Towards Fine-Grained Explainability for Heterogeneous Graph Neural Network

Tong Li[1], **Jiale Deng**[1], **Yanyan Shen**[1*], **Luyu Qiu**[2], **Yongxiang Huang**[2], **Caleb Chen Cao**[2]

[1] Shanghai Jiao Tong University
[2] Huawei Research Hong Kong
{2017lt, jialedeng, shenyy}@sjtu.edu.cn, {qiuluyu, huang.yongxiang2, caleb.cao}@huawei.com

Abstract

Heterogeneous graph neural networks (HGNs) are prominent approaches to node classification tasks on heterogeneous graphs. Despite the superior performance, insights about the predictions made from HGNs are obscure to humans. Existing explainability techniques are mainly proposed for GNNs on homogeneous graphs. They focus on highlighting salient graph objects to the predictions whereas the problem of how these objects affect the predictions remains unsolved. Given heterogeneous graphs with complex structures and rich semantics, it is imperative that salient objects can be accompanied with their influence paths to the predictions, unveiling the reasoning process of HGNs. In this paper, we develop xPath, a new framework that provides fine-grained explanations for black-box HGNs specifying a cause node with its influence path to the target node. In xPath, we differentiate the influence of a node on the prediction w.r.t. every individual influence path, and measure the influence by perturbing graph structure via a novel graph rewiring algorithm. Furthermore, we introduce a greedy search algorithm to find the most influential fine-grained explanations efficiently. Empirical results on various HGNs and heterogeneous graphs show that xPath yields faithful explanations efficiently, outperforming the adaptations of advanced GNN explanation approaches.

Introduction

Real-world graphs often come with nodes and edges in multiple types, which are known as heterogeneous graphs. Recently, heterogeneous graph neural networks (HGNs) have become one of the standard paradigms for modeling rich semantics of heterogeneous graphs in various application domains such as e-commerce, finance, and healthcare (Lv et al. 2021; Wang et al. 2022). In parallel with the proliferation of HGNs, understanding the reasons behind the predictions from HGNs is urgently demanded in order to build trust and confidence in the models for both users and stakeholders. For example, a customer would be satisfied if an HGN-based recommender system accompanies recommended items with explanations; a bank manager may want to know why an HGN flagged an account as fraudulent.

To equip HGNs with the capability of providing explanations, some researches focus on developing interpretable models that use model parameters (Li et al. 2021), gradients (Pope et al. 2019; Baldassarre and Azizpour 2019; Schnake et al. 2021) or attention scores (Hu et al. 2020; Yang et al. 2021) to find salient information for the predictions. Unfortunately, they are model-specific and fall short when accesses to the architecture or parameters of an HGN are unauthorized, especially under confidentiality and security concerns (Dou et al. 2020; Liu et al. 2021).

[*]corresponding author.

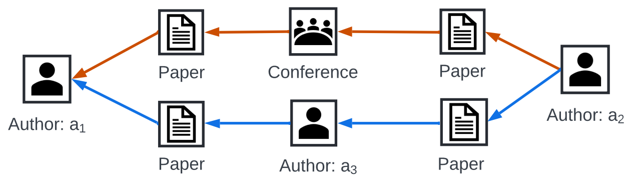


Figure 1: An explainer identifies author a_2 as the explanation for the prediction of author a_1. There are two semantically different paths showing how a_2 affects the prediction of a_1.

Alternatively, we can adapt advanced model-agnostic explanation methods (Yuan et al. 2020) proposed for GNNs (on homogeneous graphs) to HGNs. These methods attribute model predictions to graph objects, such as nodes (Vu and Thai 2020), edges (Ying et al. 2019; Luo et al. 2020; Schlichtkrull, De Cao, and Titov 2020; Wang et al. 2021b; Lin, Lan, and Li 2021) and subgraphs (Yuan et al. 2021). Their goal is to learn or search for optimal graph objects that maximize mutual information with the predictions. While such explanations answer the question "*what* is salient to the prediction", they fail to unveil "*how* the salient objects affect the prediction". In particular, there may exist multiple paths in the graph to propagate the information of the salient objects to the target object and affect its prediction. Without distinguishing these different influential paths, the answer to the "how" question remains unclear, which could compromise the utility of the explanation. This issue becomes more prominent when it comes to explaining HGNs due to the complex semantics of heterogeneous graphs.

Example 1 *Consider an HGN that performs node classification on a heterogeneous academic graph with three types of nodes, Author(A), Paper(P), Conference(C), as shown in Figure 1. The model classifies the research area of author a_1 as "AI". An explainer identified author a_2 as the* *explanation for the prediction of a_1. To explore how a_2 affects the prediction, we notice there are two simple paths P_1, P_2 in the graph that relate a_2 to a_1. P_1 indicates a_1, a_2 have co-authored with a_3 individually and P_2 shows a_1, a_2 have published papers in the same conference c. Assume the HGN can propagate a_2's information to a_1 along both paths to affect the prediction. The two paths carry different semantic meanings following* APAPA *and* APCPA *at the meta level respectively. They thus provide semantically different answers to the question "how a_2 affects the prediction of a_1 by the HGN". Without distinguishing the influence paths from a_2 (cause) to a_1 (effect), it is difficult to understand the reasoning process of the model (the "how" question).*

The goal of this paper is to provide explanations for black-box HGNs that can answer "what" and "how" questions on node classification tasks. Consider a heterogeneous graph G with node set V and edge set E and an HGN M_G that predicts the label of a target node $v_t \in V$ to be $M_G(v_t)$. To achieve the goal, we propose to explain $M_G(v_t)$ with *fine-grained explanations* in the form of (v, P), where $v \in V$ is the cause of the prediction and P is a simple path in the graph connecting v to v_t that unveils how v affects the prediction. In Example 1, both (a_2, P_1) and (a_2, P_2) are fine-grained explanations for $M_G(a_1)$, corresponding to two different ways that a_2 affects the prediction of a_1. Particularly, fine-grained explanations allow us to drill down the influence of a node on the prediction with respect to each individual influence path, and the node and path in such an explanation answer "what" and "how" questions respectively.

