u}}_{v_i,v_j})$. Then a model, denoted as θ^*, is achieved by minimizing the GNN-NCM loss:

$$\theta^* \in \arg\min_{\theta} \mathcal{L}(\widehat{\mathcal{M}}(\mathcal{G}, \theta); v) \tag{7}$$

Details of training GNN-NCMs are shown in Algorithm 1. Basically, this algorithm takes the causal structure $\mathcal{G}$ with respect to a reference node v as input and returns an optimized GNN-NCM model $\widehat{\mathcal{M}}(\mathcal{G}, \theta^*)$.

4.4 Realizing GNN Causal Explanation

The remaining question is: how to find the causal explanatory subgraph Γ from a graph G to causally explain GNN predictions? The answer is using the trained GNN-NCMs $\widehat{\mathcal{M}}(\mathcal{G}, \theta^*)$. Before that, the first step is to clarify a node's role in GNN-NCMs for explanation.

Theorem 3 (Node explainability). *Let a prediction for a graph G be explained. A node $v \in G$ is causally explainable, if $p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v)$ can be computed.*

The $\mathcal{G}$-constrained GNN-NCM is trained on interventions and can interpret the GNN predictions. Moreover, the information extracted from interventions can be used for interpreting nodes. Specifically, we define expressivity to measure the information for an explainable node.

Theorem 4 (Explainable node expressivity). *An explainable node v has expressivity defined as $exp_v(\widehat{\mathcal{M}}(\mathcal{G}, \theta)) = \sum_{y_v} y_v p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v)$.*

In other words, the node expressivity reflects how well the node is in the causal explanatory subgraph. Now we are ready to realize GNN causal explanation based on learned GNN-NCMs. Given a graph G, we start from a random node v, and build the causal structure $\mathcal{G}$ centered on v. By Algorithm 1, we can reach an optimized GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta^*)$ and obtain the v's expressivity.

We repeat this process for all nodes in the graph G and find the node v^* with the associated $\widehat{\mathcal{M}}(\mathcal{G}, \theta^*)$ yielding the highest expressivity $\exp_{v^*}(\widehat{\mathcal{M}}(\mathcal{G}, \theta^*))$. The underlying subgraph of the causal structure centered by v^* is then treated as the causal explanatory subgraph Γ. Algorithm 2 describes the learning process.

Table 1: Dataset statistics.

	Avg. #nodes	Avg. #edges	#test graphs
BA+House	11.97	18.17	500
BA+Grid	15.96	24.20	500
BA+Cycle	10.0	10.5	500
Tree+House	12	13	500
Tree+Cycle	13	13.50	500
Tree+Grid	24	27	500
Benzene	20.48	21.73	100
Fluoride carbonyl	20.66	22.03	100

5 Experiments

5.1 Experimental Setup

Datasets: Following prior works [19, 46], we use six synthetic datasets, and two real-world datasets with groundtruth explanation for evaluation. Dataset statistics are shown in Table 1.

- *Synthetic graphs*: **1) BA+House:** This graph stems from a base random Barabási-Albert (BA) graph attached with a 5-node "house"-structured motif as the groundtruth explanation; **2) BA+Grid:** This graph contains a base random BA graph and is attached with a 9-node "grid" motif as the groundtruth explanation; **3) BA+Cycle:** A 6-node "cycle" motif is appended to randomly chosen nodes from the base BA graph. The "cycle" motif is the groundtruth explanation; **4) Tree+House:** The core of this graph is a balanced binary tree. The 5-node "house" motif, as the groundtruth explanation, is attached to random nodes from the base tree. **5) Tree+Grid:** Similarly, binary tree a the core graph and a 9-node "grid" motif as the groundtruth explanation is attached; **6) Tree+Cycle:** A 6-node "cycle" motif, the groundtruth explanation, is appended to nodes from the binary tree. The label of the synthetic graph is decided by the label of nodes in the groundtruth explanation. Following existing works [19, 46], a node v's label y_v is set to be 1 if v is in the groundtruth, and 0 otherwise. Hence, in these graphs, the base graph acts as the non-causal subgraph that can cause the spurious correlation, while the attached motif can be seen as the causal subgraph, as it does not change across graphs and decides the graph label.
- *Real-world graphs:* We use two representative real-world graph datasets with groundtruth [1]. **1) Benzene:** it includes 12,000 molecular graphs extracted from the ZINC15 [31] database and the task is to identify whether a given molecule graph has a benzene ring or not. The groundtruth explanations are the nodes (atoms) forming the benzene ring. **2) Fluoride carbonyl:** This dataset contains 8,671 molecular graphs with two classes: a positive class means a molecule graph contains a fluoride (F-) and a carbonyl (C=O) functional group. The groundtruth explanation consists of combinations of fluoride atoms and carbonyl functional groups within a given molecule.

Models and parameter setting: In CXGNN, we use a feedforward neural network to parameterize GNN-NCM. The neural network consists of an input layer, two fully connected hidden layers, and an output layer. ReLU activation functions is used in all hidden layers, while a softmax activation function is applied to the output layer. The input to the network is the target node v's node effects and edge effects (see Equation 2), whose values are sampled from a standard Gaussian distribution, and the output is the predicted causal effect on v. The detailed hyperparameters are shown in Appendix D.1. The hyperparameters in the compared GNN explainers are optimized based on their source code.
Baseline GNN explainers: We compare CXGNN with both association-based and causality-inspired GNN explainers. We choose 4 representative ones: gradient-based Guidedbp [12], perturbation-based GNNExplainer [46], surrogate-based PGMExplainer [33], and causality-inspired GEM [19], RCExplainer [38], and OrphicX [20]. We use the public source code of these explainers for comparison. The causality-inspired explainers are inspired by causality concepts to infer the explanatory subgraph, but they inherently do not provide causal explanations.
Evaluation metrics: Given a set of testing graphs $\mathbb{G}$. For each test graph $G \in \mathbb{G}$, we let its groundtruth explanatory subgraph be Γ_G and the estimated explanatory subgraph by a GNN explainer be Γ. We use two common metrics, i.e., graph explanation accuracy and explanation recall from the literature [1]. In addition, to justify the superiority of our causal explainer, we introduce a third metric groundtruth match accuracy, which is the most challenging one.

- **Graph explanation accuracy:** For a graph G, the graph explanation accuracy is defined as the fraction of nodes in the estimated explanatory subgraph Γ that are contained in the groundtruth Γ_G, i.e., $|V(\Gamma) \cap V(\Gamma_G)|/|V(\Gamma_G)|$. We then report the average accuracy across all testing graphs.
- **Graph explanation recall:** Different GNN explainers output the estimated explanatory subgraph with different node sizes. When two explainers output the same number of nodes in Γ_G, the one with a smaller node size should be treated as having a better quality. To account for this, we use the explanation recall metric that is defined as $|V(\Gamma) \cap V(\Gamma_G)|/|V(\Gamma)|$ for a given graph G. We then report the average recall across all testing graphs.
- **Groundtruth match accuracy:** For a testing graph G, we count a 1 if the estimated Γ and groundtruth Γ_G exactly match, i.e., $\Gamma_G = \Gamma$, and 0 otherwise. In other words, the groundtruth match accuracy of all testing graphs $\mathbb{G}$ is defined as $\sum_{G \in \mathbb{G}} \mathbf{1}[\Gamma_G = \Gamma]/|\mathbb{G}|$, where $\mathbf{1}[\cdot]$ is an indicator function.

5.2 Results on Synthetic Datasets

Comparison results: Table 2 shows the results of all the compared GNN explainers on the 6 synthetic datasets with 500 testing graphs and 3 metrics. We have several observations. In terms of explanation accuracy, CXGNN performs comparable or slightly worse than causality-inspired methods. This is because, to ensure high accuracy, the estimated explanatory subgraph of these methods should have a large size. This can be reflected by the explanation recall, where

Table 2: Comparison results on the synthetic datasets. B.H.: BA+House; B.G.: BA+Grid; B.C.: BA+Cycle; T.H.: Tree+House; T.G.: Tree+Grid; T.C.: Tree+Cycle.

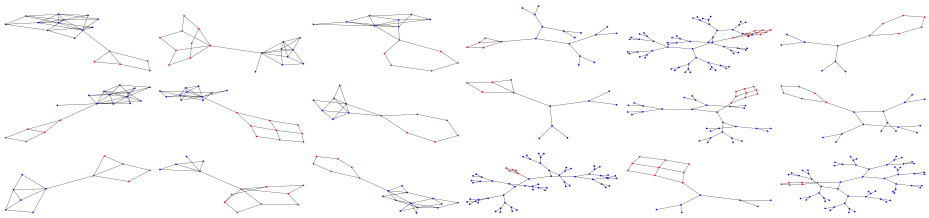
Graph explanation accuracy (%)						
	B.H.	**B.G.**	**B.C.**	**T.H.**	**T.C.**	**T.G.**
GNNExp. [46]	75.60	76.16	75.13	77.24	71.60	72.18
PGMExp. [33]	61.60	44.98	63.07	58.28	49.90	37.42
Guidedbp [12]	60.00	0.00	66.67	0.00	0.00	0.00
GEM [19]	98.2	88.19	97.91	96.23	95.51	86.96
RCExp. [38]	**100.00**	88.89	**100.00**	**100.00**	**100.00**	**100.00**
OrphicX [20]	88.00	89.00	55.65	96.20	**100.00**	99.93
CXGNN	**100.0**	**100.00**	83.33	**100.0**	82.67	**100.00**
Graph explanation recall (%)						
	B.H.	**B.G.**	**B.C.**	**T.H.**	**T.C.**	**T.G.**
GNNExp. [46]	37.62	52.72	45.08	32.18	33.05	40.60
PGMExp. [33]	30.80	31.14	37.84	24.28	23.93	21.05
Guidedbp [12]	12.40	17.94	16.18	5.99	12.98	15.38
GEM [19]	39.18	50.86	45.40	38.75	34.65	41.20
RCExp. [38]	100.00	60.00	89.52	45.45	46.60	39.13
OrphicX [20]	98.08	**97.71**	60.00	41.38	59.22	40.61
CXGNN	**100.0**	60.55	**90.00**	**61.67**	**68.15**	**49.05**
Groundtruth match accuracy (%)						
	B.H.	**B.G.**	**B.C.**	**T.H.**	**T.C.**	**T.G.**
GNNExp. [46]	0.20	2.20	2.20	0.80	0.40	0.20
PGMExp. [33]	1.00	0.00	0.00	3.80	0.00	0.00
Guidedbp [12]	1.00	0.6	0.6	0.6	0.2	0.6
GEM [19]	0.80	6.00	6.00	2.50	1.20	1.00
RCExp. [38]	100.00	0.00	49.60	0.00	0.00	0.00
OrphicX [20]	39.00	43.00	5.00	1.40	21.00	33.00
CXGNN	**100.0**	**44.00**	**67.60**	**99.40**	**61.20**	**46.00**

explanation recall is significantly reduced. Overall, the causality-inspired methods obtain higher accuracies than purely association-based methods.

