
Towards Robust Heterogeneous Graph Explanations under Structural Perturbations

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 Explaining the prediction process of Graph Neural Networks (GNNs) is critical for
2 enhancing model transparency and trustworthiness. However, real-world graphs are
3 predominantly heterogeneous and often suffer from structural noise, which severely
4 hampers the reliability of existing explanation methods. To address this challenge,
5 we propose RoHeX, a Robust Heterogeneous Graph Neural Network Explainer.
6 RoHeX begins with a theoretical analysis of how different heterogeneous GNN
7 architectures amplify noise through message passing. To mitigate this effect, we
8 introduce a denoising variational inference framework that operates on the graph
9 structure to extract robust latent representations. Furthermore, RoHeX incorporates
10 heterogeneous edge semantics into the subgraph generation process and formulates
11 explanation as an optimization problem under the graph information bottleneck
12 principle. This enables RoHeX to generate explanations that are both semantically
13 meaningful and structurally robust. Extensive experiments on multiple real-world
14 heterogeneous graph datasets demonstrate that RoHeX significantly outperforms
15 state-of-the-art baselines in terms of explanation quality and robustness to noise.

16 1 Introduction

17 Graph Neural Networks (GNNs) have emerged as powerful tools for learning from graph-structured
18 data, demonstrating strong performance across domains such as social networks [1], citation
19 graphs [2], and recommender systems [3]. Despite these successes, GNNs remain largely opaque:
20 their predictions are difficult to interpret, which limits their deployment in sensitive domains involving
21 fairness, privacy, and security [4, 5].

22 To address this, GNN explainers aim to reveal the decision rationale behind model predictions by
23 identifying critical substructures. Existing approaches fall into two categories: post-hoc methods [6,
24 7, 8], which explain a pretrained model without modifying it, and built-in methods [9, 10, 11], which
25 generate explanations during model training. Post-hoc methods are more flexible and generalizable,
26 while built-in methods often yield task-specific insights. However, both struggle under complex, noisy
27 conditions—especially in heterogeneous graphs, which are the norm in real-world settings [12, 13].

28 Heterogeneous graphs, composed of diverse node and edge types, introduce nontrivial challenges.
29 Their interwoven semantics and irregular structures complicate subgraph extraction. Moreover,
30 real-world graphs commonly exhibit noise such as irrelevant or missing edges, which exacerbate the
31 challenges in achieving robust model explainability [14, 15]. The diversity in distributions, attributes,
32 and application domains across heterogeneous graphs also limits the generalizability of existing
33 explainers. While post-hoc methods offer broader applicability, they remain vulnerable to structural
34 irregularities and noise, which can distort the model’s reasoning process. Moreover, noise amplifies
35 structural irregularities and shifts node importance, rendering conventional methods that rely on strict
36 constraints (e.g., subgraph size, connectivity, or budget) ineffective [16].

In this paper, we propose RoHeX, a Robust Heterogeneous Graph Neural Network Explainer. First, we theoretically analyze how noise amplification occurs in heterogeneous GNNs. Second, we introduce a denoising variational inference module that learns robust latent representations by filtering noise in the input graph. Third, we design a heterogeneous explanation generator based on relation-aware attention, which captures the rich semantics across node and edge types. Finally, by integrating the Graph Information Bottleneck (GIB) principle, we reframe explanation as an information-theoretic optimization problem to better handle irregular structures. We validate RoHeX’s performance through extensive experiments on multiple real-world datasets, demonstrating its superior ability to handle noise and generate high-quality explanations compared to state-of-the-art GNN explainers.

The contributions of this paper are as follows:

- We present the first systematic analysis of how noise interacts with heterogeneity in GNN explainers, theoretically proving that existing heterogeneous GNNs amplify structural noise and degrade explainability.
- We propose RoHeX, a robust heterogeneous GNN explainer that integrates denoising variational inference and a relation-aware explanation generator, effectively modeling heterogeneity while suppressing noise during explanation generation.
- Extensive experiments on multiple real-world heterogeneous graphs demonstrate that RoHeX consistently outperforms existing explainers in both explanation fidelity and robustness under noisy conditions.

2 Problem Definition

2.1 Heterogeneous Graph

A heterogeneous graph, denoted as $\mathcal{G} = (\mathbf{A}, \mathbf{X}, \mathcal{A}, \mathcal{R}, \Phi)$, encompasses multiple types of nodes $\mathcal{V}$ and edges $\mathcal{E}$, where $\mathbf{A}$ is the corresponding adjacency matrix, $\mathbf{X}$ represents node features, $\mathcal{A}$ denotes the set of node types, $\mathcal{R}$ signifies the set of edge types, and Φ represents the set of meta-paths. A meta-path $\phi \in \Phi$ is a path of edges connecting various types of nodes from node v_1 to node v_{l+1} , such as $\mathcal{A}_1 \xrightarrow{\mathcal{R}_1} \mathcal{A}_2 \xrightarrow{\mathcal{R}_2} \dots \xrightarrow{\mathcal{R}_l} \mathcal{A}_{l+1}$, where l denotes the length of the meta-path. The label set of $\mathcal{G}$ is denoted as $\mathbf{Y}$, comprising $\mathcal{C}$ categories. Meanwhile, a heterogeneous graph has two mapping functions $\psi(v) : \mathcal{V} \rightarrow \mathcal{A}$ and $\varphi(e) : \mathcal{E} \rightarrow \mathcal{R}$ that correspond to nodes and edges, respectively.

2.2 Heterogeneous Graph Neural Network Explainer

Given a trained GNN model f as the subject of explanation and a heterogeneous graph $\mathcal{G}$, the objective of the GNN explainer is to identify the most influential subgraph $\mathcal{G}_s = (\mathbf{A}_s, \mathbf{X}, \mathcal{A}_s, \mathcal{R}_s)$. Here, $\mathbf{A}_s$ represents the adjacency matrix formed by nodes $\mathcal{V}_s$ and $\mathcal{E}_s$ which significantly contribute to prediction. For the original prediction of GNN model f , it can be formalized as follows:

$$\hat{y} = \operatorname{argmax}_{c \in \mathcal{C}} P_f(c | \mathbf{A}, \mathbf{X}, \mathcal{A}, \mathcal{R}), \quad (1)$$

where $P_f(\cdot)$ represents the prediction function of the GNN model f . Current research indicates that graph structural information is crucial for classification tasks [16, 17]. Therefore, our RoHeX focuses on exploring structural noise when generating explanations. The excellent explanation should contain the most critical information to approximate the predicted labels and outcome changing when predicting the remaining part of the original graph, which is as follows:

$$\operatorname{argmax}_{c \in \mathcal{C}} P_f(c | \mathbf{A}_s, \mathbf{X}, \mathcal{A}_s, \mathcal{R}_s) = \hat{y}. \quad (2)$$

3 Methodology

Figure 1 illustrates the overall framework of RoHeX. We begin by applying a denoising variational graph encoder to obtain a robust latent representation of the input graph $\mathcal{G}$. Node embeddings sampled

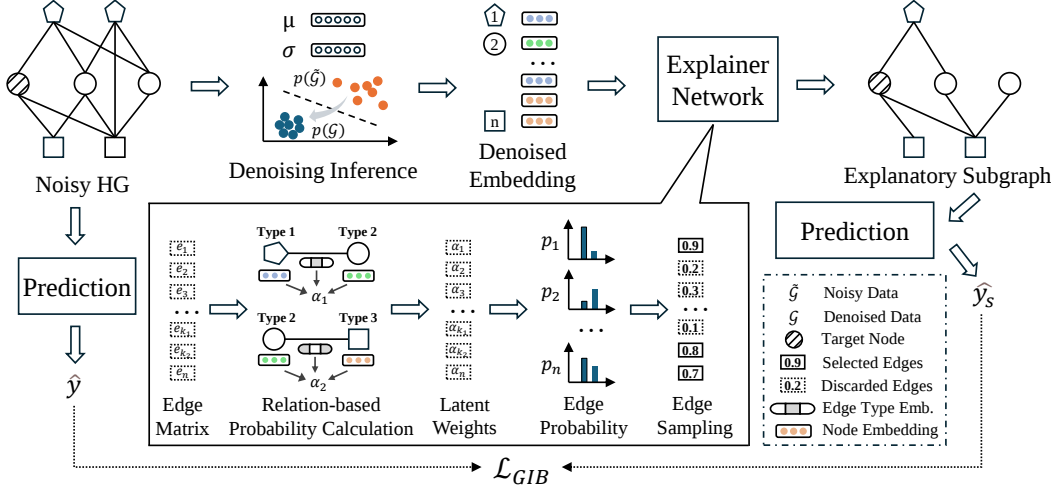


Figure 1: The architecture of our proposed RoHeX. First, the denoised node representations are obtained from the noisy graph via denoising variational inference. Then, the Explainer Network employs the heterogeneous relation-based importance computation method to obtain the weights for different edges. The top k percent of edges are selected as important edges to generate the explanatory subgraph. Finally, the generated explanatory subgraph and the original graph are respectively input into heterogeneous GNN models to obtain predictions, which are used to compute the loss function.

from the latent structural distribution are used to construct edge representations. These edge features are then fed into a heterogeneous relation-aware attention module, which estimates the importance of each edge by modeling the semantics of different edge types. Based on the learned importance scores, RoHeX generates a compact and informative explanatory subgraph. Finally, the entire process is optimized under the graph information bottleneck objective, which adaptively promotes structural sparsity and robustness against irregularities.

3.1 Controllable Structural Perturbation for Heterogeneous Graphs

Real-world graph noise often arises from missing or spurious edges and is typically modeled via random edge addition or deletion [18, 19, 20]. Building upon this idea, we introduce a controllable and heterogeneous-aware structural perturbation strategy—a heuristic but flexible method designed to simulate realistic noise, maintain comparability with prior work, and enable targeted evaluation under adversarial or highly irregular settings. Notably, this method can be changed into a graph structure attack method to achieve structural corruption.

