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.

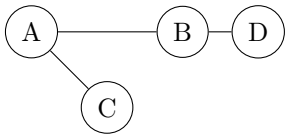

Fig. 7: A toy example

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:

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

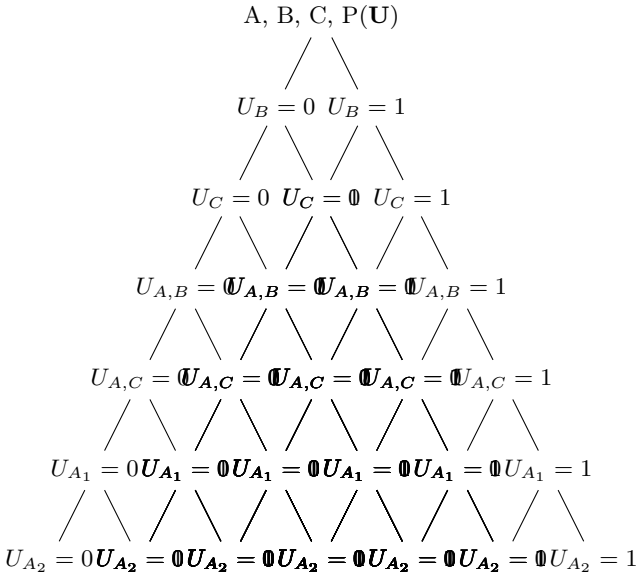

Fig. 8: Logic tree for example's SCM

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.

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

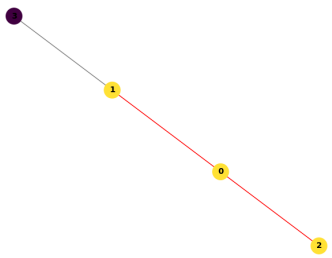

Fig. 9: Result of GNN-NCM $\widehat{\mathcal{M}}(\mathcal{G},\theta))$ for toy example

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.

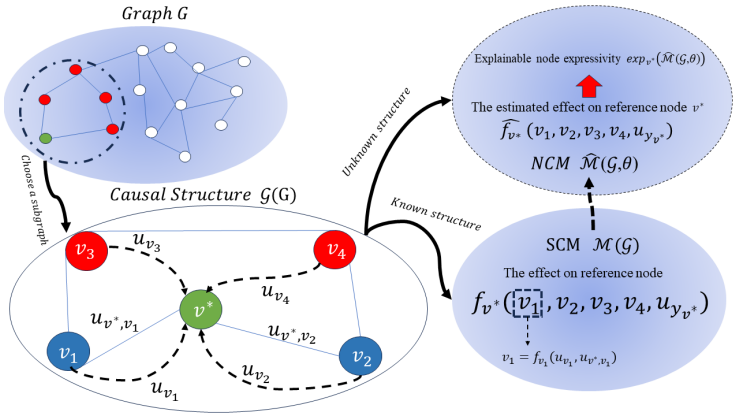

Fig. 10: Overview. Green node is the reference node v, Blue nodes and Red nodes are node v's 1-hop and 2-hop neighbors.

- 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.

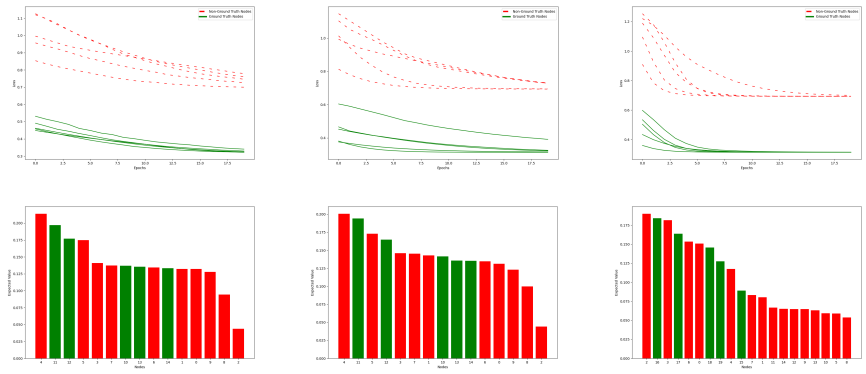

Fig. 11: More results on BA+House. Top 3 figures: Loss curves of training the GNN-NCMs on the groundtruth nodes (green curves) and non-groundtruth ones (red curves); Bottom 3 figures: Node expressivity distributions on three unsuccessful graphs. Green bars (red bars) correspond to nodes that are (NOT)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

