Generating In-Distribution Proxy Graphs for Explaining Graph Neural Networks

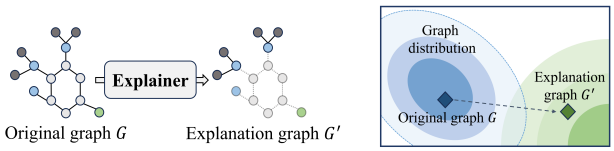
Zhuomin Chen [1] **Jiaxing Zhang** [2] **Jingchao Ni** [3] **Xiaoting Li** [4] **Yuchen Bian** [5] **Md Mezbahul Islam** [1]
Ananda Mohan Mondal [1] **Hua Wei** [6] **Dongsheng Luo** [1]

Abstract

Graph Neural Networks (GNNs) have become a building block in graph data processing, with wide applications in critical domains. The growing needs to deploy GNNs in high-stakes applications necessitate explainability for users in the decision-making processes. A popular paradigm for the explainability of GNNs is to identify explainable subgraphs by comparing their labels with the ones of original graphs. This task is challenging due to the substantial distributional shift from the original graphs in the training set to the set of explainable subgraphs, which prevents accurate prediction of labels with the subgraphs. To address it, in this paper, we propose a novel method that generates proxy graphs for explainable subgraphs that are in the distribution of training data. We introduce a parametric method that employs graph generators to produce proxy graphs. A new training objective based on information theory is designed to ensure that proxy graphs not only adhere to the distribution of training data but also preserve explanatory factors. Such generated proxy graphs can be reliably used to approximate the predictions of the labels of explainable subgraphs. Empirical evaluations across various datasets demonstrate our method achieves more accurate explanations for GNNs.

[1]Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, USA [2]New Jersey Institute of Technology, Newark, USA [3]Department of Computer Science, University of Houston, Houston, USA [4]Visa Research, USA [5]Amazon Search A9, USA [6]School of Computing and Augmented Intelligence, Arizona State University, Tempe, USA. Correspondence to: Zhuomin Chen <zchen051@fiu.edu>, Dongsheng Luo <dluo@fiu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).



(a) Explanation process (b) OOD problem

Figure 1. Examples of the explanation process and out-of-distribution problem. (a) is the explanation process for a graph learning model. The original graph G undergoes an explanation process, resulting in an explanation graph G' that highlights the most significant features and relationships; (b) shows the explanation graph G' is out of distribution where the GNN is trained.

1. Introduction

Graph Neural Networks (GNNs) have emerged as a pivotal technology for handling graph-structured data, demonstrating remarkable performance in various applications including node classification and link prediction (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018; Scarselli et al., 2008). Their growing use in critical sectors such as healthcare and fraud detection has escalated the need for explainability in their decision-making processes (Wu et al., 2022; Li et al., 2022; Zhang et al., 2024). To meet this demand, a variety of explanation methods have been recently developed to interpret the behavior of GNN models. These methods primarily concentrate on identifying a subgraph that significantly impacts the model's prediction for a particular instance (Ying et al., 2019; Luo et al., 2020).

A prominent approach to explain GNNs involves the Graph Information Bottleneck (GIB) principle (Wu et al., 2020). This principle focuses on extracting a compact yet informative subgraph from the input graph, ensuring that this subgraph retains sufficient information for the model to maintain its original prediction. A key aspect of the GIB approach is evaluating the predictive capability of such a subgraph. Typically, this is accomplished by feeding the subgraph into the GNN model and comparing its prediction against that of the complete input graph.

Although it is intuitively correct, the underlying assumption of the aforementioned approach – GNN model can

make accurate predictions on explanation subgraphs – may not always hold. As shown in Figure 1, explanation subgraphs can significantly deviate from the distribution of original graphs, leading to an Out-Of-Distribution (OOD) issue (Zhang et al., 2023b; Fang et al., 2023c;a; Zheng et al., 2024). For instance, in the MUTAG dataset (Debnath et al., 1991), each graph represents a molecule, with nodes symbolizing atoms and edges indicating chemical bonds. The molecular graphs in this dataset usually contain hundreds of edges. In contrast, the NO_2 functional group, identified as a key subgraph influencing positive mutagenicity in a molecule, comprises merely 2 edges. This stark contrast in structural properties leads to a significant difference in the distributions of explanation subgraphs and original graphs. Since the model was trained with original graphs, the reliability of predictions on subgraphs is undermined due to the distribution shifting problem.

Several pioneering studies have attempted to address this distributional challenge (Fang et al., 2023c; Zhang et al., 2023b;a). For example, CGE regards the GNN model as a transparent, fully accessible system. It considers the GNN model as a teacher network and employs an additional "student" network to predict the labels of explanation subgraphs (Fang et al., 2023c). As another example, MixupExplainer (Zhang et al., 2023b) generates a mixed graph for the explanation by blending it with a non-explanatory subgraph from a different input graph. This method posits that the mixup graph aligns with the distribution of the original input graphs. However, this claim is predicated on a rather simplistic assumption that the explanation and non-explanatory subgraphs are independently drawn. However, in real-world applications, these methods often face practical limitations. The dependence on a "white box" model in CGE and the oversimplified assumptions in MixupExplainer are not universally applicable. Instead, GNN models are usually given as "black boxes", and the graphs in real-life applications do not conform to strict independence constraints, highlighting the need for more versatile and realistic approaches to the OOD problem.

In response to these challenges, in this work, we introduce an innovative concept of proxy graphs to the realm of explainable GNNs. These proxy graphs are designed to be both explanation-preserving and distributionally aligned with the training data. By this means, the predictive capabilities of explanation subgraphs can be reliably inferred from the corresponding proxy graphs. We begin with a thorough investigation into the feasible conditions necessary for generating such in-distributed proxy graphs. Leveraging our findings, we further propose a novel architecture that incorporates variational graph auto-encoders to produce proxy graphs. Specifically, we utilize a graph auto-encoder to reconstruct the explainable subgraph and another variational auto-encoder to generate a non-explanatory subgraph. A proxy graph is then obtained by combining two output subgraphs of auto-encoders. We delineate our main contributions as follows:

- We systematically analyze and address the challenge of the OOD issue in explainable GNNs, which is pivotal for enhancing the reliability and interpretability of GNNs in real-world applications.
- We introduce an innovative parametric method that incorporates graph auto-encoders to produce in-distributed proxy graphs that are both situated in the original data distribution and preserve essential explanation information. This facilitates more precise and interpretable explanations in GNN applications.
- Through comprehensive experiments on various real-world datasets, we substantiate the effectiveness of our proposed approach, showcasing its practical utility and superiority in producing explanations.

2. Notations and Preliminary

2.1. Notations and Problem Formulation

We denote a graph G from an graph set $\mathcal{G}$ by a triplet $(\mathcal{V}, \mathcal{E}; \boldsymbol{X})$, where $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ is the node set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ is the node feature matrix, where d is the feature dimension and the i-th row is the feature vector associated with node v_i. The adjacency matrix of G is denoted by $\boldsymbol{A} \in \{0, 1\}^{n \times n}$, which is determined by the edge set $\mathcal{E}$ such that $A_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$, $A_{ij} = 0$, otherwise. In this paper, we focus on the graph classification task, and node classification can be converted to computation graph classification problem (Ying et al., 2019; Luo et al., 2020). Specifically, for graph classification task, each graph G is associated with a label $Y \in \mathcal{Y}$. The to-be-explained GNN model $f(\cdot)$ has been well-trained to classify G into its class, *i.e.*, $f : \mathcal{G} \mapsto \{1, 2, \cdots, |\mathcal{Y}|\}$.

Following the existing works (Ying et al., 2019; Luo et al., 2020; Yuan et al., 2022; Huang et al., 2024), the explanation methods under consideration in this paper are model/task agnostic and treat GNN models as black boxes — *i.e.*, the so-called *post-hoc, instance-level* explanation methods. Formally, our research problem is described as follows:

Problem 1 (Post-hoc Instance-level GNN Explanation)**.** *Given a to-be-explained GNN model $f(\cdot)$ and a set of graphs $\mathcal{G}$, the goal of post-hoc instance-level explanation is to learn a parametric function so that for an arbitrary graph $G \in \mathcal{G}$, it finds a compact subgraph $G^* \subseteq G$ that can "explain" the prediction $f(G)$. The parametric mapping $\Psi_\psi : \mathcal{G} \mapsto \mathcal{G}^*$ is called an explanation function, where $\mathcal{G}^*$ is the alphabet of G^*, and ψ is the parameter of the explanation function.*

2.2. Graph Information Bottleneck as the Objective Function

The Information Bottleneck (IB) principle, foundational in learning dense representations, suggests that optimal representations should balance minimal and sufficient information for predictions (Tishby et al., 2000). This concept has been adapted for GNNs through the Graph Information Bottleneck (GIB) approach, consolidating various post-hoc GNN explanation methods like GNNExplainer (Ying et al., 2019) and PGExplainer (Luo et al., 2020). The GIB framework aims to find a subgraph G^* to explain a GNN model's prediction on a graph G as (Zhang et al., 2023b):

$$G^* = \arg\min_{G'} I(G, G') - \alpha I(Y, G'), \tag{1}$$

where G' is a candidate explanatory subgraph, Y is the label, and α is a balance parameter. The first term is the mutual information between the original graph G and the explanatory subgraph G' and the second term is negative mutual information of Y and the explanatory subgraph G'. At a high level, the first term encourages detecting a small and dense subgraph for explanation, and the second term requires that the explanatory subgraph is label preserving.

Due to the intractability of mutual information between Y and G' in equation 1, some existing works (Wu et al., 2020; Miao et al., 2022) derived a parameterized variational lower bound of $I(Y, G')$:

$$I(Y, G') \geq \mathbb{E}_{G',Y}[\log P(Y|G')] + H(Y), \tag{2}$$

where the first term measures how well the subgraphs predict the labels. A higher value indicates that the subgraphs are, on average, more predictive of the correct labels. The second term $H(Y)$ quantifies the amount of inherent unpredictability or variability in the labels. By introducing equation 2 to equation 1, a tractable upper bound is used as the objective:

$$G^* = \arg\min_{G'} I(G, G') - \alpha \mathbb{E}_{G',Y}[\log P(Y|G')], \tag{3}$$

where $H(Y)$ is omitted due to its independence to G'.

