Guarding Graph Neural Networks for Unsupervised Graph Anomaly Detection

Yuanchen Bei, Sheng Zhou, Jinke Shi, Yao Ma, *Member, IEEE*, Haishuai Wang, and Jiajun Bu, *Member, IEEE*

***Abstract*—Unsupervised graph anomaly detection aims at identifying rare patterns that deviate from the majority in a graph without the aid of labels, which is important for a variety of real-world applications. Recent advances have utilized graph neural networks (GNNs) to learn effective node representations by aggregating information from neighborhoods. This is motivated by the hypothesis that nodes in the graph tend to exhibit consistent behaviors with their neighborhoods. However, such consistency can be disrupted by graph anomalies in multiple ways. Most existing methods directly employ GNNs to learn representations, disregarding the negative impact of graph anomalies on GNNs, resulting in suboptimal node representations and anomaly detection performance. While a few recent approaches have redesigned GNNs for graph anomaly detection under semi-supervised label guidance, *how to address the adverse effects of graph anomalies on GNNs in unsupervised scenarios and learn effective representations for anomaly detection are still underexplored.* To bridge this gap, in this article, we propose a simple, yet effective framework for guarding GNNs for unsupervised graph anomaly detection (G3AD). Specifically, G3AD first introduces two auxiliary networks along with correlation constraints to guard the GNNs against inconsistent information encoding. Furthermore, G3AD introduces an adaptive caching (AC) module to guard the GNNs from directly reconstructing the observed graph data that contains anomalies. Extensive experiments demonstrate that our G3AD can outperform 20 state-of-the-art methods on both synthetic and real-world graph anomaly datasets, with flexible generalization ability in different GNN backbones.**

***Index Terms*—Graph anomaly detection, graph learning, graph neural networks (GNNs), unsupervised anomaly detection.**

NOMENCLATURE

Abbreviation	Expansion
$\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$	Attributed graph with anomalies.
$\boldsymbol{A}$	Graph adjacency matrix.
$\boldsymbol{X}$	Node attribute matrix.
$\mathcal{V}$	Set of nodes in $\mathcal{G}$.
$\mathcal{E}$	Set of edges in $\mathcal{G}$.
$\mathcal{N}_i$	Neighborhood set of node v_i.
$n = \lvert\mathcal{V}\rvert$	Number of nodes in $\mathcal{G}$.
d	Dimension of node attributes in $\mathcal{G}$.
k	Number of graph anomalies in $\mathcal{G}$.
$\boldsymbol{Y}$	Anomaly label set of nodes.
$\boldsymbol{S}$	Anomaly score vector indicating node abnormalities.

Received 25 April 2024; revised 20 February 2025; accepted 4 May 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62476245 and in part by Zhejiang Provincial Natural Science Foundation of China under Grant LTGG23F030005. *(Corresponding author: Sheng Zhou.)*

Yuanchen Bei, Jinke Shi, and Haishuai Wang are with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310058, China (e-mail: yuanchenbei@zju.edu.cn; shijinke@zju.edu.cn; haishuai.wang@zju.edu.cn).

Sheng Zhou and Jiajun Bu are with Zhejiang Key Laboratory of Accessible Perception and Intelligent Systems, Zhejiang University, Hangzhou 310058, China (e-mail: zhousheng_zju@zju.edu.cn; bjj@zju.edu.cn).

Yao Ma is with the Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: majunyao@gmail.com).

Digital Object Identifier 10.1109/TNNLS.2025.3569526

I. INTRODUCTION

GRAPH anomaly detection aims to identify rare patterns or behaviors that significantly deviate from the majority of a graph [1], [2], [3]. It has attracted increasing attention from both academia and industry due to its practical applications, such as network intrusion detection [1], social spammer detection [4], and financial fraud detection [5]. The scarcity of anomaly labels in real-world applications, coupled with the challenge of acquiring such labels, has led to a widespread study of unsupervised graph anomaly detection [6], [7]. Pioneer works on graph anomaly detection have separately mined the attributed anomalies and topological anomalies by feature engineering and graph structure encoding, respectively [8], [9]. Recent breakthroughs in graph neural networks (GNNs) have empowered the unified modeling of attributes and topology, leading to significant advancements in unsupervised graph anomaly detection [10], [11], [12].

The success of representative GNNs, as illustrated in Fig. 1(a), has heavily relied on aggregating information from neighborhoods sharing similar patterns, for example, label homophily and feature consistency [13], [14]. Although successful, in graphs with anomalies, we argue that this foundation can be easily undermined by various graph anomalies in multiple ways [15]. Fig. 1(b)–(d) illustrates toy examples of the three main types of graph anomaly impacts on GNNs.

1) As illustrated in Fig. 1(b), the *attributed-induced anomalies* where node features are corrupted, such as the account takeover in social networks, will directly result in aggregating incorrect information from neighborhoods.
2) As illustrated in Fig. 1(c), the *topological-induced anomalies* that connect to incorrect neighborhoods, such as the fraudsters in E-commerce systems, will result in aggregating information from inconsistent neighborhoods.

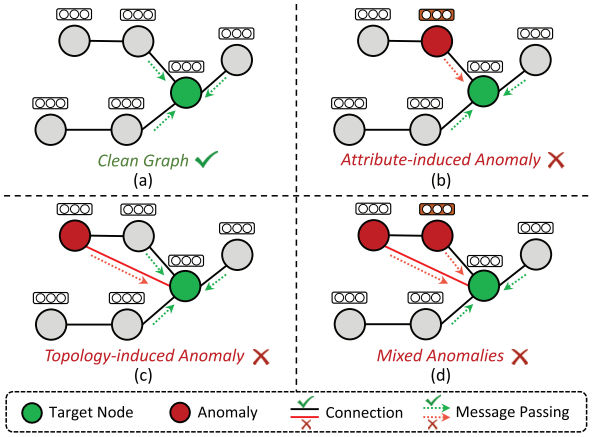

Fig. 1. Toy examples of GNN message passing on clear graphs and the graphs under three types of anomaly impacts. (a) Clean graph; (b) attribute-induced anomaly; (c) topology-induced anomaly; and (d) mixed anomalies.

3) More seriously, as illustrated in Fig. 1(d), the *mixed anomalies* that simultaneous occurrence of multiple anomaly types is a more common situation and will have a significantly more detrimental impact on GNNs.

Existing unsupervised graph anomaly detection methods have primarily focused on designing effective unsupervised anomaly scoring functions in the representation space learned directly from GNNs while *overlooking the negative impact of anomalies on the inherent GNNs themselves* [3], [16], [17]. This is crucial for accurate anomaly detection, which is indispensable to discriminative node representations. Under this limitation, the representation learning ability of GNNs is hampered from adequately capturing normal patterns in the graph, thereby leading to suboptimal representation learning and anomaly detection performance. Hence, it is crucial to alleviate the negative impact of anomalies and fully release the power of GNNs for anomaly detection.

Although important, addressing the negative impacts of graph anomalies on GNNs in unsupervised scenarios is nontrivial. An intuitive solution might involve the development of innovative GNNs that adapt to certain graph anomalies. This has attracted increasing attention, and a few very recent works have achieved tremendous success in semi-supervised scenarios [18], [19]. However, the lack of labels in unsupervised anomaly detection poses a significant challenge in guiding the redesign of GNNs, which is more commonly encountered in practice. More importantly, while anomalies do exist on the graph and exert certain effects on GNNs, they account for only a small fraction of the total. The majority of patterns remain normal and can be effectively captured by GNNs. Therefore, a complete redesign or deprecation of GNNs could risk compromising the identification of the majority of normal patterns due to the influence of a minor proportion of anomalies, especially in unsupervised contexts. Therefore, *how to reduce the impact of anomalies on GNNs while enabling them to capture the majority of normal patterns effectively and aid in unsupervised anomaly detection?*

To answer the above underexplored research question, in this article, we propose a simple, yet effective framework for guarding graph neural networks for unsupervised graph anomaly detection (G3AD). Instead of directly encoding the observed graph with anomalies, G3AD first introduces two auxiliary encoders tailored with correlation constraints to guard the GNNs against encoding inconsistent information. Subsequently, to optimize GNNs in an unsupervised manner and detect multiple types of unknown anomalies, G3AD proposes a comprehensive learning objective that includes both local attribute/topology reconstruction and global consistency alignment, which also serves as anomaly scoring. During this process, to avoid directly reconstructing the observed graph with anomalies G3AD introduces an adaptive caching (AC) module to guard the GNNs from misleading learning objectives during model training. Fig. 2 illustrates the comparison between the existing GNN-based unsupervised graph anomaly detection paradigm and the one under G3AD framework. Extensive experiments on both four synthetic and two real-world graph anomaly datasets demonstrate that G3AD outperforms 20 state-of-the-art unsupervised graph anomaly detection models. Further in-depth analysis from diverse perspectives also demonstrates the strengths of G3AD. The main contributions of this article are summarized as follows.

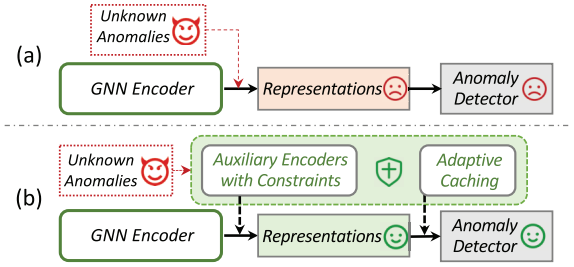

Fig. 2. Concept maps between the (a) existing GNN-based unsupervised graph anomaly detection paradigm and (b) our proposed GNN-guarded unsupervised graph anomaly detection paradigm.

1) We emphasize the negative impact of unknown anomalies on GNNs, which is crucial for graph anomaly detection under unsupervised settings while overlooked by most existing unsupervised works.
2) We propose G3AD, a simple yet effective framework to guard GNNs from encoding inconsistent information and directly reconstructing the abnormal graph in unsupervised graph anomaly detection.
3) We conduct extensive experiments on 20 widely used datasets, including both synthetic and real-world scenarios. Experimental results show that the proposed G3AD outperforms 20 state-of-the-art baselines.

In the following sections, we will first review the previous work related to our method in Section II. Second, we will give some key preliminaries of our work in Section III. Then, the detailed description of the proposed G3AD will be introduced in Section IV. To further verify the effectiveness of G3AD, we conduct various experiments in Section V. Finally, the conclusion of this article is posed in Section VI.

II. Related Works

A. Graph Neural Networks

GNNs are a series of deep learning models specifically designed for processing graph-structured data [20], [21]. The

core design of GNNs is to learn node representations by aggregating and propagating information from neighborhoods to the central nodes. This information propagation (message passing) allows GNNs to capture complex topology dependencies and contextual information among nodes [22], [23].

Typically, in GNNs, each node has an initial feature representation, and through multiple layers of message aggregation and propagation operations, the node representations are gradually updated and refined [20], [21], [24]. Representatively, graph convolution networks (GCNs) [20] adopt the graph convolution operator and stack it into multiple layers for neighbor message passing. Graph attention networks (GATs) [24] further introduce the attention mechanism to consider the different importance of neighbor nodes and dynamically assign different weights when aggregating neighbor features. GraphSAGE [21] extends and improves the graph convolution to handle large-scale graph data, with the key idea of generating node representations through sampling and aggregating neighbor nodes.