More importantly, CXGNN drastically outperforms all the compared GNN explainers in terms of groundtruth match. Such a big difference demonstrates all the association-based and causality-inspired GNN explainers are insufficient to uncover the exact groundtruth. The is due to existing GNN explainers inherently learning from *correlations* among nodes/edges in the graph, and capturing spurious correlations. Instead, our causal explainer CXGNN can do so much more accurately. This verifies the causal explainer indeed can intrinsically uncover the causal relation between the explanatory subgraph and the graph label.

Visualization results: Figure 1 visualizes the explanations results of some testing graphs in the four synthetic datasets. We note that there are different ways for the groundtruth subgraph to attach to the base synthetic graph. We can see CXGNN's output exactly matches the groundtruth in these cases, while the existing GNN explainers cannot. One reason could be that existing GNN explainers are sensitive to the spurious relation.

Loss curve: Figure 2 shows the loss curves to train our GNN-NCM on a set of nodes, where some nodes are in the groundtruth and some are not from



(a) BA+House **(b)** BA+Grid **(c)** BA+Cycle **(d)** Tree+House **(e)** Tree+Grid **(f)** Tree+Cycle

Fig. 1: Visualizing explanation results (subgraph containing the red nodes) by our CXGNN on synthetic graphs.

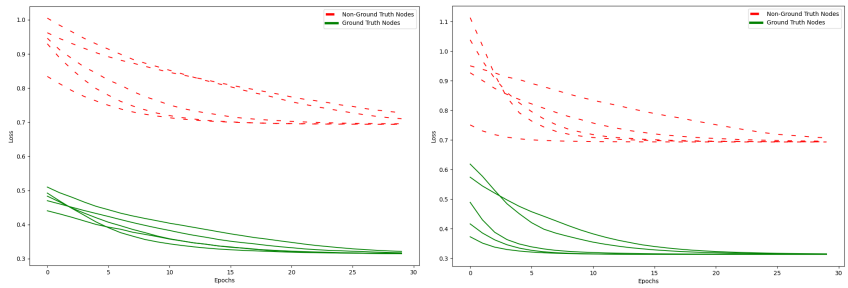


Fig. 2: Loss curves of training the GNN-NCMs on the groundtruth nodes (green curves) and non-groundtruth ones (red curves) on two random chosen graphs from BA+House. More examples in other datasets are shown in Appendix D.

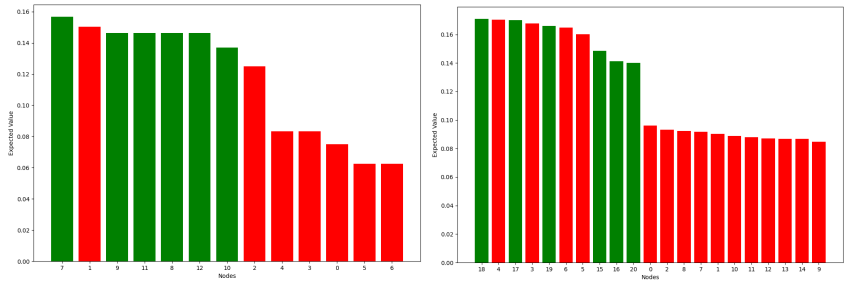


Fig. 3: Node expressivity distributions on two unsuccessful graphs from BA+Cycle. Green bars correspond to nodes that are in the groundtruth, while red bars correspond to nodes that are not. More examples in other datasets are shown in Appendix D.

BA+House. We can see the loss decreases stably for groundtruth nodes, while the loss for nodes not from the groundtruth are relatively high. This reflects our designed GNN-NCM makes it easier to learn groundtruth nodes. That being said, CXGNN indeed tends to find the causal subgraph.

Node expressivity distribution: We notice CXGNN still misses finding the groundtruth explanatory subgraph for some graphs. One possible reason could be that, theoretically, our GNN-SCM can always uncover the causal subgraph, but practically, it is challenging to train the optimal one. Here, we randomly

Table 3: Comprehensive comparison results on the real-world datasets.

	Exp. Acc. (%)		Exp. Recall (%)		GT Match Acc. (%)	
Method	**Benzene**	**F.C.**	**Benzene**	**F.C.**	**Benzene**	**F.C.**
GNNExp. [46]	66.05	44.44	18.88	14.42	0.00	0.00
PGMExp. [33]	33.33	17.78	7.51	4.98	0.00	0.00
Guidedbp [12]	0.00	0.00	9.06	8.00	0.00	0.00
GEM [19]	71.98	46.22	19.80	14.57	0.00	0.00
RCExp. [38]	0.20	0.05	10.85	2.01	0.00	0.00
OrphicX [20]	47.63	11.14	30.31	10.01	3.40	5.50
CXGNN	**73.46**	**66.67**	**21.35**	**16.43**	**66.67**	**75.00**

Fig. 4: Explanation results (subgraph containing the yellow nodes) by our CXGNN on real-world graphs. The left and right two graphs are in Benzene and F.C., respectively.

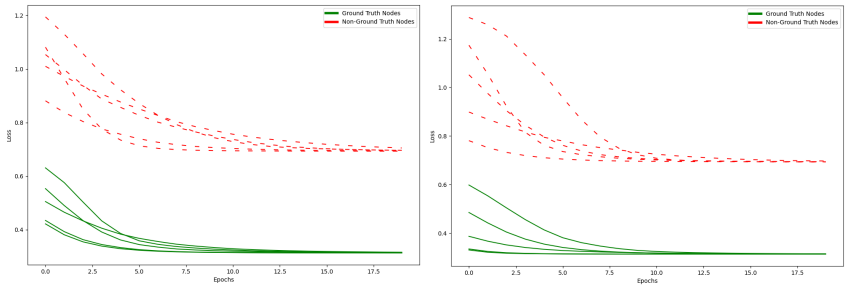


Fig. 5: Loss curves of training the GNN-NCMs on the groundtruth nodes (green curves) and non-groundtruth ones (red curves) on two graphs from the two real-world datasets, respectively. More examples are shown in Appendix D.

select 2 such unsuccessful graphs in BA+Cycle and plot their distributions on the node expressivity in Figure 3. We observe that, though the groundtruth nodes are not always having the best expressivity, they are still at the top.

5.3 Results on Real-World Datasets

Comparison results: Table 3 shows the results of all the compared explainers on the real-world datasets and three metrics. We have similar observations as those in Table 2. Especially, no existing explainers can even find one exactly matched groundtruth. Particularly, the explanation subgraphs produced by the two causality-inspired baselines can cover the majority or almost all groundtruth in synthetic datasets (hence high accuracy), and the sizes of the explanation subgraphs are slightly larger than those of the groundtruth (hence relatively large

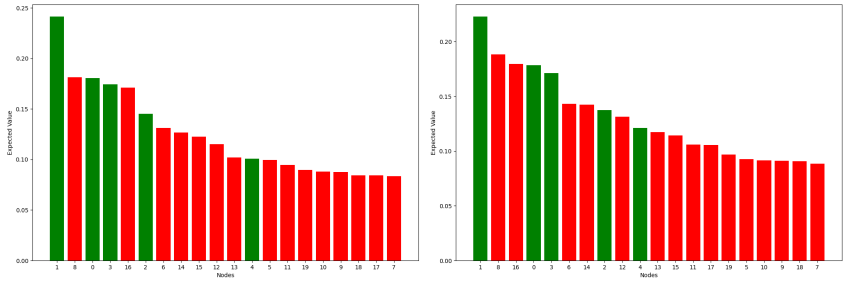


Fig. 6: Node expressivity distributions on two unsuccessful graphs from the real-world datasets, respectively. Green bars correspond to nodes that are in the groundtruth, while red bars correspond to nodes that are not. More examples are in Appendix D.

recall). However, the causality-inspired baselines are not good at exactly matching the groundtruth, i.e., groundtruth match accuracy is low overall. Note also that the exact match of CXGNN is also largely reduced (about 30%). One possible reason is that the groundtruth explanation in real-world graphs is not easy to define or even inaccurate. For instance, in MUTAG, both NO_2 and NH_2 motifs are considered as the "mutagenic" groundtruth in the literature. However, [19] found 32% of non-mutagenic graphs contain NO_2 or NH_2, implying inaccurate groundtruth. Here, we propose to also use an approximate groundtruth match accuracy, where we require the estimated subgraph to be a subset and its size is no less than 60% of the groundtruth. With this new alternative metric, its value is much larger (i.e., 67% and 75%) on the two datasets.

Visualization results: Figure 4 visualizes the explanation results of some graphs in the real-world datasets. We observe the explanatory subgraphs found by CXGNN approximately/exactly match the groundtruth.

Loss curve: Figure 5 shows the loss curves to train our GNN-NCM on a set of groundtruth and non-groundtruth ones. Similarly, the loss decreases stably for groundtruth nodes, while not for non-groundtruth ones. Again, this implies our GNN-NCM tends to find the causal subgraph.

Node expressivity distribution: We randomly select some unsuccessful graphs in real-world datasets and plot their distributions on the node expressivity in Figure 6. Still, though the groundtruth nodes do not always achieve the best expressivity, they are at the top.

6 Conclusion

GNN explanation, i.e., identifying the informative subgraph that ensures a GNN makes a particular prediction for a graph, is an important research problem. Though various GNN explainers have been proposed, they are shown to be prone to spurious correlations. We propose a *causal* GNN explainer based on the fact that a graph often consists of a causal subgraph and fulfills the goal via causal inference. We then propose to train GNN neural causal models to uncover the causal explanatory subgraph. In future work, we will study the robustness of our CXGNN under the adversarial graph perturbation attacks [17, 22, 34–37, 45].