The OOD Problem in GIB. In the existing research, the estimation of $P(Y|G')$ is typically achieved by applying the to-be-explained model $f(\cdot)$ to the input graph G' (Ying et al., 2019; Miao et al., 2022). A critical assumption in these approaches involves the GNN model's ability to accurately predict candidate explanation subgraphs G'. This assumption, however, overlooks the OOD problem, where the distribution of explanation subgraphs significantly deviates from that of the original training graphs (Zhang et al., 2023b; Fang et al., 2023a;c;b; Zheng et al., 2024; Amara et al., 2023). Formally, let $P_{\mathcal{G}}$ be the distribution of the training graphs, and $P_{\mathcal{G}'}$ be the distribution of the explanation subgraphs. The core issue in the GIB objective function is caused by $P_{\mathcal{G}} \neq P_{\mathcal{G}'}$. This distributional disparity undermines the predictive reliability of the model $f(\cdot)$, trained on $P_{\mathcal{G}}$ when applied to subgraphs from $P_{\mathcal{G}'}$. As a result, the predictive power of explanations provided by $f(G')$ is an unreliable approximation of $P(Y|G')$ in equation 3.

3. Graph Information Bottleneck with Proxy Graphs

In this section, we propose the concept of *proxy graphs* to mitigate the aforementioned OOD issue. A proxy graph not only retains the label information present in G' but also conforms to the distribution of the original graph dataset. Specifically, we assume that proxy graphs $\tilde{G}$ are drawn from the distribution $P_{\mathcal{G}}$ and reformulate the estimation of $P(Y|G')$ by marginalizing over the distribution of proxy graphs as follows:

$$P(Y|G') = \mathbb{E}_{\tilde{G}\sim P_{\mathcal{G}}}[P(Y|\tilde{G}) \cdot P(\tilde{G}|G')]. \tag{4}$$

In equation 4, we address the OOD challenge by predicting Y using a proxy graph $\tilde{G}$ instead of directly using G'. This approach is particularly effective when the conditional probability $P(Y|\cdot)$ is approximated by the model $f(\cdot)$. To facilitate the maximization of the likelihood as outlined in equation 4, we further approximate $P(\tilde{G}|G')$ with a parameterized function, denoted as $Q_{\boldsymbol{\phi}}(\tilde{G}|G')$. The formal representation is thus given by

$$P(Y|G') = \mathbb{E}_{\tilde{G}\sim P_{\mathcal{G}}}[P(Y|\tilde{G}) \cdot Q_{\boldsymbol{\phi}}(\tilde{G}|G')], \tag{5}$$

where $\boldsymbol{\phi}$ denotes model parameters.

Given the combinatorial complexity inherent in graph structures, it is computationally infeasible to enumerate all potential proxy graphs $\tilde{G}$ from the unknown distribution $P_{\mathcal{G}}$, which is necessary for calculating equation 5. To overcome this challenge, we propose to approximate $P_{\mathcal{G}}$ with the parameterized function $Q_{\boldsymbol{\phi}}(\tilde{G}|G')$, and estimate equation 5 with a Monte Carlo estimator that samples proxy graphs from $Q_{\boldsymbol{\phi}}(\tilde{G}|G')$. That is

$$P(Y|G') = \mathbb{E}_{\tilde{G}\sim Q_{\boldsymbol{\phi}}(\tilde{G}|G')}[P(Y|\tilde{G})], \tag{6}$$

with constraints

$$Q_{\boldsymbol{\phi}}(\tilde{G}|G') \approx P_{\mathcal{G}}, \quad H(Y|\tilde{G}) \approx H(Y|G'), \tag{7}$$

where the first constraint ensures that $\tilde{G}$ is sampled from a distribution that approximates $P_{\mathcal{G}}$, effectively addressing the OOD challenge. The second constraint guarantees that the label information preserved in $\tilde{G}$ is similar to that in G'.

Therefore, combining equation 3, equation 6 and equation 7, our proxy graph-induced objective function becomes:

$$\begin{aligned}&\arg\min_{G'} I(G,G') - \alpha \mathbb{E}_{G',Y}[\log \mathbb{E}_{\tilde{G}\sim Q_{\boldsymbol{\phi}}(\tilde{G}|G')}[P(Y|\tilde{G})]]\\ &\text{s.t.}\quad Q_{\boldsymbol{\phi}}(\tilde{G}|G') \approx P_{\mathcal{G}}, \quad H(Y|\tilde{G}) \approx H(Y|G').\end{aligned} \tag{8}$$

Bi-level optimization. In equation 8, we formulate a joint optimization loss function that aims to identify the optimal explanation alongside its corresponding proxy graphs. Building upon this, we refine the framework by replacing the first constraint with a distributional distance measure, specifically the Kullback-Leibler (KL) divergence, between $Q_{\boldsymbol{\phi}}(\tilde{G}|G')$ and $P_{\mathcal{G}}$. This leads to the development of a bi-level optimization model. Formally, the model is expressed as follows:

$$\begin{aligned}&\arg\min_{G'} I(G,G') - \alpha \mathbb{E}_{G',Y}[\log \mathbb{E}_{\tilde{G}\sim Q_{\boldsymbol{\phi}^*}(\tilde{G}|G')}[P(Y|\tilde{G})]]\\ &\quad\text{where } \boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathrm{KL}(Q_{\boldsymbol{\phi}}(\tilde{G}|G'), P_{\mathcal{G}}),\\ &\qquad\text{s. t. } H(Y|\tilde{G}) \approx H(Y|G').\end{aligned} \tag{9}$$

3.1. Derivation of Outer Optimization

For the outer optimization objective, we elaborate on instantiating the first term $I(G,G')$. Akin to the original graph, we denote the explanation subgraph G' by $(\mathcal{V}, \mathcal{E}', \boldsymbol{X})$, whose adjacency matrix is $\boldsymbol{A}'$. We follow (Miao et al., 2022) to include a variational approximation distribution $R(G')$ for the distribution $P(G')$. Then, we obtain its upper bound as follows:

$$I(G,G') \leq \mathbb{E}_G[\mathrm{KL}(P(G'|G)||R(G'))]. \tag{10}$$

We follow the Erdős–Rényi model (Erdős et al., 1960) and assume that each edge in G' has a probability of being present or absent, independently of the other edges. Specifically, we assume that the existence of an edge (u,v) in G' is determined by a Bernoulli variable $A'_{u,v} \sim \mathrm{Bern}(\pi'_{uv})$. Thus, $P(G'|G)$ can be decomposed as $P(G'|G) = \prod_{(u,v)\in\mathcal{E}} P(A'_{uv}|\pi'_{uv})$. The bound is always true for any prior distribution $R(G')$. We follow an existing work and assume that in the prior distribution, the existence of an edge (u,v) in G' is determined by another Bernoulli variable $A''_{u,v} \sim \mathrm{Bern}(r)$, where $r \in [0,1]$ is a hyper-parameter (Miao et al., 2022), independently to the graph G. We have $R(G') = P(|\mathcal{E}|)\prod_{(u,v)\in\mathcal{E}} P(A''_{uv})$. Thus, the KL divergence between $P(G'|G)$ and the above marginal distribution $R(G')$ becomes:

$$\begin{aligned}&\mathrm{KL}(P(G'|G)||R(G'))\\ &= \sum_{(u,v)\in\mathcal{E}} \pi'_{uv}\log\frac{\pi'_{uv}}{r} + (1-\pi'_{uv})\log\frac{1-\pi'_{uv}}{1-r} + \text{Const.}\end{aligned} \tag{11}$$

By replacing $I(G,G')$ with the above tractable upper bound, we obtain the loss function, denoted by $\mathcal{L}_{\text{exp}}$, for training the explainer $\Psi(\cdot)$.

3.2. Derivation of Inner Optimization

For the inner optimization objective, we implement the first constraint by minimizing the distribution distance between $Q_{\boldsymbol{\phi}}(\tilde{G}|G')$ and $P_{\mathcal{G}}$. Under the Erdős–Rényi assumption, the distribution loss is equivalent to the cross-entropy loss between $\tilde{G}$ given G' and G over the full adjacency matrix (Chen et al., 2023b). Considering that G is usually sparse rather than a fully connected graph, in practice, we adopt a weighted version to emphasize more connected node pairs (Wang et al., 2016). Formally, we have the following distribution loss.

$$\mathcal{L}_{\text{dist}} = \frac{\beta}{|\mathcal{E}|}\sum_{(u,v)\in\mathcal{E}} \log(\tilde{p}_{uv}) + \frac{1}{|\bar{\mathcal{E}}|}\sum_{(u,v)\in\bar{\mathcal{E}}} \log(1-\tilde{p}_{uv}), \tag{12}$$

where $\bar{\mathcal{E}}$ is the set of node pairs that are unconnected in G, $\tilde{p}_{uv}$ is the probability of node pair (u,v) in $\tilde{G}$, and β is a hyper-parameter to for the trade-off between connected and unconnected node pairs.

The second constraint requires the mutual information of Y and $\tilde{G}$ is the same as that of Y and G'. Due to the OOD problem, it is non-trivial to directly compute $P(Y|G')$ or $H(Y|G')$. Instead, we implement this constraint with a novel graph generator that $\tilde{G}$ is obtained by combining G' and a non-explanatory subgraph (Zhang et al., 2023b). In practice, we implement the non-explanatory part by perturbing the residual subgraph $G - G'$. The intuition is that if an explanation comprises label information, it is unlikely to change the prediction by manipulating the remaining non-explanatory part, which is widely adopted in the literature (Fang et al., 2023a; Zheng et al., 2024).

4. The ProxyExplainer

Based on our novel GIB with proxy graphs, in this section, we introduce a straightforward yet theoretically robust instantiation, named ProxyExplainer. As shown in Figure 2, the architecture of our model comprises two key modules: the explainer and the proxy graph generator. The explainer takes the original graph G as its input and outputs a subgraph G' as an explanation, which is optimized through the outer objective in equation 9. The proxy graph generator

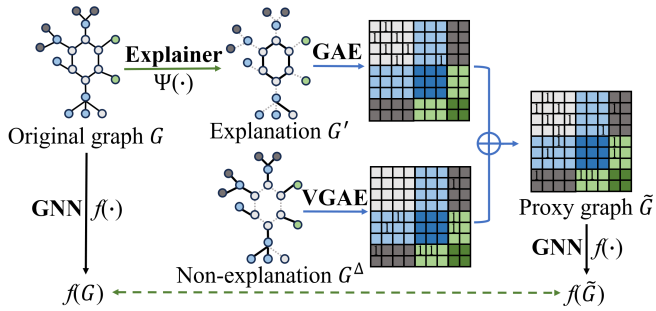


Figure 2. ProxyExplainer consists of two components: an Explainer and a Proxy Graph Generator. The Explainer takes a graph G as input and produces an explainable subgraph G', The proxy generator creates in-distribution proxy graphs that preserve the label information in G'. The proxy generator consists of a graph auto-encoder (GAE) and a variational graph auto-encoder (VGAE).

generates an in-distributed proxy graph, which is optimized with the inner objective.