Because real-world data can be widely modeled as graphs, GNNs have emerged as a significant branch within the field of neural network research [23], [25]. They possess the ability to perform learning on graph-structured data, automatically extracting features and making predictions, providing an effective solution for tasks on graph data. Based on these advantages, GNNs have achieved significant success in various application domains, such as social network analysis [26], [27], recommendation systems [28], [29], and bioinformatics [30], [31].

B. Unsupervised Graph Anomaly Detection

Graph anomaly detection technologies have been widely applied in real-world systems to ensure their robustness and security, such as financial transaction networks [19], social network applications [32], and E-commerce systems [33].

Existing graph anomaly detection methods in unsupervised manners can be largely divided into four main categories.

1) *Classical Shallow Models:* SCAN [34] can be utilized for anomaly detection based on structural similarity. MLPAE [35] detects anomalies with nonlinear transformations.
2) *Enhanced GNNs:* GAAN [36] utilizes a generative adversarial framework to train the graph encoder with real graph and anomalous fake graph samples. ALARM [37] proposes a multiview representation learning framework with multiple graph encoders and a well-designed aggregator. AAGNN [11] then enhances the GNN with an abnormality-aware aggregator.
3) *Deep Graph Autoencoder Models:* Dominant [38] first introduces a deep graph autoencoder model with a shared encoder to measure anomalies by reconstruction error. AnomalyDAE [39] introduces asymmetrical cross-modality interactions between autoencoders. ComGA [40] further designs a community-aware tailored graph autoencoder to make the representation between normal and anomalous nodes more distinguishable. Recently, CoCo [41] enhances the graph autoencoder-based anomaly detector with node-level context correlation modeling.
4) *Graph Contrastive Learning Methods:* CoLA [42] exploits the local information and introduces a self-supervised graph contrastive learning method to detect anomalies. Furthermore, ANEMONE [43] utilizes the graph contrastive learning method at multiple scales. SL-GAD [16] further performs anomaly detection from both generative and multiview contrastive perspectives. Sub-CR [17] then proposes a self-supervised method based on multiview contrastive learning with graph diffusion and attribute reconstruction. Furthermore, ARISE [44] enhances graph contrastive learning with substructure discovery. NLGAD [6] adopts multi-scale graph contrastive learning with a hybrid normality measurement.

However, the above GNN-based models directly apply the message passing without anomaly guarding, neglecting the impact of neighborhood inconsistency on GNN mechanisms, thus the performance of GNNs is harmed and limited.

C. Neighborhood Consistency in Graphs

The nature that neighboring nodes in real-world graphs tend to share consistent behaviors, such as labels or attributes [45], has been the foundation of many GNNs. As introduced in Section II-A, pioneer representative works of GNNs [20], [21], [24] have utilized the neural message passing by aggregating information from neighbors, which is a straightforward way of applying the feature consistency. Later works have proved that such message passing is equivalent to classic label propagation, where the labels are propagated along with the connection between neighboring nodes [46]. This can be viewed as utilizing the neighborhood consistency in labels, which is also widely called the homophily assumption in practice [14].

However, the homophily assumption may not always hold in real-world graphs [47], [48], [49]. To tackle this challenge, recent works have been made on the *heterophily GNN* for representation learning under graphs with low neighborhood consistency. Representatively, MixHop [50] mixes powers of the adjacency matrix for graph convolution to ease the limitation. H2GCN [51] designs a model with ego and neighbor separation, higher-order neighbors, and intermediate representations combination. LINKX [52] separately embeds the adjacency and node features with simple MLP transformations rather than the aggregation based on neighborhood consistency. GloGNN [53] performs aggregation from the whole set of nodes with both low-pass and high-pass filters.

Nevertheless, in unsupervised graph anomaly detection, these heterophily GNNs mentioned above are not directly appropriate, due to the unknown neighborhood inconsistency in attribute and topology under the unsupervised setting and the overlook of the anomaly-specific design.

III. Preliminaries

In this section, we present some key notations and definitions related to our target unsupervised graph anomaly detection task. Note that we focus on node-level graph anomaly detection in this article. For the convenience of

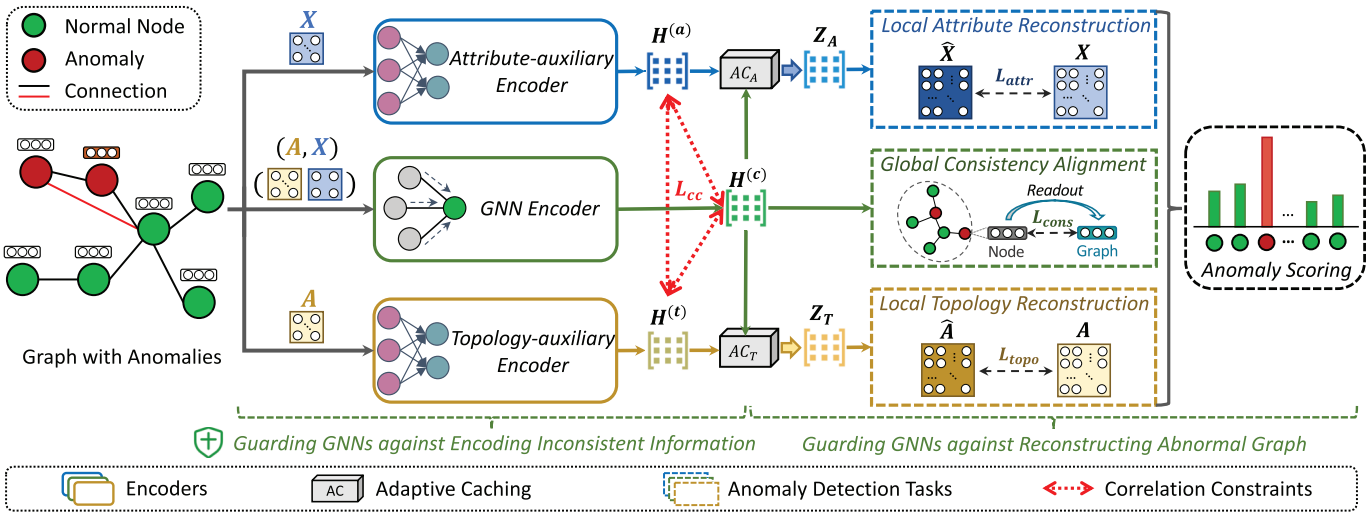

Fig. 3. Overall architecture of G3AD guarding framework contains two major novel parts: 1) Guarding GNNs against encoding inconsistent information with two auxiliary encoders with correlation constraints and 2) guarding GNNs against reconstructing abnormal graphs with adaptive information caching. Under the two guards, we comprehensively consider both local reconstruction and global alignment to detect different types of anomalies.

readers, we list the main symbols used in this article in Nomenclature.

Notations: Let $\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$ be a graph with the node set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ and the edge set $\mathcal{E}$, where $|\mathcal{V}| = n$. $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ denotes the graph adjacency matrix, for each element $A_{i,j}$ in $\boldsymbol{A}$, $A_{i,j} = 1$ indicates that there is an edge between node v_i and node v_j, and otherwise $A_{i,j} = 0$. $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ denotes the node attribute matrix, where the ith row vector $\boldsymbol{x}_i = \boldsymbol{X}[i,:] \in \mathbb{R}^d$ indicates the attribute vector of v_i with d-dimensional representation. $\mathcal{N}_i$ is the neighborhood set of a central node v_i in the graph $\mathcal{G}$.

Definition 1 (Graph Neural Networks) In form, for an L-layer GNN, the calculation process of each layer can be expressed as the aggregation and updating operators. In the aggregation phase, each central node aggregates the message from its neighbor nodes

$$\boldsymbol{m}_i^{(l+1)} = \text{Aggregator}(\{\boldsymbol{h}_j^{(l)} | j \in \mathcal{N}_i\}) \tag{1}$$

where the function Aggregator$(\cdot)$ is the message aggregation operator to aggregate information from nodes' neighborhoods, $\boldsymbol{h}_j^{(l)}$ is the representation of node v_j in the lth GNN layer, and $\boldsymbol{m}_i^{(l+1)}$ is the aggregated message for node v_i in the $(l+1)$th GNN layer. After obtaining the aggregated message, in the updating phase, each central node adopts the aggregated information to update and transform its own representations

$$\boldsymbol{h}_i^{(l+1)} = \text{Updater}(\boldsymbol{h}_i^{(l)}, \boldsymbol{m}_i^{(l+1)}) \tag{2}$$

where the function Updater$(\cdot)$ is the node representation updating operator and $\boldsymbol{h}_i^{(l+1)}$ is the updated representation of node v_i.

Definition 2 (Graph Anomalies) Given an abnormal attributed graph $\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$ containing n node instances, and k of them are graph anomalies ($k \ll n$), whose attributes, connections, or behaviors are different from most other normal nodes. On a graph with anomalies, each node v_i is associated with an anomaly label $y_i \in \boldsymbol{Y}$, where $\boldsymbol{Y}$ denotes the anomaly label set and each element $y_i \in \{0, 1\}$ denotes whether node v_i is an anomaly.

Definition 3 (Unsupervised Graph Anomaly Detection) Given an abnormal graph as Definition 2, the target of unsupervised graph anomaly detection is to learn a model $\mathcal{F}(\cdot) : \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^n$ in an unsupervised manner that outputs anomaly score vector $\boldsymbol{S} \in \mathbb{R}^n$ to measure the degree of abnormality of nodes. Specifically, the ith element s_i in the score vector $\boldsymbol{S}$ indicates the abnormality of node v_i, where a larger score means a higher abnormality. Note that the anomaly label set $\boldsymbol{Y}$ is invisible in the unsupervised setting.

IV. Methodology

In this section, we present the details of the guarding GNN for unsupervised graph anomaly detection (G3AD). The overall framework of G3AD is illustrated in Fig. 3. Generally, G3AD follows a representative GNN-based anomaly detection scheme where the GNNs learn the representations, and the representations are optimized with unsupervised objectives and measured for anomaly detection. Besides, G3AD introduces two novel guarding strategies to the paradigm, namely guarding against encoding inconsistent information and guarding against reconstructing abnormal graphs, so that the GNNs can produce effective representations and boost the anomaly detection ability. With carefully designed guarding strategies and equipped anomaly detection tasks, G3AD can outperform current GNN-based models. In the rest of this section, we will introduce the details of these guarding strategies and how G3AD serves unsupervised graph anomaly detection.

A. Guarding GNNs Against Encoding Inconsistent Information

Motivation: As discussed in previous works, GNNs are designed for capturing consistent information from neighborhoods [15]. However, in graphs with anomalies, the anomaly-induced inconsistent patterns in unsupervised graph

anomaly detection have been largely overwhelmed and will negatively disrupt the GNN performance. Therefore, G3AD first introduces two auxiliary encoders with correlation constraints to design an encoding guarding strategy for GNNs. Specifically, the auxiliary encoders are expected to encode the inconsistent information from the attribute and topology perspectives, and the GNNs are guarded against encoding the inconsistent information under the correlation constraints among the representations learned by the three encoders.