To find fine-grained explanations that are faithful to $M_G(v_t)$, there are two challenging issues to be tackled. First, we need to measure the influence of any node v on the prediction $M_G(v_t)$ with respect to a simple path P from v to v_t. A typical way is to perturb (v, P) and measure the change in the model prediction. However, traditional perturbations on graph data (Ying et al. 2019; Vu and Thai 2020) like masking nodes or edges are inapplicable to our setting because multiple fine-grained explanations for $M_G(v_t)$ may share the same perturbed graph object and the prediction change caused by the perturbation is not specific to any of the explanations. In Example 1 , masking a_2 perturbs the influences of (a_2, P_1) and (a_2, P_2) on $M_G(a_1)$ simultaneously. To address the issue, our idea is to perturb all the *walks* that participate in propagating the information of v to v_t with respect to P, where a walk corresponds to the trajectory of a possible information flow in the HGN. We develop a novel graph rewiring algorithm such that all the walks facilitating the information of v to flow to v_t along P are blocked and the other walks are not affected.

Second, we are interested in highly influential fine-grained explanations on $M_G(v_t)$, i.e., the salient part for the prediction. However, a full enumeration of possible fine-grained explanations for $M_G(v_t)$ can be prohibitively expensive or unaffordable as the number of simple paths is exponential to the number of edges. We design an efficient search algorithm to explore the space of fine-grained explanations in a greedy manner. It generally follows the breadth-first search process but expands a small number of explanations with highest influence scores in each layer. The experiments show the search is able to find faithful fine-grained explanations with high efficiency.

To summarize, this paper introduces an efficient fine-grained explanation framework named xPath for black-box HGNs on node classification tasks. The major contributions are as follows. **(1)** We propose a new fine-grained explanation scheme to explain black-box HGNs on node classification tasks. Each explanation involves a cause node to the prediction and a path specifies how the node affects the prediction. **(2)** We develop a novel graph rewiring algorithm to perturb the walks associated with the path in a fine-grained explanation and quantify the influence of a node on the prediction w.r.t. each individual path to the target node. **(3)** We design a greedy search algorithm to find top-K most influential fine-grained explanations efficiently. **(4)** Extensive experiments on various HGNs and real-world heterogeneous graphs demonstrate xPath can find explanations that are faithful to HGNs and outperform the adaptations of the state-of-the-art GNN explanation methods.[1] To the best of our knowledge, this work is the first explanation approach for black-box HGNs. The proposed fine-grained explanations distinguish influence paths with different semantics, allowing practitioners to understand the reasoning process of HGNs on complex heterogeneous graphs.

Preliminaries

Definition 1 (Heterogeneous Graph G) *A heterogeneous graph $G = (V, E, \phi, \psi)$ consists of a node set V and a directed edge set E, where $\phi : V \to T$ is the node type mapping function and $\psi : E \to S$ is the edge type mapping function. T and S denote the respective sets of predefined node types and edge types, where $|T| + |S| > 2$.*

Definition 2 (Heterogeneous GNN (HGN) M_G) *Let $M_G : V \to C$ be a trained heterogeneous graph neural network model that performs node classification on the heterogeneous graph G, where C is the label set. For any node $v_t \in V$, $M_G(v_t)$ is the model prediction.*

In this paper, we assume M_G performs message passing along the directed edges in G, which holds for most existing HGNs (Schlichtkrull et al. 2018; Fu et al. 2020; Hu et al. 2020; Lv et al. 2021). Apart from this assumption, we treat HGN as a black-box model without requiring the detailed model architecture and parameters.

Definition 3 (Fine-Grained Explanation X_G) *To explain a model prediction $M_G(v_t)$, we propose a fine-grained explanation $X_G(v_t)$ in the form of (v, P) where $v \in V \backslash \{v_t\}$ is regarded as the cause for the prediction and $P = \langle v, \cdots, v_t \rangle$ is a directed simple (acyclic) path connecting v with v_t (according to the edge set E) indicating the way in which v affects $M_G(v_t)$.*

Let $\mathcal{P}_G(v_t)$ denote all the simple paths in the graph ended with v_t. We can obtain the set of all the possible fine-grained explanations for $M_G(v_t)$ as: $\mathcal{X}_G(v_t) = \{X_G(v_t) = (v, P) \mid$

[1]Source code at https://github.com/LITONG99/xPath.

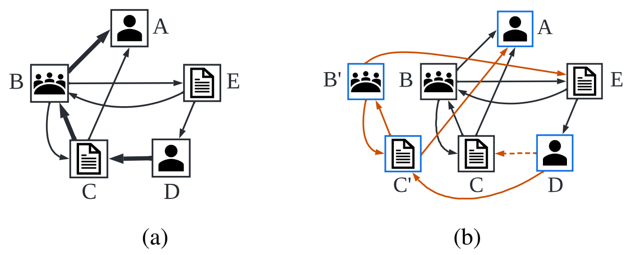


Figure 2: (a) Illustration of the fine-grained explanation ($v = D$, $P = \langle D, C, B, A\rangle$) for the prediction of A in graph G. (b) The rewired graph G_R^P with B', C' as proxies.

$P = \langle v, \cdots, v_t\rangle \in \mathcal{P}_G(v_t)\}$. Among all the fine-grained explanations in $\mathcal{X}_G(v_t)$, we are interested in the most influential ones. We denote by $\mathsf{IF}(X_G(v_t))$ the *influence* of node v on the prediction $M_G(v_t)$ w.r.t. path P, which is also called the *influence score* of $X_G(v_t)$ for simplicity.

Problem Formulation. For each prediction made by a black-box HGN, the goal of this paper is to find K fine-grained explanations with the highest influence scores.