For each edge type $r \in \mathcal{R}$, we define its deletion rate η_r^+ and false addition rate η_r^- , which represent the probabilities of deleting and adding edges of this type, respectively. The perturbed adjacency $\tilde{\mathbf{A}}_r$ for edge type r is generated as:

$$\tilde{\mathbf{A}}_{ij}^r = \begin{cases} 0 & \text{with probability } \eta_r^+ \cdot \frac{d_i^r + d_j^r}{2d^r}, \\ 1 & \text{with probability } \eta_r^- \cdot \frac{1}{d_i^r + d_j^r}, \\ \mathbf{A}_{ij} & \text{otherwise,} \end{cases} \quad (3)$$

where d_i^r is the degree of node v_i under edge type r , and $\bar{d}^r = \frac{1}{|\mathcal{V}_r|} \sum_{i \in \mathcal{V}_r} d_i^r$ denotes the average degree for type r . This degree-aware perturbation prevents the disproportionate removal of hub-node edges and limits unrealistic edge additions, thus preserving key graph structure properties. To control the overall noise intensity, we introduce a global noise budget B , representing the total number of edges to be perturbed. This budget is allocated across edge types according to their importance scores:

$$B_r = B \cdot \frac{s_r}{\sum_{r' \in \mathcal{R}} s_{r'}}, \quad \text{where } s_r = \frac{|\mathcal{E}_r|}{|\mathcal{E}|} \cdot \log |\mathcal{E}_r|, \quad (4)$$

with s_r capturing both the proportion and diversity of edge type r . We then calibrate the edge type-specific perturbation rates:

$$\eta_r^+ = \frac{B_r^+}{|\mathcal{E}_r|}, \quad \eta_r^- = \frac{B_r^-}{|\mathcal{V}_r|^2 - |\mathcal{E}_r|}, \quad \text{s.t. } B_r^+ + B_r^- = B_r. \quad (5)$$

Finally, we provide a theoretical analysis showing how degree-aware perturbation impacts node connectivity:

Theorem 3.1 (Degree-Aware Perturbation). *For node v_i , its expected perturbed degree d'_i under the above strategy satisfies:*

$$E[d'_i] = d_i + \sum_{r \in \mathcal{R}} \left(\eta_r^- \cdot \frac{(|\mathcal{V}_r| - d_i^r)}{d_i + \bar{d}^r} - \eta_r^+ \cdot \frac{d_i^{r^2}}{2\bar{d}^r} \right).$$

By incorporating degree balance constraints and noise budget allocation, our method simulates real-world noise more realistically and preserves the structural properties of the original graph.

3.2 Noise Analysis and Denoising Variational Inference

We investigate the impact of noise on different approaches for Heterogeneous Graph Neural Network (HGNN). We categorize common HGNN into two classes: meta-path-based and neighborhood aggregation-based methods. Meta-path-based methods typically require defining a meta-path ϕ , and then capturing information along different relations following the meta-path structure, aggregating this information, such as Paths2Pair [21] and MAGNET [22]. Neighborhood aggregation-based methods simultaneously consider the neighbor node types and edge types and use specific aggregation functions to combine information from different types. Common neighborhood aggregation methods include MHGCN [23] and Simple-HGN [24]. However, these two categories of methods differ in their efficiency of noise propagation [20], and we find that meta-path-based message passing methods amplify the impact of noise.

Theorem 3.2 (Noise Amplification Effect in HG). *In HG, compared to neighborhood aggregation-based methods, meta-path-based methods can significantly amplify the effect of noisy edges. Specifically, for a node v_i and a newly added noisy edge e_{ij} , the factor by which its influence changes is $\frac{d_{v_i} + k}{d_{v_i} + 1}$, where k is the degree of the new neighbor v_j under the noise and d_{v_i} is the degree of v_i . When $k > d_{v_i}$, this factor is significantly greater than 1.*

The complete proof of Theorem 3.2 is provided in Appendix D. Based on Theorem 3.2, we employ a neighborhood aggregation method to encode heterogeneous graph and mitigate noise. Given noisy graph data $\tilde{\mathcal{G}}$, our objective is to obtain a denoised version of the standard graph data $\mathcal{G}$. VGAE [25] uses variational inference to derive statistical properties of the graph. The statistical data of latent variables in VGAE can be efficiently inferred from the latent space rather than the observation space, which provides robust graph information. For the standard graph $\mathcal{G}$, it initially generates latent variables $\mathbf{Z}$ from a prior distribution $p(\mathbf{Z})$, such as a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$. Second, the data $\mathcal{G}$ is generated using a conditional distribution $p(\mathcal{G}|\mathbf{Z})$. VGAE optimizes its parameters by maximizing the likelihood of the observed data, which as follows:

$$\text{KL}(q_\Psi(\mathbf{Z}|\mathcal{G})||p_\theta(\mathbf{Z}|\mathcal{G})) + \mathcal{L}(\Psi, \theta; \mathcal{G}), \quad (6)$$

where Ψ is the encoder and θ represents the parameters to be optimized. Then, the evidence lower bound $\mathcal{L}(\Psi, \theta; \mathcal{G})$ can be expressed as follows:

$$\mathcal{L}(\Psi, \theta; \mathcal{G}) = \mathbb{E}_{q_\Psi(\mathbf{Z}|\mathcal{G})} \left[\log \frac{p_\theta(\mathbf{Z}, \mathcal{G})}{q_\Psi(\mathbf{Z}|\mathcal{G})} \right] = \mathbb{E}_{q_\Psi(\mathbf{Z}|\mathcal{G})} [\log p_\theta(\mathbf{Z}|\mathcal{G})] - \text{KL}(q_\Psi(\mathbf{Z}|\mathcal{G})||p(\mathbf{Z})). \quad (7)$$

Variational inference enhances the model’s robustness and generalization capabilities [26, 27]. However, due to the differing distributions between noisy heterogeneous graph data and standard graph data, the obtained distribution tends to align with the noisy distribution, potentially misleading the GNN explainer into generating incorrect explanatory subgraphs. Therefore, we introduce a denoising module during the process of variational inference. The original encoder part is modified to:

$$q'_\Psi(\mathbf{Z}|\mathcal{G}) = \int q_\Psi(\mathcal{G}|\tilde{\mathcal{G}})q(\tilde{\mathcal{G}}|\mathcal{G})d\tilde{\mathcal{G}}, \quad (8)$$

141 where Ψ is the encoder based on $\tilde{\mathcal{G}}$, and $q(\tilde{\mathcal{G}}|\mathcal{G}) = \prod_{r \in \mathcal{R}} q(\tilde{\mathbf{A}}_r|\mathbf{A}_r)$. During this process, the
 142 evidence lower bound is expressed as:

$$\mathcal{L}_d = \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log \frac{p_{\theta}(\mathbf{Z}, \mathcal{G})}{q'_{\Psi}(\mathbf{Z}|\mathcal{G})}]. \quad (9)$$

143 As we need to derive the distribution of the noisy graph data $\tilde{\mathcal{G}}$, this lower bound can be further
 144 refined as:

$$\begin{aligned} \mathcal{L}_d &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log \frac{p_{\theta}(\mathbf{Z}, \mathcal{G})}{q'_{\Psi}(\mathbf{Z}|\mathcal{G})}] \geq \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log \frac{p_{\theta}(\mathcal{G}, \mathbf{Z})}{q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})} \right] \\ &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log p_{\theta}(\mathcal{G}|\mathbf{Z})] - \mathbb{E}_{q(\tilde{\mathcal{G}}|\mathcal{G})} [\text{KL}(q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})||p(\mathbf{Z}))]. \end{aligned} \quad (10)$$

145 The detailed derivation is in the Appendix E. Compared to VGAE, the denoising variational inference
 146 models the posterior probability $p(\mathbf{Z}|\mathcal{G})$ using a Gaussian Mixture Model, whereas VGAE models
 147 $p(\mathbf{Z}|\mathcal{G})$ using a Gaussian distribution. Additionally, during the optimization process, there is a
 148 constraint that forces $q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})$ to approximate the standard Gaussian distribution $p(\mathbf{Z})$. Consequently,
 149 our method can significantly improve the model's robustness and produce high-quality graph data.
 150 We further employ the Monte Carlo sampling method to approximate the objective, which can be
 151 effectively optimized using gradient descent as follows:

$$\mathcal{L}_d \approx \frac{1}{K} \sum_{k=1}^K \sum_{r \in \mathcal{R}} \log \frac{p_{\theta}(\mathcal{G}_r, \mathbf{Z})}{q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{A}}_r)}, \quad (11)$$

152 where K is the number of samples sampled during the simulation. After denoising variational infer-
 153 ence, we input the sampled robust representations $\mathbf{Z}$ into the Heterogeneous Explanation Generator,
 154 where the complex semantics on the heterogeneous graph are learned. Before delving into that, we
 155 introduce the Graph Information Bottleneck.

156 3.3 Graph Information Bottleneck

157 As mentioned in the introduction, noise exacerbates the irregularity of graph structures and alters node
 158 importance. Therefore, previous methods imposing structural regularity constraints on explanatory
 159 subgraphs are infeasible under noise influence. We exploit the Graph Information Bottleneck (GIB)
 160 to enable the explainer network to adaptively handle structural irregularities. The objective of GIB
 161 is to obtain the optimal explanatory subgraph $\mathcal{G}_s$. From an information-theoretic perspective, GIB
 162 limits the amount of information carried by the explanatory subgraph $\mathcal{G}_s$, rather than imposing
 163 simple structural constraints. Simultaneously, nodes may require scattered edges across the graph to
 164 jointly explain their predictive function, rather than constraining connectedness. Consequently, GIB
 165 adaptively explores the entire graph without imposing any potentially biased restrictions. GIB can be
 166 formulated as:

$$\min_{\mathcal{G}_s \subset \mathcal{G}} -\text{I}(\hat{y}; \mathcal{G}_s) + \beta \text{I}(\mathcal{G}; \mathcal{G}_s), \quad (12)$$

167 where $\text{I}(\cdot; \cdot)$ denotes mutual information, and β controls the trade-off between the two terms. Since
 168 the information in $\mathcal{G}_s$ can be continually optimized, the explain task can be characterized as an
 169 optimization task guided by GIB.

