Dilution of Unreliable Information: Learning in Graph with Noisy Structures and Absent Attributes

Xinxin Li[†‡], Yang Liu[*†‡], Siyong Xu[¶], Weigao Wen[¶], Qing He[†‡], Xiang Ao[*†‡§]
[†] *State Key Laboratory of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences.*
[‡] *University of Chinese Academy of Sciences, Beijing 100049, China.*
[§] *Institute of Intelligent Computing Technology, Suzhou, CAS.*
[¶] *Individual Researcher.*
{lixinxin23s, liuyang2023, heqing, aoxiang}@ict.ac.cn
{siyongxu666, weigaowen}@gmail.com

Abstract*—Graph Neural Networks (GNNs) are vulnerable to perturbations in both edges and attributes by fraudsters attempting to evade detection. A low-cost and effective perturbation strategy involves establishing connections with benign users and providing as little information as possible, leading to a graph with noisy structure and absent attributes. We formulate a novel problem as Learning in Graphs with Noisy structures and Absent node attributes (LGNA), for which no existing methods are specifically designed. To mitigate this gap, we propose a reliable graph learning framework called RENA, which implements a "Dilution of Unreliable Information" approach for the LGNA task. The core principle of RENA is to utilize more reliable information to decrease the proportion of unreliable information, thus diluting its impact. Specifically, only the observed node attributes and unconnected node pairs are considered reliable, while imputed attributes and connected node pairs are deemed unreliable. We first randomly sample a large number of unconnected node pairs and fewer connected pairs to create different structural views to supervise structure learning and dilute the impact of noisy edges. Next, we apply a graph autoencoder framework, assigning higher weights to the observed attributes and lower weights to the imputed attributes during the reconstruction process, thereby diluting the impact of imputation noise. Experiments show that our method outperforms state-of-the-art baselines on LGNA scenarios and conventional incomplete graph learning tasks.

***Index Terms*—Graph Neural Network, Noisy Structures, Absent Attributes, Reliable Graph Learning**

I. Introduction

With the rapid advancement of technology, fraudulent activities have become increasingly prevalent, impacting industries such as finance [1], [2], healthcare [3], and review management [4], resulting in significant economic losses [5]. In 2023, approximately 25.5% of the global population suffered financial losses due to fraud, with total economic damages projected to reach \$1.026 trillion [6]. Interactions involving fraudsters and benign users can be modeled as graph-like data, with users and their interactions modeled as nodes and edges. GNN-based fraud detection methods [7], [8] have achieved remarkable success in identifying fraudsters.

However, these GNN-based fraud detection models are particularly susceptible to strategies employed by fraudsters seeking to evade detection. Most well-established GNNs [9]–[11] hinge on the message-passing mechanism, where the graph structure and node attributes are essential for model effectiveness. Hence, as nodes in graphs, fraudsters can potentially undermine the performance of GNN-based fraud detection models by perturbing both edges and attributes. For edge perturbation, the most intuitive manner is to *establish connections with benign users*. By doing so, the information from these benign users could be propagated to the fraudsters along with the injected edges, making it more difficult for GNN-based models to distinguish them from benign users. Regarding node attribute perturbation, on the other side, a low-cost yet effective strategy is *keeping silent*. It means providing as little valuable information as possible to conceal fraudulent intentions by deriving a low-quality initial embedding [12], [13]. Besides, some benign users are reluctant to disclose personal information, such as age and gender, beyond the required information [14]. We observe from a real-world dataset [15] that almost 80.97% of fraudsters and 74.75% of benign users have incomplete profile information.

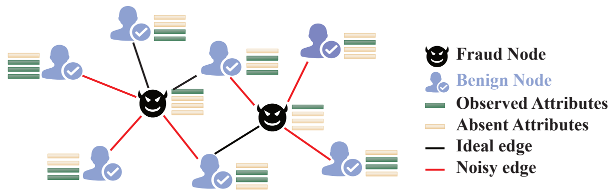


Fig. 1. Fraudsters often interact with benign users to conceal their activities and evade detection by regulatory authorities. Additionally, users are unwilling to provide optional information.

Consequently, in fraud scenarios, the graph data can be characterized as Graphs with Noisy structures and Absent node attributes (GNA). GNA represents a generalized form of graph data that encompasses noisy edges and absent attribute dimensions, as depicted in Figure 1. In such graphs, fraudulent nodes often form multiple noisy edges with benign ones while exhibiting partially blank attributes. The GNA learning task is referred to as Learning in Graphs with Noisy structures and

[*] Corresponding authors.

Absent attributes, abbreviated as **LGNA** in this paper.

To the best of our knowledge, there are no existing approaches specifically tailored for LGNA. Existing solutions related to LGNA include Graph Structure Learning (GSL) [16]–[19], graph node attribute imputation [20]–[23] and incomplete graph learning [14], [24], [25], all of which encounter critical challenges in addressing LGNA. Firstly, existing GSL methods [16], [17] heavily depend on observed structural information as supervision signals, leaving them susceptible to noisy edges in GNA. The learned structure can be suboptimal due to the misleading of noisy structures. Secondly, attribute imputation or incomplete graph learning methods utilize graph structure information to impute absent attributes, such as taking the average of the neighbor values. When the structure contains noisy edges, the imputed attributes would inevitably collect irrelevant information from its noisy neighbors. Therefore, existing solutions fail to solve the LGNA problem since they can be easily affected by unreliable information from either graph structure or node attributes.

In this paper, we aim to investigate how to accomplish LGNA more reliably. To this end, we first identify which aspects of the available information can be considered reliable and leveraged to address the problem effectively. Recall that the limitations of existing solutions primarily stem from their susceptibility to unreliable information, such as noisy edges and absent node attributes. In contrast, unconnected node pairs and observed node attributes are potential sources of reliable information. While some unconnected node pairs represent latent edges that should exist but were omitted due to data collection errors, the overall impact of these missing edges is minimal compared to the vast majority of truly unconnected node pairs, given the inherent sparsity of graph data, as validated in the Appendix A. Furthermore, we argue that observed node attributes are inherently reliable because they typically carry semantic meaning. This characteristic makes significant deviations from normal values easily detectable, regardless of whether the node belongs to a fraudster or a legitimate user. Consequently, the observed attribute values should be regarded as reliable.

Based on that analysis, we propose a **<u>RE</u>**liable graph learning framework for graphs with **<u>N</u>**oisy structures and **<u>A</u>**bsent node attributes, named **RENA**. The heart of RENA lies in diluting the negative effects caused by unreliable information and increasing the proportion of a more reliable component. Specifically, instead of relying on connected node pairs for graph structure learning [16], [21], RENA leverages reliable unconnected node pairs as supervision signals. To achieve this, we randomly sample a large number of unconnected node pairs along with a smaller set of connected node pairs, thereby reducing the influence of noisy edges and improving the quality of the learned graph structure. For node attributes, our goal is to minimize the impact of noise introduced by attribute imputation. To this end, we first impute the absent attributes and subsequently employ a graph autoencoder for attribute reconstruction. During training, we assign higher weights to observed than imputed attributes, ensuring that the learned representations are less affected by imputation noise and remain faithful to the original data.