Methodology

In this section, we first introduce the notion of walk perturbation which is crucial for computing the $\mathsf{IF}(X_G(v_t))$. We then present a novel graph rewiring algorithm for walk perturbation and discuss its properties. Finally, we describe our explanation framework xPath that can find top-K fine-grained explanations for each prediction efficiently.

Walk Perturbation

We now consider a fine-grained explanation $X_G(v_t)$ and want to measure $\mathsf{IF}(X_G(v_t))$. As described before, traditional perturbation techniques such as node masking and edge dropping fail to untangle the influences of multiple fine-grained explanations that have overlapping nodes.

Our key insight is that the HGN model performs actual information propagation from v to v_t w.r.t. P via *walks* on the graph. Each walk is an ordered node sequence connected by edges in the graph showing the trajectory of a specific information flow. In Figure 2a, two walks $\langle D, C, B, A\rangle$ and $\langle D, C, B, C, B, A\rangle$ indeed participates in propagating the information of D to A w.r.t. the path $\langle D, C, B, A\rangle$. On the contrary, the walk $\langle D, C, A\rangle$ does not follow the path to propagate information of D to A. Intuitively, if we can identify all the walks associated with P and block them to prevent the information of v from flowing to v_t w.r.t. P, the observed prediction change will imply the $\mathsf{IF}(X_G(v_t))$.

We next formally define walks and specify the walks that are associated with a simple path P.

Definition 4 (Walk W_G) *Given a heterogeneous graph G, a walk W_G is an ordered node sequence $\langle v_1, \cdots, v_L\rangle$ where $\langle v_i, v_{i+1}\rangle \in E$ for any $i \in [1, L-1]$ and $L \geq 2$. Note that different from simple paths, a walk can involve (self-)loops.*

Definition 5 (Walk $\mathcal{W}_G^P$ Associated With P) *Given a fine-grained explanation $X_G(v_t) = (v, P)$ for the prediction $M_G(v_t)$, a walk $W_G = \langle v_1, \cdots, v_L\rangle$ is associated with P iff it includes a suffix $\langle v_l, \cdots, v_L\rangle$ satisfying the following three conditions:*
(i) $v_l = v \wedge v_L = v_t \wedge \forall l \leq i < L, v_i \neq v_t$;
(ii) for all $i \in [l, L-1]$, we have $v_i = v_{i+1}$ or exactly one of the two edges $\langle v_i, v_{i+1}\rangle$, $\langle v_{i+1}, v_i\rangle$ exists in the path P;
(iii) after erasing all the loops in the suffix, we obtain exactly the same path as P.

We denote by $\mathcal{W}_G^P$ the set of all the walks associated with P and have the following theorem.

Theorem 1 *Given two different fine-grained explanations $X_{G,1}(v_t) = (v_1, P_1)$ and $X_{G,2}(v_t) = (v_2, P_2)$ where $P_1 \neq P_2$ for the prediction $M_G(v_t)$, we have $\mathcal{W}_G^{P_1} \neq \mathcal{W}_G^{P_2}$.*

Based on Theorem 1, we can distinguish the influence of v w.r.t. different simple paths connecting v to v_t, and measure the influence of v on the prediction $M_G(v_t)$ w.r.t. P by perturbing all the walks in $\mathcal{W}_G^P$. Henceforth, we may use X_G^P to denote a fine-grained explanation where v and v_t are the start and end nodes in P.

Graph Rewiring for Walk Perturbation

To perturb $\mathcal{W}_G^P$ for a fine-grained explanation $X_G(v_t) = (v, P)$, we introduce a novel graph rewiring algorithm. The algorithm takes G and P as inputs and produces a rewired graph G_R^P which guarantees the walks in $\mathcal{W}_G^P$ are blocked on G_R^P and the other walks are reserved.

Let $P = \langle v(v_0), v_1, \cdots, v_L, v_t(v_{L+1})\rangle$, $L \geq 0$. The graph rewiring algorithm involves the following two steps.

• *Step 1: creating proxy nodes.* It first creates a proxy node $\mathrm{proxy}(v_i)$ for every node v_i ($i \in [1, L]$) in P. The feature vector and the type of each proxy node are the same as those of the original node. For ease of presentation, we let $v = \mathrm{proxy}(v)$ and $v_t = \mathrm{proxy}(v_t)$ though v and v_t do not have actual proxy nodes in the graph.

• *Step 2: establishing edges.* The algorithm then establishes edges for the L proxy nodes. For v_i ($i \in [1, L]$), we denote by $\mathsf{InE}(v_i)$ and $\mathsf{OutE}(v_i)$ the original sets of in-edges and out-edges of v_i in G, respectively. For $\langle u, v_i\rangle \in \mathsf{InE}(v_i)$, if (i) it is a self-loop **or** (ii) exactly one of the two edges $\langle u, v_i\rangle$, $\langle v_i, u\rangle$ exists in P, we add a new edge from $\mathrm{proxy}(u)$ to $\mathrm{proxy}(v_i)$. For $\langle v_i, u\rangle \in \mathsf{OutE}(v_i)$, if (i) it is not a self-loop **and** (ii) both $\langle v_i, u\rangle$ and $\langle u, v_i\rangle$ are not contained in P, we add a new edge from $\mathrm{proxy}(v_i)$ to u. Each newly added edge shares the same feature vector and the type of the original edge. Finally, we remove the first edge $\langle v, v_1\rangle$ in P from the graph.

The resultant graph is the rewired graph G_R^P. The overall process is formalized in Algorithm 1.

Correctness Guarantee. We first use an example to show intuitions on why G_R^P is able to block $\mathcal{W}_G^P$ without interfering with other walks. Without loss of generality, we assume every node and edge in G has a distinct type. Informally, two walks are said to be *equivalent* if their corresponding nodes and edges share the same feature vector and type.