170 The GIB principle aims to obtain the minimum sufficient information about the graph $\mathcal{G}$. The first
 171 term maximizes the mutual information between the label and the explanatory subgraph, ensuring $\mathcal{G}_s$
 172 contains as much information about the label as possible. The second term minimizes the mutual
 173 information between the input graph and the explanatory subgraph, ensuring $\mathcal{G}_s$ contains the minimum
 174 information about the input graph. Next, we introduce the Heterogeneous Explanation Generator,
 175 describing how each term is optimized during training under the GIB principle.

176 3.4 Heterogeneous Explanation Generator

177 We begin by modeling the explanatory subgraph as a Gilbert random graph [28], where edges are
 178 conditionally independent. Following the literature [16], we define an adjacency matrix-like edge
 179 matrix E_s , where each element e_{ij} is a binary variable indicating whether the edge is included in the
 180 subgraph. When there is an edge (i, j) from v_i to v_j , $e_{ij} = 1$, otherwise $e_{ij} = 0$.

181 To capture the rich semantics of heterogeneous graphs, pairwise node interactions alone are in-
 182 sufficient. Thus, we incorporate heterogeneous semantics learning into the explanatory subgraph
 183 generation process. We incorporate heterogeneous edge type information into the attention compu-
 184 tation by extending the standard graph attention mechanism. Specifically, we assign an edge type
 185 embedding $\mathbf{r}_{\varphi(e)}$ for each edge type $\varphi(e)$, and simultaneously utilize the edge type embeddings and
 186 node embeddings to compute the attention coefficient α_{ij} :

$$\alpha_{ij} = \frac{\exp(\text{ReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{z}_i\|\mathbf{W}\mathbf{z}_j\|\mathbf{W}_r\mathbf{r}_{\varphi(e_{ij})}]))}{\sum_{k \in \mathcal{N}_i} \exp(\text{ReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{z}_i\|\mathbf{W}\mathbf{z}_k\|\mathbf{W}_r\mathbf{r}_{\varphi(e_{ik})}]))}, \quad (13)$$

187 where $\mathbf{W}_r$ is a learnable weight matrix for type embeddings. Edge type embedding is a one-hot
 188 encoding derived from each edge type. This attention coefficient α_{ij} integrates both heterogeneous
 189 node and edge type semantics, offering a more comprehensive representation.

190 Next, we define the heterogeneous random graph variable. The probability of the heterogeneous
 191 explanatory subgraph can be factorized as:

$$p(\mathcal{G}) = \prod_{(i,j) \in E_s} p(e_{ij}|\varphi(e_{ij})), \quad (14)$$

192 where $e_{ij} \sim \text{Bern}(\pi_{ij})$ and π_{ij} is the edge existence probability inferred via α_{ij} . To enable
 193 backpropagation through discrete edge selections, we adopt the reparameterization trick using a
 194 hard-concrete relaxation:

$$e_{ij} = \text{Sigmoid}\left(\frac{\log \epsilon - \log(1 - \epsilon) + \alpha_{ij}(\varphi(e_{ij}))}{\tau}\right), \quad (15)$$

$$\epsilon \sim \text{Uniform}(0, 1),$$

195 where τ is a temperature coefficient to smooth the optimization, and α_{ij} from Eq. 13 adds heteroge-
 196 neous information into the explanatory subgraph. When $\alpha_{ij} = \log \frac{\pi_{ij}}{1-\pi_{ij}}$, we have $\lim_{\tau \rightarrow 0} p(e_{ij} =$
 197 $1) = \frac{\exp(\alpha_{ij})}{1+\exp(\alpha_{ij})}$, so we can obtain the explanatory subgraph $\mathcal{G}_s$ since $p(e_{ij} = 1) = \pi_{ij}$.

198 This results in a continuous probability matrix $\mathbf{M}_p \in \mathbb{R}^{N \times N}$, where each entry $[\mathbf{M}_p]_{ij} = \pi_{ij}$
 199 denotes the likelihood of including edge (i, j) . We then construct the soft explanatory subgraph:

$$\mathcal{G}_s = (\mathbf{A}_s = \mathbf{M}_p \odot \mathbf{A}, \mathbf{X}, \mathcal{A}_s, \mathcal{R}_s). \quad (16)$$

200 To optimize the explainer, we adopt the Graph Information Bottleneck (GIB) principle, balancing
 201 predictive fidelity and information compression. The GIB objective 12 is upper-bounded as:

$$-I(\hat{y}; \mathcal{G}_s) + \beta I(\mathcal{G}; \mathcal{G}_s) \leq -\mathbb{E}_{p(\mathcal{G}_s, \hat{y})} [\log p_f(\hat{y}|\mathcal{G}_s)] + H(\hat{y}) + \beta \mathbb{E}_{p(\mathcal{G})} [\text{KL}(p_\alpha(\mathcal{G}_s|\mathcal{G})||q(\mathcal{G}_s))], \quad (17)$$

202 where f is the GNN model and α is the explain model, see Appendix E for detailed derivation. Since
 203 $H(\hat{y})$ is constant, the objective function can be expressed as follows:

$$\mathcal{L}_{GIB} = -\mathbb{E}_{p(\mathcal{G}_s, \hat{y})} [\log p_f(\hat{y}|\mathcal{G}_s)] + \beta \mathbb{E}_{p(\mathcal{G})} [\text{KL}(p_\alpha(\mathcal{G}_s|\mathcal{G})||q(\mathcal{G}_s))]. \quad (18)$$

204 The total loss combines the GIB objective with the denoising loss from the variational graph encoder:

$$\mathcal{L} = \mathcal{L}_d + \mathcal{L}_{GIB}. \quad (19)$$

205 3.5 Complexity Analysis.

206 The cost of each iteration comprises two parts: (1) the variational inference process and (2) the
 207 heterogeneous explanation generation. The time complexity of the first step is $O(N^2 + E)$, and the
 208 space complexity is $O(N)$, as this step requires storing the robust node representations. The time
 209 complexity of the second step is $O(E)$, and the space complexity is $O(E)$. Therefore, the overall
 210 time complexity of RoHeX is $O(N^2 + E)$, and the space complexity is $O(N + E)$.

211 4 Experiment

212 In this section, we evaluate the performance of the proposed RoHeX and state-of-the-art baselines on
 213 the node classification task. We then analyze the contributions of different components of RoHeX and
 214 demonstrate why RoHeX is robust to noise and capable of generating explanations that incorporate
 215 heterogeneous information.

Table 1: The comparison of RoHeX and baselines under different ratios of random structural noise. We use **bold** font to mark the best score. The second best score is marked with underline.

Dataset	Metric	Noise	PGExplainer	GNNExplainer	PGM-Explainer	V-InfoR	AMExplainer	Hete-PGE	xPath	RoHeX
DBLP	MAE	10%	1.2158±0.0062	0.8530±0.0009	1.0704±0.0007	1.1930±0.0030	1.4862±0.0026	0.8719±0.0008	<u>0.8162±0.0032</u>	0.8359±0.0029
		20%	1.2179±0.0089	0.9080±0.0007	1.2046±0.0006	1.1960±0.0025	1.5697±0.0008	0.8896±0.0005	0.9651±0.0042	0.8743±0.0018
		30%	1.2449±0.0059	1.2613±0.0011	1.3313±0.0002	1.2312±0.0026	1.7133±0.0006	1.1913±0.0006	1.1405±0.1032	0.8827±0.0034
		40%	1.2451±0.0068	1.3389±0.0008	1.3401±0.0005	1.2530±0.0010	1.9345±0.0005	0.9268±0.0006	1.2852±0.0002	0.9014±0.0015
	RMSE	10%	1.6775±0.0054	1.2968±0.0006	1.3855±0.0005	1.6481±0.0027	1.9219±0.0048	<u>1.2814±0.0004</u>	1.3175±0.0020	1.2416±0.0017
		20%	1.6815±0.0666	1.3072±0.0004	1.5280±0.0003	1.6511±0.0020	2.0419±0.0006	<u>1.2859±0.0002</u>	1.4341±0.0025	1.2750±0.0013
		30%	1.6999±0.0024	1.8470±0.0007	1.6497±0.0001	1.6781±0.0020	2.1842±0.0003	1.6532±0.0003	1.5752±0.1198	1.2792±0.0022
		40%	1.7060±0.0042	1.9043±0.0004	1.6681±0.0001	1.6885±0.0008	2.3616±0.0002	1.3100±0.0004	1.6819±0.0001	1.2889±0.0008
ACM	MAE	10%	0.7624±0.0080	0.3449±0.0003	<u>0.2155±0.0009</u>	0.7639±0.0001	0.3895±0.0005	0.8091±0.0003	0.3900±0.0001	0.2129±0.0009
		20%	0.7751±0.0162	0.3951±0.0001	<u>0.3732±0.0003</u>	0.7913±0.0004	0.6746±0.0224	0.8183±0.0005	0.3985±0.0003	0.2483±0.0010
		30%	0.7867±0.0152	0.5087±0.0003	0.5932±0.0006	0.8064±0.0003	0.7077±0.0221	0.8220±0.0002	0.4164±0.0012	0.3140±0.0019
		40%	0.7913±0.0181	0.6496±0.0002	0.7932±0.0007	0.8154±0.0005	0.7181±0.0164	0.8310±0.0001	0.4292±0.0006	0.3163±0.0015
	RMSE	10%	1.0258±0.0037	0.6831±0.0005	0.5121±0.0004	1.0145±0.0002	0.6241±0.0003	1.0740±0.0001	0.7750±0.0002	<u>0.5662±0.0012</u>
		20%	1.0307±0.0069	0.7791±0.0002	<u>0.6893±0.0002</u>	1.0366±0.0004	0.8213±0.0649	1.0778±0.0004	0.7458±0.0003	0.6177±0.0008
		30%	1.0423±0.0127	0.9506±0.0001	0.8793±0.0004	1.0705±0.0007	0.8412±0.0698	1.0731±0.0001	0.7768±0.0009	0.6669±0.0013
		40%	1.0431±0.0124	1.1306±0.0002	1.0766±0.0004	1.0786±0.0003	0.8474±0.0713	1.0856±0.0001	0.8396±0.0007	0.6909±0.0008
Freebase	MAE	10%	0.7189±0.0096	0.9012±0.0002	0.9190±0.0003	<u>0.5957±0.0322</u>	0.9312±0.0004	0.7760±0.0007	0.9006±0.0127	0.3885±0.0010
		20%	0.7237±0.0078	0.9108±0.0003	0.9401±0.0001	<u>0.6822±0.0527</u>	0.9709±0.1209	0.7812±0.0005	0.9247±0.0092	0.4441±0.0012
		30%	0.7285±0.0041	0.9126±0.0003	0.9530±0.0004	<u>0.7249±0.0329</u>	1.0089±0.1651	0.7908±0.0003	0.9263±0.0043	0.4694±0.0021
		40%	0.7370±0.0019	0.9378±0.0007	0.9587±0.0009	0.7894±0.0111	1.0531±0.0005	0.8030±0.0001	0.9301±0.0061	0.4880±0.0014
	RMSE	10%	<u>1.0616±0.0071</u>	1.2886±0.0001	1.2432±0.0001	1.0375±0.0233	1.2589±0.0004	1.1064±0.0003	1.2466±0.0056	0.8251±0.0018
		20%	1.0635±0.0051	1.2983±0.0002	1.2549±0.0001	1.1172±0.0566	1.2987±0.1848	1.1117±0.0004	1.2435±0.0108	0.8854±0.0010
		30%	1.0689±0.0039	1.2995±0.0002	1.2747±0.0002	1.1487±0.0277	1.3391±0.2381	1.1200±0.0002	1.2721±0.0034	0.9035±0.0012
		40%	1.0803±0.0018	1.3217±0.0005	1.2838±0.0004	1.1929±0.0054	1.3830±0.0004	1.1315±0.0001	1.2643±0.0072	0.9217±0.0007