1) GNN Encoder: The GNN encoder is utilized for encoding the part of the information that satisfies the consistent homophily assumption, which is **shared** by both attribute and topology. To focus on the model architecture, we adopt a two-layer GAT [24] as the GNN encoder $f_{\text{gnn}}(\cdot)$ for simplicity. It is worth noting that any other GNNs can be directly applied here. Given the input attributes $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ of node v_i and its neighbor v_j, a GAT layer learns the input attributes with the attention mechanism. The attention coefficient is computed as

$$e_{i,j} = \text{att}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \text{LeakyReLU}(\boldsymbol{a}^T \cdot [\boldsymbol{W}_{\text{enc}}\boldsymbol{x}_i \| \boldsymbol{W}_{\text{enc}}\boldsymbol{x}_j]) \quad (3)$$

where $att(\cdot)$ is a single-layer feedforward neural network, parameterized by weight vector $\boldsymbol{a} \in \mathbb{R}^{2d'}$ and $\boldsymbol{W}_{\text{enc}} \in \mathbb{R}^{d' \times d}$ with the transformed dimension d', $\|$ denotes the concatenate operation. The attention coefficient $\alpha_{i,j}$ is normalized as

$$\alpha_{i,j} = \text{Softmax}(e_{i,j}) = \frac{\exp(e_{i,j})}{\sum_{k \in \mathcal{N}_i} \exp(e_{i,k})} \quad (4)$$

where $\mathcal{N}_i$ denotes the neighbor set of node v_i. Then, the final output feature $\boldsymbol{h}_i \in \mathbb{R}^{d'}$ of node v_i can be obtained as

$$\boldsymbol{h}_i = \sum_{j \in \mathcal{N}_i} \alpha_{i,j} \boldsymbol{W}_{\text{enc}} \boldsymbol{x}_j. \quad (5)$$

For convenience, we use $\boldsymbol{H}^{(c)} \in \mathbb{R}^{n \times d'}$ denote the output representation of consistency encoder $f_{\text{gnn}}(\cdot)$. Note that G3AD can be generalized in different GNN backbones, and any other graph message passing functions [21], [54] can also be adopted here. We will discuss the generalization ability of G3AD in Section V-D.

2) Auxiliary Encoders With Correlation Constraints: In our G3AD, the attribute-auxiliary encoder and the topology-auxiliary encoder are further utilized for encoding the part of information dependent on each information source (attribute and topology) as well as the potential specific type of anomalies under the attribute/topology inconsistent situation.

Existing methods directly utilize GNNs to model the attribute and topology simultaneously. However, in the part where the consistency is destroyed by anomalies, message passing in GNNs will introduce negative noise information, even helping anomalies to *camouflage* when passing normal messages to them [18]. Therefore, here we use two *independent* MLPs without graph message passing, rather than GNN encoders based on neighborhood consistency, to better model the inconsistent attribute/topology patterns and guard the GNN encoder learning.

Specifically, the two independent MLPs $f_a(\cdot)$ and $f_t(\cdot)$ are applied to encode the original attribute matrix $\boldsymbol{X}$ and the adjacency matrix $\boldsymbol{A}$, respectively,

$$\boldsymbol{H}^{(a)} = f_a(\boldsymbol{X}), \quad \boldsymbol{H}^{(t)} = f_t(\boldsymbol{A}) \quad (6)$$

where $f_a(\cdot)$ and $f_t(\cdot)$ are two MLPs with learnable parameters $\boldsymbol{W}_a$ and $\boldsymbol{W}_t$, respectively, and $\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(t)} \in \mathbb{R}^{n \times d'}$ are the corresponding output matrices.

To ensure better guarding of the GNN encoder against those inconsistent patterns, it is desired that the representations encoded from the GNN and the auxiliary encoders are more independent of each other. Therefore, we impose the correlation constraint on the embedding space of the encoded representation to ensure all aspects of information $\boldsymbol{H}^{(a)}$, $\boldsymbol{H}^{(t)}$, and $\boldsymbol{H}^{(c)}$ are well-disentangled encoding.

Specifically, the correlation constraint between each pair of encoded representation aspects is designed as the absolute correlation coefficient measurement. Minimizing the constraint ensures the two paired vectors become more independent [55]. Take the constraint between $\boldsymbol{H}^{(a)}$ and $\boldsymbol{H}^{(t)}$ as an example, the formal expression is as follows:

$$\text{aCor}(\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(t)}) = \text{abs}\left(\frac{\text{Cov}(\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(t)})}{\sqrt{\text{Var}(\boldsymbol{H}^{(a)}) \cdot \text{Var}(\boldsymbol{H}^{(t)})}}\right) \quad (7)$$

where $\text{Cov}(\cdot)$ is the covariance between two matrices, and $\text{Var}(\cdot)$ is a matrix's own variance, and $\text{abs}(\cdot)$ is the absolute value function. $\text{aCor}(\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(c)})$ and $\text{aCor}(\boldsymbol{H}^{(t)}, \boldsymbol{H}^{(c)})$ can be calculated in the same way. $\text{Cov}(\cdot)$ is insensitive to the order of the two inputs. The overall regularization $\mathcal{L}_{\text{cc}}$ of all correlation constraints can be presented as

$$\begin{aligned}\mathcal{L}_{\text{cc}} = &\text{aCor}(\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(c)}) + \text{aCor}(\boldsymbol{H}^{(t)}, \boldsymbol{H}^{(c)}) \\ &+ \text{aCor}(\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(t)})\end{aligned} \quad (8)$$

where the calculated $\mathcal{L}_{\text{cc}}$ is utilized as a correlation regularization loss during the model training stage.

B. Guarding GNNs Against Reconstructing Abnormal Graph

Motivation: Graph anomalies have various types, including attribute anomalies, topology anomalies, and mixed anomalies. In unsupervised graph anomaly detection, we are unable to know either anomaly labels or anomaly types. Thus, G3AD aims to comprehensively detect different anomaly types in the following ways.

1) *Local Attribute Reconstruction:* The attribute reconstruction errors distinguish sparse attribute anomalies from the predominant normal nodes.
2) *Local Topology Reconstruction:* The topology reconstruction errors distinguish sparse topology anomalies from the primary normal nodes.
3) *Global Consistency Alignment:* The alignment distance between node representations simultaneously considering attribute and topology information encoded by the GNN and the global graph consistency vector can further be equipped for anomaly distinguishing.

Among these objectives, the graph reconstruction scheme has been proved to be essential for both representation optimization and anomaly detection under the unsupervised setting [38], [39], [56]. However, reconstruction targets are needed to make GNN-encoded consistent representations to fit the observed graph data with anomalies. The unknown anomalies in the abnormal graph may provide a misleading objective to GNNs for both tasks. Thus, G3AD further introduces AC

to guard the GNNs against reconstructing abnormal graphs by cooperating with the auxiliary encoders that partake the inconsistent anomaly patterns.

1) Adaptive Caching: From the consistent alignment perspective, since the consistent information encoded by the GNN encoder has already been guarded, it should be directly used for measuring the global consistency under a mixture of attribute and topology information. *Yet from the attribute/topology reconstruction perspective*, due to the graph to be reconstructed containing anomalies, it is necessary to further guard the GNN-encoded representations against directly reconstructing and preserving the abnormal graph.

Therefore, we design an AC module to concurrently leverage the GNN-encoded representations for normal part reconstruction and auxiliary-encoded representations for inconsistent part reconstruction, avoiding the force of GNNs to fit inconsistent anomaly patterns. It automatically selects appropriate information for the reconstruction of normal and abnormal parts from GNN representation and auxiliary representation under unsupervised conditions, through learnable parameters. The general AC module can be formulated as follows:

$$\boldsymbol{imp} = \mathrm{Tanh}(\tau(\boldsymbol{H}^{(1)} \| \boldsymbol{H}^{(2)})) \tag{9}$$

$$\mathrm{AC}(\boldsymbol{H}^{(1)}, \boldsymbol{H}^{(2)}) = \boldsymbol{imp}_{[:n]} \cdot \boldsymbol{H}^{(1)} + \boldsymbol{imp}_{[n:]} \cdot \boldsymbol{H}^{(2)} \tag{10}$$

where $\tau(\cdot)$ is an MLP for information fusion, $\boldsymbol{imp} \in \mathbb{R}^{2n}$ is the weight vector of each dimension of the input features, and $\boldsymbol{H}^{(1)}$ and $\boldsymbol{H}^{(2)}$ denote two information sources, such as $\boldsymbol{H}^{(a)}$, $\boldsymbol{H}^{(t)}$, and $\boldsymbol{H}^{(c)}$.

The AC module is utilized to generate two types of representations for attribute and topology reconstruction

$$\boldsymbol{Z}_A = \mathrm{AC}(\boldsymbol{H}^{(a)}, \boldsymbol{H}^{(c)}), \quad \boldsymbol{Z}_T = \mathrm{AC}(\boldsymbol{H}^{(t)}, \boldsymbol{H}^{(c)}) \tag{11}$$

where $\boldsymbol{Z}_A$ and $\boldsymbol{Z}_T$ are the cached attribute representation and cached topology representation. Compared with vanilla GNN-based representation, the caching of GNN-encoded and attribute/topology-specific information automatically selects helpful information and better alleviates anomalies' impact on the GNN-encoded representations.

2) Anomaly Detection Tasks: Given the GNN encoded representation, cached attribute representation, and cached topology representation, we aim to make full use of them for anomaly scoring in an unsupervised setting. Therefore, as the above definition of the three types of anomaly detection objectives, the detection tasks can be unfolded as follows.

a) *Local attribute reconstruction:* The reconstruction-based operator has been widely observed to be effective for unsupervised anomaly detection [7], [38], [56]. Therefore, given the cached attribute representation $\boldsymbol{Z}_A$, we utilize the node's attribute reconstruction ability for attribute anomaly mining, benefiting from the reconstruction target tending to fit the patterns of the predominant nodes while the patterns of anomalies are rare [38]. We use a two-layer GCN [20] as the attribute reconstruction function $g_a(\cdot)$ as follows:

$$\boldsymbol{R}^{(l+1)} = \mathrm{LeakyReLU}(\tilde{\boldsymbol{A}} \boldsymbol{R}^{(l)} \boldsymbol{W}_{\mathrm{re}}^{(l)}) \tag{12}$$

where $\tilde{\boldsymbol{A}} = \hat{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{A}^* \hat{\boldsymbol{D}}^{-\frac{1}{2}} \in \mathbb{R}^{n \times n}$ is the normalized adjacency matrix, $\hat{\boldsymbol{D}} \in \mathbb{R}^{n \times n}$ is the degree matrix of $\boldsymbol{A}^* = \boldsymbol{A} + \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{R}^{(l)}$ and $\boldsymbol{W}_{\mathrm{re}}^{(l)}$ are the input features and trainable parameters in the lth layer, respectively.

