Counterfactual Graph Learning for Anomaly Detection on Attributed Networks

Chunjing Xiao, Xovee Xu, *Graduate Student Member, IEEE*, Yue Lei, Kunpeng Zhang, Siyuan Liu, and Fan Zhou

***Abstract*—Graph anomaly detection is attracting remarkable multidisciplinary research interests ranging from finance, healthcare, and social network analysis. Recent advances on graph neural networks have substantially improved the detection performance via semi-supervised representation learning. However, prior work suggests that deep graph-based methods tend to learn spurious correlations. As a result, they fail to generalize beyond training data distribution. In this article, we aim to identify structural and contextual anomaly nodes in an attributed graph. Based on our preliminary data analyses, spurious correlations can be eliminated with causal subgraph interventions. Therefore, we propose a new graph-based anomaly detection model that can learn causal relations for anomaly detection while generalizing to new environments. To handle situations with varying environments, we steer the generative model to manufacture synthetic environment features, which are exerted on realistic subgraphs to generate counterfactual subgraphs. Further, these counterfactual subgraphs help a few-shot anomaly detection model learn transferable and causal relations across different environments. The experiments on three real-world attributed graphs show that the proposed approach achieves the best performance compared to the state-of-the-art baselines and learns robust causal representations resistant to noises and spurious correlations.**

***Index Terms*—Causal inference, causal representation, counterfactual, network anomaly detection, representation learning.**

I. Introduction

ANOMALY detection on an attributed network – identifying abnormal nodes in a network where node attributes are provided, is an important task in both academia and industry. Anomaly nodes in the network are considered as data objects deviating dramatically from the majority regarding structures and/or properties. Anomaly detection has a wide range of applications from detecting network attacks in cybersecurity, inspecting fraudulent transactions in finance, recognizing malicious senders in email networks, and to investigating diseases in healthcare [1]. Recently, remarkable improvements have been achieved by taking advantage of different deep learning techniques, such as Transformers [2], autoencoders [3], GANs [4] and few-show learning [5]. Despite the improvements achieved, existing methods still struggle in generalization beyond training data distribution. As a result, a well-trained model might suffer from performance degradation when applied to newly observed nodes with different environments [6].

The performance degradation can be attributed to data biases [7] and shortcut learning [8], which means that deep learning models are prone to learn dataset-dependent spurious correlations based on statistical associations [9]. These characteristics become problematic when the distribution of test data is different from training data. For example, for network anomaly detection tasks, the test samples can be newly observed nodes or those whose neighboring nodes and edges change over time. Their environments (e.g., subgraph topology and types of neighboring nodes) might be essentially different from the training samples. The model, which is trained by learning dataset-dependent correlations on training data, may not be able to obtain expected performance on test data.

Existing work has shown that, besides statistical associations between variables, learning causal relations can efficiently alleviate this performance degradation problem in the field of image processing [10], [11], [12]. Causal relations reflect the fundamental data generating mechanism, which tends to be universal and invariant across different environments [13], and provides the most transferable and confident information to unseen environments. Learning a representation exposing causal relations, which may confront different environmental changes and interventions, can make the model more robust and generalized. However, inferring causality between data variables remains extremely difficult or even impossible [14].

In this article, we take a first attempt towards causal graph learning and propose a **C**ounter**F**actual graph **A**nomaly **D**etection (**CFAD**) model, which tries to learn causal relations to train a robust model for detecting the anomalies on attributed networks. Based on the concept of Structural Causal

Manuscript received 13 July 2022; revised 19 December 2022; accepted 23 February 2023. Date of publication 28 February 2023; date of current version 15 September 2023. This work was supported in part by the National Natural Science Foundation of China under Grants 62072077 and 62176043, and in part by the Natural Science Foundation of Sichuan Province, China under Grant 2022NSFSC0505. Recommended for acceptance by Y. Tong. *(Chunjing Xiao and Xovee Xu contributed equally to this work.) (Corresponding author: Fan Zhou.)*

Chunjing Xiao and Yue Lei are with the School of Computer and Information Engineering, Henan University, Kaifeng, Henan 475001, China, and also with the Henan Key Laboratory of Big Data Analysis and Processing, Henan University, Kaifeng, Henan 475004, China (e-mail: ChunjingXiao@gmail.com; leiyue971109@gmail.com).