Acknowledgements

We thank all the anonymous reviewers for their valuable feedback and constructive comments. Behnam and Wang are partially supported by the Amazon Research Award, the National Science Foundation (NSF) under Grant Nos. 2216926, 2241713, 2331302, and 2339686. Any opinions, findings conclusions, and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References

1. Agarwal, C., Queen, O., Lakkaraju, H., Zitnik, M.: Evaluating explainability for graph neural networks. Scientific Data **10**(1), 144 (2023)
2. Ay, N., Polani, D.: Information flows in causal networks. Advances in complex systems **11**(01), 17–41 (2008)
3. Baldassarre, F., Azizpour, H.: Explainability techniques for graph convolutional networks. arXiv preprint arXiv:1905.13686 (2019)
4. Beckers, S.: Causal explanations and xai. In: Conference on Causal Learning and Reasoning. pp. 90–109. PMLR (2022)
5. Duval, A., Malliaros, F.D.: Graphsvx: Shapley value explanations for graph neural networks. In: Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II 21. pp. 302–318. Springer (2021)
6. Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking graph neural networks. Journal of Machine Learning Research (2023)
7. Fan, S., Wang, X., Mo, Y., Shi, C., Tang, J.: Debiasing graph neural networks via learning disentangled causal substructure. Advances in Neural Information Processing Systems **35**, 24934–24946 (2022)
8. Feng, Q., Liu, N., Yang, F., Tang, R., Du, M., Hu, X.: Degree: Decomposition based explanation for graph neural networks. arXiv preprint arXiv:2305.12895 (2023)
9. Funke, T., Khosla, M., Rathee, M., Anand, A.: Zorro: Valid, sparse, and stable explanations in graph neural networks. IEEE Transactions on Knowledge and Data Engineering (2022)
10. Geiger, A., Wu, Z., Lu, H., Rozner, J., Kreiss, E., Icard, T., Goodman, N., Potts, C.: Inducing causal structure for interpretable neural networks. In: International Conference on Machine Learning. pp. 7324–7338. PMLR (2022)
11. Granger, C.W.: Investigating causal relations by econometric models and cross-spectral methods. Econometrica: journal of the Econometric Society pp. 424–438 (1969)
12. Gu, J., Tresp, V.: Saliency methods for explaining adversarial attacks. arXiv preprint arXiv:1908.08413 (2019)
13. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems. pp. 1024–1034 (2017)
14. Huang, Q., Yamada, M., Tian, Y., Singh, D., Chang, Y.: Graphlime: Local interpretable model explanations for graph neural networks. IEEE Transactions on Knowledge and Data Engineering (2022)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
17. Li, J., Pang, M., Dong, Y., Jia, J., Wang, B.: Graph neural network explanations are fragile. In: Proceedings of the 41 st International Conference on Machine Learning (2024)
18. Li, W., Li, Y., Li, Z., Hao, J., Pang, Y.: Dag matters! gflownets enhanced explainer for graph neural networks. arXiv preprint arXiv:2303.02448 (2023)
19. Lin, W., Lan, H., Li, B.: Generative causal explanations for graph neural networks. In: International Conference on Machine Learning. pp. 6666–6679. PMLR (2021)
20. Lin, W., Lan, H., Wang, H., Li, B.: Orphicx: A causality-inspired latent variable model for interpreting graph neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13729–13738 (2022)
21. Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., Zhang, X.: Parameterized explainer for graph neural network. Advances in neural information processing systems **33**, 19620–19631 (2020)
22. Mu, J., Wang, B., Li, Q., Sun, K., Xu, M., Liu, Z.: A hard label black-box adversarial attack against graph neural networks. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS) (2021)
23. Pearl, J.: Causality. Cambridge university press (2009)
24. Pearl, J., Glymour, M., Jewell, N.P.: Causal inference in statistics: A primer. John Wiley & Sons (2016)
25. Pearl, J., Mackenzie, D.: The book of why: the new science of cause and effect. Basic books (2018)
26. Pereira, T., Nascimento, E., Resck, L.E., Mesquita, D., Souza, A.: Distill n'explain: explaining graph neural networks using simple surrogates. In: International Conference on Artificial Intelligence and Statistics. pp. 6199–6214. PMLR (2023)
27. Pope, P.E., Kolouri, S., Rostami, M., Martin, C.E., Hoffmann, H.: Explainability methods for graph convolutional neural networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10772–10781 (2019)
28. Schlichtkrull, M.S., Cao, N.D., Titov, I.: Interpreting graph neural networks for NLP with differentiable edge masking. In: International Conference on Learning Representations (2021), `https://openreview.net/forum?id=WznmQa42ZAx`
29. Schwab, P., Karlen, W.: Cxplain: Causal explanations for model interpretation under uncertainty. Advances in neural information processing systems **32** (2019)
30. Shan, C., Shen, Y., Zhang, Y., Li, X., Li, D.: Reinforcement learning enhanced explainer for graph neural networks. Advances in Neural Information Processing Systems **34**, 22523–22533 (2021)
31. Sterling, T., Irwin, J.J.: Zinc 15–ligand discovery for everyone. Journal of chemical information and modeling **55**(11), 2324–2337 (2015)
32. Sui, Y., Wang, X., Wu, J., Lin, M., He, X., Chua, T.S.: Causal attention for interpretable and generalizable graph classification. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 1696–1705 (2022)
33. Vu, M., Thai, M.T.: Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. Advances in neural information processing systems **33**, 12225–12235 (2020)
34. Wang, B., Gong, N.Z.: Attacking graph-based classification via manipulating the graph structure. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS) (2019)

35. Wang, B., Jia, J., Cao, X., Gong, N.Z.: Certified robustness of graph neural networks against adversarial structural perturbation. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 1645–1653 (2021)
36. Wang, B., Li, Y., Zhou, P.: Bandits for structure perturbation-based black-box attacks to graph neural networks with theoretical guarantees. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
37. Wang, B., Pang, M., Dong, Y.: Turning strengths into weaknesses: A certified robustness inspired attack framework against graph neural networks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
38. Wang, X., Wu, Y., Zhang, A., Feng, F., He, X., Chua, T.S.: Reinforced causal explainer for graph neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(2), 2297–2309 (2022)
39. Wang, X., Wu, Y., Zhang, A., He, X., Chua, T.S.: Towards multi-grained explainability for graph neural networks. Advances in Neural Information Processing Systems **34**, 18446–18458 (2021)
40. Wang, X., Shen, H.W.: Gnninterpreter: A probabilistic generative model-level explanation for graph neural networks. arXiv preprint arXiv:2209.07924 (2022)
41. Wu, Y.X., Wang, X., Zhang, A., He, X., Chua, T.S.: Discovering invariant rationales for graph neural networks. arXiv preprint arXiv:2201.12872 (2022)
42. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems **32**(1), 4–24 (2020)
43. Xia, K., Lee, K.Z., Bengio, Y., Bareinboim, E.: The causal-neural connection: Expressiveness, learnability, and inference. Advances in Neural Information Processing Systems **34**, 10823–10836 (2021)
44. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: International Conference on Learning Representations (2019)
45. Yang, H., Wang, B., Jia, J., et al.: Gnncert: Deterministic certification of graph neural networks against adversarial perturbations. In: The Twelfth International Conference on Learning Representations (2024)
46. Ying, Z., Bourgeois, D., You, J., Zitnik, M., Leskovec, J.: GNNExplainer: Generating explanations for graph neural networks. In: Advances in Neural Information Processing Systems (2019)
47. Yuan, H., Tang, J., Hu, X., Ji, S.: Xgnn: Towards model-level explanations of graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 430–438 (2020)
48. Yuan, H., Yu, H., Wang, J., Li, K., Ji, S.: On explainability of graph neural networks via subgraph explorations. In: Proceedings of the 38th International Conference on Machine Learning (ICML). pp. 12241–12252 (2021)
49. Zhang, S., Liu, Y., Shah, N., Sun, Y.: Gstarx: Explaining graph neural networks with structure-aware cooperative games. Advances in Neural Information Processing Systems **35**, 19810–19823 (2022)
50. Zhang, Y., Defazio, D., Ramesh, A.: Relex: A model-agnostic relational model explainer. In: Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society. pp. 1042–1049 (2021)

Appendix

A A GNN-SCM Example

The causal structure of a graph is a subgraph centering on a reference node and accepts the SCM structure via Definition 2. The goal of causality in a graph is to identify the subgraph with the maximum explainable node expressivity as Theorem 4 that causally explains GNN predictions. Below, we use a toy example graph to show how our explainer captures the causality in this graph.

We use a toy example to demonstrate SCMs and the intervention process in GNNs. Figure 7 shows a graph G that contains four nodes A, B, C, and D, and three edges A-B, A-C, and B-D. In GNNs, these edges contain messages passing between two nodes. For example, the message between two nodes A and B in the l-th layer of the GNN is $m^l_{A,B} = \text{MSG}(h^{l-1}_A, h^{l-1}_B, e_{A,B})$. If there exists an edge, it means that there is an interaction between nodes that has a specific value in each layer l. If there is no edge, two nodes don't share a message. If we consider node A as the reference node v, the nodes B, and C are in the 1-hop neighbors $\mathcal{N}_{\leq 1}(v)$, and node D is in the 2-hop neighbors $\mathcal{N}_{\leq 2}(v)$. This GNN-SCM induces its causal structure $\mathcal{G}$ from Graph G, as discussed in Definition 2.

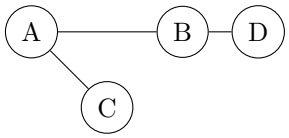


Fig. 7: A toy example

A.1 GNN-SCM construction

Following [23], we build a GNN-SCM $\mathcal{M}(\mathcal{G})$ that learns from the causal structure $\mathcal{G}$. The endogenous variables are node labels $\{y_v; v \in A, B, C, D\}$. The exogenous variables in $\mathcal{G}$ are reference node A's states: u_{A_1}, u_{A_2} (in this example we consider binary states A_1 and A_2), edges effects on reference node A: $u_{A,B}, u_{A,C}$, and neighbor nodes' effects on reference node A: u_B, u_C, u_D. All of these latent variables are assumed to accept the same probability $P(\mathbf{U})$ as a probability function defined over the domain of U since we don't want to input new specific information. $\mathcal{F}$ is a set of functions based on the observable and latent variables discussed above. One should consider that u_{v_i} and u_{v_j} are not independent.