For the cached attribute representation $\boldsymbol{Z}_A$, $g_a(\cdot)$ reconstructs the attribute matrix $\hat{\boldsymbol{X}} \in \mathbb{R}^{n \times d}$ with corresponding distance-based reconstruction loss $\mathcal{L}_{\mathrm{attr}}$, which can be computed as

$$\hat{\boldsymbol{X}} = g_a(\boldsymbol{Z}_A, \boldsymbol{A}) \tag{13}$$

$$\mathcal{L}_{\mathrm{attr}} = \|\boldsymbol{X} - \hat{\boldsymbol{X}}\|_2^2 \tag{14}$$

where $\|\cdot\|_2^2$ is the Euclidean distance.

b) *Local topology reconstruction:* Similar to the perspective of local attribute reconstruction, for the given cached topology representation Z_T, the topology reconstruction capability is also a factor for anomaly detection due to topology anomalies' rare and inconsistent local topological patterns. The reconstruction function $g_t(\cdot)$ reconstructs the adjacency matrix from $\boldsymbol{Z}_T$ with the reconstruction loss $\mathcal{L}_{\mathrm{topo}}$ as follows:

$$\hat{\boldsymbol{A}} = g_t(\boldsymbol{Z}_T) = \boldsymbol{Z}_T \cdot \boldsymbol{Z}_T' \tag{15}$$

$$\mathcal{L}_{\mathrm{topo}} = \|\boldsymbol{A} - \hat{\boldsymbol{A}}\|_2^2 \tag{16}$$

where $g_t(\cdot)$ is designed as an inner product between $\boldsymbol{Z}_T$ and its self-transposition $\boldsymbol{Z}_T'$ to make the reconstruction efficient.

c) *Global consistency alignment:* Given the GNN encoded representation $\boldsymbol{H}^{(c)}$ with both the attribute and topology information by graph message passing to detect anomalies, a naive way for evaluation is to measure the distance between node embeddings and their corresponding subgraph. However, the anomalies may occur as neighbors of normal nodes under the neighborhood inconsistency. Aligning the node with a noisy subgraph may be suboptimal. Thus, due to abnormal nodes making up only a minority of the entire graph, we turn to measure each node embedding with the graph summary vector for global consistency measurement to reduce the negative impact of neighborhood on the summary vectors.

Specifically, to conduct the alignment, we first read out the consistent GNN-encoded representations of nodes into a graph summary representation

$$\boldsymbol{E}_g = \mathrm{Readout}(\boldsymbol{H}^{(c)}) \tag{17}$$

where Readout$(\cdot)$ can be a kind of pooling operation (such as min, max, mean, and weighted pooling [42]). Here, we use the mean pooling as default for simplicity and $\boldsymbol{E}_g$ is the readout representation of the graph.

Then, we conduct the global consistency alignment task between each node's GNN-encoded representation $\boldsymbol{z}_{c,i} \in \boldsymbol{H}^{(c)}$ and the graph summary vector $\boldsymbol{E}_g$ as follows:

$$\mathcal{L}_{\mathrm{cons}} = \log\left(\sqrt{\sum_{i=1}^{n} \|\boldsymbol{z}_{c,i} - \boldsymbol{E}_g\|_2^2} + e\right) \tag{18}$$

where $\boldsymbol{z}_{c,i}$ is the consistent representation of node v_i and constant e is used to limit the lower bound of the loss for better balance $\mathcal{L}_{\mathrm{cons}}$'s numerical relationship with other loss terms.

C. Anomaly Scoring

With the three different anomaly detection tasks, we utilize the above three anomaly detection tasks for the final anomaly scoring. The normal nodes in the graph are expected to show low discrepancies, while anomalies exhibit high discrepant values due to their inconsistency, irregularity, and diversity. Therefore, here we compute the anomaly score s_i of each node v_i in multiperspectives according to

$$s_i = \underbrace{\lambda_1 \cdot \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|_2^2}_{\text{Local Attribute Reconstruction}} + \underbrace{(1-\lambda_1) \cdot \|\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i\|_2^2}_{\text{Local Topology Reconstruction}} + \underbrace{\lambda_2 \cdot log(\sqrt{\|\boldsymbol{z}_{c,i} - \boldsymbol{E}_g\|_2^2} + e)}_{\text{Global Consistency Alignment}} \tag{19}$$

where λ_1 is a hyperparameter to balance the attribute and topology reconstruction and λ_2 is also a hyperparameter that measures the effect of the global consistency alignment. Nodes with larger scores are more likely to be considered as anomalies, thus we can compute the ranking of anomalies according to the nodes' scores calculated as (19).

D. Joint Training Objective Function

Following previous works to optimize the model in the absence of labels [38], [42], to jointly train different aspects of anomaly scoring loss with the correlation constraint for GNN guarding, the training objective function of G3AD

$$\mathcal{L} = \lambda_1 \cdot \mathcal{L}_{\text{attr}} + (1-\lambda_1) \cdot \mathcal{L}_{\text{topo}} + \lambda_2 \cdot \mathcal{L}_{\text{cons}} + \mathcal{L}_{\text{cc}} \tag{20}$$

where the hyperparameters λ_1 and λ_2 are the same as mentioned in (19). In this way, G3AD is trained using the gradient descent algorithm on the overall joint training objective function.

Conclusively, the pseudocode of the overall procedure workflow of G3AD is described as the given Algorithm 1. In each training epoch, G3AD first encodes three aspects of representations by the guarded GNN encoder along with the auxiliary encoders. Furthermore, we adopt the correlation constraint to minimize correlations between the three representations. Before conducting the anomaly detection tasks, the AC module is equipped for GNN representation guarding. Then, anomaly detection tasks are conducted for anomaly scoring and joint objective loss obtaining. A backpropagation is then executed with a gradient descent algorithm to optimize the parameters of G3AD. Finally, the anomaly scores are returned for each node to evaluate their abnormality.

E. Complexity Analysis

In this subsection, we conduct the complexity analysis of G3AD. First, the time complexity of encoding is $max(\mathcal{O}(f_a(\cdot), f_t(\cdot), f_{\text{gnn}}(\cdot)))$. The complexity of $f_a(\cdot)$ and $f_t(\cdot)$ is $\mathcal{O}(|\mathcal{V}|dF)$, and the complexity of $f_{\text{gnn}}(\cdot)$ is $\mathcal{O}((|\mathcal{V}|+|\mathcal{E}|)dF)$, where F is the summation of all feature maps across different layers. Therefore, the complexity of the encoding guarding part is $\mathcal{O}((|\mathcal{V}|+|\mathcal{E}|)dF)$. Then, the complexity of the AC$(\cdot)$ module is $\mathcal{O}(|\mathcal{V}|dF)$, and the global consistency alignment can be processed simultaneously with $\mathcal{O}(|\mathcal{V}|dF)$. Finally, the time complexity of the reconstruction modules is $max(\mathcal{O}(g_a(\cdot), g_t(\cdot)))$, where $\mathcal{O}(g_a(\cdot))$ is $\mathcal{O}(|\mathcal{E}|dF)$ and $\mathcal{O}(g_t(\cdot))$ is $\mathcal{O}(|\mathcal{V}|^2)$. To sum up, the overall time complexity of G3AD is $\mathcal{O}((|\mathcal{V}|+|\mathcal{E}|)dF + max(|\mathcal{E}|dF, |\mathcal{V}|^2))$.

Algorithm 1 Overall Procedure of G3AD

Input: An abnormal graph $\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$; Training epochs T; Balance parameters λ_1 and λ_2.
Output: An anomaly score list for the nodes.

```
for epoch ∈ 1, 2, ... T do
    Obtain the attribute-specific, topology-specific, and consistency representation H^(a) = f_a(X), H^(t) = f_t(A), H^(c) = f_gnn(X, A), respectively.
    Calculate the correlation constraint loss L_cc between any two of the three encoded representations (H^(a), H^(t), H^(c)) via Eq.(8).
    Obtain the cached attribute representation Z_A = AC(H^(a), H^(c));
    Reconstruct the attribute matrix X̂ = g_a(Z_A, A), and calculate the reconstruction loss L_attr via Eq.(14);
    Obtain the cached structure representation Z_T = AC(H^(t), H^(c));
    Reconstruct the adjacency matrix Â = g_t(Z_T), and calculate the reconstruction loss L_topo via Eq.(16);
    Conduct the global consistency alignment with consistency representation H^(c), and calculate the reconstruction loss L_cons via Eq.(18);
    Minimize the joint training loss L = λ1 · L_attr + (1 − λ1) · L_topo + λ2 · L_cons + L_cc;
    Update model's learnable parameters by using stochastic gradient descent;
end for
Compute anomaly scores of nodes in the attributed network G based on Eq.(19).
```

V. Experiments

In this section, we conduct comprehensive experiments and in-depth analysis to demonstrate the effectiveness of G3AD. Specifically, we aim to answer the following research questions: **RQ1:** How does G3AD perform compared with state-of-the-art models? **RQ2:** How much do the architecture and components of G3AD contribute? **RQ3:** Can G3AD framework positively generalize in different GNN backbones? **RQ4:** How well does G3AD disentangle encode the information in each encoder? How does G3AD perform on different types of anomalies? And how to explain the anomaly scoring effectiveness of G3AD? **RQ5:** How do key hyperparameters impact G3AD's anomaly detection performance?

A. Experimental Settings

1) Datasets: We adopt six graph anomaly detection datasets on both synthetic and real-world scenarios that have been widely used in previous research [18], [42], including four *synthetic datasets*: Cora, Citeseer, Pubmed [57], and Flickr [58], and two *real-world datasets*: Weibo [59] and Reddit [60]. The statistics are shown in Table I. The details of the datasets are introduced as follows.

TABLE I
STATISTICS OF THE EXPERIMENTAL DATASETS

Dataset	# nodes	# edges	# attributes	# anomalies
Cora	2,708	5,429	1,433	150
Citeseer	3,327	4,732	3,703	150
Pubmed	19,717	44,338	500	600
Flickr	7,575	239,738	12,407	450
Weibo	8,405	407,963	400	868
Reddit	10,000	20,744,044	64	366

1) Cora[1] [57] is a classical citation network consisting of 2708 scientific publications (contains 150 injected anomalies) along with 5429 links between them. The text contents of each publication are treated as its attributes.
2) Citeseer[1] [57] is also a citation network consisting of 3327 scientific publications (contains 150 injected anomalies) with 4732 links. The node attribute in this dataset is defined the same as in the Cora dataset.
3) Pubmed[1] [57] is another citation network consisting of 19717 scientific publications (contains 600 injected anomalies) with 44338 links. The node attributes in this dataset are also defined as in the Cora dataset.
4) Flickr[2] [58] is a social network dataset acquired from the image hosting and sharing website Flickr. In this dataset, 7575 nodes denote the users (contains 450 injected anomalies), 239738 edges represent the following relationships between users, and node attributes of users are defined by their specified tags that reflect their interests on the website.
5) Weibo[3] [59] is a user-posts-hashtag graph dataset from the Tencent-Weibo platform, which collects information from 8405 platform users (contains 868 suspicious users) with 61964 hashtags. The user-user graph provided by the author is used, which connects users who used the same hashtag.
6) Reddit[4] [60] is a user-subreddit graph extracted from a social media platform, Reddit, which consists of one month of user posts on subreddits. The 1000 most active subreddits and the 10000 most active users (containing 366 banned users) are extracted as subreddit nodes and user nodes, respectively. We convert it to a user-user graph with 20744044 connections based on the co-interacted subreddit for our experiments.