4.1 Experiment Settings

Datasets and Baselines. We evaluate the effectiveness of our RoHeX on three real-world datasets, including two academic citation datasets (DBLP and ACM) and a knowledge graph dataset (Freebase). Since there are no existing robust heterogeneous explainer, we select three types of baselines: the surrogate method PGM-Explainer, the perturbation-based methods GNNExplainer, PGExplainer and AMExplainer, and the V-infor method studying robustness on homogeneous graphs. We used the heterogeneous graph path explainer, xPath, and extended our Heterogeneous Explanation Generator to PGExplainer, referred to as Hete-PGE, for comparison.

Evaluation. The evaluation of explainer performance is based on the generated explanatory subgraphs, assessing their contribution to the original prediction. We adopt two metrics: fidelity and sparsity. Fidelity measures the decrease in prediction confidence after removing the explanation from the input graph, while sparsity measures the ratio of remaining edges in the explanatory subgraph $\mathcal{G}_s$ relative to the input graph. We use the Mean Absolute Error (MAE, $\frac{1}{N} \sum_{i=1}^N |\mathbb{I}(\hat{y}_i = y_i) - \mathbb{I}(\hat{y}_i^{\mathcal{G}_s} = y_i)|$), and Root Mean Squared Error (RMSE, $\sqrt{\frac{1}{N} \sum_{i=1}^N (\mathbb{I}(\hat{y}_i = y_i) - \mathbb{I}(\hat{y}_i^{\mathcal{G}_s} = y_i))^2}$) as proxy measures for fidelity, and compare the performance of different baselines across varying sparsity levels, where N is the number of nodes or graphs, $\hat{y}_i$ is the original prediction result, and $\hat{y}_i^{\mathcal{G}_s}$ is the prediction result obtained by the explanatory subgraph.

Implementation Details. We conduct experiments under different proportions of random noise scenarios. Noise is added to both the training set and the test set to restore the real scene. To ensure randomness, the deletion rate and false addition rate are set to be equal. For the baselines, we select the best-performing parameters for heterogeneous datasets based on the original settings. We chose the most basic HGNN architecture, which only includes GCN [29] and relational learning modules, as the base model for fair comparison. For our RoHeX, we use Adam as the optimizer with a learning rate of $1e-4$. We set the hidden dimension for variational inference to 64, the output dimension to 32, and the edge weight output dimension to 32. Each experiment is repeated 5 times, and we report the mean and variance as the results. Descriptions of the variance, datasets, baselines, base HGNN model architecture, and parameter settings are provided in the Appendix F.

4.2 Overall Performance under Structural Perturbations

Table 1 shows the experimental results on the heterogeneous graphs with different budgets of structural perturbations. We find that RoHeX outperforms other baselines in most experimental results, achieving the best performance on the DBLP and Freebase datasets. Taking 30% noise ratio as an example, RoHeX shows 25.9% lower MAE and 22.4% lower RMSE than the second-best method on the DBLP dataset, 38.2% lower MAE and 24.1% lower RMSE on the ACM dataset, and 35.2% lower MAE and 15.4% lower RMSE on the Freebase dataset. xPath performs excellently in multiple

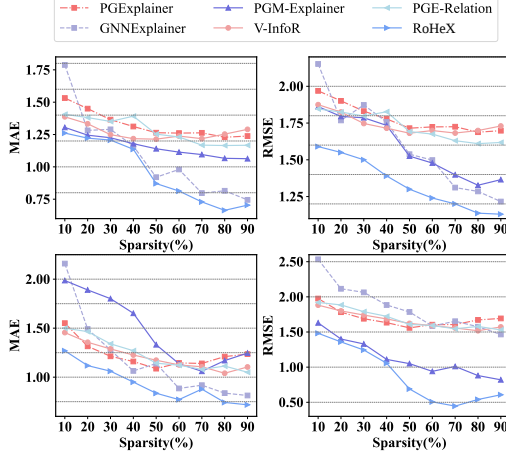


Figure 2: Fidelity-Sparsity Curve on the DBLP dataset. The first row is the result without noise, and the second row is the result with 20% noise budget.

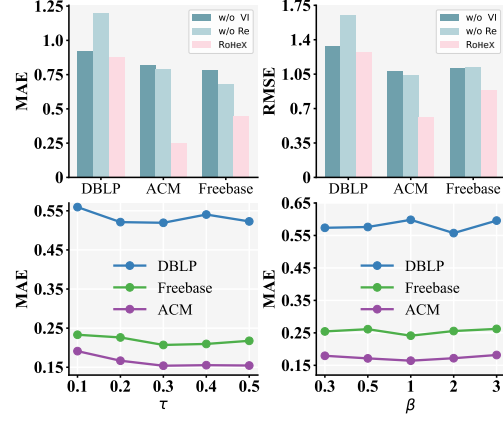


Figure 3: Ablation study on three datasets and the influence of hyperparameters τ and β on RoHeX.

scenarios, indicating that designing corresponding modules for heterogeneous graphs is essential. We can observe that Hete-PGE achieves second best performance multiple times on the DBLP dataset and outperforms many baselines on other datasets, demonstrating the effectiveness of our proposed heterogeneous explanation generator in considering rich semantics on heterogeneous graphs. Due to the similar edge type distribution in the DBLP dataset, the dataset exhibits higher heterogeneity, which enhances the module’s ability to capture heterogeneous information. Simultaneously, as a plug-and-play module, it can be conveniently extended to other parameterized explanation methods for generating explanations on heterogeneous graphs. On the medium and small-scale datasets DBLP and ACM, explanation methods based on raw features (e.g., GNNExplainer) are more susceptible to noise, potentially because raw features are more easily affected in smaller graphs. Since RoHeX generates robust graph representations, it can better mitigate the influence of noise, which is also why the latent representation-based explainer V-InfoR performs well in multiple scenarios. Under the guidance of GIB, our method can adaptively select important edges while excluding redundant and noisy edges, thereby generating the best explanations for the prediction model.

4.3 Fidelity-Sparsity Analysis

Next, we further investigate RoHeX’s performance at different sparsity levels. We provide the Fidelity-Sparsity curve on the DBLP dataset as shown in Figure 2. RoHeX consistently outperforms other baselines across all sparsity levels, indicating that our method can generate the best explanations. As the sparsity increases from 0 to 1, the overall trend of all curves is downward, i.e., decreasing error. When the sparsity is extremely low, e.g., 10%, our method significantly outperforms other baselines, suggesting that RoHeX can identify the truly critical subgraphs. We further find that although the overall performance improves as the sparsity level increases, there are still some cases where the performance drops with increasing sparsity, such as PGExplainer. We conjecture that this may be because in the subgraph generation process, when the sparsity increases to a point where all edges with high importance scores have been selected, forcing higher sparsity will begin to select unimportant edges, which can be viewed as noisy edges, leading to degraded performance. As the sparsity continues to increase, this adverse effect is offset. Since AMExplainer and xPath contain specific Sparsity settings to generate explanations, they were not included in the experiment.

4.4 Model Analysis

Perturbation Performance We analyze the perturbation performance of controllable structural perturbation for heterogeneous graphs under different noise scenarios, as shown in Table 2. Compared to the original results, our structural perturbation method significantly impacts the decision-making process of HGNN on the whole graph (Noisy), 1-hop subgraph (1-hop), and 2-hop subgraph (2-hop).

Table 2: Prediction performance of HGNN in different noise scenarios.

Dataset	Method	10%		20%		30%		40%	
		Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
DBLP	Original	92.64	92.16	92.64	92.16	92.64	92.16	92.64	92.16
	Noisy	81.58	80.34	73.41	70.26	67.18	62.02	62.99	55.91
	1-hop	57.28	55.92	54.22	52.40	52.18	49.98	50.56	48.40
	2-hop	57.64	55.65	42.07	38.55	41.76	38.06	39.01	34.01
ACM	Original	92.32	92.40	92.32	92.40	92.32	92.40	92.32	92.40
	Noisy	33.52	19.28	33.28	18.82	31.96	16.14	31.96	16.14
	1-hop	77.10	77.31	76.39	76.51	75.40	75.57	67.94	67.81
	2-hop	52.19	51.73	49.12	50.65	42.38	36.42	38.73	30.07
Freebase	Original	68.99	63.57	68.99	63.57	68.99	63.57	68.99	63.57
	Noisy	67.89	61.90	66.05	59.29	62.94	53.74	57.50	46.02
	1-hop	58.52	53.23	57.62	51.94	57.50	51.21	57.01	51.14
	2-hop	44.66	31.12	42.90	27.24	41.10	24.15	39.46	21.62

283 An interesting finding is that 30% noise is enough to cause the performance of HGNN on the ACM
284 dataset to drop to its lowest.