Our contribution could be listed as follows:

- We introduce a general graph learning problem, denoted as LGNA, which arises in the context of graph-based fraud detection, where the graph structure is noisy and node attributes are partially absent.
- We propose a reliable graph learning framework termed RENA to address the challenges in LGNA. The central idea of RENA is to mitigate the effects of unreliable structural and imputation noise by leveraging the more reliable unconnected node pairs and observed attributes as supervision signals.
- Extensive experiments on three real-world datasets validate the effectiveness of RENA for the LGNA task.

II. Related Work

We categorize graph learning tasks into four scenarios based on the characteristics of graph data: learning in graphs with absent node attributes (LGA), learning in graphs with noisy structure (LGN), learning in graphs with incomplete structure and attributes (LGI), and learning in graphs with noisy structure and absent node attributes (LGNA), each corresponding to the graph data as illustrated in Figure 2. Methods such as attribute imputation, graph structure learning, and incomplete graph learning have been proposed to address these tasks.

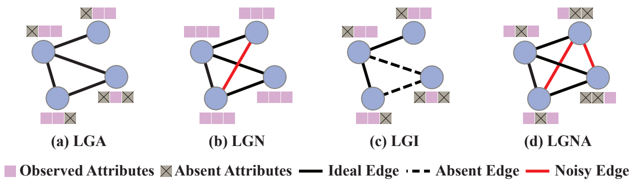


Fig. 2. Illustration of four different graph learning tasks.

A. Attribute Imputation for LGA

To address absent attributes, attribute completion aims to recover absent data based on the available information (observed attributes or graph structure). ITR [21] encodes structural information using one-hot vectors and uses this encoded structural information to impute the absent attributes. ASD-VAE [20] parameterizes the structure and attribute representations into a shared latent space. Then the shared space is decoupled into separate views, and absent values are imputed from this learned latent space. SVGA [23] applies a Gaussian Markov random field on the graph structure to capture the prior distribution of absent attributes, imposing strong regularization on latent variable distributions via structured variational inference. SAT [26] assumes that the graph structure and node attributes originate from the same latent space. It encodes them separately and reduces the discrepancy between the two using distribution matching techniques, thereby estimating the node attributes. PCFI [27] introduces channel-wise confidence to measure the certainty of the imputed attributes. Based on this confidence, it imputes attributes via channel-wise inter-node diffusion and node-wise inter-channel propagation.

B. Graph Structure Learning for LGN

A variety of methods have been proposed to learn graph structures that benefit downstream tasks. SUBLIME [18] uses a parameterized model to learn the graph structure based on complete attributes. The learned graph structure serves as the learning view, while the original structure acts as the anchor view. Then it is optimized by maximizing the agreement between these two views. WSGNN [16] uses the variational inference method to estimate absent information from observed data and collaboratively learns graph structures and node representations through a two-branch network. GraphGlow [28] models inputs and intermediate results as random variables to study their dependencies from a generative perspective. GEN [29] employs Bayesian inference to optimize graph structure with multi-order neighborhood integration.

C. Incomplete Graph learning for LGI

An incomplete graph refers to a graph where attributes and structure exhibit partially absent elements. Due to the randomness of the absent data, the attributes and structure of the graph often fail to fully match, leading to mutual interference during message passing in GNNs. T2GNN [24] addresses this challenge by utilizing a attributes teacher and a structure teacher to independently model attributes and graph structure, distilling the information into a student model. UGCL [25] employs a two-layer MLP for feature reconstruction and PPNP for structure reconstruction, optimized with a dual contrastive loss to align node representations from both pathways. D2PT [14] completes the missing attributes through long-range propagation, then generates a global graph structure based on the completed attributes, effectively passing information to stray nodes.

Despite significant progress in the areas of attribute imputation, structure learning, and incomplete graph learning, these approaches depend on utilizing reliable components in graph structure or attributes to purify unreliable components. However, when unreliable information exists in reliable components, the performance of these methods tends to be suboptimal. To bridge this gap, we propose RENA, a method specifically designed to tackle the LGNA task by leveraging reliable components to dilute unreliable ones.

III. Preliminaries

To formulate the LGNA task, we first define ideal graph data for semi-supervised node classification under ideal conditions.

Definition 1 (Ideal graph data): We consider an ideal graph as $\mathcal{G}^* = (\mathcal{V}, \mathcal{E}^*, \mathrm{A}^*, \mathrm{X}^*)$, where $\mathcal{V} = \{v_1, \cdots, v_n\}$ is the node set with size n, $\mathcal{E}^*$ is the edge set, $\mathrm{A}^* \in \{0,1\}^{n\times n}$ is the binary adjacency matrix (where $\mathrm{A}^*{}_{ij}$ = 1 means v_i and v_j are connected and vice versa) and $\mathrm{X}^* \in \mathbb{R}^{n\times d}$ is the attributes matrix (where $\mathrm{X}^*{}_i$ is d-dimensional attributes vector of node v_i). The node labels are represented by a label matrix $\mathrm{Y} \in \mathbb{R}^{n\times c}$, where c is the number of classes.

It is crucial to acknowledge that ideal graph data represents an optimal setting for graph learning. In real-world scenarios, the graph data used for model training usually contain both noisy and absent data. Specifically, the structure can contain noisy edges and some dimensions of attributes are absent, resulting in misleading and incomplete information for graph learning. Based on the above findings, the GNA data can be described as follows:

Definition 2 (GNA data): Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathrm{A}, \mathbf{X})$ denote a graph with noisy structure and absent attributes, where $\mathcal{E} = \mathcal{E}^* \cup \mathcal{E}^+$ and $\mathrm{A} = \mathrm{A}^* + \mathrm{A}^+$. Here, $\mathcal{E}^*$ is the ideal edge set and $\mathcal{E}^+$ denotes the noisy edge set, A^+ denote the adjacency matrix induced by $\mathcal{E}^+$. The absent attribute dimensions are represented by a mask matrix $\mathrm{M} \in \{0,1\}^{n\times d}$ with 0 denotes the absent dimensions. The absent node attributes are denoted by $\mathrm{X} = \mathrm{M} \odot \mathrm{X}^*$.

In the LGNA task, only the labels of a small fraction of nodes $\mathcal{V}_L \subset \mathcal{V}$ are available for model training, and the goal in the inference phase is to predict the labels of unlabeled nodes $\mathcal{V}_U \subset \mathcal{V}$, w.r.t $\mathcal{V}_U \cap \mathcal{V}_L = \emptyset$ and $\mathcal{V}_U \cup \mathcal{V}_L = \mathcal{V}$. We denote the training labels as $\mathrm{Y}_L \in \mathbb{R}^{n_L\times c}$, where $n_L = |\mathcal{V}_L|$ is the number of training nodes.

IV. Methodology

A. Overview

Most prior works [14], [18], [20] focus on either attribute imputation, structural noise, or incomplete graphs, but none have simultaneously tackled both absent attributes and structural noise. To bridge this gap, we propose a method called RENA, designed to dilute the impact of unreliable information. Figure 3 illustrates the overall workflow of RENA, which consists of two key components: Multi-View Structure Dilution (MSD) and Attributes Reconstruct Dilution (ARD). In MSD, we randomly sample a large number of unconnected node pairs and a smaller number of connected pairs to construct multiple structural views as supervision signals for structure learning, decreasing the proportion of connected node pairs, and thereby diluting the influence of noisy edges on structure learning. Next, in ARD, we introduce a graph autoencoder framework that assigns different weights to observed attributes and imputed attributes during attribute reconstruction, aiming to dilute the influence of imputation noise. Finally, we fine-tune the encoder with labels for downstream tasks.