Example 2 *Figure 2a shows a path $P = \langle D, C, B, A\rangle$ in graph G. The rewired graph G_R^P in Figure 2b blocks P by*

Algorithm 1: Rewiring algorithm

```
Input: $G, P = \langle v, v_1, \cdots, v_L, v_t\rangle$
 1: $V \leftarrow V \cup \bigcup_{i=1}^{L}\{\text{proxy}(v_i)\}$
 2: for $i \in [1, L]$ do
 3:   for $\langle u, v_i\rangle \in \mathsf{InE}(v_i)$ do
 4:     if $u = v_i \vee \langle u, v_i\rangle \in P \vee \langle v_i, u\rangle \in P$ then
 5:       $E \leftarrow E \cup \{\langle \text{proxy}(u), \text{proxy}(v_i)\rangle\}$;
 6:     end if
 7:   end for
 8:   for $\langle v_i, u\rangle \in \mathsf{OutE}(v_i)$ do
 9:     if $v_i \neq u \wedge \langle u, v_i\rangle \notin P \wedge \langle v_i, u\rangle \notin P$ then
10:       $E \leftarrow E \cup \{\langle \text{proxy}(v_i), u\rangle\}$;
11:     end if
12:   end for
13: end for
14: $E \leftarrow E \backslash \{\langle v, v_1\rangle\}$;
15: Return $G$;
```

deleting $\langle D, C\rangle$ and disconnects $\text{proxy}(B)$ *from A. Consider a walk $W_1 = \langle D, C, B, C, B, A\rangle \in \mathcal{W}_G^P$. To mimic W_1 on G_R^P, we go from D to* $\text{proxy}(C)$ *but after that we can only walk between proxy nodes without having a chance to transit to A. This implies W_1 is blocked on G_R^P.*

Consider two walks $W_2 = \langle E, B, A\rangle$ and $W_3 = \langle D, C, A\rangle$ which are not associated with P. It is clear that W_2 does not contain $\langle D, C\rangle$ and it still exists in G_R^P. For W_3, it contains an edge $e = \langle C, A\rangle$ after $\langle D, C\rangle$ which satisfies (i) e is not a self-loop and (ii) both e and its reverse edge are not contained in P. Hence, we can find a walk on G_R^P that transits from D to $\text{proxy}(C)$ *and goes back to A, i.e., $\langle D, \text{proxy}(C), A\rangle$, which is equivalent to W_3.*

Definition 6 (Equivalent Walks) *Let $W = \langle v_1, \cdots, v_L\rangle$ be a walk on G and $W' = \langle v'_1, \cdots, v'_L\rangle$ be a walk on G_R^P. W, W' are equivalent iff (i) for any $i \in [1, L]$, v_i, v'_i have the same feature vector and the node type, and (ii) for any $i \in [1, L-1]$, $\langle v_i, v_{i+1}\rangle, \langle v'_i, v'_{i+1}\rangle$ share the same feature vector and the edge type. We denote as $W = W'$.*

Let $\mathcal{W}_G^{v_t}$ denote the set of all the walks on G which end with node v_t. We have the following theorem.

Theorem 2 *For any fine-grained explanation $X_G(v_t) = (v, P)$ for the prediction $M_G(v_t)$, Algorithm 1 produces a rewired graph G_R^P that satisfies: (i) $\mathcal{W}_G^P \cap \mathcal{W}_{G_R^P}^{v_t} = \emptyset$; and (ii) $\mathcal{W}_G^P \cup \mathcal{W}_{G_R^P}^{v_t} = \mathcal{W}_G^{v_t}$.*

The above theorem conveys the correctness of our rewiring algorithm in two aspects. First, all the walks that are associated with P in G are blocked on G_R^P. Second, for any walk on G which ends with v_t and is not associated with P, we can find an equivalent walk on G_R^P.

Fine-Grained Explainability for HGNs

Before introducing our explanation framework, we first formulate the notion of *influence score* of a fine-grained explanation. Given a fine-grained explanation X_G^P for the prediction $M_G(v_t)$, we use the rewired graph G_R^P produced by

Algorithm 2: Finding top-K fine-grained explanations

```
Input: $v_t$, $M_G$, maximum iteration $L_{max}$, sample size $m$,
candidate size $b$
 1: $B \leftarrow \{\langle v_t\rangle\}; U \leftarrow \emptyset; i \leftarrow 0$;
 2: while $B \neq U$ and $i < L_{max}$ do
 3:   $B \leftarrow U; U \leftarrow \emptyset; i \leftarrow i + 1$;
 4:   for $P \in B$ and $|P| = i$ do
 5:     Randomly sample $m$ one-step extended paths from
        $P$ and add them to $U$;
 6:   end for
 7:   $U \leftarrow b$ paths in $B \cup U$ with highest influence scores;
 8: end while
 9: Return top-$K$ fine-grained explanations based on $B$;
```

Algorithm 1 to perturb all the walks in $\mathcal{W}_G^P$. We compute the model predictions on v_t given G and G_R^P, i.e., $M_G(v_t)$ and $M_{G_R^P}(v_t)$. The influence score of X_G^P is measured based on the significance of the prediction change.

Definition 7 (Influence Score of X_G^P) *Let $M_G(v_t)$ and $M_{G_R^P}(v_t)$ denote the model predictions on v_t given G and G_R^P, respectively. The influence score of X_G^P is defined as:*

$$\mathsf{IF}(X_G^P) = (-1)^{\mathbb{1}_{y=y'}} + (M_G(v_t)[y] - M_{G_R^P}(v_t)[y]), \qquad (1)$$

where $y = \arg\max_c M_G(v_t)$ and $y' = \arg\max_c M_{G_R^P}(v_t)$.