For the five synthetic datasets, we adopted synthetic anomalies to validate models [38], [42]. Following the widely used anomaly injection approach in previous advances [16], [17], [42], [43], we inject a combined set of both topological and attributed anomalies for each experimental synthetic dataset with the following manner.

1) *Injection of Topological Anomalies:* To obtain topological anomalies, the topological structure of networks is perturbed by generating small cliques composed of nodes that were originally not related. The insight is that in a small clique, a small group of nodes is significantly more interconnected with each other than the average, which can be considered a typical situation of topological anomalies in real-world graphs. Specifically, to create cliques, we begin by defining the clique size p and the number of cliques q. When generating a clique, we randomly select p nodes from the set of nodes $\mathcal{V}$ and connect them fully. This implies that all the selected p nodes are considered topological anomalies. To generate q cliques, we repeat this process q times. This results in a total of $p \times q$ topological anomalies. Following the previous works, the value of p is fixed as 15, and the value of q is set to 5, 5, 20, 20, 15 for Cora, Citeseer, Pubmed, ACM, and Flickr, respectively.
2) *Injection of Attributed Anomalies:* We inject attributed anomalies by disturbing the attributes of nodes. To generate an attributed anomaly, a node v_i is randomly selected as the target, and then another k nodes $(v_1^c, \ldots, v_k^c)$ are sampled as a candidate set $\mathcal{V}^c$. Next, we compute the Euclidean distance between the attribute vector $\mathbf{x}_c$ of each $v^c \in \mathcal{V}^c$ and the attribute vector $\mathbf{x}_i$ of v_i. We then select the node $v_j^c \in \mathcal{V}^c$ that has the largest Euclidean distance to v_i and change $\mathbf{x}_i$ to $\mathbf{x}_j^c$. Following the previous works, the value of k is set to 50 in this article.

2) Compared Baselines: We compare our proposed G3AD with 20 representative state-of-the-art models, which can be categorized into five main categories.

a) *Shallow detection methods:*
   i) *SCAN* [34] is a classic clustering method that can be applied for anomaly detection, which clusters vertices based on structural similarity to detect anomalies.
   ii) *MLPAE* [35] utilizes autoencoders on both anomalous and benign data with shallow nonlinear dimensionality reduction on the node attribute.

b) *Enhanced GNNs:*
   i) *GAAN* [36] is a generative adversarial framework with a graph encoder to obtain real graph nodes' representation and fake graph nodes' representation and a discriminator to recognize whether two connected nodes are from the real or fake graph.
   ii) *ALARM* [37] is a multiview representation learning framework with multiple graph encoders and a well-designed aggregator between them.
   iii) *AAGNN* [11] is an abnormality-aware GNN, which utilizes subtractive aggregation to represent each node as the deviation from its neighbors.

c) *Graph autoencoder-based models:*
   i) *GCNAE* [61] is a classic variational graph autoencoder with the graph convolutional network as its backbone and utilizes the reconstruction loss for unsupervised anomaly detection.
   ii) *GATAE* [24] is a graph autoencoder based on GATs with the same architecture as GCNAE.

[1]https://linqs.soe.ucsc.edu/datac
[2]http://socialcomputing.asu.edu/pages/datasets
[3]https://github.com/zhao-tong/Graph-Anomaly-Loss
[4]http://files.pushshift.io/reddit

iii) *Dominant* [38] is a deep graph autoencoder-based method with a shared encoder. It detects the anomalies by computing the weighted sum of reconstruction error terms of each node.
iv) *AnomalyDAE* [39] is a dual-graph autoencoder method based on the GAT, and the cross-modality interactions between network structure and node attribute are asymmetrically introduced on the node attribute reconstruction side.
v) *ComGA* [40] is a community-aware attributed graph anomaly detection framework with a designed tailored deep graph convolutional network.
vi) *CoCo* [41] is a correlation-enhanced graph autoencoder with correlation analysis between the local and global contexts of each node.

d) *Graph contrastive learning methods:*
i) *CoLA* [42] is a graph contrastive learning method. It detects anomalies by evaluating the agreement between each node and its neighboring subgraph sampled by the random walk-based algorithm.
ii) *ANEMONE* [43] is a multiscale graph contrastive learning method, which captures the anomaly pattern by learning the agreements between node instances at the patch and context levels concurrently.
iii) *SL-GAD* [16] is a state-of-the-art anomaly detection model with generative and multiview contrastive perspectives, which captures the anomalies from both the attribute and the structure space.
iv) *Sub-CR* [17] is a self-supervised learning method that employs the graph diffusion-based multiview contrastive learning along with attribute reconstruction.
v) *NLGAD* [6] is a multiscale graph contrastive learning network with a hybrid normality evaluation strategy.

e) *Heterophily GNNs:*
i) *MixHop* [50] is a graph convolutional network with the mixed aggregation of multihop neighbors during a single message passing operation. We construct the unsupervised autoencoder architecture for it and the rest heterophily models to fit the unsupervised graph anomaly detections.
ii) *H2GCN* [51] is a heterophily GNN by the separate encoding of ego and neighbor embeddings with higher-order neighbors and intermediate representations.
iii) *LINKX* [52] is an MLP-based model for heterophily graph modeling, which separately embeds the adjacency and node features with simple MLP operations.
iv) *GloGNN* [53] is a method that considers both the homophily and heterophily properties on the graph with the combination of both low-pass and high-pass filters over the whole node set.

3) Evaluation Metrics and Hyperparameter Settings: We evaluate the models with area under the ROC curve (ROC-AUC) and average precision (AP), two widely adopted metrics in previous works [7], [38], [43], to comprehensively evaluate the anomaly detection performance in different aspects. A higher AUC and AP value indicates better detection performance. Note that we run all experiments five times with different random seeds and report the average results with standard deviation to prevent extreme cases.

4) Hyperparameter Settings: In the experiments on different datasets, the embedding size is fixed to 64, and the embedding parameters are initialized with the Xavier method [62]. The loss function is optimized with the Adam optimizer [63]. The learning rate of G3AD is searched from $[5\times10^{\{-2,-3,-4\}}, 2\times10^{\{-2,-3,-4\}}, 1\times10^{\{-2,-3,-4\}}]$. For the balance parameter of G3AD, we first tune λ_1 from [0, 1], then once λ_1 is determined, we select an appropriate λ_2 based on the fixed λ_1. For fair comparisons, all experiments are conducted on the CentOS system equipped with NVIDIA RTX-3090 GPUs.

B. Anomaly Detection Performance (RQ1)

We first compare the main performance results between G3AD and the baseline models. The performance comparison results on synthetic and real-world datasets are reported in Tables II and III, respectively. From these results, we have the following observations.

G3AD can achieve significant performance improvements over state-of-the-art methods. Specifically, for AUC metrics, G3AD achieves the best performance on all synthetic and real-world datasets among all baselines. For AP metrics, G3AD can also generally perform well and have the best or second-best performance on half of the datasets. Overall, G3AD has the top rank among all baselines on average in two metrics. The superior performance verifies the guarding schemes with correlation constraints, and the AC under the local and global anomaly detection tasks can help improve anomaly distinguishability.

Directly using the consistency GNNs or heterophily GNNs has suboptimal detection performance. We can find from the results that the performance of enhanced consistency-based GNNs (the second category) and heterophily-based GNNs (the fifth category) both have a certain gap with G3AD. Thus, it further verifies that, under such a disrupted phenomenon induced by unknown anomalies, G3AD can provide an effective way to guard the consistent homophily and utilize discrepant heterophily patterns.

There are differences in the performance of baselines between synthetic and real-world datasets. It can be found from the table that the graph contrastive learning-based methods are the best-performed baselines on synthetic datasets in general, while enhanced GNNs and graph autoencoders achieve better results on real-world datasets. One possible reason is that the contrastive objective function of these models is related onefold, which is insufficient for directly applying to the irregular real-world datasets, and then enhanced GNNs and graph autoencoders.

Anomaly detection on real-world datasets is significantly harder than on synthetic datasets. The performance gap in real-world datasets between the models is large compared

TABLE II

OVERALL UNSUPERVISED ANOMALY DETECTION COMPARISON RESULTS ON SYNTHETIC ANOMALY DATASETS (MEAN ± STANDARD DEVIATION IN PERCENTAGE OVER FIVE TRIAL RUNS). THE BEST AND SECOND-BEST RESULTS IN EACH COLUMN ARE HIGHLIGHTED IN BOLD FONT AND UNDERLINED, RESPECTIVELY