B. Multi-View Structure Dilution

In the LGNA task, we model the graph structure using observed node attributes, rather than those after message passing [14]. The reason is that if these absent attributes aggregate information from irrelevant neighbors through the noisy structure, the observed attributes can therefore introduce noise, making it more difficult to model the graph structure correctly. Specifically, we estimate the edge weight between any node pair (u, v) as Eq. (1):

$$a_{uv} = \frac{1}{m}\sum_{i=1}^{m} \cos(\mathrm{w}_i \odot \mathrm{x}_u, \mathrm{w}_i \odot \mathrm{x}_v), \tag{1}$$

where $\mathrm{W} = [\mathrm{w}_1, \cdots, \mathrm{w}_m] \in \mathbb{R}^{d\times m}$ represent m learnable weight vectors, $\mathbf{x}_u$ denotes the observed attributes of node

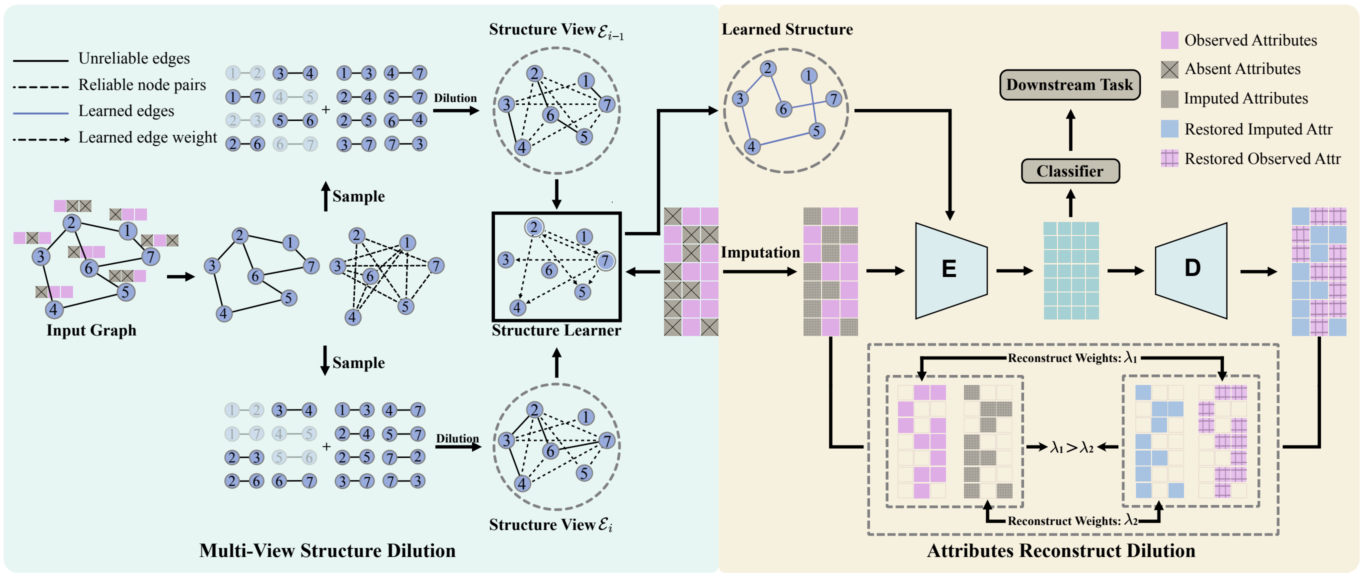


Fig. 3. Overview of the RENA framework, which is composed of two key components: Multi-View Structure Dilution (MSD) and Attributes Reconstruct Dilution (ARD). In MSD, we randomly sample connected and unconnected node pairs to construct multiple structural views, diluting the impact of noisy edges and guiding structure learning. In ARD, the decoder reconstructs attributes, assigning higher weights to observed attributes and lower weights to imputed ones to dilute the imputation noise. Finally, a classifier is introduced, and both the encoder and classifier are fine-tuned with labels for downstream tasks.

u, $\odot$ represents the Hadamard product. A common approach for supervising the learning of a_{uv} is using connected node pairs [30]. However, in noisy graph structures, connected node pairs may be noisy edges, and if we use connected node pairs, they would introduce bias into the learned graph structure. In contrast, unconnected node pairs are assumed to remain unconnected in noisy graph structures. Although some unconnected node pairs represent latent edges that should exist but were omitted due to data collection errors or other factors, the overall impact of these latent edges is minimal compared to the vast majority of truly unconnected node pairs, given the inherent sparsity of the graph data. For example, in the T-Finance dataset [31], the number of unconnected node pairs is approximately 729 times larger than that of connected node pairs. Even when considering all connected edges as latent edges, truly unconnected edges still far outnumber these latent edges. These assumptions are validated by experimental results as discussed in Appendix A. Therefore, we focus on leveraging a large set of reliable unconnected node pairs for graph structural learning.

We randomly sample p unconnected node pairs as negative samples to supervise the learning of $a_{u'v'}$, where u' and v' represent an unconnected node pair (u', v'), the loss for negative samples is defined as Eq. (2):

$$\mathcal{L}_{\text{BCE}}(a_{u'v'}, e_{u'v'}) = -\log(1 - \sigma(a_{u'v'})), \tag{2}$$

where the weight $e_{u'v'}$ between u' and v' is set to 0 because of the absence of a connection, and $\sigma(\cdot)$ is a non-linear activation. If we only sample unconnected node pairs, the model will run the risk of converging to a trivial solution by predicting all edge weights as 0. To mitigate this, we randomly sample q connected node pairs as positive samples to supervise the learning of $a_{u''v''}$, where u'' and v'' represent a connected node pair (u'', v''), the loss for positive samples is calculated as Eq. (3) :

$$\mathcal{L}_{\text{BCE}}(a_{u''v''}, e_{u''v''}) = -\log(\sigma(a_{u''v''})), \tag{3}$$

where $e_{u''v''} = 1$, which represents the weight between connected nodes u'' and v''. The sampled edges $\{e_{u'v'}\}$ and $\{e_{u''v''}\}$ form a structure view $\mathcal{S} = \{e_{u'v'}, e_{u''v''}\}$. The loss is optimized as Eq. (4) :

$$\mathcal{L}_{\mathcal{S}} = -\frac{1}{|\mathcal{S}|}\Big(\sum_{e_{uv}\in\mathcal{S}} e_{uv}\log(a_{uv}) + (1-e_{uv})\log(1-a_{uv})\Big). \tag{4}$$

Owing to the presence of noise in the structure, the positive samples sampled$\{e_{u''v''}\}$ may contain noisy edges. To further reduce the proportion of noisy edges in the structure view, we repeatedly sample positive samples T times from the connected node pairs to construct multiple structure views (MSV), which is defined as Eq. (5):

$$\begin{aligned} &\text{MSV} = \{\mathcal{S}_1, \ldots, \mathcal{S}_{T-1}, \mathcal{S}_T\}, \\ &\text{where} \quad \mathcal{S}_i = \{e_{u'v'}, e^i_{u''v''}\}, \quad i = 1, \ldots, T, \end{aligned} \tag{5}$$

where T is the number of structural views, $e^i_{u''v''}$denotes the positive samples drawn from the i-th random sampling, and $\mathcal{S}_i$ is the i-th structure view. Let $r = p/q$, where r is the dilution ratio of unconnected node pairs to connected ones. By increasing the value of r, we can decrease the proportion of unreliable node pairs in a structural view, thereby diluting the impact of noisy edges on structural learning. The structure learning loss is optimized by minimizing Eq. (6):

$$\mathcal{L}_{\text{MSV}} = \frac{1}{T}\sum_{i=1}^{T}\mathcal{L}_{\mathcal{S}_i}. \tag{6}$$