As discussed in Theorem 1, we construct the GNN-SCM $\mathcal{M}(\mathcal{G})$ based on graph G as $\mathcal{M}(\mathcal{G}) =: (U, V, F, P(\mathbf{U}))$, where:

$$\mathcal{M}(\mathcal{G}) = \begin{cases} U := \begin{cases} u_{A_1}, u_{A_2} \\ u_{A,B}, u_{A,C} \\ u_B, u_C, u_D \end{cases} , D_u = \{0,1\} \\ V := \{A, B, C, D\} \\ \mathcal{F} := \begin{cases} f_A(B, C, D, u_{A_1}, u_{A_2}) = f_A(B, u_{A_1}, u_{A_2}) \wedge f_A(C, u_{A_1}, u_{A_2}) \wedge f_A(D) \\ f_A(B, u_{A_1}, u_{A_2}) = (((\neg B \oplus U_{A_1}) \vee U_{A,B}) \oplus u_{A_2}) \\ f_A(C, u_{A_1}, U_{A_2}) = (((\neg C \oplus U_{A_1}) \vee U_{A,C}) \oplus u_{A_2}) \\ f_A(D) = \neg D \\ f_B(u_B, u_{A,B}) = \neg u_B \wedge \neg u_{A,B} \\ f_C(u_C, u_{A,C}) = \neg u_C \wedge \neg u_{A,C} \\ f_D(u_D) = \neg u_D \end{cases} \\ P(\mathbf{U}) := \begin{cases} P(u_{A_1}) = P(u_{A_2}) = P(u_{A,B}) = P(u_{A,C}) \\ = P(u_{A,D}) = P(u_B) = P(u_C) = P(u_D) \end{cases} = 1/8 \end{cases} \tag{8}$$

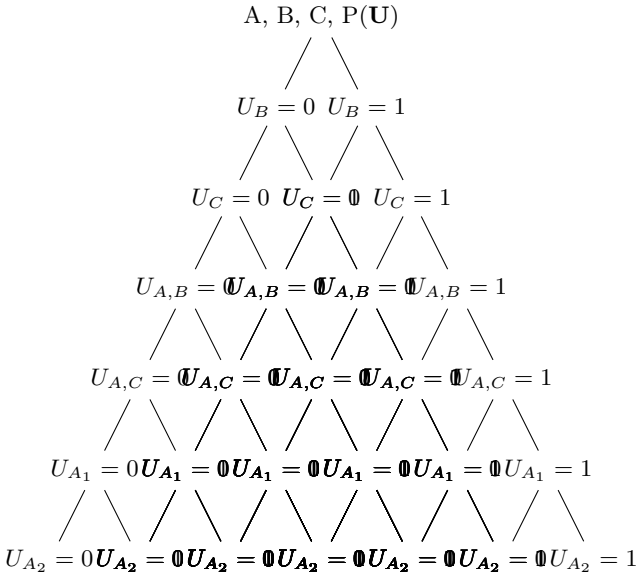


Fig. 8: Logic tree for example's SCM

In the GNN-SCM $\mathcal{M}(\mathcal{G})$, the set of functions $\mathcal{F}$ should be the exact form of the interactions between variables. The argument of each function is the input that this function will do one of the logical operations OR, AND, NOT, or XOR

on them. For example, $f_A(B, C, D, u_{A_1}, u_{A_2}) = f_A(B, u_{A_1}, u_{A_2}) \wedge f_A(C, u_{A_1}, u_{A_2}) \wedge f_A(D)$ calculates the value of effects on reference observable variable A when we assume that all observable variables B, C, D, and each state A_1, or A_2 are cause of reference observable variable A to accept a specific value A_1, or A_2. In this setting, all of these variables should be feasible and have value. *For simplicity, we consider all observable variables as binary, and the probability of these states as uniform distribution.*

In the graph provided, intervention $do(C = 1)$ means forcing the value of node C's label to be 1, and the probability $P(A = 1|do(C = 1))$ determines the respective causal effect for a treatment $A = 1$. In the GNN-SCM $\mathcal{M}(\mathcal{G})$, this effect is denoted as u_C. Since C has an edge to the reference node A, there is also a latent variable $u_{A,C}$, and its causal effect can be calculated by $P(do(C = 1)|e_{A,C} = 1)$. In this example, there is one node D that does not have an edge to A. So, here we just calculate its causal effect on node A by latent variable u_D.

For the causal effect calculation, we need a truth table showing induced values of $\mathcal{M}(\mathcal{G})$. The logic tree we used for this table is shown in Figure 8, which has seven layers since there are seven distinct latent variables(u_B, u_C, $u_{A,B}$, $u_{A,C}$, u_{A_1}, u_{A_2}) each accepting the value of 0 (it's not the cause) or 1 (it is the cause).

A.2 SCM tables

By interpreting the variables and their values from the logic tree, the truth table will be in four different states. If none of the 1-hop neighborhood nodes' observable variable affects reference node A, if one of them (node B or C) has an effect, or otherwise both of them affect the reference node. Probabilities in P($\mathbf{U}$) are labeled from p_0 to p_{63} for convenience, which are 7 binary variables(layers in the logic tree).

In all provided table rows, $u_D = 0$ and $D = 1$, meaning D is not the cause and the latent variable of it is 1. However, there is no edge between nodes A and D, we need to mention this in our calculations. For simplicity, we just showed the cases that $u_D = 0$, but there are the same truth tables with $u_D = 1$ and $D = 0$ by probabilities p_{64} to p_{127}. Given the probabilities from the truth tables, we can define them as follows:

$$P(\mathbf{U}) := \text{Unif}(0, 1) \implies p_0 = p_1 = p_2 = \ldots = p_{63} = \ldots = p_{127} = 1/128$$

A.3 GNN-SCM results

The capability of the tables shows that our specified GNN-SCM $\mathcal{M}(\mathcal{G})$ can calculate all queries from each PCH layer [25]. In continue, we will calculate an example for each layer:

An association layer query such as $P(A = 1|C = 1)$ which is the probability of observable variable A to be 1 given observable variable C to be 1, can be computed as:

u_B	u_C	u_D	$u_{A,B}$	$u_{A,C}$	u_{A_1}	u_{A_2}	**B**	**C**	**D**	**A**	**P(U)**
0	0	0	0	0	0	0	1	1	1	$\neg(B \wedge C)$	p_0
0	0	0	0	0	1	0	1	1	1	$(B \wedge C)$	p_1
0	0	0	0	0	0	1	1	1	1	$(B \wedge C)$	p_2
0	0	0	0	0	1	1	1	1	1	$\neg(B \wedge C)$	p_3
0	0	0	1	0	0	0	0	1	1	1	p_4
0	0	0	1	0	1	0	0	1	1	1	p_5
0	0	0	1	0	0	1	0	1	1	0	p_6
0	0	0	1	0	1	1	0	1	1	0	p_7
0	0	0	0	1	0	0	1	0	1	1	p_8
0	0	0	0	1	1	0	1	0	1	1	p_9
0	0	0	0	1	0	1	1	0	1	0	p_{10}
0	0	0	0	1	1	1	1	0	1	0	p_{11}
0	0	0	1	1	0	0	0	0	1	1	p_{12}
0	0	0	1	1	1	0	0	0	1	1	p_{13}
0	0	0	1	1	0	1	0	0	1	0	p_{14}
0	0	0	1	1	1	1	0	0	1	0	p_{15}

Table 4: Example's SCM truth table where $u_B = 0$, and $u_C = 0$ (node B and C are the cause of node A to get a specific value). Probabilities in $P(U)$ are labeled from p_0 to p_{15} for convenience.

u_B	u_C	u_D	$u_{A,B}$	$u_{A,C}$	u_{A_1}	u_{A_2}	**B**	**C**	**D**	**A**	**P(U)**
1	0	0	0	0	0	0	0	1	1	$\neg(B \wedge C)$	p_{16}
1	0	0	0	0	1	0	0	1	1	$(B \wedge C)$	p_{17}
1	0	0	0	0	0	1	0	1	1	$(B \wedge C)$	p_{18}
1	0	0	0	0	1	1	0	1	1	$\neg(B \wedge C)$	p_{19}
1	0	0	1	0	0	0	0	1	1	1	p_{20}
1	0	0	1	0	1	0	0	1	1	1	p_{21}
1	0	0	1	0	0	1	0	1	1	0	p_{22}
1	0	0	1	0	1	1	0	1	1	0	p_{23}
1	0	0	0	1	0	0	0	0	1	1	p_{24}
1	0	0	0	1	1	0	0	0	1	1	p_{25}
1	0	0	0	1	0	1	0	0	1	0	p_{26}
1	0	0	0	1	1	1	0	0	1	0	p_{27}
1	0	0	1	1	0	0	0	0	1	1	p_{28}
1	0	0	1	1	1	0	0	0	1	1	p_{29}
1	0	0	1	1	0	1	0	0	1	0	p_{30}
1	0	0	1	1	1	1	0	0	1	0	p_{31}

Table 5: Example's SCM truth table where $u_B = 1$, and $u_C = 0$ (only node C is not the cause of node A to get a specific value). Probabilities in $P(U)$ are labeled from p_{16} to p_{31} for convenience.

$$P(A = 1|C = 1) = \frac{P(A = 1, C = 1)}{P(C = 1)}$$
$$= \frac{p1 + p2 + p4 + p5 + p17 + p18 + p20 + p21}{p0 + p1 + p2 + p3 + p4 + p5 + p6 + p7 + p16 + p17 + p18 + p19 + p20 + p21 + p22 + p23} = 0.5 \tag{9}$$

u_B	u_C	u_D	$u_{A,B}$	$u_{A,C}$	u_{A_1}	u_{A_2}	**B**	**C**	**D**	**A**	**P(U)**
0	1	0	0	0	0	0	1	0	1	$\neg(B \wedge C)$	p_{32}
0	1	0	0	0	1	0	1	0	1	$(B \wedge C)$	p_{33}
0	1	0	0	0	0	1	1	0	1	$(B \wedge C)$	p_{34}
0	1	0	0	0	1	1	1	0	1	$\neg(B \wedge C)$	p_{35}
0	1	0	1	0	0	0	0	0	1	1	p_{36}
0	1	0	1	0	1	0	0	0	1	1	p_{37}
0	1	0	1	0	0	1	0	0	1	0	p_{38}
0	1	0	1	0	1	1	0	0	1	0	p_{39}
0	1	0	0	1	0	0	1	0	1	1	p_{40}
0	1	0	0	1	1	0	1	0	1	1	p_{41}
0	1	0	0	1	0	1	1	0	1	0	p_{42}
0	1	0	0	1	1	1	1	0	1	0	p_{43}
0	1	0	1	1	0	0	0	0	1	1	p_{44}
0	1	0	1	1	1	0	0	0	1	1	p_{45}
0	1	0	1	1	0	1	0	0	1	0	p_{46}
0	1	0	1	1	1	1	0	0	1	0	p_{47}