The first term in Eq. (1) measures the change in the predicted label and the second term focuses on the change in the probability of the predicted label. A higher $\mathsf{IF}(X_G^P)$ means blocking all the walks in $\mathcal{W}_G^P$ will change the prediction more significantly, indicating X_G^P is more critical to the prediction. Moreover, if blocking the walks in $\mathcal{W}_G^P$ does not change the prediction label (i.e., y equals y'), we have $\mathsf{IF}(X_G^P) \in [-2, 0]$. Otherwise, we have $\mathsf{IF}(X_G^P) \in [0, 2]$. Note that $\mathsf{IF}(X_G^P) < -1$ means blocking all the walks associated with P actually increases the probability of the predicted label. We regard such explanations as invalid.

Given a prediction $M_G(v_t)$, our explanation framework aims to find K fine-grained explanations with the highest influence scores. While each fine-grained explanation is exclusively specified by its simple path, i.e., the start node of the path is the cause for the prediction, it is intractable to enumerate all the simple paths ending with v_t and compute their influence scores. To avoid exhaustive search, we develop a greedy search algorithm, which is summarized in Algorithm 2. The core idea is to explore simple paths in order of their lengths. We maintain a set B of at most b simple paths. B is initialized by a trivial path $\langle v_t\rangle$ of length zero, which only used to aid the algorithm but not considered as a real simple path. At the i-th iteration ($i \in [0, L_{max})$), for each path of length i in B, we randomly extend it by one-step m times (based on the edges in G) and obtain m paths of length $i + 1$. We use U to maintain all the newly explored paths of length $i + 1$. We then find at most b paths in $B \cup U$ with the highest influence scores and update B accordingly. The iteration is repeated until all the paths in B have been extended or the maximum iteration L_{max} is reached, where

L_{max} can be set as the number of model layers. Finally, we use the reverse of K most-influential paths to form the top-K fine-grained explanations.

Time Complexity Analysis. Algorithm 2 computes influence scores for at most bmL_{max} fine-grained explanations. The time cost of graph rewiring is O(pd), where p is the maximum length of paths in explanations and d is the maximum node's degree in G. The time cost of computing prediction change is O($|M_G|$), where $|M_G|$ denotes the number of model parameters. Hence, the total time complexity of Algorithm 1 is O($bmL_{max}(pd+|M_G|)$). In practice, the values of b, m, L_{max}, p are small constant numbers and $p \le L_{max}$. The empirical time cost of finding top-K fine-grained explanations is linearly proportional to the model inference time.

Experiments

Experimental Settings

Datasets. We conduct experiments using three public heterogeneous graph datasets for node classification tasks. (1) **ACM** (Wang et al. 2021a) is an academic network with node types: *paper* (P), *author* (A) and *subject* (S). The task is to classify papers into three topics. (2) **DBLP** (Wang et al. 2021a) is a bibliography graph with node types: *author* (A), *paper* (P), *conference* (C) and *term* (T). The task is to classify authors into four research areas. (3) **IMDB**[2] is a movie graph with node types: *movie* (M), *director* (D), and *actor* (A). The statistics of the datasets are in Table 1.

HGN Models and Training Details. We evaluate the performance of explanation methods on three advanced HGNs: a 2-layer SimpleHGN (Lv et al. 2021), a 3-layer SimpleHGN and a 2-layer HGT (Hu et al. 2020), which are abbreviated as **SIM2**, **SIM3** and **HGT**. The node embedding size is set to 32 in all the models. To train these models, we split each dataset into training/validation/test sets and we randomly reserve 1000/1000 samples for validation/test. Note that HGT contains more parameters and needs more training samples to reach good performance. While SimpleHGN is parameter-efficient, its performance increases insignificantly with more training samples. We train HGT with 2000 samples and train SimpleHGN with 60 samples per label for training efficiency. Table 1 provides the test accuracy.

Compared Explanation Methods. Existing explainability techniques are proposed for GNNs and there is no off-the-self ground-truth explanations for the predictions from HGNs. Hence, we implement two basic explanation methods and adapt four state-of-the-art model-agnostic GNN explanation approaches as comparison methods. (1) **Local** only considers graph structure and selects nodes with the shortest distances to the target node. It is task-agnostic and generates the same explanation for all the HGNs. (2) **Attention** utilizes the attention scores computed by HGNs during message aggregation. For a node, we consider all the paths to the target node. We multiply the attention scores of the edges in each path, and add them up as node importance to the prediction. It is a model-specific explanation method. (3) **PGM-Explainer** (Vu and Thai 2020) perturbs node features to construct a dataset characterizing local data distribution and generates node-level explanations in form of PGMs. To adapt it to HGNs, we perturb a node by replacing its feature vector with the average feature vector of the nodes with the same type. (4) **ReFine** (Wang et al. 2021b) is a learning-based method which provides edge-level explanations. It uses a graph encoder followed by an MLP layer to produce the probability of an edge in the explanation, and trains the model using fidelity loss and contrastive loss. We use HGNs to instantiate the graph encoder. (5) **Gem** (Lin, Lan, and Li 2021) provides edge-level explanations. It involves a distillation process to obtain ground-truth explanations in advance and trains a model that predicts the probabilities of edges in the explanations. We use HGNs to implement the node embedding module. (6) **SubgraphX** (Yuan et al. 2021) provides subgraph-level explanations. It performs the Monte Carlo tree search to explore different subgraphs and measures their importance. To adapt it to HGNs, we simply ignore all the type information during the search process.

[2]https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset

Dataset	#nodes	#node types	#edges	#edge types	SIM2	SIM3	HGT
ACM	11246	3	46098	7	90.4	91.3	89.7
DBLP	26128	4	265694	10	95.8	95.5	87.3
IMDB	11616	3	45828	7	61.6	61.1	72.4

Table 1: The statistics of the datasets and the test accuracy(%) of the HGNs.