After structure learning, the learned similarity between node pairs is utilized to filter noisy edges. We define $\text{A}^s_{uv} = a_{uv}$ in Eq.(1) and A^s is the similarity-based adjacency matrix. The final optimized graph structure $\hat{\text{A}}$ is defined by (7), where the threshold is a filter parameter.

$$\hat{\text{A}}_{uv} = \begin{cases} 1 & \text{if } \text{A}^s_{uv} + \text{A}_{uv} > \text{threshold} \\ 0 & \text{otherwise} \end{cases}. \tag{7}$$

Algorithm 1 RENA: Dilution of Unreliable Information

```
    Input: $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \text{A}, \text{X})$: A graph with noisy structure and absent attributes, $q$: Number of connected node pairs, $r$: Dilution ratio, $T$: Number of views, $g_E$: Encoder, $g_D$: Decoder, $f$: Linear classifier, $n_{epoch}$: Number of training epochs, W: Learnable weight vector.
    Output: Predicted label $\hat{\text{Y}}$
 1: // Step 1: Multi-View Structure Dilution (MSD)
 2: Sample $p = r \times q$ unconnected node pairs randomly
 3: for $i = 1$ to $T$ do
 4:     Sample $q$ connected node pairs
 5:     Construct structure view $\mathcal{E}_i$
 6: end for
 7: for $e = 1$ to $n_{epoch}$ do
 8:     Compute $a_{uv}$ for any two nodes $u$ and $v$ using Eq. (1)
 9:     Train W by minimizing Eq. (6)
10: end for
11: Obtain $\hat{\text{A}}$ using Eq. (7)
12: // Step 2: Attributes Reconstruct Dilution (ARD)
13: Obtain $\overline{\text{X}}$ using Eq. (8)
14: for $e = 1$ to $n_{epoch}$ do
15:     Obtain $\hat{\text{X}}$ based on Eq.(9)
16:     Train $g_E$ and $g_D$ by minimizing Eq. (10)
17: end for
18: // Step 3: Fine-Tuning
19: for $e = 1$ to $n_{epoch}$ do
20:     Predict $\hat{\text{Y}}$ based on Eq. (11) and (12)
21:     Fine-tune $g_E$ and $f$ by minimizing Eq. (13)
22: end for
23: return $\hat{\text{Y}}$
```

C. Attributes Reconstruct Dilution

An intuitive approach to handling absent attributes is through imputation. However, both classical and recent imputation methods are prone to introducing unreliable information when the structure contains noisy edges. Classical imputation methods, such as zero or mean imputation [32], k-nearest neighbor [33], and SVD matrix completion [34], all of which can introduce irrelevant information [21]. Recent advancements utilize graph structural information [20]–[22] to impute absent attributes or maximize the mutual information between structure and the attributes [23]. However, when the graph structure contains noisy edges, the imputed attributes would inevitably collect irrelevant information from its noisy neighbors. This unreliable information propagates through the graph structure, further affecting the embedding of observed node attributes. Therefore, instead of designing an imputation method, our goal is to mitigate the impact of unreliable information introduced by imputation on observed attributes.

To this end, we apply a graph autoencoder [35] assigning a higher weight to observed attributes during the reconstruction process. This ensures that the restored node attributes primarily reflect reliable observed attributes, thereby mitigating the influence of imputation noise. Firstly, we impute the absent attributes using the k-Nearest Neighbors mean as Eq. (8) :

$$\overline{X}_{ij} = \begin{cases} \frac{1}{K}\sum_{u\in N_{i,j}} X_{uj} & \text{if } X_{ij} \text{ is absent} \\ X_{ij} & \text{otherwise} \end{cases}, \tag{8}$$

where $N_{i,j}$ represents top-K nearest nodes to node i, whose j-th dimension is not absent. The distance is computed using the Euclidean distance on the observed attributes. Then, a graph autoencoder is applied to reconstructed attributes:

$$\hat{\text{X}} = g_D(\text{Z}, \hat{\text{A}}), \quad \text{Z} = g_E(\overline{\text{X}}, \hat{\text{A}}), \tag{9}$$

where g_E and g_D are GNN-based encoder and decoder, respectively. Z is the node embedding in the latent space. $\overline{\text{X}}$ is the input of g_E, of which the absent parts are imputed as Eq.(8). $\hat{\text{A}}$ is the adjacency matrix as Eq.(7), and $\hat{\text{X}}$ represents the reconstructed attributes, i.e. the output of g_D. The loss for attributes reconstruct dilution is demonstrated as Eq. (10):

$$\mathcal{L}_{Rec} = \frac{1}{n_{train}}(\lambda_1 \left\|\overline{\text{X}}_{obs} - \hat{\text{X}}_{obs}\right\|^2 + \lambda_2 \left\|\overline{\text{X}}_{imp} - \hat{\text{X}}_{imp}\right\|^2), \tag{10}$$

where $\lambda_1 > \lambda_2$, the terms "obs" and "imp" represent the indices of the observed and imputed attributes, respectively.

The attribute reconstruction dilution method is self-supervised and lacks label information. To address this, we add a linear classifier after the encoder and fine-tune its parameters using label information. Let f represent the linear classifier and CE be the cross-entropy loss. The fine-tuning process is then formulated as Eq. (11) to Eq. (13), where $\text{Y}_L = \{y_i\}_{i=1}^{n_L}$ is the groundtruth labels for training nodes, and $\hat{\text{Y}} = \{\hat{y}_i\}_{i=1}^{n}$ is the predicted labels. The overall training algorithm is summarized in Algorithm 1.

$$\text{Z} = g_E(\overline{\text{X}}, \hat{\text{A}}), \tag{11}$$

$$\hat{\text{Y}} = f(\text{Z}), \tag{12}$$

$$\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{n_L} y_i \log(\hat{y}_i). \tag{13}$$

TABLE I
STATISTICS OF DATASETS

Datasets	Nodes	Edges	Attributes	Classes
Amazon	11,944	4,398,392	25	2
T-Finance	39,357	21,222,543	10	2
YelpChi	45,954	3,846,979	32	2

TABLE II
PERFORMANCE IN THE LGNA TASK. THE BEST PERFORMANCE IS SHOWN IN **RED CELLS AND BOLDED**, AND THE SECOND RUNNERS ARE SHOWN IN GREEN CELLS AND UNDERLINED. "OOM" MEANS OUT-OF-MEMORY.