Model	Synthetic Datasets							
	Cora		Citeseer		Pubmed		Flickr	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP
SCAN	0.6614±0.0140	0.0859±0.0044	0.6764±0.0063	0.0731±0.0032	0.7342±0.0037	0.1334±0.0032	0.6503±0.0120	0.3035±0.0227
MLPAE	0.7560±0.0101	0.3528±0.0188	0.7404±0.0131	0.3124±0.0169	0.7477±0.0066	0.2508±0.0103	0.7466±0.0041	0.3484±0.0087
GAAN	0.7917±0.0118	0.3271±0.0124	0.8066±0.0036	0.3495±0.0101	0.7842±0.0042	0.0973±0.0018	0.7463±0.0043	0.3552±0.0119
ALARM	0.8271±0.0223	0.2503±0.0379	0.8325±0.0121	0.3027±0.0559	0.8287±0.0044	0.1059±0.0020	0.6086±0.0034	0.0726±0.0013
AAGNN	0.7590±0.0056	0.3744±0.0171	0.7202±0.0140	0.2345±0.0303	0.6693±0.0377	0.1210±0.0283	0.7454±0.0033	0.3506±0.0087
GCNAE	0.7959±0.0104	0.3544±0.0350	0.7678±0.0114	0.3321±0.0243	0.7933±0.0081	0.2956±0.0083	0.7471±0.0056	0.3359±0.0125
GATAE	0.8479±0.0098	0.3894±0.0239	0.8233±0.0126	0.2962±0.0219	0.8440±0.0061	0.3777±0.0103	0.7489±0.0050	0.3578±0.0111
Dominant	0.8773±0.0134	0.3090±0.0438	0.8523±0.0051	0.3999±0.0092	0.8516±0.0034	0.1177±0.0019	0.6129±0.0035	0.0734±0.0013
AnomalyDAE	0.8594±0.0068	0.3586±0.0148	0.8092±0.0059	0.3487±0.0103	0.7838±0.0042	0.0960±0.0016	0.7418±0.0051	0.3635±0.0107
ComGA	0.7382±0.0162	0.1539±0.0242	0.7004±0.0150	0.0921±0.0085	0.7161±0.0065	0.0630±0.0015	0.6658±0.0033	0.1896±0.0120
CoCo	0.8642±0.0109	0.3938±0.0201	0.8336±0.0406	0.3175±0.0292	0.8053±0.0291	0.1826±0.0425	0.7590±0.0078	**0.3683±0.0105**
CoLA	0.8866±0.0091	0.4638±0.0503	0.8112±0.0314	0.2355±0.0175	0.9295±0.0047	0.4344±0.0134	0.5612±0.0224	0.0694±0.0047
ANEMONE	0.8997±0.0048	**0.5417±0.0290**	0.8444±0.0178	0.3244±0.0361	0.9261±0.0080	0.4311±0.0313	0.5579±0.0271	0.0712±0.0062
SL-GAD	0.8159±0.0238	0.3361±0.0337	0.7287±0.0201	0.2122±0.0273	0.8732±0.0066	0.3906±0.0313	0.7240±0.0087	0.3146±0.0123
Sub-CR	0.8968±0.0118	0.4771±0.0102	0.9060±0.0095	0.4751±0.0254	0.9251±0.0041	**0.4936±0.0137**	0.7423±0.0038	0.3611±0.0145
NLGAD	0.7369±0.0233	0.2308±0.0225	0.7559±0.0220	0.2177±0.0380	0.7331±0.0141	0.2862±0.0323	0.5045±0.0120	0.0676±0.0085
MixHop	0.7796±0.0107	0.2412±0.0300	0.7401±0.0122	0.3146±0.0195	0.7726±0.0088	0.2212±0.0126	0.7447±0.0060	0.3517±0.0229
H2GCN	0.7827±0.0104	0.3479±0.0285	0.7361±0.0169	0.3064±0.0139	0.7658±0.0065	0.2340±0.0037	0.7463±0.0042	0.3526±0.0135
LINKX	0.7601±0.0098	0.3605±0.0242	0.7416±0.0117	0.3187±0.0180	0.7543±0.0070	0.2526±0.0103	0.7466±0.0042	0.3636±0.0065
GloGNN	0.7563±0.0083	0.3470±0.0269	0.7419±0.0124	0.3214±0.0220	0.7479±0.0067	0.2511±0.0103	0.7430±0.0013	0.3583±0.0054
G3AD (ours)	**0.9687±0.0013**	0.4990±0.0191	**0.9688±0.0049**	**0.5357±0.0115**	**0.9321±0.0013**	0.2179±0.0072	**0.7693±0.0070**	0.3270±0.0100

TABLE III

OVERALL UNSUPERVISED ANOMALY DETECTION COMPARISON RESULTS ON REAL-WORLD ANOMALY DATASETS

Model	Real-world Datasets			
	Weibo		Reddit	
	AUC	AP	AUC	AP
SCAN	0.7011±0.0000	0.1855±0.0000	0.4978±0.0000	0.0364±0.0000
MLPAE	0.8946±0.0028	0.6696±0.0102	0.5108±0.0310	0.0359±0.0029
GAAN	0.9249±0.0000	**0.8104±0.0000**	0.5683±0.0001	0.0493±0.0001
ALARM	0.9226±0.0000	0.8071±0.0000	0.5644±0.0003	0.0466±0.0001
AAGNN	0.8066±0.0027	0.6679±0.0009	0.5442±0.0299	0.0400±0.0030
GCNAE	0.8449±0.0032	0.5650±0.0030	0.5037±0.0015	0.0346±0.0001
GATAE	0.9361±0.0079	0.7848±0.0361	0.5643±0.0212	0.0448±0.0030
Dominant	0.8423±0.0117	0.6163±0.0237	0.5752±0.0056	0.0570±0.0019
AnomalyDAE	0.8881±0.0165	0.6681±0.0874	0.4315±0.0001	0.0319±0.0000
ComGA	0.9248±0.0006	0.8097±0.0009	0.4317±0.0001	0.0320±0.0001
CoCo	0.9257±0.0005	0.8095±0.0023	0.5388±0.0510	0.0403±0.0060
CoLA	0.4842±0.0238	0.1006±0.0155	0.5149±0.0233	0.0393±0.0020
ANEMONE	0.3607±0.0120	0.0863±0.0147	0.4952±0.0188	0.0387±0.0026
SL-GAD	0.4298±0.0073	0.0899±0.0036	0.5488±0.0142	0.0424±0.0032
Sub-CR	0.6404±0.0070	0.4900±0.0103	0.5327±0.0156	0.0379±0.0013
NLGAD	0.3259±0.0026	0.0792±0.0027	0.5631±0.0095	0.0433±0.0009
MixHop	0.8612±0.0018	0.6278±0.0061	0.5400±0.0175	0.0398±0.0028
H2GCN	0.8546±0.0020	0.5604±0.0088	0.5476±0.0113	0.0401±0.0010
LINKX	0.8018±0.0094	0.2822±0.0088	0.5576±0.0162	0.0421±0.0020
GloGNN	0.9128±0.0251	0.7465±0.1280	0.5436±0.0364	0.0468±0.0085
G3AD (ours)	**0.9514±0.0135**	0.7434±0.0687	**0.6207±0.0022**	**0.0600±0.0008**

TABLE IV

COMPONENT ABLATION STUDY RESULTS ON G3AD

Variant	Flickr	Weibo	Reddit
G3AD (ours)	**0.7693±0.0070**	**0.9514±0.0135**	**0.6207±0.0022**
G3AD-*w/o AR*	0.5727±0.0047	0.7935±0.0887	0.6199±0.0007
G3AD-*w/o TR*	0.7466±0.0041	0.9258±0.0009	0.5136±0.0105
G3AD-*w/o CA*	0.7686±0.0074	0.9460±0.0108	0.6104±0.0186
G3AD-*w/o AR&CA*	0.5722±0.0048	0.8307±0.1005	0.6199±0.0006
G3AD-*w/o TR&CA*	0.7466±0.0040	0.9254±0.0008	0.5415±0.0321
G3AD-*w/o AR&TR*	0.5533±0.0222	0.8058±0.0262	0.4218±0.0139

to synthetic datasets, especially the graph contrastive learning methods. This shows that the widely used anomaly injection scheme may lack diversity and be inadequate to simulate irregular patterns in the real world, [64] also has the same observation. Thus, we suggest that the model effectiveness examination should be conducted on both synthetic and real-world datasets for a more objective evaluation.

C. Ablation Study (RQ2)

In this subsection, we aim to conduct a fine-grained ablation study to analyze the contribution of components and architecture of our G3AD.

1) Component Study: To verify the effectiveness of the components, we conduct an ablation study on three variants: G3AD-*w/o AR*, G3AD-*w/o TR*, G3AD-*w/o CA*, and their combinations, which remove the local attribute reconstruction, local topology reconstruction, and global consistency alignment, respectively. From the results in Table IV, we can observe that G3AD notably outperforms all three variants and three ablation combinations in general. Specifically, first, the removal of attribute reconstruction and topology reconstruction has a greater impact, which shows that the modeling of graph properties is important for anomaly discrimination. Therefore, well-representing attributes and topology under anomaly discrepancy are significant for anomaly detection. Second, the effect of different modules varies from dataset, which is related to the diversity of anomaly definitions in

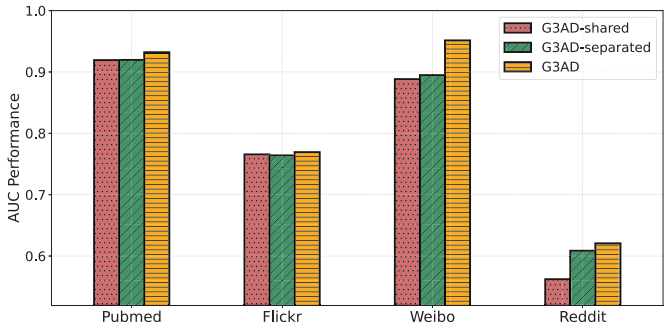

Fig. 4. Architecture ablation study results on G3AD.

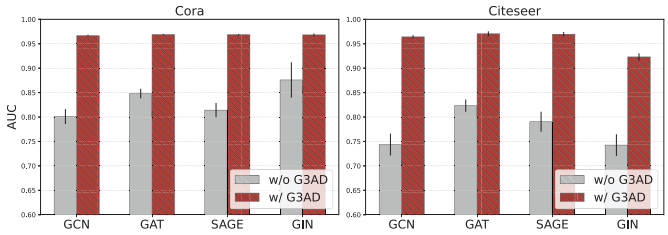

Fig. 5. Generalization study results of G3AD in different GNN backbones. (a) Cora. (b) Citeseer.

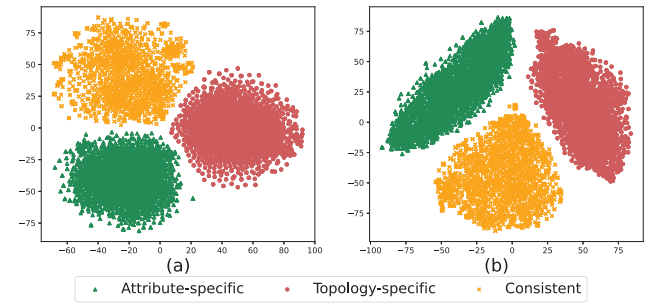

Fig. 6. Visualization of the distribution of three aspects of embeddings after G3AD encoding guarding. (a) Cora. (b) Citeseer.

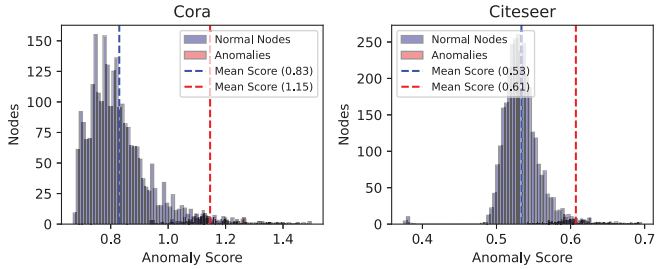

Fig. 7. Posthoc analysis of anomaly score distribution predicted by G3AD.

different scenarios. Thus, the comprehensive consideration of topology-specific, attribute-specific, and consistent patterns is beneficial to the model.

2) Architecture Study: To study the effectiveness of G3AD's architecture, we further study the guarding architecture of G3AD with its two architecture variants: 1) *G3AD-shared* removes the auxiliary encoders and shares all input information with a single GNN, which ignores the guarding of anomalies for the GNN, and 2) *G3AD-separated* replaces the three encoders with two parallel encoders without any information sharing, which is the simplest way without considering the consistent attribute-topology correlation that remained in the major normal nodes. The results are shown in Fig. 4. From the results, we have the following observations. First, compared to G3AD-*shared* variant, G3AD gains significant improvements, which proves the necessity of auxiliary guarding under abnormal graphs. Second, the improvements over G3AD-*separated* demonstrate that simply separating attribute and topology information will miss the consistent patterns that remained in most normal nodes and lead to suboptimal performance. Therefore, the guarding architecture of G3AD is effective for unsupervised anomaly detection.