4.1. The Explainer

For the sake of optimization, we follow the existing works to relax the element in $\boldsymbol{A}$ from binary values to continuous values in the range $[0, 1]$ (Luo et al., 2020; 2024). We adopt a generative explainer due to its effectiveness and efficiency (Chen et al., 2023a). Specifically, we decompose the to-be-explained model $f(\cdot)$ into two functions that $f_{\text{enc}}(\cdot)$ learns node representations and $f_{\text{cls}}(\cdot)$ predicts graph labels based on node embeddings. Formally, we have $\boldsymbol{Z} = f_{\text{enc}}(\boldsymbol{X}, \boldsymbol{A})$ and $\hat{Y} = f_{\text{cls}}(\boldsymbol{Z})$, where $\boldsymbol{Z}$ is the node representation matrix and $\hat{Y}$ is prediction. Routinely, $f_{\text{cls}}(\cdot)$ consists of a pooling layer followed by a classification layer and $f_{\text{enc}}(\cdot)$ consists of the other layers. Following the independence assumption in Section 3.1, we approximate Bernoulli distributions with binary concrete distributions (Maddison et al., 2017; Jang et al., 2017). Specifically, the probability of sampling an edge (u, v) is computed by an MLP parameterized by ψ, denoted by $g_\psi(\cdot)$. Formally, we have

$$\begin{aligned} &\omega_{uv} = g_\psi([\boldsymbol{z}_u; \boldsymbol{z}_v]), \\ &\epsilon \sim \text{Uniform}(0, 1), \\ &A'_{uv} = \sigma((\log \epsilon - \log(1-\epsilon) + \omega_{uv})/\tau), \end{aligned} \tag{13}$$

where $[\boldsymbol{z}_u; \boldsymbol{z}_v]$ is the concatenation of node representations $\boldsymbol{z}_u$ and $\boldsymbol{z}_v$, ϵ is a independent variable, $\sigma(\cdot)$ is the Sigmoid function, and τ is a temperature hyper-parameter for approximation. According to (Luo et al., 2020), the parameter π'_{uv} of Bernoulli distribution in equation 11 can be obtained by $\pi'_{uv} = \frac{\exp(\omega_{uv})}{1+\exp(\omega_{uv})}$.

4.2. The Proxy Graph Generator

As shown in our analysis in Section 3.2, we demonstrate the synthesis of a proxy graph through the amalgamation of the explanation subgraph G' and the perturbation of its non-explanatory subgraph, represented as $G^\Delta = (\mathcal{V}, \mathcal{E}^\Delta, \boldsymbol{X})$. We define the edge set of $G^\Delta, \mathcal{E}^\Delta$, as the differential set $\mathcal{E} - \mathcal{E}'$. Correspondingly, the adjacency matrix $\boldsymbol{A}^\Delta$ is derived through $\boldsymbol{A} - \boldsymbol{A}'$. Building upon this foundation, and as illustrated in Figure 2, our proposed framework introduces a dual-structured mechanism comprising two distinct graph auto-encoders(GAE) (Bank et al., 2023). To be more specific, we first include an encoder to learn a latent matrix $\boldsymbol{Z}'$ according to $\boldsymbol{A}'$ and $\boldsymbol{X}'$, and a decoder network recovers $\boldsymbol{A}'$ based on $\boldsymbol{Z}'$. Formally, we have:

$$\boldsymbol{Z}' = \text{ENC}_1(\boldsymbol{A}', \boldsymbol{X}), \quad \tilde{\boldsymbol{A}}' = \text{DEC}(\boldsymbol{Z}'), \tag{14}$$

where ENC_1 and DEC are the encoder network and decoder network. Our framework is flexible to the choices of these two networks. $\tilde{\boldsymbol{A}}' \in \mathbb{R}^{n \times n}$ is the reconstructed adjacency matrix.

To introduce perturbations into the non-explanatory segment, our approach employs a Variational Graph Auto-Encoder (VGAE), a generative model adept at creating varied yet structurally coherent graph data. This capability is pivotal in generating nuanced variations of the non-explanatory subgraph. The VGAE operates by first encoding the non-explanatory subgraph G^Δ into a latent probabilistic space, characterized by Gaussian distributions. This process is articulated as:

$$\boldsymbol{\mu}^\Delta = \text{ENC}_1(\boldsymbol{A}^\Delta, \boldsymbol{X}), \quad \boldsymbol{\sigma}^\Delta = \text{ENC}_2(\boldsymbol{A}^\Delta, \boldsymbol{X}), \tag{15}$$

where ENC_1 and ENC_2 are encoder networks that learn the mean $\boldsymbol{\mu}^\Delta$ and variance $\boldsymbol{\sigma}^\Delta$ of the Gaussian distributions, respectively. Following this, the latent representations $\boldsymbol{Z}^\Delta$ are sampled from these distributions, ensuring that each generated instance is a unique variation of the original. The decoder network then reconstructs the perturbed non-explanatory subgraph from these sampled latent representations:

$$\boldsymbol{Z}^\Delta \sim \mathcal{N}(\boldsymbol{\mu}^\Delta, \text{diag}(\boldsymbol{\sigma}^\Delta)^2), \quad \tilde{\boldsymbol{A}}^\Delta = \text{DEC}(\boldsymbol{Z}^\Delta), \tag{16}$$

where $\tilde{\boldsymbol{A}}^\Delta \in \mathbb{R}^{n \times n}$ represents the adjacency matrix of the perturbed non-explanatory subgraph. This novel use of VGAE facilitates the generation of diverse yet representative perturbations, crucial for enhancing the interpretability of explainers in our proxy graph framework. The adjacency matrix of a proxy graph is then obtained by

$$\tilde{\boldsymbol{A}} = \tilde{\boldsymbol{A}}' + \tilde{\boldsymbol{A}}^\Delta. \tag{17}$$

Loss function. To train the proxy graph generator, we introduce a standard Gaussian distribution as the prior for

the latent space representations in the VGAE, specifically $\boldsymbol{Z} \sim \mathcal{N}(0, \mathcal{I})$, where $\mathcal{I}$ represents the identity matrix. Then, the loss function is as follows.

$$\mathcal{L}_{\text{proxy}} = \mathcal{L}_{\text{dist}} + \lambda \mathcal{L}_{\text{KL}}, \tag{18}$$

where $\mathcal{L}_{\text{dist}}$ is equivalent to cross-entropy between $\tilde{G}$ and G (Chen et al., 2023b). $\mathcal{L}_{\text{KL}}$ represents the KL divergence between the distribution of the latent representations $\boldsymbol{Z}^{\Delta}$ and the assumed Gaussian prior. This term is crucial for regulating the variational aspect of the VGAE, ensuring that the generated perturbations are meaningful and controlled. λ is a hyper-parameter.

Alternate Training. To train the explainer and the proxy graph generator networks, we follow existing works (Zheng et al., 2024) to use an alternate training schedule that trains the proxy graph generator network M times and then trains the explainer network once. M is a hyper-parameter determined by grid search. The detailed algorithm description of our model is shown in Appendix B.

5. Related Work

GNN Explanation. The goal of explainability in GNNs is to ensure transparency in graph-based tasks. Recent works have been directed towards elucidating the rationale behind GNN predictions. These explanation methods can be broadly classified into two categories: instance-level and model-level approaches (Yuan et al., 2022). In this study, we focus on instance-level explanations, which aim to clarify the specific reasoning behind individual predictions made by GNNs. These methods are critical for understanding the decision-making process on a case-by-case basis to enhance the explainability of GNNs. For example, GNNExplainer (Ying et al., 2019) excludes certain edges and node features to observe the changes in prediction. However, its single-instance focus limits its applicability to provide a global understanding of the to-be-explained model (Chen et al., 2023a). PGExplainer (Luo et al., 2020; 2024) introduces a parametric neural network to learn edge weights. Thus, once training is complete, it can explain new graphs without retraining. ReFine (Wang et al., 2021) integrates a pre-training phase that focuses on class comparisons and is fine-tuned to refine context-specific explanations. GStarX (Zhang et al., 2022) assigns an importance score to each node by caculating the Hamiache and Navarro values of the structure to obtain explanatory subgraphs. GFlowExplainer (Li et al., 2023) uses a generator to construct a TD-like flow matching condition to learn a policy for generating explanations by adding nodes sequentially.

Distribution Shifting in Explanations. The distribution shifting problem in post-hoc explanations has been increasingly recognized in explainable AI fields (Chang et al., 2019; Qiu et al., 2022). For example, FIDO (Chang et al., 2019) works on enhancing image classifier explanations, focusing on relevant contextual details that agree with the training data's distribution. A recent study tackles the distribution shifting problem in image explanations by introducing a module that assesses the similarity between altered data and the original dataset distribution (Qiu et al., 2022). In the graph domain, an ad-hoc strategy to mitigate distribution shifting is to initially reduce the size constraint coefficient during the explanation process (Fang et al., 2023b). MixupExplainer (Zhang et al., 2023b) and RegExplainer (Zhang et al., 2023a) propose non-parametric solutions by mixing up the explanation subgraph with a non-explainable part from another graph. However, these methods operates under the assumption that the explanatory and non-explanatory subgraphs in mixed graphs are independent, which may not hold in many real-life graphs.

6. Experiments

We present empirical results that illustrate the effectiveness of our proposed method. These experiments are mainly designed to explore the following research questions:

- **RQ1:** Can the proposed framework outperform other baselines in identifying explanations for GNNs?
- **RQ2:** Is the distribution shifting severe in explanation subgraphs? Can the proposed approach alleviate that?
- **RQ3:** How does each component of ProxyExplainer impact the overall performance in generating explanations?

6.1. Experimental Settings

To evaluate the performance of ProxyExplainer, we use six benchmark datasets with ground-truth explanations. These include four real-world datasets: MUTAG (Kazius et al., 2005), Benzene (Sanchez-Lengeling et al., 2020),Alkane-Carbonyl (Sanchez-Lengeling et al., 2020), and Fluoride-Carbonyl (Sanchez-Lengeling et al., 2020), along with two synthetic datasets: BA-2motifs (Luo et al., 2020) and BA-3motifs (Chen et al., 2023b). We take GradCAM (Pope et al., 2019), GNNExplainer(Ying et al., 2019), PGExplainer(Luo et al., 2020), ReFine (Wang et al., 2021), and MixupExplainer (Zhang et al., 2023b) for comparison. We follow the experimental setting in previous works (Ying et al., 2019; Luo et al., 2020; Sanchez-Lengeling et al., 2020) to train a Graph Convolutional Network (GCN) model (Kipf & Welling, 2017) with three layers. Experiments on another representative GNN, Graph Isomorphism Network (GIN) (Xu et al., 2019), can be found in Appendix D.3. We use the Adam optimizer (Kingma & Ba, 2014) with the inclusion of a weight decay $5e-4$. Detailed information

Table 1. Explanation accuracy in terms of AUC-ROC on edges.