285 **Ablation Study** We investigate the contributions of different components in RoHeX. Specifically,
286 we study (a) the effectiveness of the denoising variational inference module, and (b) the effectiveness
287 of the relation-based importance module. We use 'w/o VI' to denote the model without the denoising
288 variational inference module, and 'w/o Re' to denote the model without the relation-based importance
289 module. For the latter case, we replace it with the common concatenation operation, i.e., $\alpha_{ij} =$
290 $\text{MLP}[(\mathbf{z}_i, \mathbf{z}_j)]$. The experiments are conducted under 20% noise budget, and the first row of Figure 3
291 shows the results after ablation. We find that without the denoising variational inference module,
292 the model relies on the original features and graph structure for prediction, failing to mitigate the
293 influence of noise, leading to performance degradation. When the model loses the ability to learn
294 heterogeneous relationships, the process of generating explanation subgraphs struggles to recognize
295 the complex semantics in heterogeneous graphs. All edges are treated as the same type, and the
296 model explains solely based on node interactions. This demonstrates the necessity of our proposed
297 relation importance module.

298 **Hyperparameter Analysis** We further analyze the impact of two parameters τ and β on model
299 performance. τ controls the approximation degree of e_{ij} distribution to the Bernoulli distribution,
300 ranging within $[0.1, 0.5]$. β balances the information recovery strength and information filtering
301 strength in the optimization objective, and we select values from $\{0.3, 0.5, 1, 2, 3\}$. The second row
302 of Figure 3 shows the effects of these hyperparameters on RoHeX across three datasets. We can
303 observe that the best results of τ all appear around 0.3. That is, when $\tau = 0.3$, the continuity and
304 approximation degree in Eq. 15 reach the best trade-off. Secondly, RoHeX is not very sensitive to β
305 that controls the constraint strength in Eq. 18, validating that our used GIB constraint can adapt to
306 different data scenarios and achieve superior performance.

307 5 Conclusion

308 In this work, we focus on the problem of explaining heterogeneous graph neural network under
309 noise. We are the first to study this problem, theoretically proving that heterogeneous graph neural
310 network have an amplifying effect on noise, and propose RoHeX to mitigate the influence of
311 noise and obtain explanatory subgraphs based on heterogeneous relations. Specifically, RoHeX
312 employs denoising variational inference to obtain robust graph representations and parameterizes the
313 explanatory subgraph generation process with heterogeneous semantics. It integrates type information
314 to capture the complexity of diverse node and edge types. Extensive experiments on real-world
315 datasets demonstrate RoHeX's superiority over other state-of-the-art baselines. For future work, we
316 plan to further extend RoHeX to dynamic graphs by incorporating dynamic information into the
317 explanation generation process, further broadening RoHeX's applicability.

References

- [1] Yuwei Cao, Hao Peng, Zhengtao Yu, and S Yu Philip. Hierarchical and incremental structural entropy minimization for unsupervised social event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 8255–8264, 2024.
- [2] Yifan Lu, Mengzhou Gao, Huan Liu, Zehao Liu, Wei Yu, Xiaoming Li, and Pengfei Jiao. Neighborhood overlap-aware heterogeneous hypergraph neural network for link prediction. *Pattern Recognition*, 144:109818, 2023.
- [3] Yi Zhang, Yiwen Zhang, Yuchuan Zhao, Shuiguang Deng, and Yun Yang. Dual variational graph reconstruction learning for social recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2024.
- [4] Peter Müller, Lukas Faber, Karolis Martinkus, and Roger Wattenhofer. Graphchef: Decision-tree recipes to explain graph neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.
- [5] Senzhang Wang, Jun Yin, Chaozhuo Li, Xing Xie, and Jianxin Wang. V-infor: a robust graph neural networks explainer for structurally corrupted graphs. *Advances in Neural Information Processing Systems*, 36, 2024.
- [6] Minh N Vu and My T Thai. Limitations of perturbation-based explanation methods for temporal graph neural networks. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 618–627. IEEE, 2023.
- [7] Tamara Pereira, Erik Nascimento, Lucas E Resck, Diego Mesquita, and Amauri Souza. Distill n’explain: explaining graph neural networks using simple surrogates. In *International Conference on Artificial Intelligence and Statistics*, pages 6199–6214. PMLR, 2023.
- [8] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(7):6968–6972, 2022.
- [9] Sangwoo Seo, Sungwon Kim, and Chanyoung Park. Interpretable prototype-based graph information bottleneck. *Advances in Neural Information Processing Systems*, 36, 2024.
- [10] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. Protgnn: Towards self-explaining graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9127–9135, 2022.
- [11] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 430–438, 2020.
- [12] Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and S Yu Philip. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. *IEEE Transactions on Big Data*, 9(2):415–436, 2022.
- [13] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. Heterogeneous network representation learning: A unified framework with survey and benchmark. *IEEE Transactions on Knowledge and Data Engineering*, 34(10):4854–4873, 2020.
- [14] James Fox and Sivasankaran Rajamanickam. How robust are graph neural networks to structural noise? *arXiv preprint arXiv:1912.10206*, 2019.
- [15] Enyan Dai, Wei Jin, Hui Liu, and Suhang Wang. Towards robust graph neural networks for noisy graphs with sparse labels. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 181–191, 2022.
- [16] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

- [17] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2847–2856, 2018.
- [18] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1399–1407, 2019.
- [19] Jiaqi Ma, Shuangrui Ding, and Qiaozhu Mei. Towards more practical adversarial attacks on graph neural networks. *Advances in neural information processing systems*, 33:4756–4766, 2020.
- [20] Mengmei Zhang, Xiao Wang, Meiqi Zhu, Chuan Shi, Zhiqiang Zhang, and Jun Zhou. Robust heterogeneous graph neural networks against adversarial attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4363–4370, 2022.
- [21] Jinquan Hang, Zhiqing Hong, Xinyue Feng, Guang Wang, Guang Yang, Feng Li, Xining Song, and Desheng Zhang. Paths2pair: Meta-path based link prediction in billion-scale commercial heterogeneous graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5082–5092, 2024.
- [22] Xin-Cheng Wen, Cuiyun Gao, Jiaxin Ye, Yichen Li, Zhihong Tian, Yan Jia, and Xuan Wang. Meta-path based attentional graph learning model for vulnerability detection. *IEEE Transactions on Software Engineering*, 2023.
- [23] Pengyang Yu, Chaofan Fu, Yanwei Yu, Chao Huang, Zhongying Zhao, and Junyu Dong. Multiplex heterogeneous graph convolutional network. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2377–2387, 2022.
- [24] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 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*, pages 1150–1160, 2021.
- [25] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.
- [26] Haoyi Fan, Fengbin Zhang, Yuxuan Wei, Zuoyong Li, Changqing Zou, Yue Gao, and Qionghai Dai. Heterogeneous hypergraph variational autoencoder for link prediction. *IEEE transactions on pattern analysis and machine intelligence*, 44(8):4125–4138, 2021.
- [27] Daniel Im Im, Sungjin Ahn, Roland Memisevic, and Yoshua Bengio. Denoising criterion for variational auto-encoding framework. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.
- [28] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.
- [29] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- [30] Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting graph neural networks for nlp with differentiable edge masking. In *International Conference on Learning Representations*, 2020.
- [31] Jiaxing Zhang, Dongsheng Luo, and Hua Wei. Mixupexplainer: Generalizing explanations for graph neural networks with data augmentation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3286–3296, 2023.
- [32] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

- 412 [33] Minh Vu and My T Thai. Pgm-explainer: Probabilistic graphical model explanations for graph
413 neural networks. *Advances in neural information processing systems*, 33:12225–12235, 2020.
- 414 [34] Shengyao Lu, Keith G Mills, Jiao He, Bang Liu, and Di Niu. Goat: Explaining graph neural
415 networks via graph output attribution. In *The Twelfth International Conference on Learning*
416 *Representations*, 2024.
- 417 [35] Tong Li, Jiale Deng, Yanyan Shen, Luyu Qiu, Huang Yongxiang, and Caleb Chen Cao. Towards
418 fine-grained explainability for heterogeneous graph neural network. In *Proceedings of the AAAI*
419 *Conference on Artificial Intelligence*, volume 37, pages 8640–8647, 2023.
- 420 [36] Wei Zhang, Xiaofan Li, and Wolfgang Nejdl. Adversarial mask explainer for graph neural
421 networks. In *Proceedings of the ACM on Web Conference 2024*, pages 861–869, 2024.
- 422 [37] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Het-
423 erogeneous graph attention network. In *The world wide web conference*, pages 2022–2032,
424 2019.
- 425 [38] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. Magnn: Metapath aggregated graph
426 neural network for heterogeneous graph embedding. In *Proceedings of the web conference*
427 *2020*, pages 2331–2341, 2020.
- 428 [39] Xiaocheng Yang, Mingyu Yan, Shirui Pan, Xiaochun Ye, and Dongrui Fan. Simple and efficient
429 heterogeneous graph neural network. In *Proceedings of the AAAI conference on artificial*
430 *intelligence*, volume 37, pages 10816–10824, 2023.
- 431 [40] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and
432 Max Welling. Modeling relational data with graph convolutional networks. In *The semantic*
433 *web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018,*
434 *proceedings 15*, pages 593–607. Springer, 2018.
- 435 [41] Lingfan Yu, Jiajun Shen, Jinyang Li, and Adam Lerer. Scalable graph neural networks for
436 heterogeneous graphs. *arXiv preprint arXiv:2011.09679*, 2020.
- 437 [42] Zeyu Sun, Wenjie Zhang, Lili Mou, Qihao Zhu, Yingfei Xiong, and Lu Zhang. Generalized
438 equivariance and preferential labeling for gnn node classification. In *Proceedings of the AAAI*
439 *Conference on Artificial Intelligence*, volume 36, pages 8395–8403, 2022.
- 440 [43] Sucheta Dawn. Graph neural networks for homogeneous and heterogeneous graphs: Algorithms
441 and applications. 2025.
- 442 [44] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a
443 collaboratively created graph database for structuring human knowledge. In *Proceedings of the*
444 *2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008.
- 445 [45] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint*
446 *arXiv:1412.6980*, 2014.