D. Generalization Study (RQ3)

To investigate the generalization ability of the proposed G3AD framework in different GNN backbones, we evaluate the improvements of G3AD (w/ G3AD) for different GNN backbones (w/o G3AD). Specifically, we adopt the representative GCN [20], GAT [24], SAGE [21], and GIN [54] for G3AD in the GNN encoder, respectively.

The generalization study results are illustrated in Fig. 5. From the results, we can find that G3AD can stably bring significant improvements across each GNN backbone. Specifically, equipped with our G3AD framework, the GNNs have 16.12% and 23.62% average AUC improvements on the Cora and Citeseer datasets, respectively. Furthermore, the improvements are evident across all GNN backbones. It has achieved at least a 10% gain on each GNN, with an average improvement of 25.16% specifically on GCNs. These results highlight the generalization capability of G3AD in improving various GNN backbones across different datasets.

E. Case Study (RQ4)

In this subsection, we aim to conduct case studies on G3AD with synthetic datasets to analyze the correlation-constrained GNN encoding guarding via embedding visualization and analyze the performance of G3AD on different types of anomalies.

1) Embedding Visualization: We first conduct the distribution visualization of attribute-specific, topology-specific, and consistent GNN embeddings after the correlation constrained guarding from the attribute encoder $f_a(\cdot)$, the topology encoder $f_t(\cdot)$, and the GNN encoder $f_{\text{gnn}}(\cdot)$ by reducing their dimension to two with the t-SNE [65] method. The visualization results are illustrated in Fig. 6, which shows that the three aspects of representations are well distinguished from each other in the embedding space. Therefore, it illustrates that the three aspects of representations have been well disentangled by G3AD to guard the GNN encoding.

2) Performance on Different Anomaly Types: To further analyze different types of induced anomalies, we conduct a case study of G3AD compared with the overall top-3 performed baselines on two synthetic datasets. We report the detection performance on synthetic attributed anomalies, synthetic structure anomalies, and synthetic mixed (both attribute and structure perspectives) anomalies. The split performance results are shown in Table V. From the results, we find that the best baseline Sub-CR obviously tends to detect

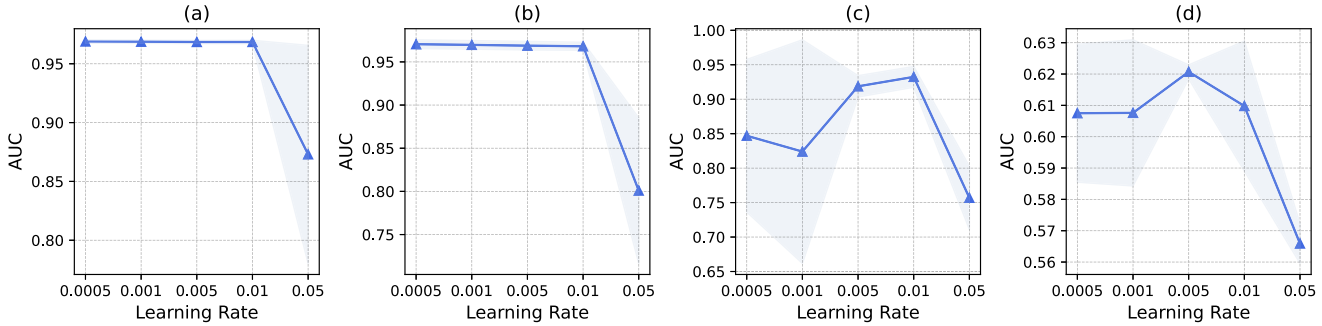

Fig. 8. Parameters study on G3AD with different learning rates. (a) Cora. (b) Citeseer. (c) Weibo. (d) Reddit.

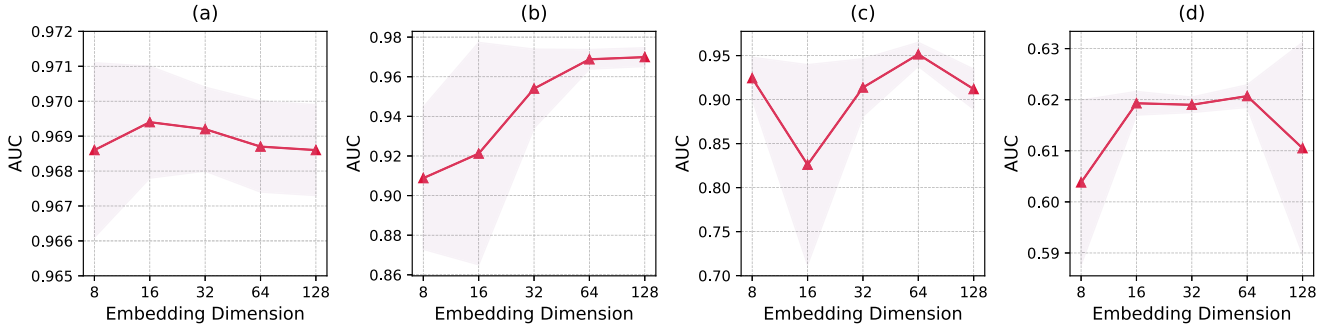

Fig. 9. Parameters study on G3AD with different embedding dimensions. (a) Cora. (b) Citeseer. (c) Weibo. (d) Reddit.

TABLE V
CASE STUDY OF G3AD ON ATTRIBUTE-INDUCED, TOPOLOGY-INDUCED, AND MIXED ANOMALY DETECTION PERFORMANCE WITH TOP-3 PERFORMED BASELINES

	Anomaly	Attribute	Topology	Mixed
Cora	CoLA	0.8458±0.0234	0.9273±0.0217	0.8866±0.0091
	ANEMONE	0.8305±0.0151	0.9688±0.0086	0.8997±0.0048
	Sub-CR	**0.9806**±0.0042	0.8240±0.0328	0.8968±0.0118
	G3AD (ours)	0.9623±0.0023	**0.9752±0.0013**	**0.9687±0.0013**
Citeseer	CoLA	0.7437±0.0267	0.8785±0.0426	0.8112±0.0314
	ANEMONE	0.7519±0.0220	0.9368±0.0152	0.8444±0.0178
	Sub-CR	0.9510±0.0099	0.8733±0.0358	0.9060±0.0095
	G3AD (ours)	**0.9637±0.0037**	**0.9740±0.0073**	**0.9688±0.0049**

attribute anomalies, and the second-best performed model, ANEMONE, has an inclination toward detecting topology anomalies. Our G3AD can generally balance and achieve good performance on all types of anomalies due to the explicit guarding of representation learning, guarding of directly abnormal graph reconstruction, and comprehensive consideration of different kinds of anomaly characteristics.

3) Anomaly Score Distribution: To provide an explanation foundation for G3AD anomaly detection, we conduct a post hoc analysis by visualizing the anomaly score distribution predicted by G3AD. This distribution is crucial for both the gradient descent training phase and the anomaly discrimination process of the model. The visualization results are depicted in Fig. 7 for nodes in the Cora and Citeseer datasets. In the figure, we differentiate between normal nodes and anomalies using distinct colors: blue for normal nodes and red for anomalies. Additionally, we indicate the mean anomaly scores for both normal nodes and anomalies with dashed lines. From the results, the distinct separation between the normal and anomaly node distributions in both datasets shows the effectiveness of G3AD in anomaly detection. The clear distinction in anomaly scores allows for accurate identification and discrimination of anomalies.

F. Parameter Analysis (RQ5)

In this subsection, we aim to study the impact of different hyperparameters on G3AD with two synthetic datasets and two real-world datasets, including the learning rate, the embedding dimension, the balanced parameters λ_1 and λ_2, and the readout function in (17).

1) Effectiveness of Learning Rate: Then, the study of the effectiveness of the learning rate can be found in Fig. 8. From the results, we can find that the performance of G3AD on synthetic datasets is less sensitive to changes in learning rate. As long as the learning rate is not set too large, the model's performance is similar under relatively small learning rates. For real-world datasets, the appropriate learning rate range will be smaller than that of synthetic datasets, but a learning rate value of around 0.005 can generally achieve relatively good anomaly detection performance.

2) Effectiveness of Embedding Dimension: Furthermore, we have also analyzed the impact of different embedding dimensions on G3AD, as illustrated in Fig. 9. It can be seen from the study results that, except for the Cora dataset, which can achieve optimal performance on a smaller 16-D space, the other three datasets can achieve optimal performance on a 64-D representation space in general, which is no need for us to continue to increase the representation dimension to achieve better performance.

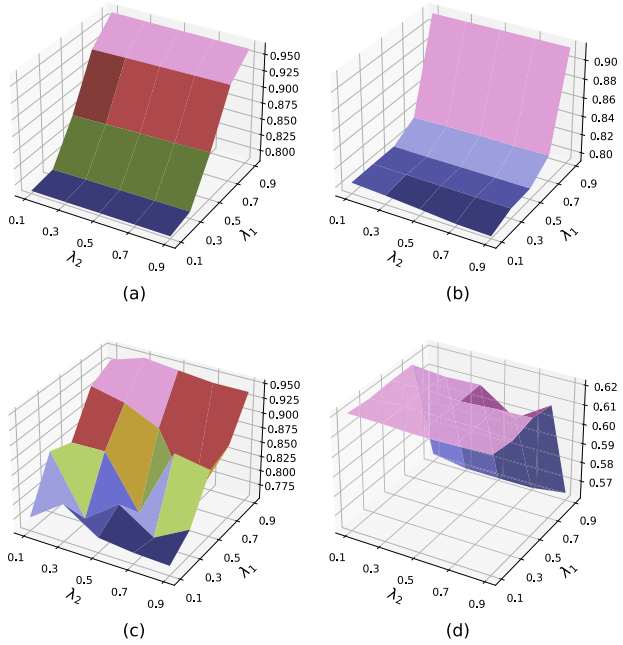

Fig. 10. Parameters study results with different combinations of λ_1 and λ_2. (a) Cora. (b) Citeseer. (c) Weibo. (d) Reddit.

TABLE VI
COMPARISON RESULTS ON DIFFERENT READOUT FUNCTIONS IN THE GLOBAL CONSISTENCY ALIGNMENT

Readout	Flickr	Weibo	Reddit
Mean	**0.7693±0.0070**	**0.9514±0.0135**	0.6207±0.0022
Min	0.7679±0.0084	0.9227±0.0422	**0.6212±0.0017**
Max	0.7682±0.0085	0.9075±0.0947	0.6194±0.0011
Attention	0.7684±0.0084	0.8664±0.1089	0.6186±0.0023

3) Effectiveness of Balanced Parameters: We first investigate the effect of the different combinations of key balanced parameters λ_1 (balances the effect between topology and attribute reconstruction) and λ_2 (tunes the impact of consistency alignment) on G3AD. The parameter study results are shown in Fig. 10 on the Cora, Citeseer, Weibo, and Reddit datasets. From the results, we can observe that the performance of G3AD varies with respect to balanced parameters λ_1 and λ_2. More specifically, first, changes in parameter λ_1 may bring more volatility than parameter λ_2. This is explainable since the consistency is related to both attribute and topology, which is more robust and stable [66]. Furthermore, with a fixed λ_2 value, a relatively large value of λ_1 can make G3AD perform better, which means that the impact of the attribute reconstruction needs more attention and consideration.