Table 6: Example's SCM truth table where $u_B = 0$, and $u_C = 1$ (only node B is not the cause of node A to get a specific value). Probabilities in $P(U)$ are labeled from p_{32} to p_{47} for convenience.

u_B	u_C	u_D	$u_{A,B}$	$u_{A,C}$	u_{A_1}	u_{A_2}	**B**	**C**	**D**	**A**	**P(U)**
1	1	0	0	0	0	0	0	0	1	$\neg(B \wedge C)$	p_{48}
1	1	0	0	0	1	0	0	0	1	$(B \wedge C)$	p_{49}
1	1	0	0	0	0	1	0	0	1	$(B \wedge C)$	p_{50}
1	1	0	0	0	1	1	0	0	1	$\neg(B \wedge C)$	p_{51}
1	1	0	1	0	0	0	0	0	1	1	p_{52}
1	1	0	1	0	1	0	0	0	1	1	p_{53}
1	1	0	1	0	0	1	0	0	1	0	p_{54}
1	1	0	1	0	1	1	0	0	1	0	p_{55}
1	1	0	0	1	0	0	0	0	1	1	p_{56}
1	1	0	0	1	1	0	0	0	1	1	p_{57}
1	1	0	0	1	0	1	0	0	1	0	p_{58}
1	1	0	0	1	1	1	0	0	1	0	p_{59}
1	1	0	1	1	0	0	0	0	1	1	p_{60}
1	1	0	1	1	1	0	0	0	1	1	p_{61}
1	1	0	1	1	0	1	0	0	1	0	p_{62}
1	1	0	1	1	1	1	0	0	1	0	p_{63}

Table 7: Example's SCM truth table where $u_B = 1$, and $u_C = 1$ (node B and C are not the cause of node A to get a specific value). Probabilities in $P(U)$ are labeled from p_{48} to p_{63} for convenience.

An intervention layer query such as $P(A = 1|do(C = 1))$ which is the probability of A being 1 after this intervention on C. It seeks to understand the causal effect of setting C to 1 on outcome A . $do(C = 1)$ represents an intervention where the variable C is actively set to 1 to control the variable directly, essen-

tially breaking its usual causal edges with other variables. This query can be computed as:

$$
\begin{aligned}
&P(A=1|do(C=1)) = P(A=(B\wedge C)|C=1) \vee P(A=1|C=0)\\
&= \frac{p1+p2+p17+p18}{p0+p1+p2+p3+p4+p5+p6+p7+p16+p17+p18+p19+p20+p21+p22+p23}\\
&+ \frac{p24+p25+p28+p29+p36+p37+p40+p41+p44+p45+p52+p53+p56+p57+p60+p61}{p8+p9+...+p14+p15+p24+p25+p26+p27+p28+p29+p30+p31+p32+...+p63}\\
&= 0.25+0.33=0.58
\end{aligned} \tag{10}
$$

For the implementation, we need the observed data generated from the graph. One generated data can be: $\{A:True, B:False, C:False, D:True, u_{A_1}:1, u_{A_2}:0, u_{A,B}:1, u_{A,C}:1, u_B:0, u_C:1, u_D:0, A|C:1\}$. The Fraction of samples that satisfy each function of the GNN-SCM $\mathcal{M}(\mathcal{G})$: $\{f_A:0.156458, f_B:0.249319, f_C:0.25009, f_D:0.50059\}$. These values show the range and central tendency of the probabilities across the 500 trials, indicating a degree of variability in the outcomes based on the random generation of the probabilities.

A.4 Example's GNN-NCM

With respect to literature, an NCM is as expressive as an SCM, and all NCMs are SCMs. The causal diagram constraints are the bias between SCMs and NCMs. In our specified GNN-SCM $\widehat{\mathcal{M}}(\mathcal{G})$, and GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta))$, all causal information(observable, and latent variables) comes from the causal structure defined in Definition. 2. The respective GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta)$ is constructed as a proxy of the exact GNN-SCM $\mathcal{M}(\mathcal{G})$. Based on Equation. 3, reference node A is chosen as the target node for causal structure $\mathcal{G}$. This GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta)$ is an inductive bias type of the GNN-SCM as $\widehat{\mathcal{M}}(\mathcal{G},\theta) =: \langle \widehat{\mathbf{U}}, V, \widehat{\mathcal{F}}, P(\widehat{\mathbf{U}})\rangle$. The construction of the corresponding GNN-NCM that induces the same distributions for our example dataset is as follows:

$$
\widehat{\mathcal{M}}(\mathcal{G},\theta)) = \begin{cases} \widehat{\mathbf{U}} := \{\widehat{\mathbf{U}}\}, D_{\widehat{\mathbf{U}}} = [0,1] \\ V := \{A,B,C,D\} \\ \widehat{\mathcal{F}} := \begin{cases} \widehat{f}_A(\widehat{\mathbf{U}}) =? \\ \widehat{f}_B(A,\widehat{\mathbf{U}}) =? \\ \widehat{f}_C(A,\widehat{\mathbf{U}}) =? \\ \widehat{f}_D(A,\widehat{\mathbf{U}}) =? \end{cases} \\ P(\widehat{\mathbf{U}}) :=? \end{cases} \tag{11}
$$

We know the observable variables V values, but there is no clue about the exact values of latent variables U, so we have to estimate them by functions $\widehat{\mathcal{F}}$ based on the information in the given causal structure $\mathcal{G}$. First, we have to build the causal structure $\mathcal{G}$ given example graph G.

$$
\begin{aligned}
\mathcal{G}(G) &= \{\mathbf{V}_v = \{A,B,C,D\}\},\\
&\mathbf{U}_v = \{\mathbf{U}_{v_i} : \{u_B, u_C, u_D, u_{A_1}, u_{A_2}\} \cup \{\mathbf{U}_{v,v_i} : \{u_{A,B}, u_{A,C}\}\}
\end{aligned} \tag{12}
$$

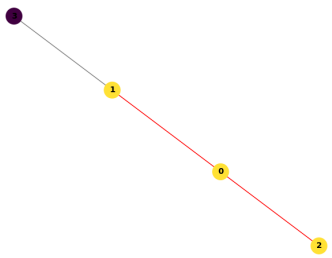


Fig. 9: Result of GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta))$ for toy example

Second, given the probabilities from the truth tables, we have:

$$P(\widehat{\mathbf{U}}) := \text{Unif}(0,1) \implies p_0 = p_1 = p_2 = \ldots = p_{63} = 1/256$$

At last, based on Algorithm 1, we train the GNN-NCM to find the functions. The GNN-NCM extends the GNN-SCM to utilize the power of neural networks in capturing complex patterns in the dataset. The process begins with sampling from a prior distribution to simulate the unobserved confounders in the causal process. Since the reference node in the GNN-SCM was node A, we assign value 1 to it and want to see how GNN-NCM finds the causal effects on this node in graph G. The result of the Algorithm 2, are:

When the target node is A, the expected probability is 0.24082797765731812
When the target node is B, the expected probability is 0.2353899081548055
When the target node is C, the expected probability is 0.18209974467754364
When the target node is D, the expected probability is 0.12958189845085144

Hence, the final causal explanatory subgraph Γ based on the results is the GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta))$ with target node A is:

$$\begin{aligned}
&\mathcal{G}(G) = \big\{\mathbf{V}_v = \{A = (B \wedge C), B = 1, C = 1, D = 0\}\big\},\\
&\mathbf{U}_v = \big\{\mathbf{U}_{v_i} : \{u_B = 0, u_C = 0, u_D = 1, u_{A_1} = 0, u_{A_2} = 1\}\\
&\qquad \cup \{\mathbf{U}_{v,v_i} : \{u_{A,B} = 1, u_{A,C} = 1\}\big\}
\end{aligned} \tag{13}$$

The GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta))$ structure is:

$$\widehat{\mathcal{M}}(\mathcal{G},\theta)) = \begin{cases}
\widehat{\mathbf{U}} := \Big\{\mathbf{U}_{v_i} : \{u_B = 0, u_C = 0, u_D = 1, u_{A_1} = 0, u_{A_2} = 1\}\\
\qquad \cup \{\mathbf{U}_{v,v_i} : \{u_{A,B} = 1, u_{A,C} = 1\}\Big\}\\
V := \{A = (B \wedge C), B = 1, C = 1, D = 0\}\\
\widehat{\mathcal{F}} := \begin{cases}
\widehat{f}_A(\widehat{\mathbf{U}}) = \widehat{f}_A(\widehat{\mathbf{u}}_{A_1} = 0, \widehat{\mathbf{u}}_{A_2} = 1, \widehat{\mathbf{u}}_{A,B} = 1, \widehat{\mathbf{u}}_{A,C} = 1)\\
\widehat{f}_B(A, \widehat{\mathbf{U}}) = \widehat{f}_B(\widehat{\mathbf{u}}_B = 0, \widehat{\mathbf{u}}_{A,B} = 1)\\
\widehat{f}_C(A, \widehat{\mathbf{U}}) = \widehat{f}_C(\widehat{\mathbf{u}}_C = 0, \widehat{\mathbf{u}}_{A,C} = 1)\\
\widehat{f}_D(A, \widehat{\mathbf{U}}) = \widehat{f}_D(\widehat{\mathbf{u}}_D = 1)
\end{cases}\\
P(\widehat{\mathbf{U}}) := \begin{cases}
P(u_{A_1}) = P(u_{A_2}) = P(u_{A,B}) = P(u_{A,C})\\
= P(u_{A,D}) = P(u_B) = P(u_C) = P(u_D)
\end{cases} = 1/8
\end{cases} \tag{14}$$

B More Background on Causality

According to the literature, causality interprets the information by the Pearl Causal Hierarchy (PCH) layers [25].