Method	**Dataset**	**Amazon**			**T-Finance**			**YelpChi**		
	Metric	**F1-macro**	**AUC**	**GMean**	**F1-macro**	**AUC**	**GMean**	**F1-macro**	**AUC**	**GMean**
Baselines	GCN	0.6388	0.7851	0.5970	0.7138	0.8138	0.5543	0.5341	0.5668	0.4742
	GAT	0.5867	0.5582	0.5135	0.6121	0.5919	0.4442	0.5075	0.5205	0.1833
	GraphSAGE	0.7375	0.84.52	0.6617	0.7918	0.8587	0.6495	0.5756	0.6614	0.5241
	ASD-VAE	0.6676	0.8028	0.4758	OOM	OOM	OOM	OOM	OOM	OOM
	PCFI	0.4751	0.7911	0.0000	OOM	OOM	OOM	0.5470	0.5760	0.5243
	SUBLIME	0.4750	0.4251	0.0000	OOM	OOM	OOM	OOM	OOM	OOM
	DPT	0.5980	0.9074	0.8121	0.6401	0.8968	0.6421	0.2695	0.6429	0.5139
	D2PT	0.6114	0.8995	0.7547	0.4311	0.8389	0.6187	0.2872	0.6634	0.3578
	T2GNN	0.7180	0.8648	0.7843	<u>0.8090</u>	0.9091	0.7805	<u>0.6818</u>	<u>0.8290</u>	**0.7521**
	GDN	0.8085	0.8881	0.7586	0.7185	<u>0.9124</u>	0.5797	0.6234	0.7421	0.6389
	GHRN	0.7778	0.9003	<u>0.8300</u>	0.7382	0.8853	<u>0.8088</u>	0.5245	0.5956	0.5624
	PMP	<u>0.8493</u>	<u>0.9343</u>	0.8082	0.7219	0.7320	0.6744	0.6200	0.7233	0.6335
Ours	RENA	**0.8639**	**0.9583**	**0.8980**	**0.8636**	**0.9429**	**0.8823**	**0.7004**	**0.8296**	<u>0.7403</u>

V. EXPERIMENTS

In this section, we validate the effectiveness of RENA in LGNA to address the following research questions:

- **RQ1:** How does RENA perform in comparison to state-of-the-art methods in the LGNA task?
- **RQ2:** Does RENA achieve superior performance over state-of-the-art methods in the LGI task? [1]
- **RQ3:** How do the key components impact the performance of RENA?
- **RQ4:** How do different hyperparameter settings influence the performance of RENA?

A. Experimental Setup

1) Datasets.: We adopt three fraud datasets for evaluation. The Amazon dataset [36] includes product reviews under the Musical Instrument category. The nodes in the Amazon dataset represent users with 25-dimensional attributes. The edges represent connections between two users, such as reviewing the same product at least once, having at least one instance of the same star rating within one week, or sharing top-5% mutual review TF-IDF similarities.

The YelpChi dataset [37] includes hotel and restaurant reviews, which are filtered as spam and recommended as legitimate by Yelp. The nodes present reviews with 32-dimensional attributes. The edges represent connections between two reviews, such as those posted by the same user, under the same product, or posted in the same month under the same product.

The T-Finance dataset [31] aims to find the anomaly accounts in transaction networks. The nodes are unique anonymized accounts with 10-dimensional attributes related to registration days, logging activities, and interaction frequency. The edges in the graph represent two accounts that have transaction records. Human experts annotate nodes as anomalies if they fall into categories like fraud, money laundering, and online gambling. For Amazon and YelpChi datasets, we treat different types of edges as a single type for processing. More statistics are shown in Table I.

[1]RENA can also effectively handle LGA and LGN tasks; however, due to space constraints, we validate its effectiveness on the more complex LGI task.

2) Compared Methods: We evaluated our approach against five groups of baseline models: (1) traditional GNNs, such as GCN [9], GAT [10], and GraphSAGE [11]; (2) two models for attribute imputation, namely ASD-VAE [20] and PCFI [27]; (3) a model for structure learning, SUBLIME [18]; (4) three GNNs designed for incomplete graphs, including D2PT [14], DPT [14], and T2GNN [24]; and (5) three methods designed for fraud detection, including GDN [38], GHRN [39], PMP [40]. Comprehensive details can be found in Appendix B.

3) Experimental Settings: For all methods, nodes are randomly split into training, validation, and test sets in a 40%, 20%, and 40% ratio. Each experiment is repeated with five random seeds, and the average performance is reported. To simulate real-world noise, we randomly sample 30% of the number of original edges from the unconnected pairs, add connections for each sampled pair to form noisy edges, and mask 30% of the attributes for each node. For fair comparison, all baselines are implemented using their official code and original configurations. RENA employs a two-layer GAT with 256 hidden units for both the encoder and decoder and is optimized using the Adam [41] optimizer, with a learning rate of 0.005, a weight decay of 0.0005. And the number of learnable weight vectors is $m = 2$. For all methods, we save the model according to the best F1 score on the validation set as [31]. More details can be found in Appendix C.

4) Implementation: RENA is implemented in PyTorch 2.3.0 with Python 3.9. All experiments are conducted on an Ubuntu 22.04.4 LTS server with 72 cores, 512 GB of RAM, and an NVIDIA A100 GPU(80 GB).

5) Metrics: We choose three widely used metrics to measure the performance of all the methods, namely F1-macro, AUC [42], and GMean. F1-macro is the unweighted mean of the F1-score of each class, which neglects the imbalance ratio between normal and anomaly labels. AUC is the area under the ROC Curve. GMean calculates the geometric mean of the

TABLE III

PERFORMANCE IN INCOMPLETE GRAPHS. THE BEST PERFORMANCE IS SHOWN IN **RED CELLS AND BOLDED**, AND THE SECOND RUNNERS ARE SHOWN IN <u>GREEN CELLS AND UNDERLINED</u>. "OOM" MEANS OUT-OF-MEMORY.

Method	Dataset	Amazon			T-Finance			YelpChi		
	Metric	F1-macro	AUC	GMean	F1-macro	AUC	GMean	F1-macro	AUC	GMean
Baselines	GCN	0.6961	0.8290	0.6358	0.7906	0.8890	0.7212	0.5479	0.5777	0.4978
	GAT	0.6262	0.6095	0.5030	0.5881	0.6437	0.4199	0.5364	0.5584	0.3800
	GraphSAGE	0.7359	0.7899	0.6270	0.8031	0.8693	0.7306	0.5747	0.6617	0.5224
	ASD-VAE	0.6644	0.7998	0.1099	OOM	OOM	OOM	OOM	OOM	OOM
	PCFI	0.5704	0.8182	0.7189	OOM	OOM	OOM	0.5725	0.6150	0.5659
	SUBLIME	0.4750	0.4522	0.0000	OOM	OOM	OOM	OOM	OOM	OOM
	DPT	0.5990	0.9081	0.8246	0.6898	0.9073	0.8033	0.1990	0.5710	0.4522
	D2PT	0.5636	0.8798	0.7633	0.6988	0.9085	0.8055	0.2298	0.5945	0.5507
	T2GNN	0.7512	0.8594	0.7450	**0.8706**	<u>0.9320</u>	<u>0.8776</u>	**0.7114**	<u>0.8281</u>	**0.7621**
	GDN	0.8130	0.8589	0.7448	0.7485	0.8997	0.6394	0.6362	0.7829	0.6908
	GHRN	0.7813	0.9072	0.8329	0.8088	0.9055	0.8356	0.5189	0.5922	0.5637
	PMP	**0.8569**	<u>0.9364</u>	<u>0.8357</u>	0.6512	0.6475	0.5761	0.5810	0.7202	0.5046
Ours	RENA	<u>0.8394</u>	**0.9535**	**0.8952**	<u>0.8621</u>	**0.9473**	**0.8892**	<u>0.6913</u>	**0.8322**	<u>0.7574</u>

True Positive Rate (TPR) and True Negative Rate (TNR).