Evaluation Metrics. A faithful explanation should involve graph objects (e.g., nodes, edges, subgraphs) that are necessary and sufficient to recover the original prediction. Without ground-truth explanations, we follow the previous work (Yuan et al. 2020) and measure faithfulness by two fidelity metrics: the *accuracy fidelity* F^{acc} measures the prediction change and the *probability fidelity* F^{prob} studies the probability change on the original predicted label. Specifically, let $\mathcal{T}$ denote test samples correctly predicted by the model. For a test sample (v_t, y), we induce a graph $\tilde{G}$ based on all the involving graph objects in its explanation. We then compute the prediction $M_{\tilde{G}}(v_t)$ and the predicted label $\tilde{y}$ based on $\tilde{G}$. Formally, we compute metrics by:

$$F^{acc} = \frac{1}{|\mathcal{T}|} \sum_{(v_t,y)\in\mathcal{T}} (1 - \mathbb{1}_{y=\tilde{y}}), \tag{2}$$

$$F^{prob} = \frac{1}{|\mathcal{T}|} \sum_{(v_t,y)\in\mathcal{T}} (M_G(v_t)[y] - M_{\tilde{G}}(v_t)[y]). \tag{3}$$

Remark. xPath focuses on evaluating the influence of a node on the prediction w.r.t. a specific path. It is unreasonable to regard the reported top-K fine-grained explanations with the highest influence scores as a subgraph composed of K paths. This is because our rewiring algorithm only perturbs the walks that are associated with the path in a fine-grained explanation, rather than treating the path as a subgraph and excluding all the involving graph objects from the graph. However, the two fidelity metrics actually treat our top-K fine-grained explanations as a subgraph explanation and measure

Dataset	Metric	Model	Local	Attention	PGM-Explainer	ReFine	GEM	SubgraphX	xPath
ACM	F^{acc}	SIM2	$\underline{2.7}$	4.2	18.1±0.31	17.2±1.25	4.2±0.00	3.8±0.49	**0.2*****±0.09**
		SIM3	5.4	6.5	31.7±0.34	26.0±0.14	26.9±0.01	$\underline{4.6\pm0.20}$	**4.1±0.23**
		HGT	$\underline{0.7}$	**0.2**	46.7±0.52	4.2±1.67	1.9±0.00	1.4±0.33	5.5±0.19
	F^{prob}	SIM2	0.7	1.2	6.8±0.08	6.2±0.31	1.2±0.00	$\underline{0.6\pm1.23}$	**-0.9*****±0.02**
		SIM3	5.4	2.4	11.4±0.02	10.9±0.05	11.3±0.00	$\underline{1.5\pm0.31}$	**0.7*** **±0.09**
		HGT	$\underline{0.6}$	**0.003**	35.2±0.34	3.3±0.99	1.6±0.00	1.0±0.07	1.9±0.30
DBLP	F^{acc}	SIM2	12.1	6.8	2.3±0.15	4.7±0.10	11.4±0.15	$\underline{0.7\pm0.57}$	**0.6±0.09**
		SIM3	7.0	$\underline{4.9}$	31.9±0.26	14.5±0.75	32.1±0.00	5.9±0.05	**0.3*****±0.05**
		HGT	3.0	2.5	6.2±0.24	7.7±1.12	6.1±0.00	$\underline{0.8\pm0.06}$	**0.3*****±0.00**
	F^{prob}	SIM2	4.2	2.2	0.8±0.05	0.9±0.10	3.8±0.00	**-0.8±3.43**	$\underline{-0.5\pm0.01}$
		SIM3	2.7	$\underline{1.7}$	11.3±0.03	5.3±0.67	11.4±0.01	2.1±0.31	**-0.4*** **±0.01**
		HGT	3.9	3.5	5.9±0.22	6.2±0.68	6.3±0.00	**-2.3±0.07**	$\underline{-1.0\pm0.03}$
IMDB	F^{acc}	SIM2	6.3	6.3	6.0±0.93	13.4±0.81	18.8±8.49	$\underline{2.4\pm0.38}$	**0.9*****±0.08**
		SIM3	10.1	10.1	9.4±1.15	16.8±0.82	17.1±1.20	$\underline{2.8\pm0.05}$	**2.5±0.00**
		HGT	$\underline{0.1}$	$\underline{0.1}$	3.4±0.68	4.9±0.66	13.5±2.20	1.4±0.02	**0.0±0.00**
	F^{prob}	SIM2	1.3	1.3	1.0±0.20	3.1±0.15	5.1±2.68	$\underline{-1.7\pm2.31}$	**-2.8±0.01**
		SIM3	2.3	2.3	2.0±0.57	4.8±0.45	5.1±0.20	$\underline{-1.6\pm0.51}$	**-2.0*****±0.01**
		HGT	$\underline{0.02}$	$\underline{0.02}$	4.8±0.22	4.0±0.24	13.3±2.57	0.1±0.14	**-2.8*****±0.00**

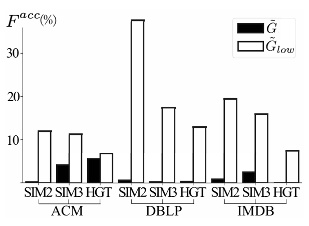
Table 2: Comparison results on fidelity metrics(%). The best results are in bold and the second-best results are underlined. And $*$ denotes statistically significant improvement (measured by t-test with p-value < 0.01) over all baselines.

the effectiveness of the subgraph induced by the paths. In this regard, the two metrics do not act fairly to xPath.