Definition 4 (PCH layers). *The PCH layers* L_i *for* $i \in 1,2,3$ *are:* L_1 *association layer,* L_2 *intervention layer, and* L_3 *counterfactual layer.*

Definition 5 ($\mathcal{G}$-Consistency). *Let* $\mathcal{G}$ *be the causal structure induced by SCM* $\mathcal{M}^*$*. For any SCM* $\mathcal{M}$*, we say* $\mathcal{M}$ *is* $\mathcal{G}$*-consistent w.r.t* $\mathcal{M}^*$ *if* $\mathcal{M}$ *imposes the same constraints over the interventional distributions as the true* $\mathcal{M}^*$*.*

Definition 6 ($\mathcal{G}$-Constrained NCM). *Let* $\mathcal{G}$ *be the causal structure induced by SCM* $\mathcal{M}^*$*. We can construct NCM* $\widehat{\mathcal{M}}$ *as follows: 1) Choose* $\widehat{\mathbf{U}}$ *s.t.* $\widehat{U}_{\mathbf{C}} \in \widehat{\mathbf{U}}$*, where any pair* $(V_i, V_j) \in \mathbf{C}$ *is connected with a bidirected arrow in* $\mathcal{G}$ *and is maximal; 2) For each* $V_i \in \mathbf{V}$*, choose* $\boldsymbol{Pa}(V_i) \subseteq \mathbf{V}$ *s.t. for every* $V_j \in \mathbf{V}$*,* $V_j \in \boldsymbol{Pa}(V_i)$ *iff there is a directed arrow from* V_j *to* V_i *in* $\mathcal{G}$*. Any NCM in this family is said to be* $\mathcal{G}$*-constrained.*

Theorem 5. *Any* $\mathcal{G}$*-constrained NCM* $\widehat{\mathcal{M}}(\boldsymbol{\theta})$ *is* $\mathcal{G}$*-consistent.*

C Proofs

In this section, we provide proofs of the theorems in the main body of the paper.

C.1 Proof of Theorem 1

Theorem 1 (GNN-SCM). *For a GNN operating on a graph* G*, there exists an SCM* $\mathcal{M}(\mathcal{G})$ *w.r.t. the causal structure* $\mathcal{G}$ *of the graph* G*.*

Proof. A GNN is a neural network operating on a graph $G = (\mathcal{V}, \mathcal{E})$ including set of nodes $\mathcal{V}$ and set of edges $\mathcal{E}$. Recall the GNN background in Section 3, the GNN learning mechanism for a node v in the l-th layer can be summarized as:

$$\mathrm{GNN}(v) \equiv \begin{cases} \text{node embeddings } h_v^{l-1} \text{ from previous layer } l-1, \text{ for } u \in \{v\} \cup \mathcal{N}(v) \\ \text{message } m_{u,v}^l = \mathrm{MSG}(h_u^{l-1}, h_v^{l-1}, e_{u,v}) \text{ for current layer } l \\ \text{aggregated message } h_v^l = \mathrm{AGG}(m_{u,v}^l | u \in \mathcal{N}(v)) \text{ for current layer } l \end{cases} \tag{15}$$

where the above process is iteratively performed k times for a k-layer GNN. In doing so, each node v will leverage the information from all its within k neighborhoods. We denote the k-layer GNN learning for v as:

$$\mathrm{GNN}(G) = \big\{\text{node embed: } \{h_v\} \cup \{h_u : u \in \mathcal{N}_{\leq k}(v)\}, \text{message: } \{m_{u,v}, u \in \mathcal{N}_{\leq k}(v)\}\big\},$$

where h_v is v' node feature and we omit the dependence on node v for notation simplicity.

By definition from literature, an SCM $\mathcal{M}$ is a four-tuple $\mathcal{M} \equiv (\mathbf{U}, \mathbf{V}, \mathcal{F}, P(\mathbf{U}))$. In this specification of ours, an SCM $\mathcal{M}(\mathcal{G})$ is a $\mathcal{G}$-consistent four-tuple model based on a set of observable variables $\mathbf{V}$, a set of latent variables $\mathbf{U}$, a set of functions $\mathcal{F}$, and the probability of latent variables $P(\mathbf{U})$. Recall in Definition 2, where the causal structure $\mathcal{G}(G)$ of a given graph G (centered on a reference node v) was defined. Now, the correspondence of the GNN learning on G and the causal structure $\mathcal{G}$ centered on a node v can be written as:

$$\mathcal{G}(G) \equiv \text{GNN}(G) \iff \begin{cases} \text{obs. vars: } \{y_v\} \cup \{y_{v_i} : v_i \in \mathcal{N}_{\leq k}(v)\} \equiv \text{node embed. } \{h_v\} \cup \{h_u : u \in \mathcal{N}_{\leq k}(v)\} \\ \text{lat. vars: } \{\mathbf{U}_{v_i} : v_i \in \mathcal{N}_{\leq k}(v)\} \cup \{\mathbf{U}_{v,v_i} : e_{v,v_i} \in \mathcal{E}\} \equiv \text{ msg: } \{m_{u,v}, u \in \mathcal{N}_{\leq k}(v)\} \\ \text{probability of latent variables } P(\mathbf{U}) \equiv \text{Dom}(\{h_v\}) \end{cases} \tag{16}$$

Hence, there exists a GNN-SCM $\mathcal{M}(\mathcal{G})$ that induces the causal structure $\mathcal{G}$ of G, as below:

$$\mathcal{G}(G) = \begin{cases} \text{node set } \mathcal{V}(\mathcal{G}) \\ \text{edge set } \mathcal{E}(\mathcal{G}) \\ \text{node effect } = \{\mathbf{U}_{v_i}\} \\ \text{edge effect } = \{\mathbf{U}_{v,v_i}\} \end{cases} \iff \mathcal{M}(\mathcal{G}) \equiv \begin{cases} \mathbf{U} : \{\mathbf{U}_{v_i} : v_i \in \mathcal{N}_{\leq k}(v)\} \cup \{\mathbf{U}_{v,v_i} : e_{v,v_i} \in \mathcal{E}\} \\ \mathbf{V} : \{y_v\} \cup \{y_{v_i}; v_i \in \mathcal{N}_{\leq k}(v)\} \\ \mathcal{F} : \{f_1, f_2, \ldots, f_n\} \in \mathcal{F};\ f(\mathbf{U}) \to \mathbf{V} \\ P(\mathbf{U}) : P(\mathbf{U}_{v_i}), P(\mathbf{U}_{v,v_i}) \end{cases} \tag{17}$$

□

C.2 Proof of Theorem 2

Theorem 2 (GNN-NCM). *Given causal structure $\mathcal{G}$ of a graph G and the underlying GNN-SCM $\mathcal{M}(\mathcal{G})$, there exists a $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ that enables any inferences consistent with $\mathcal{M}(\mathcal{G})$.*

Proof. From the literature, we know there exists a SCM $\mathcal{M}$ that includes exact values of observable and latent variables through studying the causes and effects within the SCM structure. First, we show a lemma that demonstrates the inheritance of neural causal models (NCMs) (see its definition in Section B) from SCMs, which are built upon Definition 5, Definition 6 and Theorem 5.

Lemma 1 ([43]). *All NCMs $\widehat{\mathcal{M}}(\theta)$ (parameterized by θ) are SCMs (i.e., $\widehat{\mathcal{M}}(\theta) \prec M$). Further, any $\mathcal{G}$-constrained $\widehat{\mathcal{M}}(\theta)$ (see Definition 6) has the same empirical observations as the SCM $\mathcal{M}$, which means $\mathcal{G}$-constrained NCMs can be used for generating any distribution associated with the SCMs.*

By Lemma 1, we know a $\mathcal{G}$-constrained NCM $\widehat{\mathcal{M}}(\theta)$ inherits all properties of the respective SCM $\mathcal{M}$ and ensures causal inferences via $\mathcal{G}$-constrained NCM. In our context, we need to build the corresponding $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ for the GNN-SCM defined in Equation 17. With it, we ensure all $\mathcal{G}$-constrained GNN-NCMs $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ are GNN-SCMs ($\widehat{\mathcal{M}}(\mathcal{G}, \theta) \prec \mathcal{M}(\mathcal{G})$), meaning these GNN-NCMs can be used for performing causal inferences on the causal structure $\mathcal{G}(G)$. First, based to the four-tuple SCM $\mathcal{M} \equiv (\mathbf{U}, \mathbf{V}, \mathcal{F}, P(\mathbf{U}))$, a $\mathcal{G}$-constrained NCM $\widehat{\mathcal{M}}(\theta) = (\widehat{\mathbf{U}}, \mathbf{V}, \widehat{\mathcal{F}}(\theta), P(\widehat{\mathbf{U}}))$ can be defined. In our scenario, we can define the set of functions $\widehat{\mathcal{F}}(\theta)$ of the $\mathcal{G}$-constrained GNN-NCM as:

$$\widehat{\mathcal{F}}(\theta) = \{\widehat{f}_{v_i}(\widehat{\mathbf{u}}_{v_i}, \widehat{\mathbf{u}}_{v_i,v_j}; \theta_{v_i}) : v_i \in \mathcal{V}(\mathcal{G})\} \approx \{f_1, f_2, \ldots, f_n\} \in \mathcal{F};\ f(\mathbf{U}_{v_i}) \to v_i,$$

From Theorem 1, there exists a GNN-SCM $\mathcal{M}(\mathcal{G})$ for a GNN operating on a graph G. Also the causal structure $\mathcal{G}(G)$ in Definition 2 naturally satisfies Definition 6. Then, with respect to Equation 3, our $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ based on underlying GNN-SCM $\mathcal{M}(\mathcal{G})$ is defined as:

$$\mathcal{M}(\mathcal{G}) \iff \widehat{\mathcal{M}}(\mathcal{G}, \theta) \equiv \begin{cases} \widehat{\mathbf{U}} := \{\widehat{\mathbf{U}}_{v_i} : v_i \in \mathcal{N}_{\leq k}(v)\} \cup \{\widehat{\mathbf{U}}_{v,v_i} : e_{v,v_i} \in \mathcal{E}\} \\ \mathbf{V} := \{y_v\} \cup \{y_{v_i}; v_i \in \mathcal{N}_{\leq k}(v)\} \\ \widehat{\mathcal{F}}(\theta) := \{\widehat{f}_{v_i}(\widehat{\mathbf{u}}_{v_i}, \widehat{\mathbf{u}}_{v_i,v_j})(\theta_{v_i}) : v_i \in V\} \\ P(\widehat{\mathbf{U}}) := \{\widehat{\mathbf{U}}_{v_i} \sim \text{Unif}(0,1) : v_i \in \mathbf{V}\} \cup \{T_{k,v_i} \sim \mathcal{N}(0,1)\} \end{cases}$$

□

C.3 Proof of Theorem 3

Theorem 3 (Node explainability). *Let a prediction for a graph G be explained. A node $v \in G$ is causally explainable, if $p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v)$ can be computed.*

Proof. In a graph classification task, GNN predicts a graph label $\widehat{y}_G$ for a graph G with label y_G. The GNN explanation measures how accurately did the GNN classify the graph by finding the groundtruth explanation Γ_G in the graph G. In other words, the graph explanation demands the nodes in Γ_G should be as accurate as possible. That is, if $y_v = \widehat{y}_v; \forall v \in \Gamma_G$, then GNN explanation explained G's prediction accurately.