B. Performance on LGNA (RQ1)

To answer RQ1, we conducted comparisons against all baselines on the three datasets, as summarized in Table II. We observe that: (a) RENA outperforms all baselines over all evaluation metrics on Amazon and T-Finance datasets, and comparable performance on the YelpChi dataset. This is because RENA supervises the learning of graph structure by focusing on unconnected node pairs, and diluting the influence of connected node pairs on the learned graph structure. Additionally, RENA effectively dilutes the impact of imputation noise on observed attributes by reweighing observed and imputed attributes during the reconstruction process; (b) Methods designed to handle incomplete graphs, such as D2PT, DPT, and T2GNN, generally outperform those focused solely on absent attributes or structural noise across all metrics on the Amazon dataset. These methods typically rely on either a reliable structure or attributes to optimize the model. When both attributes are absent and structures are noisy, neither provides reliable information; (c) DPT outperforms D2PT on most metrics in both the Amazon and T-Finance datasets. This is likely because D2PT uses long-range information to impute attributes, which are then used to construct the global graph structure. However, when the structure contains noise, it aggregates irrelevant neighborhood information, thus affecting the observed attributes and the modeling of the graph structure.

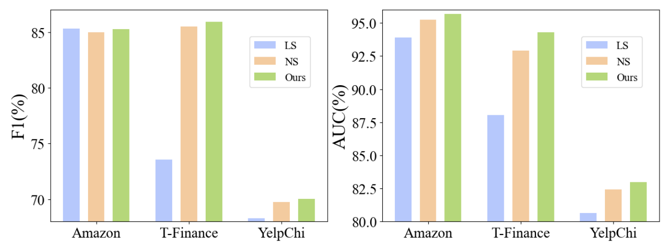
C. Performance on LGI (RQ2)

To answer RQ2, we evaluated our method in the LGI task, where both the structure and attributes are absent. Specifically, we randomly masked 30% of edges and attributes for each node to create incomplete graphs like [24]. Table III reports the results, and we have the following findings: (a) Although RENA is specifically tailored for the LGNA task, it achieves state-of-the-art or comparable performance, demonstrating its generalizability to LGI tasks. This generalizability stems from the ability to adjust the dilution ratio r according to the characteristics of the graph structure in different tasks. For example, in the LGI task, where some edges in the graph are absent, connected node pairs are more reliable, while the unconnected ones are unreliable. We can increase the proportion of connected node pairs to dilute the influence of unconnected node pairs on the learned structure. (b) In contrast, DPT, D2PT and T2GNN cannot adjust their graph structure optimization methods based on the characteristics of graph structures specific to tasks, limiting their generalizability to handle different tasks. As shown in Table III, DPT, D2PT, and T2GNN exhibit performance differences between the LGI and LGNA tasks in the T-Finance and YelpChi datasets.

D. Ablation Study (RQ3)

To answer RQ3, we verify the contributions of the key components in RENA: MSD and ARD. To verify MSD, we replace the optimized structure $\hat{\mathrm{A}}$ with the original noisy structure A (NS) or the similarity-based structure A^s (LS), respectively, and compare the results with our method. Figure 4(a) shows the results and we have the following findings: (a) RENA achieved the best performance compared with the two variants. The reason is that the similarity of noisy edges in A^s is expected to be estimated lower than ideal edges so that noisy edges can be filtered out. (b) Replacing the optimized structure with the original noisy structure results in less performance degradation than the similarity-based structure, as the computed similarity is mainly used for filtering noisy edges rather than recovering the full structure.

To validate the dilution effect of ARD on imputation noise, we conducted experiments under different ratios of λ_1 to λ_2. The results in Figure 4(b) demonstrate better performance as the ratio of λ_1 to λ_2 exceeds 1. This is because attribute imputation introduces unreliable information, propagating through the structure and weakening the embedding of the observed node attributes. When λ_1 is larger, the model focuses more on the observed attributes, thus diluting the impact of imputation


(a) LS and NS are the similarity-based structure and the original noisy structure

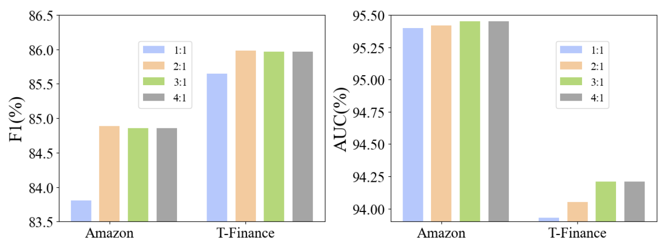
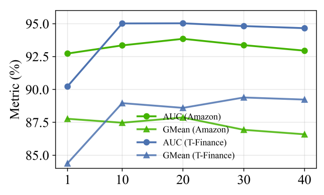
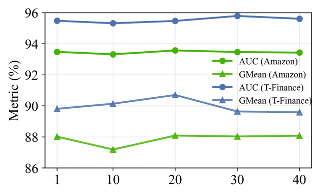
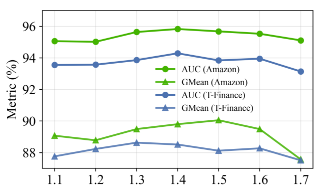

(b) Ratios of λ_1 to λ_2 during the reconstruction process

Fig. 4. Ablation of Multi-View Structure Dilution (a) and Attributes Reconstruct Dilution (b) on three datasets.


(a) Dilution ratio


(b) Number of structural views


(c) Threshold

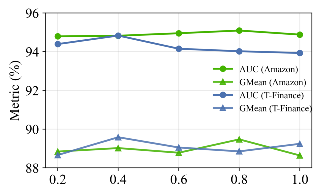

(d) Ratio of sampled connected pairs

Fig. 5. Sensitivity analysis of r, T, threshold, and ratios of unconnected to connected node pairs on Amazon and T-Finance datasets.

noise on the observed attributes. We further validate the effectiveness of reweighing observed and imputed attributes in ARD across different imputation methods, with results and additional analysis provided in the AppendixD.

E. Sensitivity Analysis (RQ4)

To answer RQ4, we further evaluate the sensitivity of RENA from four aspects: dilution ratio r, which is the ratio of unconnected node pairs to connected node pairs in a structural view, number of structural views T, threshold, and the ratio of sampled connected node pairs p. Due to limited space, we only report the results on the Amazon and T-Finance datasets, as shown in Figure 5[2]. Other datasets demonstrate similar sensitivity patterns.

Figure 5(a) shows the results of dilution ratio r. From the results, we can draw the following conclusions: (a) Although there may be noisy edges in the structural view, increasing the dilution ratio can reduce the impact of noisy edges on model performance. (b) A larger ratio leads to a significant increase in unconnected node pairs compared to connected ones, thereby impairing the ability of the model to accurately learn the weights of the edges.