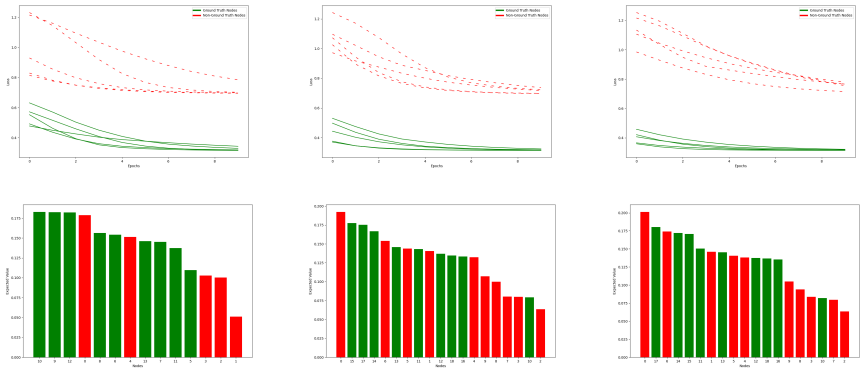

Fig. 12: More results on BA+Grid.

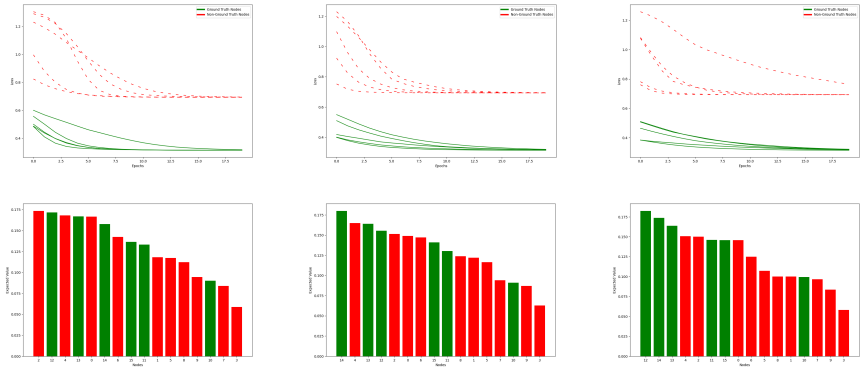

Fig. 13: More results on BA+Cycle.

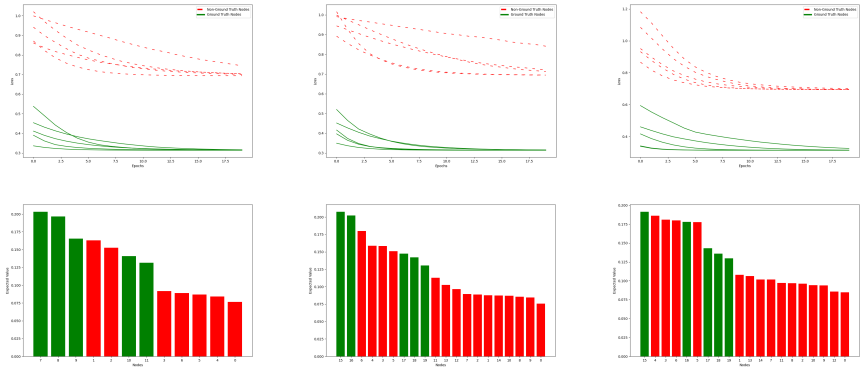

Fig. 14: More results on Tree+House.