Based on Theorem 2, $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ induces causal structure $\mathcal{G}$ based on the reference node $v \in \mathcal{V}(\mathcal{G})$. So, the trained $\mathcal{G}$-constrained GNN-NCM estimates node effect $\mathbf{U}_{v_i}$, and edge effect $\mathbf{U}_{v,v_i}$ defined in Equation 2. Note that all the effects are respective to the reference node v, and if v changes, the causal structure $\mathcal{G}$ will be also changed, and as a result $\mathcal{G}$-constrained GNN-NCM will be completely different.

According to Equation. 5, a $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}, \theta)$ calculates $p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v)$ as the expected value of all the causal effects from the neighbor nodes v_i, i.e. $\text{do}(v_i)$, on the reference node v:

$$\forall v \in \mathcal{V}(G), (\exists v_i \in \mathcal{N}_{\leq k}(v)) \implies \left(p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v) \geq 0\right) \tag{18}$$

As the expected value was calculated for v, we can explain v's node label based on the outcome of $p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v)$. □

C.4 Proof of Theorem 4

Theorem 4 (Explainable node expressivity). *An explainable node v has expressivity defined as $exp_v(\widehat{\mathcal{M}}(\mathcal{G}, \theta)) = \sum_{y_v} y_v p^{\widehat{\mathcal{M}}(\mathcal{G},\theta)}(y_v)$.*

Proof. We know there is a value for the expected effect on an explainable node $v \in \mathcal{V}(G)$ as $p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v)$. This probability was calculated by the trained $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta)$.

For our purpose, we only consider the association and intervention layers. The association layer is about the observable information provided by the data, while the intervention layer in this paper is an explanation via doing interventions.

$$\begin{aligned} &\text{Association Layer } \mathrm{L}_1: \quad G=(\mathcal{V},\mathcal{E}),\ y_G \in \mathcal{Y},\ \Gamma_G \\ &\text{Intervention Layer } \mathrm{L}_2: \quad \mathrm{do}(v_i)=(y_v|y_{v_i}=x),\ \mathbf{U}_{v_i},\ \mathbf{U}_{v,v_i} \end{aligned} \tag{19}$$

The explanation methods using information in the association layer is called association-based explanation. Instead, the $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)$ is trained on interventions—a node v_i in the neighborhood of the reference node v provides the causal explanation information based on the intervention $\mathrm{do}(v_i)$—and leveraging this interventional layer information can causality interpret the GNN predictions. To align this intrinsic explanation information from causal effects, we introduce the term *expressivity*. Remember in probability theory, where the expected value of a random variable provides a measure of the central tendency of a probability distribution. For a discrete random variable Y with a probability distribution $p(y)$, the expected value of Y , denoted $\mathbb{E}(Y)$, is defined as: $\mathbb{E}(Y_v) = \sum_y y \cdot p(y)$, where y ranges over all possible values of Y, and $p(y)$ is the probability that Y takes the value y. According to Equation 5, $p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v)$ includes all the causal effect from the neighhor nodes on y_v. Hence, the expected value of of the random variable node label Y_v will be defined as:

$$\mathbb{E}_{\mathrm{L}_2}(Y_v) = \sum_{y_v} y_v \cdot p^{\widehat{\mathcal{M}}(\mathcal{G}(G),\theta)}(y_v),$$

where the subscript L_2 means the expectation leverages the interventional layer information. Note that this expected value is only feasible for the reference node (i.e., v) upon which the $\mathcal{G}$-constrained GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta)$ was built. This expected value is treated as the expressivity of the explainable node v that is denoted as $\exp_v(\widehat{\mathcal{M}}(\mathcal{G},\theta))$. □

D Experiments

D.1 More experimental setup

CXGNN: The hyperparameter settings were determined through a systematic search and validation process to optimize the model's performance. The following hyperparameters were selected based on cross-validation:

- Learning rate: We test learning rates—0.001, 0.01, 0.1—to find the optimal value, and the learning rate of 0.01 yielded the best results.

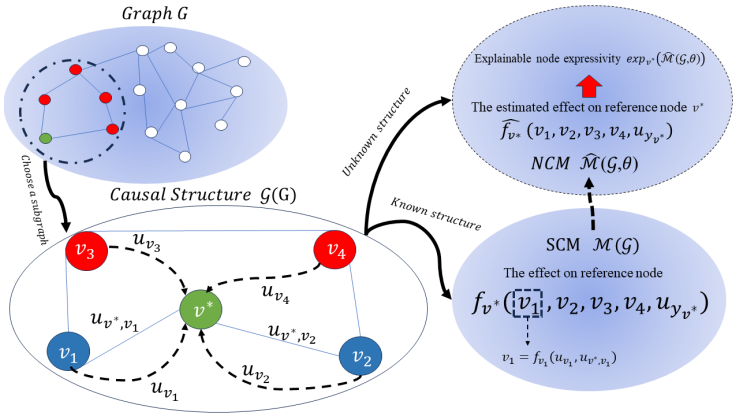


Fig. 10: Overview. Green node is the reference node v, Blue nodes and Red nodes are node v's 1-hop and 2-hop neighbors.

- Number of hidden layers: We considered architectures with 1, 2, and 3 hidden layers. A network with 2 hidden layers outperformed the others in terms of both accuracy and convergence speed.
- Hidden layer size: We tested various hidden layer sizes, including 32, 64, and 128 neurons per layer. A hidden layer size of 64 neurons struck a balance between model complexity and performance.
- Batch size: We tested different batch sizes, ranging from 32 to 128. A batch size of 64 was found to be suitable.
- In addition, as the baseline GNN is GCN [16] that is a 2-layer neural network. We hence use $k = 2$ in CXGNN.

GNNExplainer: Its hyperparameters are detailed as follows:

- A dictionary to store coefficients for the entropy term and the size term for learning edge mask and node feature mask. The chosen settings are:
  - edge: entropy: 1.0, and size: 0.005
  - feature: entropy: 0.1, and size: 1.0
- The number of epochs for training the explanation model: 200.
- Learning Rate: Used in the Adam optimizer for training, set to 0.01.
- The number of hops to consider when explaining a node prediction. It is equal to the number of layers in the GNN.

PGMexplainer: It is for explaining predictions made by GNNs using Probabilistic Graphical Models (PGMs), provides these hyperparameters:

- Number of perturbed graphs to generate: 10 for graph-level explanations
- How node features are perturbed: mean, for graph-level explanations.
- The probability that a node's features are perturbed: 0.5
- The threshold for the chi-square independence test: 0.05
- Threshold for the difference in predicted class probability: 0.1
- Number of nodes to include in PGM: all nodes given by the chi-square test are kept.

Guidedbp: It is a form of Guided Backpropagation for explaining graph predictions, contains several hyperparameters and method-specific parameters:

- The loss function used to train the model: cross entropy.
- The number of hops for the k-hop subgraph, is implicitly set to the number of layers in the GNN (i.e., $k = 2$ in our results).

GEM: The GEM method has the below hyperparameters:

- Optimization Parameters:
  - Learning Rate: 0.1, Gradient Clipping: 2, Batch size: 20, Number of Epochs: 100, Optimizer: "Adam"
- Model Parameters:
  - Hidden Dimension: 20, Output Dimension: 20, Number of Graph Convolution Layers: 2
- Explainer Parameters:
  - Iterations to find alignment matrix: 1000, Number of mini-batches: 10

RCExp: The reinforced causal explainer for graph neural networks has the below hyperparameters:

- Optimization Parameters:
  - Learning Rate: 0.01, Weight Decay: 0.005, Number of Epochs: 100, Optimizer: "Adam"
- Model Parameters:
  - Hidden Dimension: 64, 32, Output Dimension: 2, Number of Graph Convolution Layers: 2
- Explainer Parameters:
  - Output size of edge action rep generator: 64, Edge attribute dimension: 32

Orphicx: The causality-inspired latent variable Model for interpreting graph neural networks has the below hyperparameters:

- Optimization Parameters:
  - Learning Rate: 0.0005, Weight Decay: 0.01, Number of Epochs: 100, Optimizer: "Adam", Early Stopping Patience: 20
- Model Parameters:
  - Hidden Dimension: 32, Decoder Hidden Dimension: 16, Dropout rate: 0.5, Output Dimension: 108, Number of Graph Convolution Layers: 2
- Explainer Parameters: Number of causal factors: 5

D.2 More experimental results

More results on the synthetic graphs and real-world graphs in terms of loss curves and node expressivity distributions are shown in the Figure 11-Figure 18.

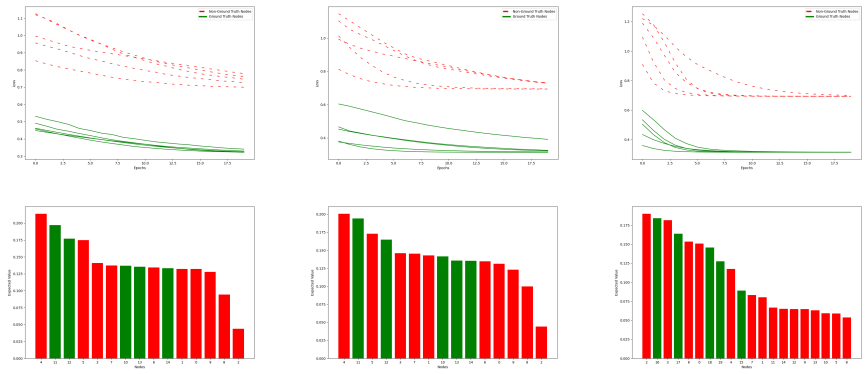


Fig. 11: More results on BA+House. Top 3 figures: Loss curves of training the GNN-NCMs on the groundtruth nodes (green curves) and non-groundtruth ones (red curves); Bottom 3 figures: Node expressivity distributions on three unsuccessful graphs. Green bars (red bars) correspond to nodes that are (NOT) in the groundtruth. Same meaning for all the below figures.