	MUTAG	Benzene	Alkane-Carbonyl	Fluoride-Carbonyl	BA-2motifs	BA-3motifs
GradCAM	$0.727_{\pm 0.000}$	$0.740_{\pm 0.000}$	$0.448_{\pm 0.000}$	$0.694_{\pm 0.000}$	$0.714_{\pm 0.000}$	$0.709_{\pm 0.000}$
GNNExplainer	$0.682_{\pm 0.009}$	$0.485_{\pm 0.001}$	$0.551_{\pm 0.003}$	$0.574_{\pm 0.002}$	$0.644_{\pm 0.007}$	$0.511_{\pm 0.002}$
PGExplainer	$0.832_{\pm 0.032}$	$0.793_{\pm 0.054}$	$0.660_{\pm 0.036}$	$0.702_{\pm 0.018}$	$0.734_{\pm 0.117}$	$0.796_{\pm 0.010}$
ReFine	$0.612_{\pm 0.004}$	$0.606_{\pm 0.002}$	$0.768_{\pm 0.001}$	$0.571_{\pm 0.000}$	$0.698_{\pm 0.001}$	$0.629_{\pm 0.005}$
MixupExplainer	$0.863_{\pm 0.103}$	$0.611_{\pm 0.032}$	$0.811_{\pm 0.006}$	$0.706_{\pm 0.013}$	$0.906_{\pm 0.059}$	$0.859_{\pm 0.019}$
ProxyExplainer	$\mathbf{0.977}_{\pm 0.009}$	$\mathbf{0.845}_{\pm 0.036}$	$\mathbf{0.934}_{\pm 0.005}$	$\mathbf{0.758}_{\pm 0.068}$	$\mathbf{0.935}_{\pm 0.008}$	$\mathbf{0.960}_{\pm 0.008}$

Table 2. Explanation accuracy in terms of AP on MUTAG and BA-2motifs.

	MUTAG	BA-2motifs
GradCAM	$0.247_{\pm 0.000}$	$0.664_{\pm 0.000}$
GNNExplainer	$0.232_{\pm 0.001}$	$0.608_{\pm 0.004}$
PGExplainer	$0.611_{\pm 0.024}$	$0.682_{\pm 0.117}$
ReFine	$0.227_{\pm 0.001}$	$0.619_{\pm 0.002}$
MixupExplainer	$0.647_{\pm 0.083}$	$0.787_{\pm 0.073}$
ProxyExplainer	$\mathbf{0.756}_{\pm 0.107}$	$\mathbf{0.839}_{\pm 0.036}$

Table 3. Fidelity evaluation on MUTAG and BA-2motifs.

	MUTAG		BA-2motifs	
	$Fid_{\alpha_1,+} \uparrow$	$Fid_{\alpha_2,-} \downarrow$	$Fid_{\alpha_1,+} \uparrow$	$Fid_{\alpha_2,-} \downarrow$
GradCAM	$0.004_{\pm 0.000}$	$0.162_{\pm 0.000}$	$0.072_{\pm 0.000}$	$0.107_{\pm 0.000}$
GNNExp.	$0.031_{\pm 0.001}$	$0.148_{\pm 0.001}$	$0.057_{\pm 0.002}$	$0.132_{\pm 0.001}$
PGExp.	$0.034_{\pm 0.011}$	$0.148_{\pm 0.005}$	$0.065_{\pm 0.017}$	$0.126_{\pm 0.009}$
ReFine	$0.003_{\pm 0.000}$	$0.160_{\pm 0.001}$	$0.060_{\pm 0.005}$	$0.125_{\pm 0.001}$
MixupExp.	$0.037_{\pm 0.006}$	$0.146_{\pm 0.003}$	$0.074_{\pm 0.005}$	$0.112_{\pm 0.003}$
ProxyExp.	$\mathbf{0.040}_{\pm 0.002}$	$\mathbf{0.145}_{\pm 0.001}$	$\mathbf{0.086}_{\pm 0.003}$	$\mathbf{0.106}_{\pm 0.002}$

regarding datasets and baselines is delineated in Appendix C.

To evaluate the quality of explanations, we approach the explanation task as a binary classification of edges. Edges that are part of ground truth subgraphs are labeled as positive, while all others are deemed negative. The importance weights given by the explanation methods are interpreted as prediction scores. An effective explanation technique is one that assigns higher weights to edges within the ground truth subgraphs compared to those outside of them. We utilize the AUC-ROC for quantitative assessment (Ying et al., 2019; Luo et al., 2020).

6.2. Quantitative Evaluation (RQ1)

To answer RQ1, we compare the proposed method, ProxyExplainer, to other baselines. Each experiment was conducted 10 times using random seeds, and the average AUC scores as well as standard deviations are presented in Table 1.

The results demonstrate that ProxyExplainer provides the most accurate explanations across all datasets. Specifically, it improves the AUC scores by an average of 10.6% on real-world datasets and 7.5% on synthetic datasets over the leading baselines. Comparisons with baseline methods highlight the advantages of our proposed explanation framework. Besides, ProxyExplainer captures underlying explanatory factors consistently across diverse datasets. For instance, MixupExplainer exhibits proficiency on the synthetic BA-2motifs dataset but performs poorly on the real-world Benzene dataset. The reason is that MixupExplainer relies on the independence assumption of explanation and non-explanation subgraphs, which may not hold in real-world datasets. In contrast, ProxyExplainer consistently demonstrates high performance across different datasets, showcasing its robustness and adaptability.

Considering the importance of precision for the positive class in our context, we further adopt AP to evaluate the performance. As shown in Table 2, with AP scores, ProxyExplainer consistently outperforms the other baselines in two benchmark datasets MUTAG and BA-2motifs.

Additionally, some existing works, such as SubgraphX (Yuan et al., 2021), adopt faithfulness-based metrics for evaluation. However, these metrics are problematic due to the OOD problem (Zheng et al., 2024; Amara et al., 2023). So we use the robust fidelity metrics ($Fid_{\alpha_1,+}$, $Fid_{\alpha_2,-}$) as described in (Zheng et al., 2024) with default settings ($\alpha_1 = 0.1, \alpha_2 = 0.9$) to evaluate model faithfulness. Table 3 demonstrates that ProxyExplainer is consistently outperforms all baselines in both $Fid_{\alpha_1,+}$ and $Fid_{\alpha_2,-}$.

6.3. Alleviating Distribution Shifts (RQ2)

In this section, we assess ProxyExplainer's ability to generate in-distribution proxy graphs. Due to the intractable of direct computation, we follow the previous work (Chen et al., 2023b) to utilize Maximum Mean Discrepancy (MMD) between distributions of multiple graph statistics, including degree distributions, clustering coefficients, and spectrum distributions, between the generated proxy graphs and original graphs. Specifically, we utilize Gaussian Earth Mover's Distance kernel when computing MMDs. Smaller MMD

Table 4. MMD results between the ground truth explanations and original graphs (GT); PGExplainer explanations and original graphs (PGE); proxy graphs in our methods and original graphs (Proxy).

	MUTAG			Benzene			Alkane-Carbonyl		
Metric	GT	PGE	Proxy	GT	PGE	Proxy	GT	PGE	Proxy
Deg.	0.614	0.468	**0.123**	0.843	0.393	**0.236**	0.872	0.665	**0.177**
Clus.	**0.003**	**0.003**	0.009	0.009	**0.002**	0.004	**0.011**	**0.011**	**0.011**
Spec.	0.414	0.341	**0.186**	0.295	0.163	**0.101**	0.596	0.447	**0.049**
Sum.	1.032	0.813	**0.317**	1.147	0.558	**0.341**	1.479	1.123	**0.237**
	Fluoride-Carbonyl			BA-2motifs			BA-3motifs		
Metric	GT	PGE	Proxy	GT	PGE	Proxy	GT	PGE	Proxy
Deg.	0.638	0.488	**0.196**	0.759	0.496	**0.060**	0.541	0.149	**0.092**
Clus.	**0.012**	**0.012**	**0.012**	**0.447**	0.463	0.584	0.262	0.382	**0.245**
Spec.	0.351	0.315	**0.100**	0.245	0.256	**0.091**	0.217	0.063	**0.062**
Sum.	1.000	0.815	**0.308**	1.451	1.215	**0.735**	1.020	0.594	**0.399**

values indicate similar graph distributions. For comparison, we also include the Ground truth explanations and the ones generated by PGExplainer.

The results are shown in Table 4. "GT" denotes the MMDs between the ground truth explanations and original graphs. "PGE" represents the MMDs between explanations generated by PGExplainer and original graphs. "Proxy" denotes the MMDs between proxy graphs in our method and original graphs. We have the following observations. First, the MMDs between ground truth and original graphs are usually large, verifying our motivation that a model trained on original graphs may not have correct predictions on the OOD explanation subgraphs. Second, the explanations generated by a representative work, PGExplainer, are often OOD from original graphs, indicating that the original GIB-based objective function may be sub-optimal. Third, in most cases, proxy graphs generated by our method are with smaller MMDs, demonstrating their in-distribution property.

6.4. Ablation Studies (RQ3)

In this section, we conduct ablation studies to investigate the roles of different components. Specifically, we consider the following variants of ProxyExplainer: (1) w/o G^{Δ}: in this variant, we remove the non-explanatory subgraph generator (VGAE), which is the bottom half as shown in Figure 2; (2) w/o $\mathcal{L}_{\text{KL}}$: in this variant, we remove the KL divergence from the training loss in ProxyExplainer; (3) w/o $\mathcal{L}_{\text{dist}}$: in this variant, we remove the distribution loss from ProxyExplainer. The results of the ablation study on MUTAG and BA-2motifs are reported in Figure 3.

Figure 3 illustrates a notable performance drop for all variants, indicating that each component contributes positively to the effectiveness of ProxyExplainer. Especially, in the real-life dataset MUTAG, without the in-distribution constraint, w/o $\mathcal{L}_{\text{dist}}$ is much worse than ProxyExplainer, indicating the vital role of in-distributed proxy graphs in our framework. Extensive ablation studies on other datasets can be found in the Appendix D.2.

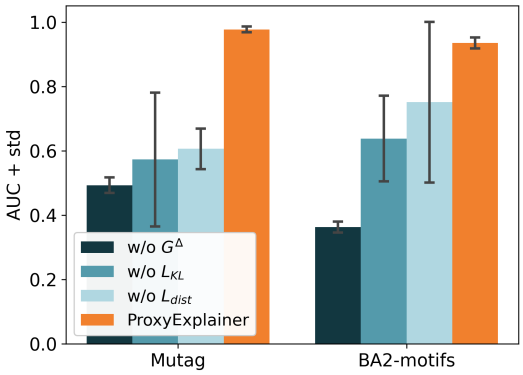


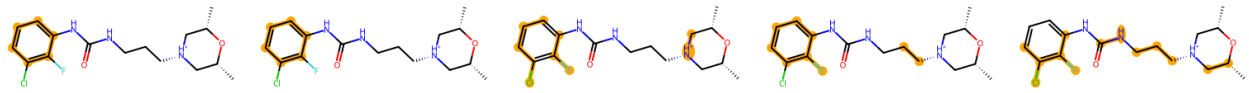
Figure 3. Ablation Studies on MUTAG and BA-2motifs.