447 A Related Work

448 **GNN Explainability.** Recently, various approaches have been proposed to explain the predictions
 449 of GNN, these approaches can be categorized into post-hoc and built-in method. Common post-hoc
 450 methods include perturbation-based [6, 30] and surrogate model-based [7, 8] approaches. Mixu-
 451 pExplainer [31] extends the existing GIB framework by introducing label-independent subgraphs
 452 during the sampling of explanation subgraphs, thereby obtaining explanations while mitigating the
 453 distribution shift phenomenon. GNNExplainer [32] learns masks for features and edges by optimizing
 454 the masks to obtain the optimal explanation. PGExplainer [16] employs a parametric neural network
 455 approach to learn the importance of each edge and ultimately selects edges with high importance
 456 scores to construct the explanatory subgraph. PGM-Explainer [33] adopts a Bayesian network formu-
 457 lation, naturally expressing the dependencies between nodes in the form of conditional probabilities.
 458 Common built-in methods include prototype learning-based [9, 10] and graph generation-based [11]
 459 approaches. PGIB [9] integrates prototypes into the Graph Information Bottleneck framework, allow-
 460 ing it to learn prototypes based on key subgraphs in the input graph, thereby providing a more accurate
 461 explanation of the prediction process. GOAt [34] generates explanatory subgraphs by decomposing
 462 the model’s output into a series of scalar products involving node and edge features, and calculating
 463 the contribution of each feature to these scalar products, thereby highlighting the edges that are
 464 important for the prediction outcome. xPath [35] provides fine-grained explanations by identifying
 465 cause nodes and their influence paths through a novel graph rewiring algorithm, thereby offering
 466 detailed insights into how specific nodes affect model predictions. AMExplainer [36] leverages ad-
 467 versarial networks to optimize for both sparsity and prediction accuracy in explanations, significantly
 468 enhancing the clarity and efficiency of model interpretability.

469 **Heterogeneous Graph Neural Networks** Heterogeneous Graph Neural Networks can be catego-
 470 rized into meta-path-based methods and neighborhood aggregation-based methods. Meta-path-based
 471 methods typically decompose heterogeneous graphs into multiple homogeneous subgraphs using
 472 predefined meta-paths, thereby capturing specific types of heterogeneous semantics. Message passing
 473 is then performed within each subgraph, and the messages are subsequently aggregated. Common
 474 methods in this category include HAN [37], MAGNN [38], and SeHGNN [39]. On the other hand,
 475 neighborhood aggregation-based methods usually aggregate information directly from neighbors and
 476 apply specific aggregation strategies based on node types. Examples of methods in this category
 477 include RGCN [40], NARS [41], and Simple-HGN [24].

478 B Proof of Theorem 3.1

479 Let node v_i ’s original degree under edge type r be denoted as d_i^r . The deletion rate for edge type r is
 480 η_r^+ , and the false addition rate is η_r^- . The average degree for edge type r is defined as:

$$\bar{d}^r = \frac{1}{|\mathcal{V}_r|} \sum_{i \in \mathcal{V}_r} d_i^r \quad (20)$$

481 where $\mathcal{V}_r$ represents the set of nodes involved in edge type r . For node v_i , the expected number of
 482 edges deleted under edge type r is:

$$\mathbb{E}[\Delta d_{i,+}^r] = \sum_{j \in \mathcal{N}_r(i)} \eta_r^+ \cdot \frac{d_i^r + d_j^r}{2\bar{d}^r} \quad (21)$$

483 where $\mathcal{N}_r(i)$ represents the set of neighbors of node v_i under edge type r . Assuming the degree
 484 distribution of neighbors is uniform, it can be approximated as:

$$\mathbb{E}[\Delta d_{i,+}^r] \approx d_i^r \cdot \eta_r^+ \cdot \frac{d_i^r + \bar{d}^r}{2\bar{d}^r} \quad (22)$$

485 Further simplification gives:

$$\mathbb{E}[\Delta d_{i,+}^r] \approx \eta_r^+ \cdot \frac{d_i^{r^2}}{2\bar{d}^r} \quad (23)$$

486 For node v_i , the expected number of false added edges under edge type r is:

$$\mathbb{E}[\Delta d_{i,-}^r] = \sum_{j \notin \mathcal{N}_r(i)} \eta_r^- \cdot \frac{1}{d_i^r + d_j^r} \quad (24)$$

Assuming the degree distribution of non-neighbor nodes is uniform, it can be approximated as:

$$\mathbb{E}[\Delta d_{i,-}^r] \approx (|\mathcal{V}_r| - d_i^r) \cdot \eta_r^- \cdot \frac{1}{d_i^r + \bar{d}^r} \quad (25)$$

For node v_i , the total degree change across all edge types is:

$$\mathbb{E}[\Delta d_i] = \sum_{r \in R} (\mathbb{E}[\Delta d_{i,-}^r] - \mathbb{E}[\Delta d_{i,+}^r]) \quad (26)$$

Substituting the above results, we get:

$$\mathbb{E}[\Delta d_i] = \sum_{r \in R} \left(\eta_r^- \cdot \frac{(|\mathcal{V}_r| - d_i^r)}{d_i^r + \bar{d}^r} - \eta_r^+ \cdot \frac{d_i^{r^2}}{2\bar{d}^r} \right) \quad (27)$$

The expected degree of node v_i after perturbation is:

$$\mathbb{E}[d'_i] = d_i + \mathbb{E}[\Delta d_i] \quad (28)$$

Substituting the expression for $\mathbb{E}[\Delta d_i]$, we get:

$$\mathbb{E}[d'_i] = d_i + \sum_{r \in R} \left(\eta_r^- \cdot \frac{(|\mathcal{V}_r| - d_i^r)}{d_i^r + \bar{d}^r} - \eta_r^+ \cdot \frac{d_i^{r^2}}{2\bar{d}^r} \right) \quad (29)$$

To maintain the degree distribution characteristics of the graph, we introduce a degree-balancing constraint, which states that the expected degree after perturbation should be as close as possible to the original degree. Specifically, we require:

$$|\mathbb{E}[d'_i] - d_i| \leq \epsilon \quad (30)$$

where ϵ is a small constant representing the allowed degree deviation. By adjusting the deletion rate η_r^+ and false addition rate η_r^- , we can satisfy this constraint.

The above derivation shows that the degree-balancing constraint can effectively control the impact of noise injection on the graph structure. For high-degree nodes, the deletion rate η_r^+ has a larger weight $\frac{d_{ir}}{2\bar{d}^r}$, thereby reducing the number of edges to be deleted. For low-degree nodes, the false addition rate η_r^- has a larger weight $\frac{1}{d_{ir} + \bar{d}^r}$, thereby increasing the number of edges to be added. This design ensures that the perturbed graph structure maintains degree distribution characteristics similar to the original graph, thereby improving the reasonableness of noise injection.

Table 3: Statistics of the graphs before and after perturbation.

Dataset	Budget	$\Delta \bar{d}$	JS Divergence
DBLP	10%	0.0831	0.0875
	20%	0.8337	0.0905
	30%	1.7506	0.1114
	40%	2.6675	0.1407
ACM	10%	4.0125	0.2156
	20%	9.0189	0.2126
	30%	14.0254	0.2131
	40%	19.0318	0.2188
Freebase	10%	0.6556	0.2029
	20%	0.3112	0.1516
	30%	0.0332	0.1167
	40%	0.3776	0.0897

C Effectiveness of Controllable Structural Perturbation and Theorem 3.1

We propose controllable structural perturbation on heterogeneous graphs to simulate real-world noise. Our focus is on statistical perturbation effects rather than individual node changes. The

Table 4: The performance of RoHeX under different distribution perturbations.

Dataset	Budget	10%		20%		30%		40%	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
DBLP	Uniform	0.7940	1.3462	0.8877	1.3708	0.9391	1.4185	0.9486	1.3891
	Real	0.8359	1.2416	0.8743	1.2750	0.8827	1.2792	0.9014	1.2889
	Difference	5.01%	-8.42%	-1.53%	-7.51%	-6.39%	-10.89%	-5.24%	-7.77%
ACM	Uniform	0.2225	0.5648	0.2544	0.5986	0.3362	0.7516	0.3743	0.7333
	Real	0.2129	0.5662	0.2483	0.6177	0.3140	0.6669	0.3163	0.6909
	Difference	-4.51%	0.25%	-2.46%	3.09%	-7.07%	-12.70%	-18.34%	-6.14%
Freebase	Uniform	0.4331	0.7906	0.4564	0.9125	0.5186	0.9758	0.5247	0.9818
	Real	0.3885	0.8251	0.4441	0.8854	0.4694	0.9035	0.4880	0.9217
	Difference	-11.47%	4.18%	-2.78%	-3.06%	-10.48%	-8.01%	-7.51%	-6.53%

uniform assumption was introduced to simplify the derivation, a practice widely adopted in prior works [42, 43]. Assuming uniform distribution effectively captures perturbation impacts. To further validate this, we analyzed the degree distributions of the DBLP, ACM, and Freebase datasets before and after perturbation, as shown in Table 3. The results indicate that our method achieves significant noise effect while only slightly altering the degree distribution (with JS divergence < 0.22), as demonstrated in Table 2 of the main text.

Additionally, we conducted experiments comparing the performance of the explainer under the uniform assumption and the actual degree distribution. The results, presented in Table 4, confirm that the uniform assumption has minimal impact on model performance compared to using the real distribution.

D Proof of Theorem 3.2

In Graph Neural Network, a node representation is typically updated by aggregating information from its neighboring nodes. This process can be described as a message passing mechanism, where each node receives messages from neighboring nodes and updates its representation based on these messages. To avoid cases where the influence is overly amplified during the aggregation process, the messages from neighboring nodes are typically normalized. A common normalization approach is to multiply each neighbor message by the inverse of its degree. Assuming that each node influence on neighbors is equal, a higher-degree node will distribute its influence evenly among all neighbors. Therefore, the influence received by each neighbor should be proportional to the inverse of the node degree. In contrast, in random walk models, the transition probability between nodes is inversely proportional to the node degree. That is, the probability of a node reaching a particular neighbor is the inverse of its degree.

Given a heterogeneous graph $\mathcal{G}$, let v_i be a node with degree d_{v_i} . A noisy edge e_{ij} is added to the graph, where v_j is a new neighbor with degree d_{v_j} and k specific-type neighbors that match a given meta-path ϕ .