Figure 5(b) shows the results of T. The results indicate that when T is set too low (for example, below 20 in the Amazon and T-Finance datasets), the number of connected node pairs is inadequate to effectively guide the model to learn edge weights. In contrast, when T exceeds 20, more noisy edges appear, leading to performance degradation. Figure 5(c) demonstrates that the best performance is achieved when the threshold is 1.4 on amazon and T-Finance, with performance degradation occurring when the threshold is either larger or smaller than 1.4. This is because when the threshold is set below 1.4, noisy edges remain in the optimized structure, whereas when the threshold exceeds 1.4, some reliable edges are filtered out, both leading to decreased model performance. Figures 5(d) shows the impact of the ratio of sampled node pairs. We observed that the model reaches optimal performance at a ratio of 0.8 on Amazon and at a ratio of 0.4 on T-Finance. When the ratio exceeds 0.8 on Amazon or 0.4 on T-Finance, the sampled edges contain more noisy edges, whereas when the ratio is lower than 0.4, the number of sampled edges is insufficient to effectively supervise model training.

[2]Only AUC and G-Mean are reported, as the relatively low F1-score impairs the clarity of joint visualization.

VI. Conclusion

In this work, we present the problem of graph learning with noisy structure and absent attributes (a.k.a LGNA). Our proposed approach, RENA, mitigates the impact of unreliable information by utilizing more reliable unconnected node pairs and observed attributes as supervision signals. A large number of unconnected node pairs and a small set of connected node pairs are sampled to create multiple structural views to supervise the graph structure learning. Then, a graph autoencoder is employed to reconstruct node attributes, with higher weights assigned to observed attributes. Finally, the encoder is fine-tuned with labels for downstream tasks. Extensive experiments validate the effectiveness of RENA in both LGNA and incomplete graph learning tasks.

Acknowledgments

The research work is supported by National Key R&D Plan No.2022YFC3303302, the National Natural Science Foundation of China under Grant (No. U2436209, 62476263), the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDB0680201, Beijing Nova Program 20230484430, the Innovation Funding of ICT, CAS under Grant No. E461060. Yang Liu is also supported by the Postdoctoral Fellowship Program of CPSF under Grant Number GZB20240761.

References

[1] C. Liu, Q. Zhong, X. Ao, L. Sun, W. Lin, J. Feng, Q. He, and J. Tang, "Fraud transactions detection via behavior tree with local intention calibration," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3035–3043.

[2] Q. Zhong, Y. Liu, X. Ao, B. Hu, J. Feng, J. Tang, and Q. He, "Financial defaulter detection on online credit payment via multi-view attributed heterogeneous information network," in *Proceedings of the web conference 2020*, 2020, pp. 785–795.

[3] M. Kumar, R. Ghani, and Z.-S. Mei, "Data mining to predict and prevent errors in health insurance claims processing," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 65–74.

[4] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, 2020, pp. 1569–1572.

[5] X. Zhu, X. Ao, Z. Qin, Y. Chang, Y. Liu, Q. He, and J. Li, "Intelligent financial fraud detection practices in post-pandemic era," *The Innovation*, vol. 2, no. 4, 2021.

[6] Global Anti-Scam Alliance, "The Global State of Scams - 2023," 2023. [Online]. Available: https://www.gasa.org/_files/ugd/7bdaac_b0d2ac61904941aeb4cbf0217aa355d2.pdf

[7] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data mining and knowledge discovery*, vol. 29, pp. 626–688, 2015.

[8] P. Boniol, T. Palpanas, M. Meftah, and E. Remy, "Graphan: Graph-based subsequence anomaly detection," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 2941–2944, 2020.

[9] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[10] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[11] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[12] H. Jiang, J. Pei, D. Yu, J. Yu, B. Gong, and X. Cheng, "Applications of differential privacy in social network analysis: A survey," *IEEE transactions on knowledge and data engineering*, vol. 35, no. 1, pp. 108–127, 2021.

[13] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, 2009, pp. 135–146.

[14] Y. Liu, K. Ding, J. Wang, V. Lee, H. Liu, and S. Pan, "Learning strong graph neural networks with weak information," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 1559–1571.

[15] H. Xuanwen, Y. Yang, W. Yang, W. Chunping, Z. Zhisheng, X. Jiarong, C. Lei, and V. Michalis, "Dgraph: A large-scale financial dataset for graph anomaly detection," in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[16] D. Lao, X. Yang, Q. Wu, and J. Yan, "Variational inference for training graph neural networks in low-data regime through joint structure-label estimation," in *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, 2022, pp. 824–834.

[17] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 66–74.

[18] Y. Liu, Y. Zheng, D. Zhang, H. Chen, H. Peng, and S. Pan, "Towards unsupervised deep graph structure learning," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1392–1403.

[19] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Learning discrete structures for graph neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 1972–1982.

[20] X. Jiang, Z. Qin, J. Xu, and X. Ao, "Incomplete graph learning via attribute-structure decoupled variational auto-encoder," in *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024, pp. 304–312.

[21] W. Tu, S. Zhou, X. Liu, Y. Liu, Z. Cai, E. Zhu, C. Zhang, and J. Cheng, "Initializing then refining: A simple graph attribute imputation network." in *IJCAI*, 2022, pp. 3494–3500.

[22] W. Tu, B. Xiao, X. Liu, S. Zhou, Z. Cai, and J. Cheng, "Revisiting initializing then refining: an incomplete and missing graph imputation network," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[23] J. Yoo, H. Jeon, J. Jung, and U. Kang, "Accurate node feature estimation with structured variational graph autoencoder," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 2336–2346.

[24] C. Huo, D. Jin, Y. Li, D. He, Y.-B. Yang, and L. Wu, "T2-gnn: Graph neural networks for graphs with incomplete features and structure via teacher-student distillation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, 2023, pp. 4339–4346.

[25] S. Fu, Q. Peng, Y. He, B. Du, and X. You, "Towards unsupervised graph completion learning on graphs with features and structure missing," in *2023 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2023, pp. 1019–1024.

[26] X. Chen, S. Chen, J. Yao, H. Zheng, Y. Zhang, and I. W. Tsang, "Learning on attribute-missing graphs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 2, pp. 740–757, 2020.

[27] D. Um, J. Park, S. Park, and J. Y. Choi, "Confidence-based feature imputation for graphs with partially known features," *arXiv preprint arXiv:2305.16618*, 2023.

[28] W. Zhao, Q. Wu, C. Yang, and J. Yan, "Graphglow: Universal and generalizable structure learning for graph neural networks," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 3525–3536.

[29] R. Wang, S. Mou, X. Wang, W. Xiao, Q. Ju, C. Shi, and X. Xie, "Graph structure estimation neural networks," in *Proceedings of the web conference 2021*, 2021, pp. 342–353.

[30] Z. Guo, L. Xia, Y. Yu, Y. Wang, Z. Yang, W. Wei, L. Pang, T.-S. Chua, and C. Huang, "Graphedit: Large language models for graph structure learning," *arXiv preprint arXiv:2402.15183*, 2024.

[31] J. Tang, J. Li, Z. Gao, and J. Li, "Rethinking graph neural networks for anomaly detection," in *International Conference on Machine Learning*. PMLR, 2022, pp. 21 076–21 089.

[32] P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal, "Pattern classification with missing data: a review," *Neural Computing and Applications*, vol. 19, pp. 263–282, 2010.

[33] G. E. Batista, M. C. Monard *et al.*, "A study of k-nearest neighbour as an imputation method." *His*, vol. 87, no. 251-260, p. 48, 2002.