4) Effectiveness of Readout Function: We further discuss the readout function selection of consistency alignment in (17), which includes mean, min, max, and attention (scoring by an MLP) pooling as in Table VI. From the results, we can find that the simple nonparameter operation with mean pooling can achieve relatively good performance. The possible reason is that due to the unbalanced number of normal and anomalies, simple mean pooling can reflect the global consistency from the representations of the majority of normal nodes.

VI. CONCLUSION

In this article, we study the problem of negative anomaly impact on GNNs in unsupervised graph anomaly detection, which is largely neglected by previous works. To address this issue, we propose G3AD, a simple but effective framework to guard GNNs against anomaly impacts in unsupervised settings. Specifically, G3AD introduces two guarding strategies with comprehensive anomaly detection perspectives. First, G3AD guards the GNN encoder against encoding inconsistent information to enhance the node representation quality for anomaly distinguishing. Then, we comprehensively include both local reconstruction and global alignment as the objectives for better detecting multiple anomalies. During this process, to guard the GNN-encoded representations against directly reconstructing the abnormal graph, G3AD further equips the representations with AC. Extensive experiments demonstrate that G3AD outperforms the state-of-the-art baselines.

REFERENCES

[1] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining Knowl. Discov.*, vol. 29, no. 3, pp. 626–688, 2015.
[2] T. Zhao, T. Jiang, N. Shah, and M. Jiang, "A synergistic approach for graph anomaly detection with pattern mining and feature learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2393–2405, Jun. 2022.
[3] X. Ma et al., "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12012–12038, Dec. 2021.
[4] S. Fakhraei, J. Foulds, M. Shashanka, and L. Getoor, "Collective spammer detection in evolving multi-relational social networks," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 1769–1778.
[5] D. Wang et al., "A semi-supervised graph attentive network for financial fraud detection," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Beijing, China, Nov. 2019, pp. 598–607.
[6] J. Duan et al., "Normality learning-based graph anomaly detection via multi-scale contrastive learning," in *Proc. 31st ACM Int. Conf. Multimedia*, Oct. 2023, pp. 7502–7511.
[7] Y. Bei et al., "Reinforcement neighborhood selection for unsupervised graph anomaly detection," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2023, pp. 11–20.
[8] T. Kim, S. C. Suh, H. Kim, J. Kim, and J. Kim, "An encoding technique for CNN-based network anomaly detection," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Jun. 2018, pp. 2960–2965.
[9] S. Carta, A. S. Podda, D. R. Recupero, and R. Saia, "A local feature engineering strategy to improve network anomaly detection," *Future Internet*, vol. 12, no. 10, p. 177, Oct. 2020.
[10] T. Zhao, C. Deng, K. Yu, T. Jiang, D. Wang, and M. Jiang, "GNN-based graph anomaly detection with graph anomaly loss," in *Proc. 2nd Int. Workshop Deep Learn. Graphs: Methods Appl.*, 2020, pp. 1–7.
[11] S. Zhou, Q. Tan, Z. Xu, X. Huang, and F.-L. Chung, "Subtractive aggregation for attributed network anomaly detection," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 3672–3676.
[12] X. Zhou et al., "Reconstructed graph neural network with knowledge distillation for lightweight anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 11817–11828, Sep. 2024.
[13] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annu. Rev. Sociol.*, vol. 27, no. 1, pp. 415–444, Aug. 2001.
[14] Y. Ma, X. Liu, N. Shah, and J. Tang, "Is homophily a necessity for graph neural networks?," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2021.
[15] L. Yang et al., "Graph neural networks beyond compromise between attribute and topology," in *Proc. ACM Web Conf.*, Apr. 2022, pp. 1127–1135.
[16] Y. Zheng, M. Jin, Y. Liu, L. Chi, K. T. Phan, and Y. P. Chen, "Generative and contrastive self-supervised learning for graph anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12220–12233, Dec. 2021.

[17] J. Zhang, S. Wang, and S. Chen, "Reconstruction enhanced multi-view contrastive learning for anomaly detection on attributed networks," in *Proc. 31st Int. Joint Conf. Artif. Intell. (IJCAI)*, 2022, pp. 2376–2382.

[18] Z. Chai et al., "Can abnormality be detected by graph neural networks?," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Vienna, Austria, Jul. 2022, pp. 1945–1951.

[19] J. Tang, J. Li, Z. Gao, and J. Li, "Rethinking graph neural networks for anomaly detection," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2022, pp. 21076–21089.

[20] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2016.

[21] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2017, pp. 1025–1035.

[22] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 249–270, Mar. 2020.

[23] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Mar. 2020.

[24] P. V. ković, G. Cucurull, A. Casanova, A. Romero, P. Lió, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2017.

[25] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100379.

[26] S. Kumar, A. Mallik, A. Khetarpal, and B. S. Panda, "Influence maximization in social networks using graph embedding and graph neural network," *Inf. Sci.*, vol. 607, pp. 1617–1636, Aug. 2022.

[27] L. Jain, R. Katarya, and S. Sachdeva, "Opinion leaders for information diffusion using graph neural network in online social networks," *ACM Trans. Web*, vol. 17, no. 2, pp. 1–37, May 2023.

[28] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 639–648.

[29] H. Chen et al., "Macro graph neural networks for online billion-scale recommender systems," in *Proc. ACM Web Conf.*, May 2024, pp. 3598–3608.

[30] X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang, "Graph neural networks and their current applications in bioinformatics," *Frontiers Genet.*, vol. 12, Jul. 2021, Art. no. 690049.

[31] H.-C. Yi, Z.-H. You, D.-S. Huang, and C. K. Kwoh, "Graph representation learning in bioinformatics: Trends, methods and applications," *Briefings Bioinf.*, vol. 23, no. 1, Jan. 2022, Art. no. bbab340.

[32] R. Yu, H. Qiu, Z. Wen, C. Lin, and Y. Liu, "A survey on social media anomaly detection," *ACM SIGKDD Explorations Newslett.*, vol. 18, no. 1, pp. 1–14, 2016.

[33] Y. Deldjoo, T. D. Noia, and F. A. Merra, "A survey on adversarial recommender systems: From attack/defense strategies to generative adversarial networks," *ACM Comput. Surveys*, vol. 54, no. 2, pp. 1–38, Mar. 2021.

[34] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger, "SCAN: A structural clustering algorithm for networks," in *Proc. ACM Int. Conf. Knowl. Discov. Data Min. (KDD)*, 2007, pp. 824–833.

[35] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proc. MLSDA 2nd Workshop Mach. Learn. Sensory Data Anal.*, Dec. 2014, pp. 4–11.

[36] Z. Chen, B. Liu, M. Wang, P. Dai, J. Lv, and L. Bo, "Generative adversarial attributed network anomaly detection," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manag.*, Oct. 2020, pp. 1989–1992.

[37] Z. Peng, M. Luo, J. Li, L. Xue, and Q. Zheng, "A deep multi-view framework for anomaly detection on attributed networks," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 6, pp. 2539–2552, Jun. 2022.

[38] K. Ding, J. Li, R. Bhanushali, and H. Liu, "Deep anomaly detection on attributed networks," in *Proc. SIAM Int. Conf. Data Mining*. Philadelphia, PA, USA: SIAM, 2019, pp. 594–602.

[39] H. Fan, F. Zhang, and Z. Li, "Anomalydae: Dual autoencoder for anomaly detection on attributed networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 5685–5689.

[40] X. Luo et al., "ComGA: Community-aware attributed graph anomaly detection," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, Feb. 2022, pp. 657–665.

[41] R. Wang et al., "Context correlation discrepancy analysis for graph anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 37, no. 1, pp. 174–187, Jan. 2025.

[42] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly detection on attributed networks via contrastive self-supervised learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2378–2392, Jun. 2022.

[43] M. Jin, Y. Liu, Y. Zheng, L. Chi, Y.-F. Li, and S. Pan, "ANEMONE: Graph anomaly detection with multi-scale contrastive learning," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 3122–3126.

[44] J. Duan, B. Xiao, S. Wang, H. Zhou, and X. Liu, "ARISE: Graph anomaly detection on attributed networks via substructure awareness," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 12, pp. 18172–18185, Dec. 2024.

[45] T. Wang, D. Jin, R. Wang, D. He, and Y. Huang, "Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily," in *Proc. AAAI Conf. Artif. Intell.*, Jun. 2022, pp. 4210–4218.

[46] H. Wang and J. Leskovec, "Combining graph convolutional neural networks and label propagation," *ACM Trans. Inf. Syst.*, vol. 40, no. 4, pp. 1–27, Oct. 2022.

[47] J. Zhu et al., "Graph neural networks with heterophily," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 12, pp. 11168–11176.

[48] L. Wu, H. Lin, Z. Liu, Z. Liu, Y. Huang, and S. Z. Li, "Homophily-enhanced self-supervision for graph structure learning: Insights and directions," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 12358–12372, Sep. 2023.

[49] J. Chen, S. Chen, J. Gao, Z. Huang, J. Zhang, and J. Pu, "Exploiting neighbor effect: Conv-agnostic gnn framework for graphs with heterophily," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 10, pp. 13383–13396, Oct. 2023.

[50] S. Abu-El-Haija et al., "MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 21–29.

[51] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2020, pp. 7793–7804.

[52] D. Lim et al., "Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2021, pp. 20887–20902.

[53] X. Li et al., "Finding global homophily in graph neural networks when meeting heterophily," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2022, pp. 13242–13256.

[54] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks," in *Proc. Int. Conf. Learn. Represent.*, Oct. 2018.

[55] A. G. Asuero, A. Sayago, and A. González, "The correlation coefficient: An overview," *Crit. Rev. Anal. Chem.*, vol. 36, no. 1, pp. 41–59, 2006.

[56] A. Roy et al., "GAD-NR: Graph anomaly detection via neighborhood reconstruction," in *Proc. 17th ACM Int. Conf. Web Search Data Mining*, Mar. 2024, pp. 576–585.

[57] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, p. 93, Sep. 2008.

[58] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jun. 2009, pp. 817–826.

[59] T. Zhao, C. Deng, K. Yu, T. Jiang, D. Wang, and M. Jiang, "Error-bounded graph anomaly loss for GNNs," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 1873–1882.

[60] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1269–1278.

[61] T. Kipf and M. Welling, "Variational graph auto-encoders," in *Proc. NIPS Workshop Bayesian Deep Learn.*, Jan. 2016.

[62] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[63] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[64] K. Liu et al., "BOND: Benchmarking unsupervised outlier node detection on static attributed graphs," in *Proc. 36th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track*, Jan. 2022, pp. 27021–27035.

[65] L. Van der Maaten and G. E. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, Jan. 2008.

[66] J. Zhu, J. Jin, D. Loveland, M. T. Schaub, and D. Koutra, "How does heterophily impact the robustness of graph neural networks? Theoretical connections and practical implications," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 2637–2647.