6.5. Case Studies

In this part, we conduct case studies to qualitatively compare the performances of ProxyExplainer against others. We adopt examples from Benzene and BA-2motifs in this part. We show visualization results in Figure 4 and Figure 5. Explanations are highlighted with bold black and bold orange edges, respectively. From the results, our ProxyExplainer stands out by generating more compelling explanations compared to baselines. Specifically, ProxyExplainer maintains clarity without introducing irrelevant edges and exhibits more concise results compared to alternative methods. The visualized performance underscores ProxyExplainer's ability to provide meaningful and focused subgraph explanations. More visualizations on these two datasets can be found in the Appendix D.5.

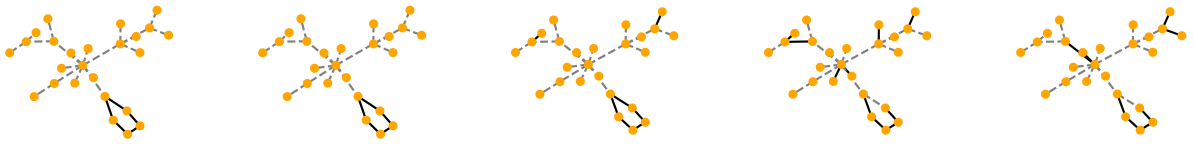
7. Conclusion

In this paper, we systematically investigate the OOD problem in the de facto framework, GIB, for learning explanations in GNNs, which is highly overlooked in the literature. To address this issue, we extend the GIB by innovatively including in-distributed proxy graphs. On top of that, we derive a tractable objective function for practical implementations. We further present a new explanation framework that utilizes two graph auto-encoders to generate proxy graphs. We conduct comprehensive empirical studies on both benchmark synthetic datasets and real-life datasets to demonstrate the effectiveness of our method in alleviating the OOD problem and achieving high-quality explanations. There are several topics we will investigate in the future. First, the OOD problem also exists in obtaining model-level explanations and counterfactual explanations. We will extend our



(a) Ground Truth (b) ProxyExplainer (c) MixupExplainer (d) PGExplainer (e) GNNExplainer

Figure 4. Visualization of explanation results from different explanation models on Benzene. The generated explanations are highlighted with bold orange edges.



(a) Ground Truth (b) ProxyExplainer (c) MixupExplainer (d) PGExplainer (e) GNNExplainer

Figure 5. Visualization of explanation results from different explanation models on BA-2motifs. The generated explanations are highlighted with bold black edges.

method to these research problems. Second, we will also analyze explainable learning methods with proxies in other data structures, such as image, language, and time series.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgments

This project was partially supported by NSF grant IIS-2331908. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

References

Amara, K., El-Assady, M., and Ying, R. Ginx-eval: Towards in-distribution evaluation of graph neural network explanations. *arXiv preprint arXiv:2309.16223*, 2023.

Bank, D., Koenigstein, N., and Giryes, R. Autoencoders. *Machine learning for data science handbook: data mining and knowledge discovery handbook*, pp. 353–374, 2023.

Chang, C.-H., Creager, E., Goldenberg, A., and Duvenaud, D. Explaining image classifiers by counterfactual generation. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1MXz20cYQ.

Chen, J., Amara, K., Yu, J., and Ying, R. Generative explanations for graph neural network: Methods and evaluations. *arXiv preprint arXiv:2311.05764*, 2023a.

Chen, J., Wu, S., Gupta, A., and Ying, Z. D4explainer: In-distribution explanations of graph neural network via discrete denoising diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=GJtP1ZEzua.

Debnath, A. K., Lopez de Compadre, R. L., Debnath, G., Shusterman, A. J., and Hansch, C. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34 (2):786–797, 1991.

Erdős, P., Rényi, A., et al. On the evolution of random graphs. *Publ. math. inst. hung. acad. sci*, 5(1):17–60, 1960.

Fang, J., Liu, W., Gao, Y., Liu, Z., Zhang, A., Wang, X., and He, X. Evaluating post-hoc explanations for graph neural networks via robustness analysis. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a.

Fang, J., Liu, W., Zhang, A., Wang, X., He, X., Wang, K., and Chua, T.-S. On regularization for explaining graph neural networks: An information theory perspective, 2023b. URL https://openreview.net/forum?id=5rX7M4wa2R_.

Fang, J., Wang, X., Zhang, A., Liu, Z., He, X., and Chua, T.-S. Cooperative explanations of graph neural networks. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pp. 616–624, 2023c.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Huang, R., Shirani, F., and Luo, D. Factorized explainer for graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, 2024.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Kazius, J., McGuire, R., and Bursi, R. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of medicinal chemistry*, 48(1):312–320, 2005.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Li, W., Li, Y., Li, Z., HAO, J., and Pang, Y. DAG matters! GFlownets enhanced explainer for graph neural networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=jgmuRzM-sb6.

Li, Y., Zhou, J., Verma, S., and Chen, F. A survey of explainable graph neural networks: Taxonomy and evaluation metrics. *arXiv preprint arXiv:2207.12599*, 2022.

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

Luo, D., Zhao, T., Cheng, W., Xu, D., Han, F., Yu, W., Liu, X., Chen, H., and Zhang, X. Towards inductive and efficient explanations for graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=S1jE5L5gl.

Miao, S., Liu, M., and Li, P. Interpretable and generalizable graph learning via stochastic attention mechanism. In *International Conference on Machine Learning*, pp. 15524–15543. PMLR, 2022.

Pope, P. E., Kolouri, S., Rostami, M., Martin, C. E., and 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.

Qiu, L., Yang, Y., Cao, C. C., Zheng, Y., Ngai, H., Hsiao, J., and Chen, L. Generating perturbation-based explanations with robustness to out-of-distribution data. In *Proceedings of the ACM Web Conference 2022*, pp. 3594–3605, 2022.

Sanchez-Lengeling, B., Wei, J., Lee, B., Reif, E., Wang, P., Qian, W., McCloskey, K., Colwell, L., and Wiltschko, A. Evaluating attribution for graph neural networks. *Advances in neural information processing systems*, 33: 5898–5910, 2020.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

Sterling, T. and Irwin, J. J. ZINC 15 - ligand discovery for everyone. *J. Chem. Inf. Model.*, 55(11):2324–2337, 2015.

Tishby, N., Pereira, F. C., and Bialek, W. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Wang, D., Cui, P., and Zhu, W. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1225–1234, 2016.

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

Wu, B., Li, J., Yu, J., Bian, Y., Zhang, H., Chen, C., Hou, C., Fu, G., Chen, L., Xu, T., et al. A survey of trustworthy

graph learning: Reliability, explainability, and privacy protection. *arXiv preprint arXiv:2205.10014*, 2022.

Wu, T., Ren, H., Li, P., and Leskovec, J. Graph information bottleneck. *Advances in Neural Information Processing Systems*, 33:20437–20448, 2020.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

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

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

Yuan, H., Yu, H., Gui, S., and Ji, S. Explainability in graph neural networks: A taxonomic survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Zhang, H., Wu, B., Yuan, X., Pan, S., Tong, H., and Pei, J. Trustworthy graph neural networks: Aspects, methods, and trends. *Proceedings of the IEEE*, 112(2):97–139, 2024. doi: 10.1109/JPROC.2024.3369017.

Zhang, J., Chen, Z., Luo, D., Wei, H., et al. Regexplainer: Generating explanations for graph neural networks in regression tasks. In *The Second Learning on Graphs Conference*, 2023a.

Zhang, J., Luo, D., and Wei, H. Mixupexplainer: Generalizing explanations for graph neural networks with data augmentation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3286–3296, 2023b.

Zhang, S., Liu, Y., Shah, N., and Sun, Y. Gstarx: Explaining graph neural networks with structure-aware cooperative games. In *Advances in Neural Information Processing Systems*, volume 35, pp. 19810–19823, 2022.

Zheng, X., Shirani, F., Wang, T., Cheng, W., Chen, Z., Chen, H., Wei, H., and Luo, D. Towards robust fidelity for evaluating explainability of graph neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=up6hr4hIQH.

A. Notations

In Table 5, we summarized the main notations we used and their descriptions in this paper.

Table 5. Symbols and their descriptions.

Symbols	Descriptions
$\mathcal{G}$	A set of graphs
$G, \mathcal{V}, \mathcal{E}$	Graph instance, node set, edge set
v_i	The i-th node
$\boldsymbol{X}$	Node feature matrix
$\boldsymbol{A}$	Adjacency matrix
$\boldsymbol{Z}$	Node representation matrix
Y	Label of graph G
$\mathcal{Y}$	A set of labels
G^*	Optimal explanatory subgraph
$\mathcal{G}^*$	A set of G^*
G'	Candidate explanatory subgraph
G^Δ	Non-explanatory graph
$\tilde{G}$	Proxy graph of G' with a fixed distribution
$\boldsymbol{h}, \boldsymbol{h}', \tilde{\boldsymbol{h}}$	Graph embeddings
d	Dimension of node feature
$f(\cdot)$	To-be-explained GNN model
$\Psi_\psi(\cdot)$	Explanation function
ψ	Parameter of the explanation function
$P_{\mathcal{G}}$	Distribution of original training graphs
$P_{\mathcal{G}'}$	Distribution of explanation subgraphs
$Q_{\boldsymbol{\phi}}$	Parameterized function of $P(\tilde{G}\|G')$
$\boldsymbol{\phi}$	Model parameters of $Q_{\boldsymbol{\phi}}$
$\boldsymbol{\phi}^*$	Optimal $\boldsymbol{\phi}$
α	Balance parameter between $I(G, G')$ and $I(Y, G')$
$\bar{\mathcal{E}}$	The set of node pairs that are unconnected in G
$\tilde{p}_{uv}$	Probability of node pair (u, v) in $\tilde{G}$
β	A hyper-parameter to get a trade-off between connected and unconnected node pairs
$f_{\text{enc}}(\cdot)$	The front part of GNNs that learns node representations
$f_{\text{cls}}(\cdot)$	The back part of GNNs that predicts graph labels based on node embeddings
$\sigma(\cdot)$	Sigmoid function
τ	Temperature hyper-parameter for approximation
λ	A hyper-parameter in Proxy loss function
$\mathcal{L}_{\text{dist}}$	Distribution loss between $\tilde{G}$ and G
$\mathcal{L}_{\text{KL}}$	KL divergence between distribution of $\boldsymbol{Z}^\Delta$ and its prior
$\mathcal{L}_{\text{proxy}}$	Proxy loss
$\mathcal{L}_{\text{exp}}$	Explainer loss

B. Algorithm

We take as input a set of graphs $\mathcal{G} = \{G_i\}_{i=0}^N$. For each graph G_i, we use an explainer model to identify a subgraph G_i' as the explanation and then compute the non-explanatory graph G_i^Δ, which is obtained by removing edges in G_i that exist in G_i'. We use GAE to reconstruct a subgraph, denoted by $\tilde{G}_i'$ from the subgraph G_i' and another VGAE to generate a new subgraph, denoted by $\tilde{G}_i^\Delta$, from the non-explainable subgraph G_i^Δ. The proxy graph, $\tilde{G}_i$, is obtained by combining them, whose adjacency matrix can be denoted by $\tilde{\boldsymbol{A}}_i = \tilde{\boldsymbol{A}}_i' + \tilde{\boldsymbol{A}}_i^\Delta$. Here $\tilde{\boldsymbol{A}}_i'$ and $\tilde{\boldsymbol{A}}_i^\Delta$ are adjacency matrices of $\tilde{G}_i'$ and $\tilde{G}_i^\Delta$, respectively. We alternatively train the explainer model and proxy graph generator as shown in Algorithm 1.