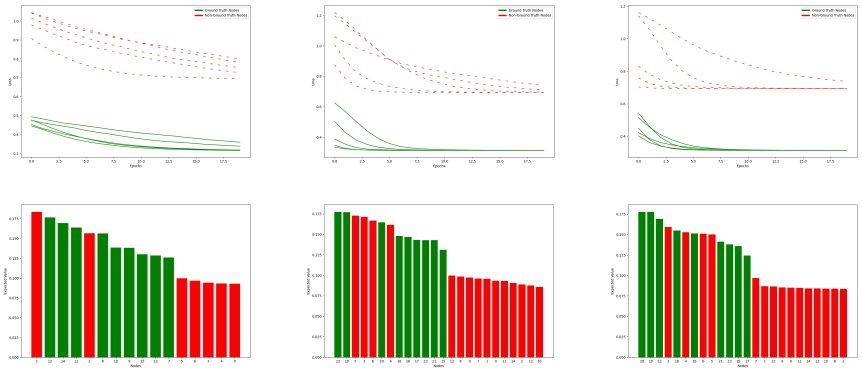

Fig. 15: More results on Tree+Grid.

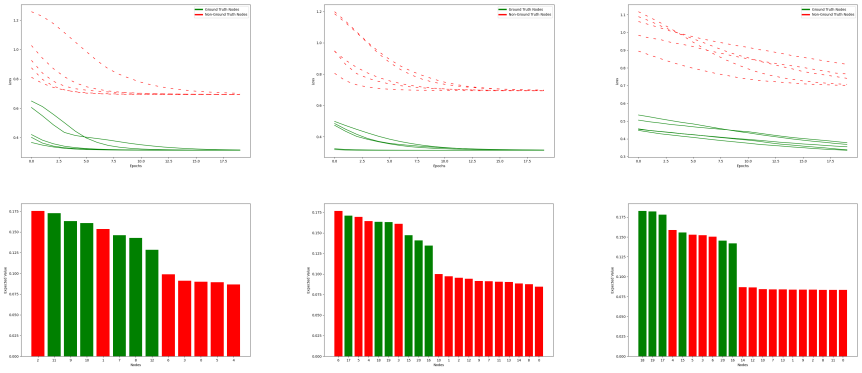

Fig. 16: More results on Tree+Cycle.

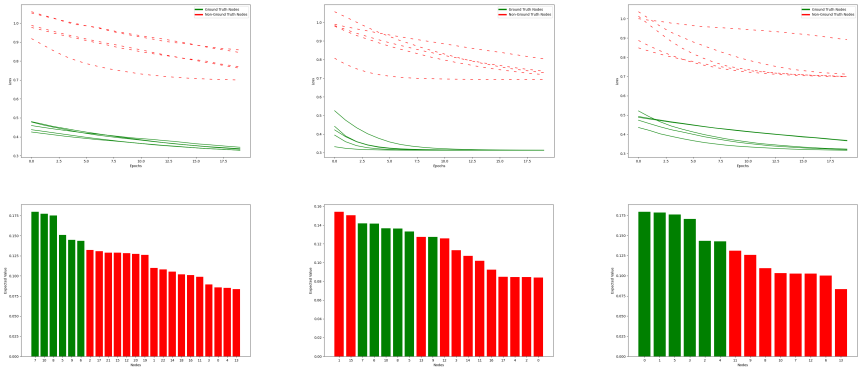

Fig. 17: More results on Benzene.

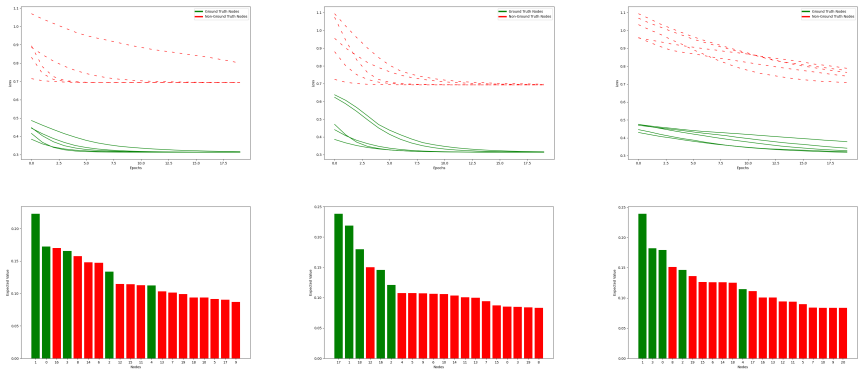

Fig. 18: More results on Fluoride Carbonyl.

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.