[34] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *The Journal of Machine Learning Research*, vol. 11, pp. 2287–2322, 2010.

[35] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang, "Graphmae: Self-supervised masked graph autoencoders," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 594–604.

[36] J. J. McAuley and J. Leskovec, "From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 897–908.

[37] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, 2015, pp. 985–994.

[38] Y. Gao, X. Wang, X. He, Z. Liu, H. Feng, and Y. Zhang, "Alleviating structural distribution shift in graph anomaly detection," in *Proceedings of the sixteenth ACM international conference on web search and data mining*, 2023, pp. 357–365.

[39] ——, "Addressing heterophily in graph anomaly detection: A perspective of graph spectrum," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 1528–1538.

[40] W. Zhuo, Z. Liu, B. Hooi, B. He, G. Tan, R. Fathony, and J. Chen, "Partitioning message passing for graph fraud detection," in *The Twelfth International Conference on Learning Representations*, 2024.

[41] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[42] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 233–240.

APPENDIX

A. Details About Sampling Latent Edges

To validate our hypothesis that latent edges have a minimal impact on graph learning, we designed the following experiment. Specifically, we add 10% and 30% of connected pairs to the unconnected set, viewing them as latent edges with potential semantic relationships, and then sampled from this unconnected set to construct MSV. The results are shown in Table V, which validates our hypothesis. This is due to the significantly smaller number of these latent edges compared to the truly unconnected node pairs, resulting in a low probability of sampling these latent edges and, consequently, their minimal influence on graph learning. For example, when considering 30% of the connected node pairs as latent edges, the sampling probabilities of these latent edges in the Amazon, T-Finance, and Yelp datasets are only 1.94%, 0.84%, and 0.11%, respectively.

TABLE V
THE IMPACT OF DIFFERENT LATENT EDGES RATIO ON GRAPH LEARNING

Rate	Amazon		T-Finance		YelpChi	
	F1	AUC	F1	AUC	F1	AUC
0%	84.69	93.54	84.87	93.56	69.73	82.55
10%	84.44	94.00	84.70	92.62	69.47	82.04
30%	82.87	93.78	85.13	93.28	69.77	82.15

B. Baselines Information

In this section, we provide a brief overview of the baseline methods used in our experiments.

- GCN [9], GAT [10] and GraphSAGE [10] are three traditional GNNs.
- SUBLIME [18] fuses incomplete attributes and a complete structure in a shared latent space, using decoupling to recover missing attributes.
- ASD-VAE [20] combines incomplete attributes and a complete structure within a shared latent space, utilizing a decoupling mechanism to restore missing attributes.
- PCFI [27] employs both intra-channel and inter-channel strategies to effectively impute missing attributes.
- D2PT [14] utilizes long-range propagation and constructs a global graph structure to tackle the incomplete graph.
- DPT [14], a variant of D2PT, focuses solely on long-distance propagation for efficient message passing.
- T2GNN [24] employs attribute and structure teachers to collaboratively guide the learning of the student model.
- GDN [38] effectively addresses anomaly detection by dynamically adjusting to structural distribution shifts.
- GHRN [39] prune inter-class edges by emphasizing and delineating the high-frequency components of the graph.
- PMP [40] adaptively aggregates information from heterophilic and homophilic neighbors.

C. Hyperparameters

All hyperparameters are tuned based on the F1 score of the validation set. For RENA, the parameter q, is tuned from $\{0.2, 0.4, 0.6, 0.8, 1.0\}$, the parameter r, is tuned from $\{1, 10, 20, 30, 40\}$ and T, is tuned from $\{1, 10, 20, 30, 40\}$. The threshold used to generate optimized structure is tuned from $\{1.1,\ 1.2,\ 1.3,\ 1.4,\ 1.5,\ 1.6,\ 1.7\}$. We fix the value of λ_2 to 1, and tuned the value of λ_1 from $\{1,\ 2,\ 3,\ 4\}$. A 2-layer GAT is deployed as graph encoder and decoder, whose number of heads is tuned from $\{16, 32\}$.

D. Effectiveness of ARD and Theoretical Analysis

We conducted experiments with three different imputation methods to verify that reweighting observed and imputed attributes in ARD effectively dilutes the impact of imputation noise. The results in Table IV demonstrate that our reweighting strategy outperforms assigning equal weights. To have a deeper insights, we analyze the gradient of $\mathcal{L}_{\text{Rec}}$ w.r.t. $\hat{X}$:

$$\frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \hat{X}} = \begin{cases} \frac{2\lambda_1}{n_{\text{train}}} \left(\hat{X}_{ij} - X_{ij} \right), & \text{if } X_{ij} \in X_{obs}, \\ \frac{2\lambda_2}{n_{\text{train}}} \left(\hat{X}_{ij} - X_{ij} \right), & \text{if } X_{ij} \in X_{imp}. \end{cases}$$

Setting $\lambda_1 > \lambda_2$ increases the gradient from the observed attributes, making the model focus more on reconstructing them and thus reducing imputation noise.

TABLE IV
PERFORMANCE UNDER DIFFERENT RATIOS OF OBSERVED TO IMPUTED ATTRIBUTES. THE BEST PERFORMANCE IS SHOWN IN BOLDED. "OBS : IMP" REFERS TO THE WEIGHT RATIO OF OBSERVED ATTRIBUTES TO IMPUTED ATTRIBUTES DURING RECONSTRUCTION.

Imputation	**Ratio**	**Amazon**			**T-Finance**			**YelpChi**		
	Obs:Imp	**F1-macro**	**AUC**	**GMean**	**F1-macro**	**AUC**	**GMean**	**F1-macro**	**AUC**	**GMean**
Zero	1:1	0.8318	0.9415	0.8718	0.8437	0.9330	0.8797	0.6016	0.7310	0.6620
	2:1	0.8344	**0.9491**	**0.8782**	0.8415	0.9359	0.8787	0.6180	**0.7333**	**0.6688**
	3:1	**0.8392**	0.9451	0.8740	0.8422	**0.9376**	0.8792	0.6193	0.7275	0.6424
	4:1	0.8344	0.9440	0.8690	**0.8442**	0.9365	**0.8804**	**0.6203**	0.7270	0.6582
Mean	1:1	**0.8280**	0.9425	0.8721	0.8463	0.9326	0.8717	0.6883	0.8113	0.7267
	2:1	0.8149	0.9470	0.8846	0.8481	0.9382	**0.8818**	0.6867	0.8113	0.7295
	3:1	0.8163	0.9464	0.8800	0.8532	0.9386	0.8803	0.6874	0.8140	0.7331
	4:1	0.8164	**0.9481**	**0.8830**	**0.8559**	**0.9389**	0.8800	**0.6890**	**0.8142**	**0.7370**
Long_Range	1:1	0.8305	0.9509	0.8767	0.8502	0.9340	0.8792	0.6618	0.7830	0.7109
	2:1	0.8262	0.9516	0.8825	**0.8556**	0.9333	0.8778	0.6615	0.7785	0.7090
	3:1	**0.8347**	0.9527	0.8846	0.8477	0.9345	**0.8821**	**0.6677**	**0.7868**	**0.7126**
	4:1	0.8268	**0.9534**	**0.8851**	0.8459	**0.9346**	0.8806	0.6632	0.7845	0.7117