Algorithm 1 Algorithm of ProxyExplainer

Input: A set of graphs $\mathcal{G} = \{G_i\}_{i=0}^{N}$, with each $G_i = (\mathcal{V}_i, \mathcal{E}_i, \boldsymbol{X}_i)$,a pretrain to-be-explained model $f(\cdot)$, hyper parameters α, λ, M, epochs E.
Initialize an explainer function $\Psi_\psi(\cdot)$
epoch $\leftarrow 0$
while epoch $< E$ **do**
  for $G_i \in \mathcal{G}$ **do**
    $G_i' \leftarrow \Psi_\psi(G_i)$
    $G_i^\Delta \leftarrow G_i - G_i'$
    Compute $\tilde{\boldsymbol{A}}_i'$ with equation 14
    Compute $\boldsymbol{Z}^\Delta$ and $\tilde{\boldsymbol{A}}_i^\Delta$ with equation 16
    Compute proxy loss $\mathcal{L}_{\text{proxy}}$ with equation 18
    Update parameters in proxy graph generator with backpropagation
    if epoch$\%M == 0$ **then**
      Compute explainer loss $\mathcal{L}_{\text{exp}}$
      Update parameters in the explainer with backpropagation
    end if
  end for
  epoch $\leftarrow$ epoch $+ 1$
end while

C. Full Experimental Setups

We conduct all experiments on a Linux machine with 8 Nvidia A100-PCIE GPUs, each with 40GB of memory. The CUDA version is 12.4 and the driver version is 550.54.15. We use Python 3.9 and Torch 2.0.1 in our project. The code is available at https://github.com/realMoana/ProxyExplainer.

C.1. Datasets

MUTAG (Kazius et al., 2005). MUTAG dataset includes 4,337 molecular graphs, each of them is classified into two groups based on its mutagenic effect on the Gram-negative bacterium S. Typhimurium. This classification comes from several specific virulence gland groups with mutagenicity in molecular mapping by Kazius et al.(Kazius et al., 2005).
Benzene (Sanchez-Lengeling et al., 2020). Benzene consists 12,000 molecular graphs from the ZINC15(Sterling & Irwin, 2015) database, which can be classified into two classes. The main goal is to determine if a Benzene ring is exited in each molecule. In cases where multiple Benzene rings are present, each Benzene ring is treated as a distinct explanation.
Alkane-Carbonyl (Sanchez-Lengeling et al., 2020). The Alkane-Carbonyl dataset consists of a total of 4,326 molecule graphs that are categorized into two different classes. Positive samples refer to molecules that have both alkane and carbonyl functional groups. The ground-truth explanation includes alkane and carbonyl functional groups in a given molecule.
Fluoride-Carbonyl (Sanchez-Lengeling et al., 2020). The Fluoride-Carbonyl dataset has 8,671 molecular graphs. The ground-truth explanation is based on the particular combination of fluoride atoms and carbonyl functional groups present in each molecule.
BA-2motifs (Luo et al., 2020). The BA-2motifs dataset consists 1,000 synthetic graphs, each derived from a basic Barabasi-Albert (BA) model. The dataset is divided into two categories: one part of the graphs add patterns that mimic the structure of a house, and the remaining integrate five-node cyclic patterns. The classification of these graphs depends on the specific patterns.
BA-3motifs (Chen et al., 2023b). BA-3motifs is an extended dataset inspired by the BA-2motifs and contains 3,000 synthetic graphs. Each base graph is accompanied by one of three different patterns: house, cycle, or grid.

C.2. Baselines

To evaluate our model, we well-train the GCN model in to ensure that it takes good performance in graph classification tasks. The results are displayed in Table 7. For a comprehensive comparison, we incorporate various post-hoc explanation methods, including GradCAM, GNNExplainer, PGExplainer, ReFine, and MixupExplainer.

Table 6. Statistics of molecule datasets for graph classification with ground-truth explanations.

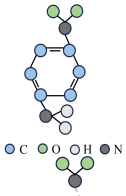
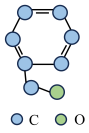
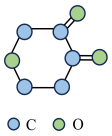
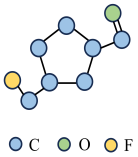
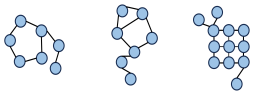
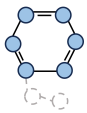
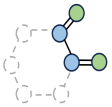
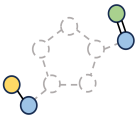
	MUTAG	Benzene	Alkane-Carbonyl	Fluoride-Carbonyl	BA-2motifs	BA-3motifs
Graphs	4,337	12,000	4,326	8,671	1,000	3,000
Average nodes	29.15	20.58	21.13	21.36	25.00	21.92
Average edges	60.83	43.65	44.95	45.37	25.48	29.51
Node features	14	14	14	14	10	4
Original graph						
Ground truth explanation						
	NH_2, NO_2	Benzene Ring	Alkane,C=O	F^-,C=O	House, cycle	House,cycle, grid

Table 7. The graph-classification task performances of the GCN model

	MUTAG	Benzene	Alkane-Carbonyl	Fluoride-Carbonyl	BA-2motifs	BA-3motifs
Train Acc	0.850	0.930	0.979	0.951	0.999	0.997
Val Acc	0.834	0.927	0.986	0.956	1.0	0.997
Test Acc	0.804	0.915	0.975	0.951	1.0	0.977

- **GradCAM (Pope et al., 2019).** This method utilizes gradients as a weighting mechanism to merge various feature maps. It operates on heuristic assumptions and cannot elucidate node classification models.

- **GNNExplainer (Ying et al., 2019).** It learns soft masks for edges and node features, and aims to elucidate predictions through mask optimization. GNNExplainer integrates these masks with the original graph through element-wise multiplications. The masks are optimized by enhancing the mutual information between the original graph and the modified graph prediction results.

- **PGExplainer (Luo et al., 2020).** This method extends the idea of GNNExplainer by assuming that the graph is a random Gilbert graph. PGExplainer generates each edge embedding by combining the embeddings of its constituent nodes, then uses these edge embeddings to determine a Bernoulli distribution to indicate whether to mask an edge or not, and utilizes a Gumbel-Softmax approach to model the Bernoulli distribution for end-to-end training.

- **ReFine (Wang et al., 2021).** ReFine identifies the edge probabilities for the entire category by maximizing the mutual information and contrastive loss between categories. In fine-tuning, it uses the edge probabilities from the previous stage to sample edges, and find explanations that maximize mutual information for specific instances.

- **MixupExplainer (Zhang et al., 2023b).** This method combines original explanatory subgraphs with randomly sampled, label-independent base graphs in a non-parametric way to mitigate the common OOD issue which found in previous methods.

D. Extensive Experiments

D.1. Extensive Distribution Analysis

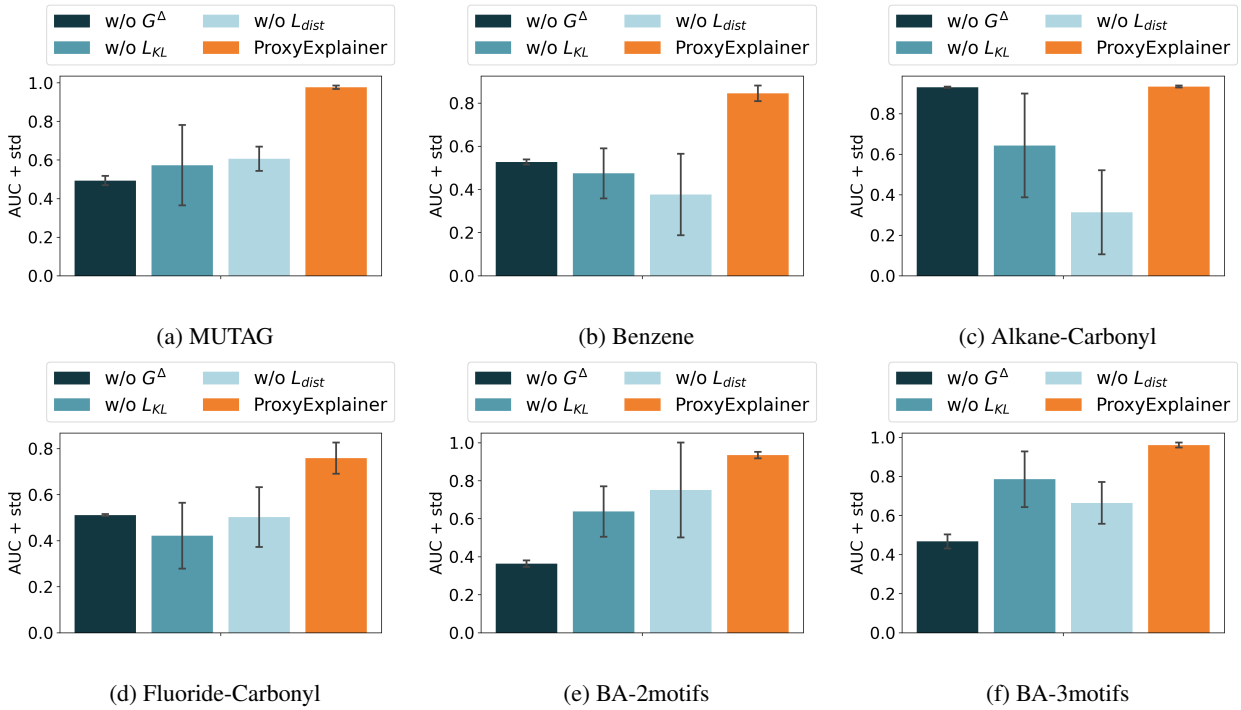
As shown in previous work (Zhang et al., 2023b), another way to approximate the distribution differences of graphs is to compare their vector embeddings. Here, we adopt the same setting and use Cosine similarity and Euclidean distance to intuitively approximate the similarities between graph distributions. Table 8 presents the computed Cosine similarity and Euclidean distance between the distribution embeddings of the original graph $\boldsymbol{h}$, the ground truth explanation subgraph $\boldsymbol{h}'$, and the generated proxy graph $\tilde{\boldsymbol{h}}$. Notably, our proxy graph embedding $\tilde{\boldsymbol{h}}$ exhibits higher Cosine similarity scores and lower Euclidean distance with the original graph embedding $\boldsymbol{h}$ compared to the ground truth explanation embedding $\boldsymbol{h}'$. We observe an average improvement of 19.5% in Cosine similarity and 35.6% in Euclidean distance. Particularly in the BA-2motifs dataset, there is a significant improvement of 60.4% in Cosine similarity and 51.8% in Euclidean distance. These findings underscore the effectiveness of our ProxyExplainer method in mitigating distribution shifts caused by induction bias in the to-be-explained GNN model $f(\cdot)$, thereby enhancing explanation performance.