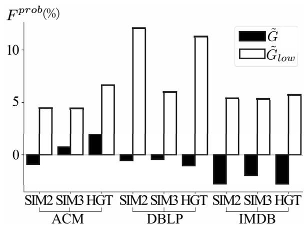
Implementation Details. We implemented xPath with PyTorch. For the four advanced explanation approaches, we used their original source codes and incorporated simple adaptations as described before to make them fit heterogeneous graphs. For xPath, we tune the hyperparameters (b, m) of ACM to $(5, 5)$, DBLP with SIM2 to $(10, 10)$, and others to $(2, 10)$. Since explanation sparsity is highly related to fidelity scores, we control the number of nodes involved in the explanations. As suggested by the previous work (Ying et al. 2019), we set the node number to 5 to avoid overwhelming users. As the comparison methods may not find explanations involving exactly 5 nodes, we tune their hyperparameters to find explanations with roughly 5 nodes that achieve the best fidelity. We conducted all the experiments on a server equipped with Intel(R) Xeon(R) Silver 4110 CPU, 128GB Memory, and a Nvidia GeForce RTX 2080 Ti GPU (12GB Memory). Each experiment was repeated 5 times and the average performance was reported.

Evaluation of Effectiveness

Comparison Results. Table 2 shows the fidelity performance of all the comparison methods. On IMDB, Local and Attention have the same results because these basic methods give the same explanations. For both metrics, lower fidelity scores indicate the graph objects in the explanations are more useful to retain the original predictions, and hence the explanations are more faithful to the model. We have the following key observations. First, xPath achieves the lowest fidelity scores in most cases, demonstrating its effectiveness in identifying explanations that are faithful to the models. Interestingly, xPath reports negative probability fidelity values in some cases, which indicates xPath is capable of filtering out noisy information and identifying information flows that are critical to the predictions. When explaining HGT on ACM dataset, xPath performs worse than Attention. We investigate the test samples in ACM where the explanations from xPath fail to provide the original predictions, and find that the paths in these explanations do not contain nodes of the *author* type. This is because the *subject* nodes have much larger degrees than the *author* nodes and the paths following *paper-subject-paper* dominate all the paths ending with a target paper. If many of these paths are indeed influential to HGT, they will occupy the top-K position, making top-K explanations lack *author* type information. To verify our claim, for each test sample in ACM, we look up paths examined during the search process for those following *author-paper* and *paper-author-paper*, and use the one with the highest influence score to replace the K-th path identified by xPath. This simple strategy achieves better results (F^{acc}=0.0% and F^{prob}=-2.0%) than Attention. This indicates xPath is extensible to explore the correlations among explanations towards better fidelity. On probability fidelity, xPath performs slightly worse than SubgraphX on DBLP. This is because the influence score is defined to be less sensitive to the change in probabilities and xPath pays more attention to avoiding label change, i.e., better accuracy fidelity than SubgraphX. Second, among the adaptations of four advanced explanation methods, SubgraphX is superior in all cases. This is consistent with the intuition that high-order explanations(e.g., subgraphs) are more expressive and powerful than low-order ones(e.g., nodes, edges). PGM-Explainer is sensitive to graph data. Its fidelity scores increase sharply on ACM and DBLP than IMDB because the node neighborhoods in ACM and DBLP is larger, which increases the difficulty of obtaining accurate local data distributions via sampling-based perturbations. GEM and ReFine fail to find good explanations, which suggests that edge-based explanations are insufficient to retain the rich semantics that are useful to the HGN predictions. We also notice that Attention is a strong baseline thanks to the knowledge of model details.



(a) Accuracy fidelity



(b) Probability fidelity

Figure 3: Effectiveness of influence scores in differentiating faithful explanations from unfaithful ones.

Local provides task-agnostic explanations and its performance is unstable over different model architectures. Third, xPath performs steadily on different HGNs and tasks. On DBLP, the four existing explanation methods report much higher accuracy fidelity on SIM3 than on SIM2. We notice that SIM3 involves larger computation graphs than SIM2 (about two orders of magnitude in the number of nodes), and existing explanation methods fall short in the presence of intricate graph structures. xPath achieves best fidelity when explaining SIM3 on all three tasks, showing the potential of applying xPath on complex HGNs and large graphs.

Effectiveness of Influence Scores. We study whether proposed influence score (Eq. 1) can differentiate faithful explanations from unfaithful ones. We consider all the paths examined during the search process and induce another graph $\tilde{G}_{low}$ based on the K paths with the lowest influence scores. Figure 3 provides the fidelity scores computed based on $\tilde{G}$ and $\tilde{G}_{low}$. In all the cases, $\tilde{G}_{low}$ consistently result in poor fidelity. The performance gap between two groups confirms the effectiveness of the influence score function in distinguishing different fine-grained explanations.

Evaluation of Efficiency

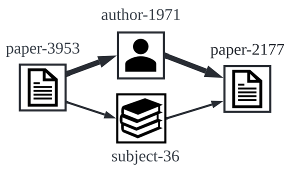
We report the running time in computing explanations for all the test samples in Table 3. xPath is much more efficient than the existing explanation methods, thanks to the greedy search algorithm. The sampling process in PGM-Explainer and the Monte Carlo search in SubgraphX incur high time costs when the target node has a large neighborhood, e.g., SIM3 on DBLP. GEM is efficient in many cases but also suffers in explaining SIM3 due to large node neighborhood, because it calculates the probability of edges between pairwise nodes in the neighborhood. ReFine generally runs faster than PGM-Explainer and SubgraphX. Note that the time cost of the learning-based methods GEM and ReFine mainly comes from training the explanation generators and the time for finding explanations for test samples can be amortized.

Case Study

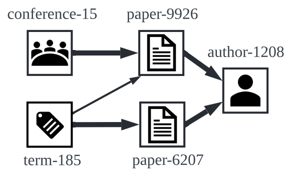
We conduct two case studies to provide intuitions on how xPath answers the "what" and "how" questions via fine-grained explanations. As shown in Figure 4a, when explaining the prediction of *paper*-2177, xPath identifies two influential fine-grained explanations with paths ⟨*paper*-3953, *author*-1971, *paper*-2177⟩ and ⟨*paper*-3953, *subject*-36, *paper*-2177⟩, whose influence scores are 1.04 and -1.00, respectively. This means the model made the prediction mainly because *paper*-2177, *paper*-3953 are written by the same author. Figure 4b shows the neighborhood of the target node *author*-1208 in DBLP. xPath finds two nodes *conference*-15 and *term*-185 that are important for the prediction. The most influential path for *conference*-15 is ⟨*conference*-15, *paper*-9926, *author*-1208⟩, while that for *term*-185 is ⟨*term*-185, *paper*-6207, *author*-1208⟩. xPath highlights two cause nodes and their influential paths, indicating the model made the prediction of *author*-1208 because the author has written *paper*-9926 in *conference*-15 and published *paper*-6207 involving *term*-185. Arguably, xPath can provide explanations with legible semantics, which is a desirable property in explaining HGNs on complex heterogeneous graphs.