For meta-path-based methods:

(a) Before adding the noisy edge, the influence of v_i is assumed to be a combination of the influences from its d_{v_i} existing neighbors $v_1, v_2, \dots, v_{d_{v_i}}$ in $\mathcal{G}$. The influence of each neighbor v_n on v_i can be represented as:

$$I_{\text{ori1}} = \sum_{n=1}^{d_{v_i}} \frac{1}{d_{v_i}} \quad (31)$$

(b) After adding the noisy edge (v_i, v_j) , v_i is directly connected to v_j , and the influence of v_j will propagate to its k neighbors. The influence on each neighbor of v_j changes in the following manner v_i :

$$I_{\text{new1}} = \sum_{i=1}^{d_{v_i}} \frac{1}{d_{v_i} + 1} + \frac{k}{d_{v_i} + 1} = \frac{d_{v_i}}{d_{v_i} + 1} + \frac{k}{d_{v_i} + 1} = \frac{d_{v_i} + k}{d_{v_i} + 1} \quad (32)$$

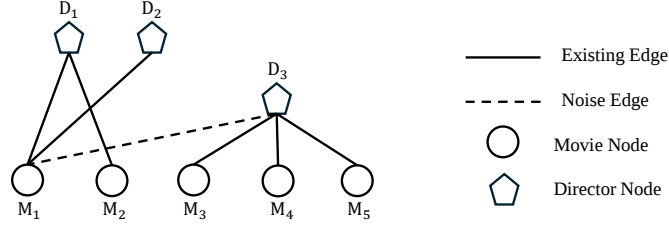


Figure 4: The illustrative example for noise against HGNNs on Movie-Director graph.

For neighborhood aggregation-based methods:

(a) Before adding the noisy edge, the influence on the direct neighbors of v_i is given by:

$$I_{\text{ori2}} = \sum_{n=1}^{d_{v_i}} \frac{1}{d_{v_i}} \quad (33)$$

(b) After adding the noisy edge (v_i, v_j) , the neighbors of v_i increase to $d_{v_i} + 1$, and the influence on each of its neighbors changes to:

$$I_{\text{new2}} = \sum_{n=1}^{d_{v_i}+1} \frac{1}{d_{v_i}+1} \quad (34)$$

The multiplicative relationship ξ of the influence propagation between the meta-path-based method and the neighborhood aggregation method is:

$$\xi = \frac{\frac{I_{\text{new1}}}{I_{\text{ori1}}}}{\frac{I_{\text{new2}}}{I_{\text{ori2}}}} = \frac{\frac{d_{v_i}+k}{d_{v_i}+1}}{1} = \frac{d_{v_i}+k}{d_{v_i}+1} \quad (35)$$

Consequently, when $k > d_{v_i}$, the multiplicative factor ξ is significantly greater than 1. This indicates that in general heterogeneous graphs, meta-path-based approaches are far more susceptible to the influence of noisy edges compared to neighborhood aggregation-based approaches. This substantiates that meta-path-based methods can significantly amplify the effect of noisy edges to a greater extent than neighborhood aggregation methods.

We also provide the illustrative example for noise against HGNNs on Movie-Director graph in Figure 4. The meta-path used is M-D-M. In neighborhood aggregation-based HGNNs, the noise-introduced nodes are not directly considered as 1-hop neighbors of M_1 . Consequently, under the influence of noise, the 2-hop neighbors M_3, M_4, M_5 can only affect M_1 through the 1-hop neighbor D_3 . However, in meta-path-based HGNNs, all neighbors under the M-D-M meta-path are aggregated with equal weight $\frac{1}{5}$, thereby enlarging the effect of the noisy edge $\langle M_1, D_3 \rangle$ to $\frac{3}{5}$ (the total weight of the noisy neighbors M_3, M_4, M_5).

E Detailed Derivation

First, we give the detailed derivation of Eq. 7. We introduce the Kullback-Leibler (KL) divergence. The KL divergence is a measure used to quantify the difference between two probability distributions. Let us consider two continuous random variables with probability distributions P and Q , and their corresponding probability density functions denoted as $p(x)$ and $q(x)$, respectively. If we aim to approximate $p(x)$ using $q(x)$, the KL divergence can be expressed as:

$$\text{KL}(P||Q) = \int p(x) \log \frac{p(x)}{q(x)} dx. \quad (36)$$

562 Because the logarithmic function is convex, the value of KL divergence is nonnegative. Then, Eq. 7
 563 can be written as:

$$\begin{aligned}
 \mathcal{L}(\Psi, \theta; \mathcal{G}) &= \mathbb{E}_{q_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log \frac{p_{\theta}(\mathbf{Z}, \mathcal{G})}{q_{\Psi}(\mathbf{Z}|\mathcal{G})} \right] \\
 &= \mathbb{E}_{q_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log p_{\theta}(\mathbf{Z}|\mathcal{G}) \cdot \frac{p(\mathbf{Z})}{q_{\Psi}(\mathbf{Z}|\mathcal{G})} \right] \\
 &= \mathbb{E}_{q_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log p_{\theta}(\mathbf{Z}|\mathcal{G})] - \text{KL}(q_{\Psi}(\mathbf{Z}|\mathcal{G}) || p(\mathbf{Z})).
 \end{aligned} \tag{37}$$

564 Second, the lower bound of denoising variational inference in Eq. 10 can be derived as:

$$\begin{aligned}
 \mathcal{L}_d &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log \frac{p_{\theta}(\mathbf{Z}, \mathcal{G})}{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} \right] \geq \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log \frac{p_{\theta}(\mathcal{G}, \mathbf{Z})}{q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})} \right] \\
 &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log p_{\theta}(\mathcal{G}|\mathbf{Z}) + \log p(\mathbf{Z}) - \log q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})] \\
 &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log p_{\theta}(\mathcal{G}|\mathbf{Z})] - \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log \frac{q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})}{p(\mathbf{Z})} \right] \\
 &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log p_{\theta}(\mathcal{G}|\mathbf{Z})] - \mathbb{E}_{q(\tilde{\mathcal{G}}|\mathcal{G})} \mathbb{E}_{q_{\Psi}(\mathbf{Z}|\mathcal{G})} \left[\log \frac{q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})}{p(\mathbf{Z})} \right] \\
 &= \mathbb{E}_{q'_{\Psi}(\mathbf{Z}|\mathcal{G})} [\log p_{\theta}(\mathcal{G}|\mathbf{Z})] - \mathbb{E}_{q(\tilde{\mathcal{G}}|\mathcal{G})} [\text{KL}(q_{\Psi}(\mathbf{Z}|\tilde{\mathcal{G}})) || p(\mathbf{Z})].
 \end{aligned} \tag{38}$$

565 Third, we derive an upper bound for GIB in Eq. 17. We decompose the mutual information:

$$\mathcal{I}(\hat{y}; \mathcal{G}_s) = \mathbb{E}_{p(\hat{y}, \mathcal{G}_s)} \left[\log \frac{p(\hat{y}, \mathcal{G}_s)}{p(\hat{y})p(\mathcal{G}_s)} \right], \mathcal{I}(\mathcal{G}; \mathcal{G}_s) = \mathbb{E}_{p(\mathcal{G}, \mathcal{G}_s)} \left[\log \frac{p(\mathcal{G}, \mathcal{G}_s)}{p(\mathcal{G})p(\mathcal{G}_s)} \right]. \tag{39}$$

566 The GIB objective can be written as:

$$\begin{aligned}
 & -\mathcal{I}(\hat{y}; \mathcal{G}_s) + \beta \mathcal{I}(\mathcal{G}; \mathcal{G}_s) \\
 &= -\mathbb{E}_{p(\hat{y}, \mathcal{G}_s)} \left[\log \frac{p(\hat{y}, \mathcal{G}_s)}{p(\hat{y})p(\mathcal{G}_s)} \right] + \beta \mathbb{E}_{p(\mathcal{G}, \mathcal{G}_s)} \left[\log \frac{p(\mathcal{G}, \mathcal{G}_s)}{p(\mathcal{G})p(\mathcal{G}_s)} \right] \\
 &= -\mathbb{E}_{p(\hat{y}, \mathcal{G}_s)} \left[\log \frac{p(\hat{y}|\mathcal{G}_s)p(\mathcal{G}_s)}{p(\hat{y})p(\mathcal{G}_s)} \right] + \beta \mathbb{E}_{p(\mathcal{G}, \mathcal{G}_s)} \left[\log \frac{p(\mathcal{G}_s|\mathcal{G})p(\mathcal{G})}{p(\mathcal{G})p(\mathcal{G}_s)} \right] \\
 &= -\mathbb{E}_{p(\hat{y}, \mathcal{G}_s)} \left[\log \frac{p(\hat{y}|\mathcal{G}_s)}{p(\hat{y})} \right] + \beta \mathbb{E}_{p(\mathcal{G}, \mathcal{G}_s)} \left[\log \frac{p(\mathcal{G}_s|\mathcal{G})}{p(\mathcal{G}_s)} \right] \\
 &= -\mathbb{E}_{p(\mathcal{G}_s)} \mathbb{E}_{p(\hat{y}|\mathcal{G}_s)} \left[\log \frac{p(\hat{y}|\mathcal{G}_s)}{p(\hat{y})} \right] + \beta \mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p(\mathcal{G}_s|\mathcal{G})}{p(\mathcal{G}_s)} \right].
 \end{aligned} \tag{40}$$

567 Using Jensen's inequality and assuming that $p_f(\hat{y}|\mathcal{G}_s)$ is an approximation of $p(\hat{y}|\mathcal{G}_s)$, we can get:

$$\begin{aligned}
 -\mathbb{E}_{p(\mathcal{G}_s)} \mathbb{E}_{p(\hat{y}|\mathcal{G}_s)} \left[\log \frac{p(\hat{y}|\mathcal{G}_s)}{p(\hat{y})} \right] &\leq -\mathbb{E}_{p(\mathcal{G}_s)} \mathbb{E}_{p(\hat{y}|\mathcal{G}_s)} [\log p_f(\hat{y}|\mathcal{G}_s)] - \mathbb{E}_{p(\hat{y})} [\log p(\hat{y})] \\
 &= -\mathbb{E}_{p(\mathcal{G}_s, \hat{y})} [\log p_f(\hat{y}|\mathcal{G}_s)] + \mathcal{H}(\hat{y}).
 \end{aligned} \tag{41}$$