Table 8. The Cosine similarity score and Euclidean distance between the distribution embeddings of the original graph $\boldsymbol{h}$, explanation subgraph $\boldsymbol{h}'$, and our proxy graph $\tilde{\boldsymbol{h}}$ on different datasets.

	MUTAG	Benzene	Alkane-Carbonyl	Fluoride-Carbonyl	BA-2motifs	BA-3motifs
Avg. Cosine($\boldsymbol{h}, \boldsymbol{h}'$) $\uparrow$	0.883	0.835	0.889	0.904	0.571	0.686
Avg. Cosine($\boldsymbol{h}, \tilde{\boldsymbol{h}}$) $\uparrow$	**0.985**	**0.905**	**0.938**	**0.908**	**0.916**	**0.918**
Avg. Euclidean($\boldsymbol{h}, \boldsymbol{h}'$) $\downarrow$	0.975	1.010	0.940	0.806	1.210	1.199
Avg. Euclidean($\boldsymbol{h}, \tilde{\boldsymbol{h}}$) $\downarrow$	**0.368**	**0.767**	**0.719**	**0.779**	**0.583**	**0.613**

D.2. Extensive Ablation Study

In Figure 6, we conduct a comprehensive ablation study across all datasets to examine the impact of different components. The findings illustrate the effectiveness of these components and their positive contribution to our ProxyExplainer.



(a) MUTAG (b) Benzene (c) Alkane-Carbonyl

(d) Fluoride-Carbonyl (e) BA-2motifs (f) BA-3motifs

Figure 6. Ablation study on all datasets.

D.3. Extensive Experiment with GIN

In order to show the robustness of our ProxyExplainer in explaining different GNN models, we first pre-train GIN on two real-world datasets and two synthetic datasets to ensure that GIN has the ability to classify graphs accurately. The results are displayed in Figure 7. Then we use both baseline methods in the previous experiment and our method ProxyExplainer to provide explanations for the pre-trained GIN model. As seen in Figure 8, it is noticeable that ProxyExplainer achieves the best performance on these datasets. Specifically, it improves the AUC scores of 2.9% on Alkane-Carbonyl, 7.9% on Fluoride-Carbonyl, 45.9% on BA-2motifs, and 6.1% on BA-3motifs. From the analysis, we can see that our ProxyExplainer can identify accurate explanations across different datasets.

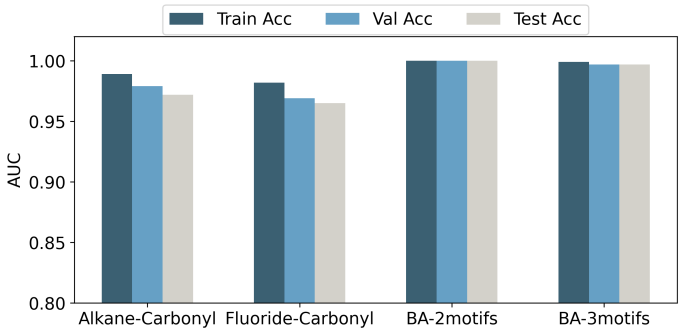
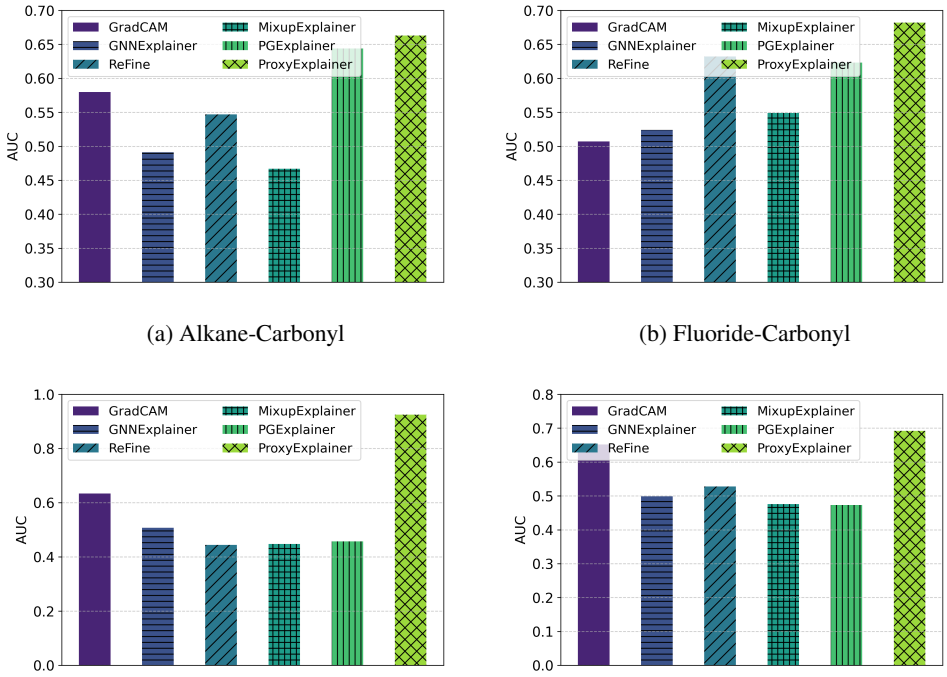


Figure 7. The graph-classification task performances of the GIN model.



(a) Alkane-Carbonyl (b) Fluoride-Carbonyl

(c) BA-2motifs (d) BA-3motifs

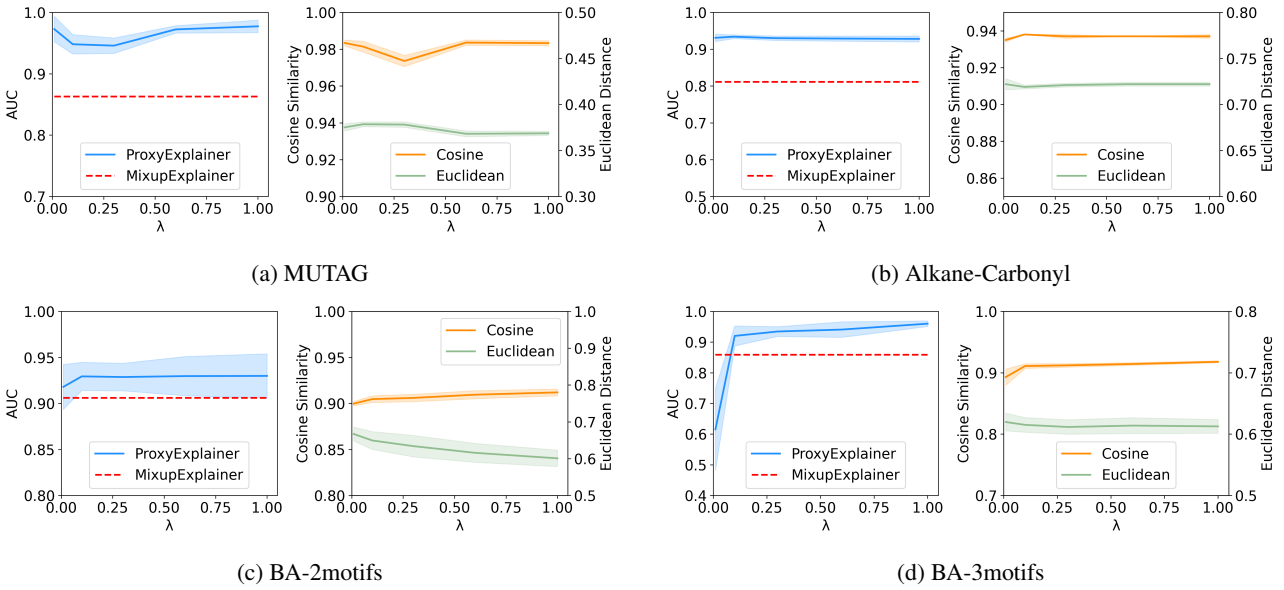
Figure 8. Explanation accuracy in terms of AUC-ROC on edges based on GIN.

D.4. Parameter Sensitive Analysis

In this section, we analyze the influence of parameters including λ, which controls the KL divergence during the proxy graph generation process, and the dimension D of node latent embedding. We vary λ from 0.01 to 1.0. For D, we vary it among

$\{32, 64, 128, 256, 512, 1024\}$. We only change the value of a specific parameter while fixing the remaining parameters to their respective optimal values.