Dataset	Model	PGM-Explainer	ReFine	GEM	Sub-graphX	xPath
ACM	SIM2	19.6	4.6	2.6	3.5	0.2
	SIM3	18.6	5.8	45.5	13.5	0.6
	HGT	16.7	3.6	2.6	3.9	0.2
DBLP	SIM2	12.4	5.9	9.7	7.5	0.6
	SIM3	46.0	11.0	84.5	27.9	1.2
	HGT	11.3	4.7	3.3	8.0	0.2
IMDB	SIM2	3.1	3.0	1.5	2.6	0.1
	SIM3	10.2	4.8	22.6	9.0	0.1
	HGT	5.5	3.5	3.0	5.2	0.1

Table 3: The time of computing explanations (in hours).



(a)



(b)

Figure 4: Case studies: (a) explaining SIM3 on *paper*-2177 in ACM; (b) explaining SIM3 on *author*-1208 in DBLP.

Conclusion

In this paper, we study the problem of explaining black-box HGNs on node classification tasks. We propose a new explanation framework named xPath which provides fine-grained explanations in the form of a node associated with its influence path to the target node. The node tells what is important to the prediction and the influence path indicates how the prediction is affected by the node. In xPath, we develop a novel graph rewiring algorithm to perform walk-level perturbation and measure the influence score of any fine-grained explanation without the knowledge of model details. We further introduce a greedy search algorithm to find top-K most influential explanations efficiently. Extensive experimental results show that xPath can provide explanations that are faithful to various HGNs with high efficiency, outperforming the adaptations of known explainability techniques.

Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful reviews and the deep learning computing framework MindSpore[3] for the support on this work. This work is supported by the National Key Research and Development Program of China (2022YFE0200500), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and SJTU Global Strategic Partnership Fund (2021 SJTU-HKUST).

References

Baldassarre, F.; and Azizpour, H. 2019. Explainability Techniques for Graph Convolutional Networks. In *International Conference on Machine Learning (ICML) Workshops, 2019 Workshop on Learning and Reasoning with Graph-Structured Representations*.

Dou, Y.; Liu, Z.; Sun, L.; Deng, Y.; Peng, H.; and Yu, P. S. 2020. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 315–324.

Fu, X.; Zhang, J.; Meng, Z.; and King, I. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of The Web Conference 2020*, 2331–2341.

Hu, Z.; Dong, Y.; Wang, K.; and Sun, Y. 2020. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, 2704–2710.

Li, J.; Peng, H.; Cao, Y.; Dou, Y.; Zhang, H.; Yu, P.; and He, L. 2021. Higher-order attribute-enhancing heterogeneous graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*.

Lin, W.; Lan, H.; and Li, B. 2021. Generative causal explanations for graph neural networks. In *International Conference on Machine Learning*, 6666–6679. PMLR.

Liu, Y.; Ao, X.; Qin, Z.; Chi, J.; Feng, J.; Yang, H.; and He, Q. 2021. Pick and choose: a GNN-based imbalanced learning approach for fraud detection. In *Proceedings of the Web Conference 2021*, 3168–3177.

Luo, D.; Cheng, W.; Xu, D.; Yu, W.; Zong, B.; Chen, H.; and Zhang, X. 2020. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33: 19620–19631.

Lv, Q.; Ding, M.; Liu, Q.; Chen, Y.; Feng, W.; He, S.; Zhou, C.; Jiang, J.; Dong, Y.; and Tang, J. 2021. Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1150–1160.

Pope, P. E.; Kolouri, S.; Rostami, M.; Martin, C. E.; and Hoffmann, H. 2019. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10772–10781.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Berg, R. v. d.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, 593–607. Springer.

Schlichtkrull, M. S.; De Cao, N.; and Titov, I. 2020. Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking. In *International Conference on Learning Representations*.

Schnake, T.; Eberle, O.; Lederer, J.; Nakajima, S.; Schutt, K. T.; Mueller, K.-R.; and Montavon, G. 2021. Higher-Order Explanations of Graph Neural Networks via Relevant Walks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01): 1–1.

Vu, M.; and Thai, M. T. 2020. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. *Advances in neural information processing systems*, 33: 12225–12235.

Wang, X.; Bo, D.; Shi, C.; Fan, S.; Ye, Y.; and Philip, S. Y. 2022. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. *IEEE Transactions on Big Data*.

Wang, X.; Liu, N.; Han, H.; and Shi, C. 2021a. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1726–1736.

Wang, X.; Wu, Y.; Zhang, A.; He, X.; and Chua, T.-S. 2021b. Towards multi-grained explainability for graph neural networks. *Advances in Neural Information Processing Systems*, 34: 18446–18458.

Yang, Y.; Guan, Z.; Li, J.; Zhao, W.; Cui, J.; and Wang, Q. 2021. Interpretable and efficient heterogeneous graph convolutional network. *IEEE Transactions on Knowledge and Data Engineering*.

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

Yuan, H.; Yu, H.; Gui, S.; and Ji, S. 2020. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445*.

Yuan, H.; Yu, H.; Wang, J.; Li, K.; and Ji, S. 2021. On explainability of graph neural networks via subgraph explorations. In *International Conference on Machine Learning*, 12241–12252. PMLR.

[3]https://www.mindspore.cn/