Table 8: Dominant time complexity of the compared explainers. N, E, d, h, T, L, K are #nodes, #edges, #node features, #neurons, #training epochs, #layers, and #samples. $h = 64 \ll d =\sim 1000$.

GNNExp.	$O(T * L * N * d^2)$
PGMExp.	$O(T * L * N * d^2 + K * (N + E))$
Guidedbp	$O(T * L * N)$
GEM	$O(T * L * N * d^2)$
RCExp.	$O(T * L * (N + E))$
OrphicX	$O(T * L * N * d^2)$
CXGNN	$O(T * L * N * h^2)$

E Complexity Analysis

Within a GNN-NCM, we train a feed-forward network. With an L-layer network and each layer has h neurons, by training K epochs, the time complexity for a graph with n nodes is $O(T * L * N * h^2)$. Note that training GNN-NCMs for all nodes independently can be easily paralleled via multi-threads/processors. We also show the dominant complexity of compared GNN explainers in Table 8. Though computing GNN-NCM per node, we can see CXGNN is still more efficient than most of the SOTA explainers (GNNExp., PGMExp., OrphicX, GEM).

F Discussion

Potential risk of overfitting. In our experiments, we tuned the number of hidden layers and hidden neurons and observed that deeper/wider networks indeed could cause overfitting. Through hyperparameters tuning, we found 2

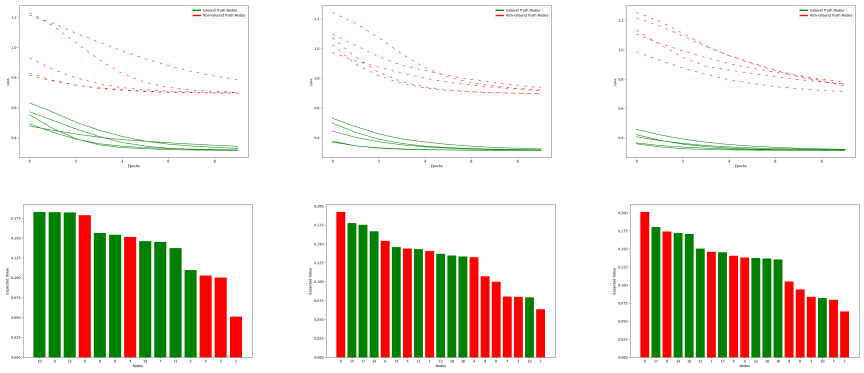


Fig. 12: More results on BA+Grid.

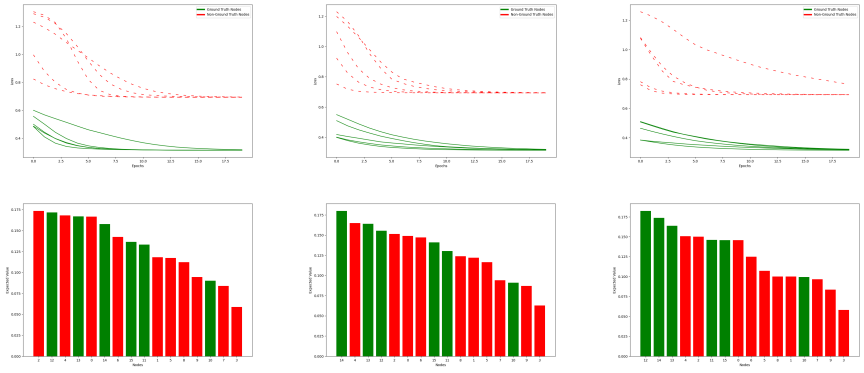


Fig. 13: More results on BA+Cycle.

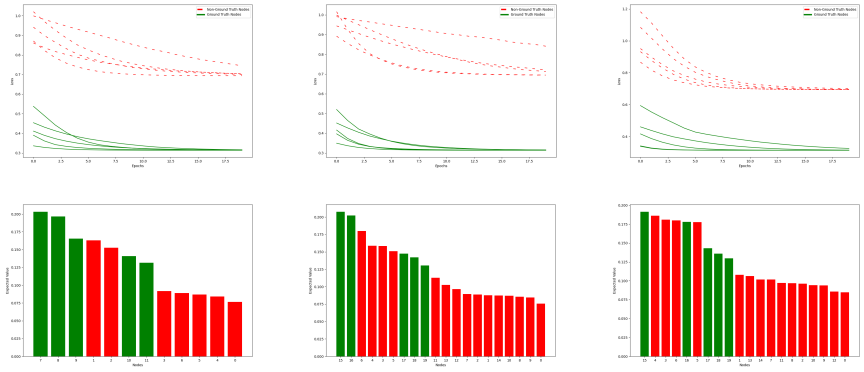


Fig. 14: More results on Tree+House.

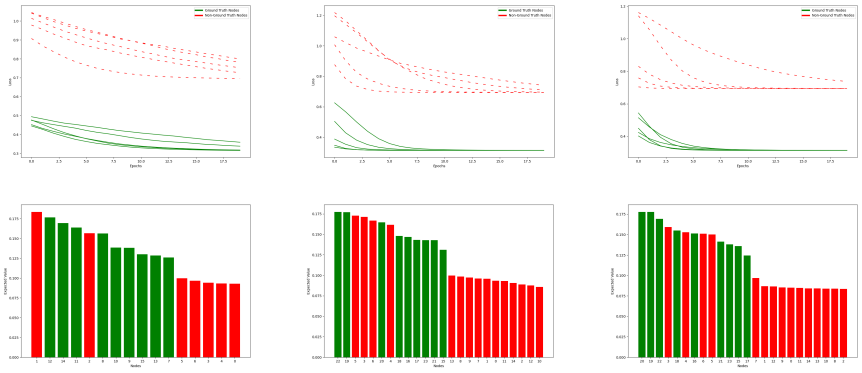


Fig. 15: More results on Tree+Grid.

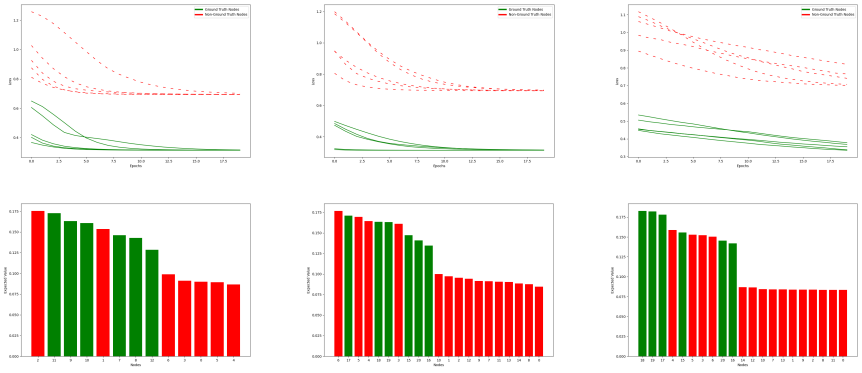


Fig. 16: More results on Tree+Cycle.

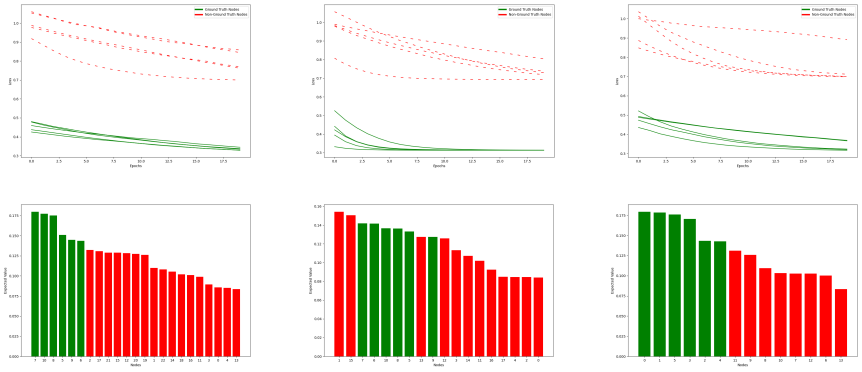


Fig. 17: More results on Benzene.

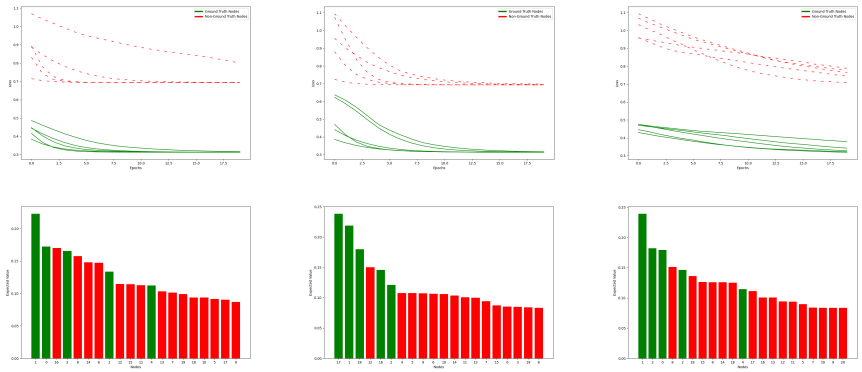


Fig. 18: More results on Fluoride Carbonyl.

hidden layers with each layer having 64 neurons that can well balance between model complexity and performance.

Practical issue of applying CXGNN to large graphs. We admit directly running CXGNN in large graphs could have a scalability issue. One solution to speed up the computation is using multi-threads/processors as all nodes can be run independently in CXGNN. Note that all existing GNN explainers also face the same scalability issue, even worse than ours as shown in Table 8. We acknowledge it is valuable future work to design scalable GNN causal explainers.

True causal subgraph is not present. Our explainer and causality-inspired ones are all based on the common assumption that a graph consists of the causal subgraph that interprets the prediction. If real-world applications do not satisfy this assumption, all these explainers may not work well.

Complexity comparison between NCM and not using NCM (i.e., SCM). Computing the cause-effect in a graph via SCM is computationally intractable. The complexity is exponential to the number of node/edge latent variables. Instead, training an NCM to learn the cause-effect is in the polynomial time.