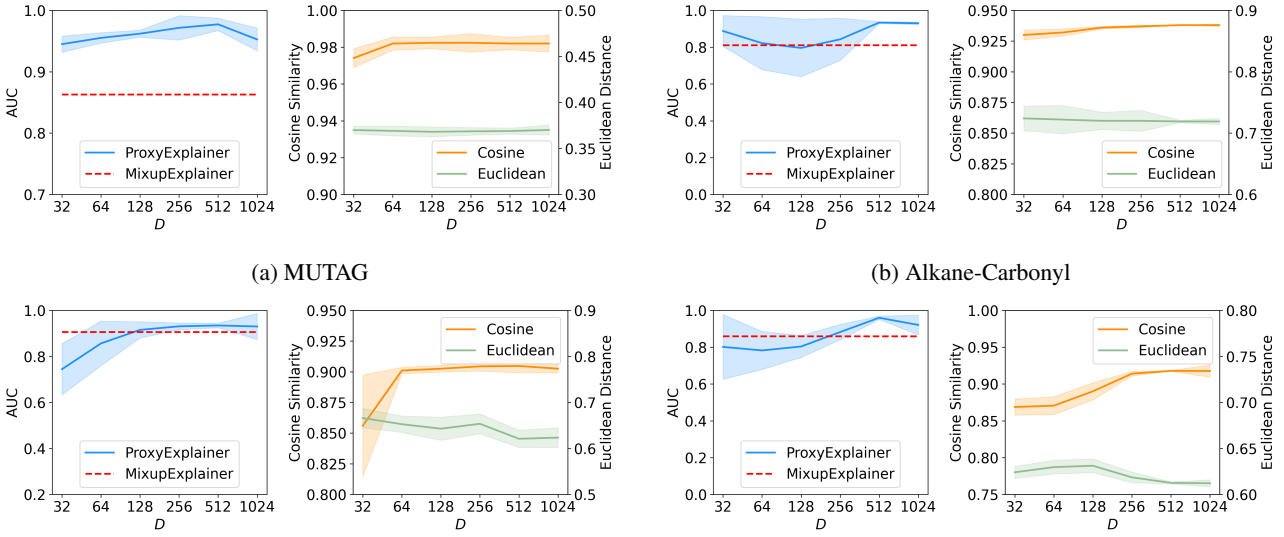
Figure 9 shows the performance of our ProxyExplainer with respect to λ on two real-world datasets (MUTAG and Alkane-Carbonyl) and two synthetic datasets (BA-2motifs and BA-3motifs). From Figure 9, we can see that ProxyExplainer consistently outperforms the best baseline, MixupExplainer, for $\lambda \in [0.25, 1.0]$. This indicates that ProxyExplainer is stable. Figure 10 presents how the dimension of node latent embedding affects performance in ProxyExplainer, the results show that ProxyExplainer can reach the best performance at $D = 512$.



(a) MUTAG (b) Alkane-Carbonyl

(c) BA-2motifs (d) BA-3motifs

Figure 9. Parameter analysis of λ on four datasets. The left side of each graph shows the explanation performance. The right side shows the Distance Analysis between $\boldsymbol{h}$ and $\tilde{\boldsymbol{h}}$.



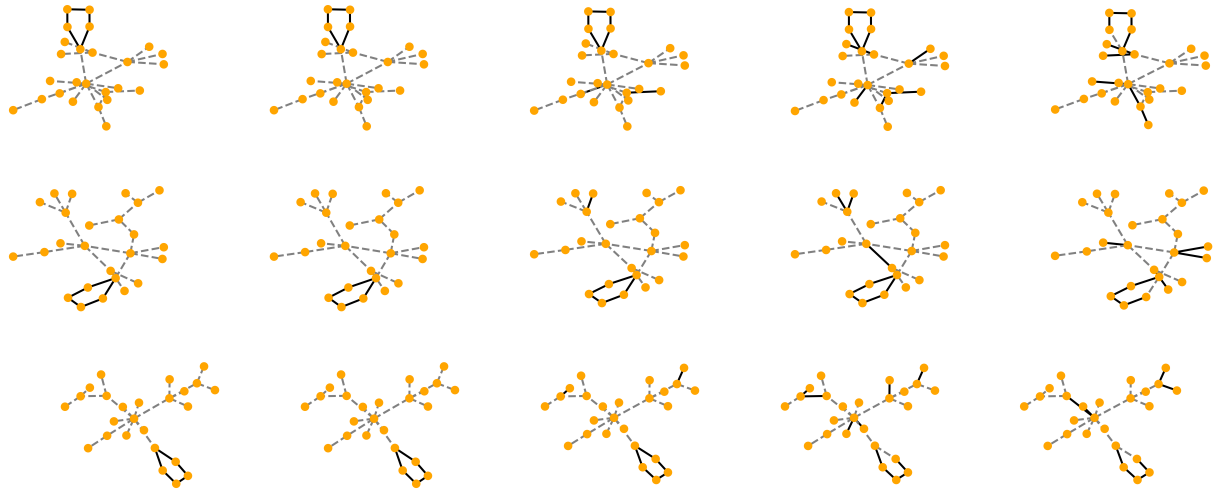
(a) MUTAG (b) Alkane-Carbonyl

(c) BA-2motifs (d) BA-3motifs

Figure 10. Parameter analysis of the dimension of node latent embedding. The left side of each graph shows the explanation performance and the right side displays the Cosine score and Euclidean distance between $\boldsymbol{h}$ and $\tilde{\boldsymbol{h}}$.

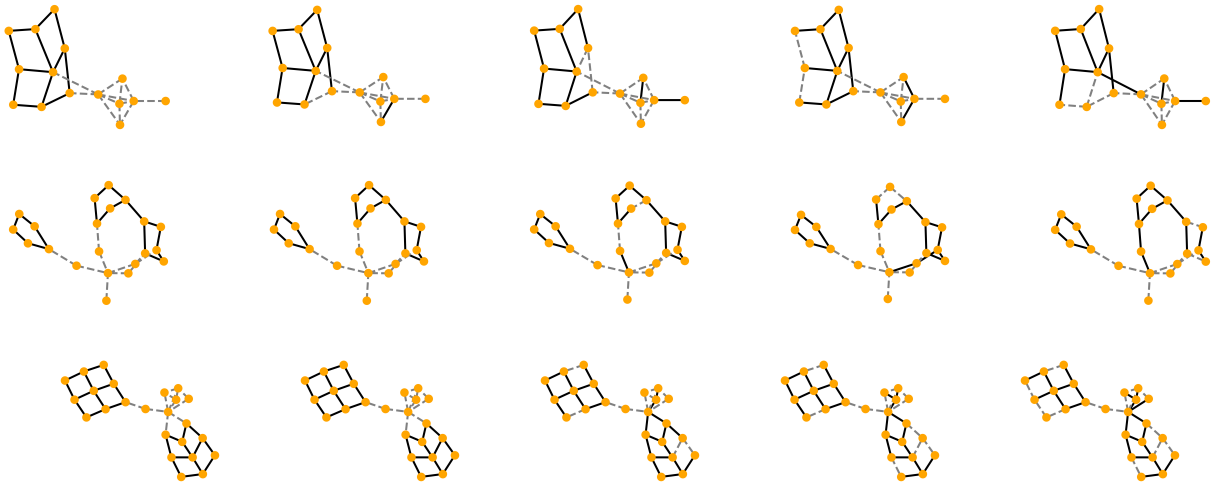
D.5. Extensive Case Study

We show more visualization examples of explanation graphs on BA-2motifs, BA-3motifs, and Benzene in Figure 11, Figure 12, and Figure 13, respectively.



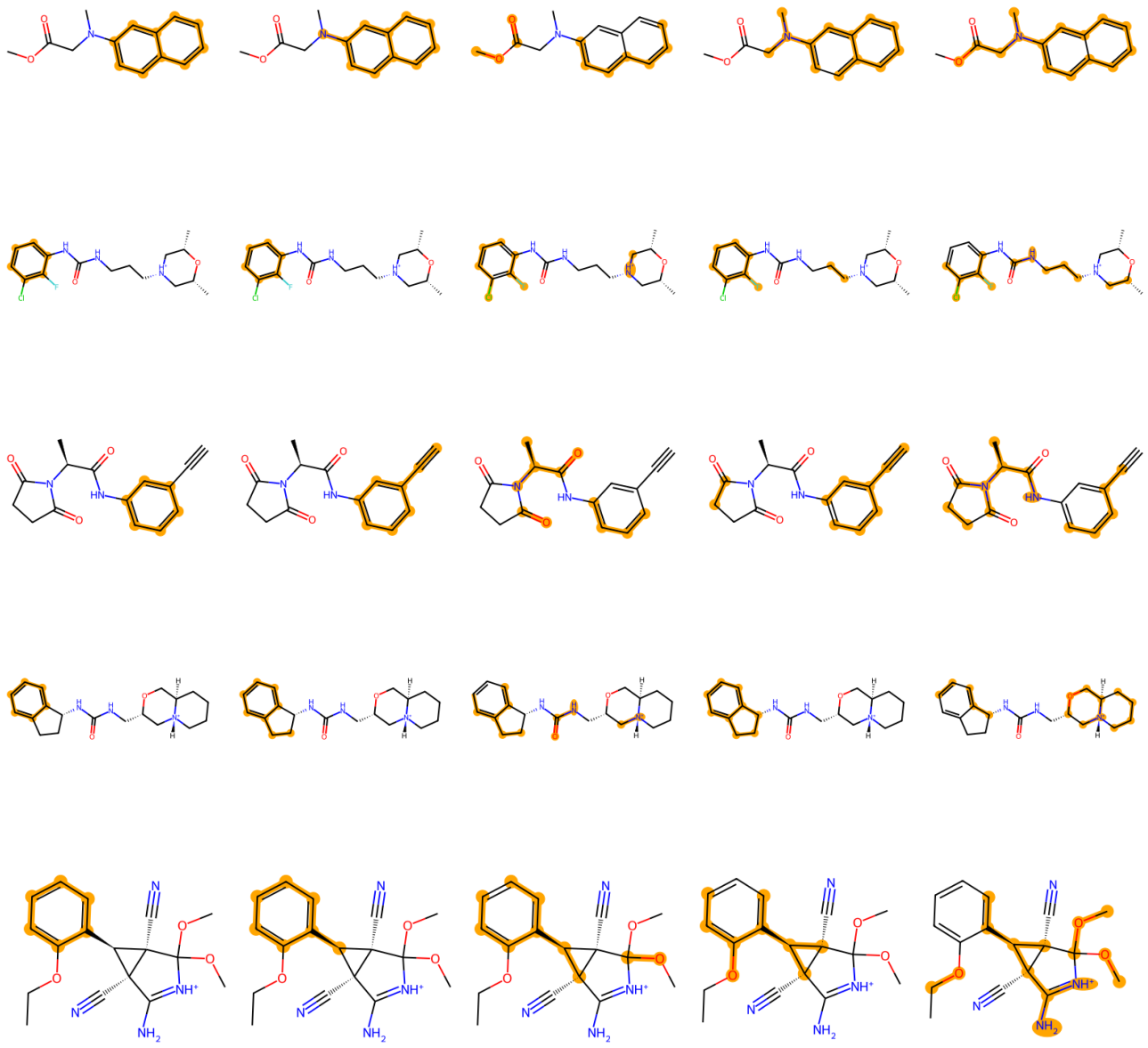
(a) Ground Truth (b) ProxyExplainer (c) MixupExplainer (d) PGExplainer (e) GNNExplainer

Figure 11. Visualization of explanation on BA-2motifs.



(a) Ground Truth (b) ProxyExplainer (c) MixupExplainer (d) PGExplainer (e) GNNExplainer

Figure 12. Visualization of explanation on BA-3motifs.



(a) Ground Truth (b) ProxyExplainer (c) MixupExplainer (d) PGExplainer (e) GNNExplainer

Figure 13. Visualization of explanation on Benzene.