568 We introduce explain models:

$$\begin{aligned}
 & \beta \mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p(\mathcal{G}_s|\mathcal{G})}{p(\mathcal{G}_s)} \right] \\
 &= \mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p_{\alpha}(\mathcal{G}_s|\mathcal{G})}{p(\mathcal{G}_s)} \cdot \frac{p(\mathcal{G}_s|\mathcal{G})}{p_{\alpha}(\mathcal{G}_s|\mathcal{G})} \right] \\
 &= \mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p_{\alpha}(\mathcal{G}_s|\mathcal{G})}{p(\mathcal{G}_s)} \right] + \mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p(\mathcal{G}_s|\mathcal{G})}{p_{\alpha}(\mathcal{G}_s|\mathcal{G})} \right].
 \end{aligned} \tag{42}$$

569 The second term is the KL divergence:

$$\mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p(\mathcal{G}_s|\mathcal{G})}{p_{\alpha}(\mathcal{G}_s|\mathcal{G})} \right] = \mathbb{E}_{p(\mathcal{G})} [\text{KL}(p(\mathcal{G}_s|\mathcal{G}) || p_{\alpha}(\mathcal{G}_s|\mathcal{G}))] \geq 0. \tag{43}$$

Therefore,

$$I(\mathcal{G}; \mathcal{G}_s) \leq \mathbb{E}_{p(\mathcal{G})} \mathbb{E}_{p_\alpha(\mathcal{G}_s|\mathcal{G})} \left[\log \frac{p_\alpha(\mathcal{G}_s|\mathcal{G})}{q(\mathcal{G}_s)} \right] = \mathbb{E}_{p(\mathcal{G})} [\text{KL}(p_\alpha(\mathcal{G}_s|\mathcal{G}) \| q(\mathcal{G}_s))]. \quad (44)$$

Combined with our previous derivation of the first term, we can get:

$$-I(\hat{y}; \mathcal{G}_s) + \beta I(\mathcal{G}; \mathcal{G}_s) \leq -\mathbb{E}_{p(\mathcal{G}_s, \hat{y})} [\log p_f(\hat{y}|\mathcal{G}_s)] + H(\hat{y}) + \beta \mathbb{E}_{p(\mathcal{G})} [\text{KL}(p_\alpha(\mathcal{G}_s|\mathcal{G}) \| q(\mathcal{G}_s))]. \quad (45)$$

F Experiment Supplement

Table 5: Statistics of Datasets.

Dataset	DBLP	ACM	Freebase
Nodes	26,128	10,942	43,854
Edges	239,566	547,872	151,034
Node Types	4	4	4
Edge Types	6	8	6
Classes	4	3	3

F.1 Datasets

We conduct experiments on three real-world datasets. According to the Heterogeneous Graph Benchmark [24] settings, we randomly split the nodes with proportions of 24%, 6%, and 70% for training, validation, and testing, respectively. The statistics of the three datasets are shown in Table 5.

- **DBLP**¹ is a computer science bibliography network that contains four types of nodes: Paper (P), Author (A), Term (T), and Venue (V). The authors in this network are from four research areas (*Database*, *Data Mining*, *Artificial Intelligence*, and *Information Retrieval*).
- **ACM**² is a citation network that contains four types of nodes: Paper (P), Author (A), Term (T), and Subject (S). The papers in this network are divided into three classes (*Database*, *Wireless Communication*, and *Data Mining*).
- **Freebase** [44] is a knowledge graph that contains four types of nodes: Movie (M), Actor (A), Director (D) and Writer (W).

F.2 Baselines

Next, we provide details on the baselines used in our experiments.

- **PGExplainer** [16] is a parameterized explainer that learns a mask for each edge to obtain edge importance scores.
- **GNNExplainer** [32] maximizes the mutual information between the model’s prediction on the original input and the masked input by masking features and edges.
- **PGM-Explainer** [33] employs a Bayesian network-based approach, treating vertices in the input graph as random variables to fit the GNN model’s predicted label.
- **V-InfoR** [5] utilizes a parametric method, learning edge masks on the latent representations to identify important edges.
- **AMExplainer** [36] optimizes GNN explanations by leveraging adversarial networks to achieve both sparsity and prediction accuracy, ensuring compact and faithful explanation sets.
- **Hete-PGE** is an extension of PGExplainer, where we replace the initial concatenation with a relation-based attention learning module to enable learning of heterogeneous semantics.
- **xPath** provides fine-grained explanations by identifying causal nodes and their influence paths through a novel graph rewiring algorithm, offering detailed insights into the model’s decision-making process.

¹<https://dblp.uni-trier.de>

²<http://dl.acm.org/>

603 F.3 Base Heterogeneous Graph Neural Network

Table 6: Node classification result using our heterogeneous Graph Neural Network.

Dataset	DBLP	ACM	Freebase
Micro-F1	92.64±0.14	92.32±0.12	68.99±0.20
Macro-F1	92.16±0.19	92.40±0.11	63.57±0.36

604 In the experiment, we use a basic heterogeneous Graph Neural Network, which encodes the input
 605 graph through 2 layers of GCN, and then used a layer of attention learning module to learn different
 606 heterogeneous relations. For a heterogeneous graph, the feature spaces of different types of nodes are
 607 usually different. We use a mapping function to map the features of different types into a common
 608 feature space, as shown below:

$$\mathbf{z}_v = \mathbf{W}_m \mathbf{x}_v^A + \mathbf{b}_m, \quad (46)$$

609 where $A \in \mathcal{A}$ is the node type of node v , $\mathbf{W}_m$ is a learnable weight, and $\mathbf{b}_m$ is the bias. Then, in the
 610 shared space, we use GCN to obtain the node embeddings:

$$\mathbf{Z}^{(l)} = \text{GCN}(\mathbf{Z}^{(l-1)}, \mathbf{A}), \mathbf{Z}^{(0)} = \mathbf{Z}_v. \quad (47)$$

611 To learn the heterogeneous semantics of the heterogeneous graph, we introduce a type vector γ_v and
 612 learn relation information through an attention module:

$$\begin{aligned} \gamma_i^q &= \mathbf{W}_r^q \gamma_i, \gamma_j^k = \mathbf{W}_r^k \gamma_j, \\ \text{score}_{ij} &= \gamma_i^q \gamma_j^k, \end{aligned} \quad (48)$$

613 where $\mathbf{W}_r^q$ and $\mathbf{W}_r^k$ are learnable weights. The attention of the nodes can be computed as follows:

$$\begin{aligned} q_i &= \mathbf{W}_q^z \mathbf{z}_i, k_j = \mathbf{W}_k^z \mathbf{z}_j, \\ \hat{\alpha}_{ij} &= \frac{\exp(\text{LeakyReLU}(a^T [q_i \parallel k_j]))}{\sum_{j' \in \mathcal{N}_i} \exp(\text{LeakyReLU}(a^T [q_i \parallel k_{j'}]))}. \end{aligned} \quad (49)$$

614 where $\mathbf{W}_q^z$ and $\mathbf{W}_k^z$ are learnable weights. The final prediction can be expressed as:

$$\begin{aligned} \text{score}_{ij} &= \hat{\alpha}_{ij} + \beta \text{score}_{ij}^{\gamma}, \\ \mathbf{Z}_H^{(l)} &= \text{LayerNorm}(\mathbf{Z}_H^{(l-1)} + \text{score}_{ij} \cdot \mathbf{Z}_H^{(l-1)}), \\ \hat{Y} &= P_f(\mathbf{Z}_H^{(l)}; \theta_p). \end{aligned} \quad (50)$$

615 where θ_p is the parameter of the predictor. The basic prediction results are shown in Table 6.

616 The experiments are conducted on an L20 GPU with 48GB of memory. Our CPU is an Intel(R)
 617 Xeon(R) Platinum 8457C. We utilized PyTorch version 1.13.1 and DGL version 1.1.1.

618 F.4 Impact of Noise

619 We measure the impact of noise on the graph distribution through the variation of the KL divergence.
 620 The result is in Table 7, and it can be observed that noise significantly disrupts the graph distribution.

Table 7: The impact of noise with different budgets on graph distribution.

Noise	10%	20%	30%	40%	50%	60%	70%	80%	90%
KL	0.11108	0.31823	0.54854	0.80690	1.09782	1.43823	2.01607	2.15205	2.40943

621 F.5 Parameter Setting

622 For the base heterogeneous Graph Neural Network, we use Adam [45] as the optimizer, LeakyReLU
 623 with a negative slope $s = 0.2$ as the activation function, a learning rate of $1e-4$, and a dropout rate of 0
 624 for Freebase and 0.5 for other datasets. The hidden dimension is set to 256. Our training is performed
 625 for 100 epochs.

626 **G Limitations**

627 **Node Feature Noise.** This paper focuses on structural noise. Both denoising variational inference
628 and heterogeneous explanation generator are proposed to mitigate the impact of structural noise.
629 Therefore, RoHeX cannot be directly used to explain heterogeneous graphs containing feature noise.
630 We will extend RoHeX to achieve robustness at the node feature level in future work.

631 **Dynamic Graph.** RoHeX can be used to explain different levels of tasks on heterogeneous graphs
632 and can also be applied to homogeneous graphs. Due to the lack of dynamic information extraction
633 module, RoHeX cannot be applied to dynamic graphs.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The abstract and introduction accurately reflect the paper's contributions and scope.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: See Appendix G.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: See Appendix B, D, and E.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: See Section 4.1 and Appendix F.5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The code will be open-sourced after the paper is accepted.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: See Appendix F.1 and F.5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: See Section 4.2.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: See Appendix F.3.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We comply.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: See Section 1.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: See Appendix F.1 and F.2.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: We does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

947 Justification: This paper does not involve LLMs as any important, original, or non-standard
948 components.
949 Guidelines:
950 • The answer NA means that the core method development in this research does not
951 involve LLMs as any important, original, or non-standard components.
952 • Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>)
953 for what should or should not be described.