Xovee Xu and Fan Zhou are with the University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, China (e-mail: xovee.xu@gmail.com; fan.zhou@uestc.edu.cn).

Kunpeng Zhang is with the University of Maryland, College Park, MD 20742 USA (e-mail: kpzhang@umd.edu).

Siyuan Liu is with the Department of Supply Chain & Information Systems, Pennsylvania State University, State College, PA 16802 USA (e-mail: siyuan@psu.edu).

Digital Object Identifier 10.1109/TKDE.2023.3250523

Models [15], [16], we interpret the generation of a node's representation by graph neural networks (GNNs) as a causal process. The research of causal explanations for GNNs [17] and our preliminary data analyses suggest that spurious correlations can be eliminated by splitting the neighbors of a node into important ones which can influence the semantics of this node, and non-important ones which can, to an extent, affect predictive outputs but hardly change the node's abnormality, e.g., topology and node roles in the graph. Hence, we decompose this process into two mechanisms: *core* feature generation and *environment* feature generation. According to the learning process of GNNs that node representations are produced by aggregating neighboring nodes in the graph [18], we assume that the core feature is extracted based on itself and its important neighborhood, which comprises the *core subgraph*, and the environment feature is extracted based on other insignificant neighborhood, which comprises the *environment subgraph*. We steer a generative model to manufacture synthetic environment subgraphs and then generate counterfactual subgraphs, i.e., unseen combinations of core subgraphs and synthetic environment subgraphs. Subsequently, we train an anomaly detection model based on the generated counterfactual subgraphs, such that the model can learn transferable and causal relations across different environments.

Specifically, in the CFAD framework, we first devise a Granger causality-based [19] causal explanation method to extract the core subgraph for a given node without requiring annotation information, according to the idea that causal explanations can extract a subgraph which is considered the main cause of the corresponding prediction [17], [20]. Then, we design a generator that takes random noises as input and outputs a group of synthetic node representations, each representation is corresponding to a node in the environment subgraph. Next, we combine the synthetic representations of environment nodes with the real representations of core nodes and decode the combined representations into a subgraph, which we call the *counterfactual subgraph*. Finally, we build a counterfactual subgraph-based contrasting loss and consistency regularization term for a few-shot anomaly detection model, which can accurately detect anomalies by learning causal relations across different environments using a few labeled anomalies. The main contribution of our work is threefold:

- We contribute to the literature by proposing a novel counterfactual graph-based anomaly detection model CFAD, which can learn causal relations to enhance the anomaly detection performance with varying environments. To the best of our knowledge, this is the first model to learn causal representations and eliminate spurious correlations in anomaly detection.
- We generate counterfactual subgraphs by steering a generative model to perform interventions on real subgraphs, and illustrate the effectiveness and efficiency of counterfactual subgraphs by utilizing them to train an anomaly detection model. We theoretically and empirically show that CFAD's counterfactual subgraphs and generative interventions can help alleviate the spurious correlations between nodes in graph and learn robust causal node representations for better computing the anomaly scores.

TABLE I
MATHEMATICAL NOTATIONS

Symbol	Description
$\mathbf{A}$	Adjacency matrix.
$\mathcal{E}, e$	Set of edges, edge.
G	Attributed network $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$.
G^c	Core subgraph.
G^e	Environment subgraph.
G^l	Local subgraph, defined as the l-hop neighbors of the target node v. $G^l = G^c \cup G^e$.
K	Number of core nodes in core subgraph.
$\mathcal{L}$	Loss function.
s_i	Anomaly score of node v_i.
$\mathcal{V}, v$	A set of nodes, a node.
$\mathbf{X}, \mathbf{x}$	Features for all nodes and one node in an attributed network G, respectively.
$\mathbf{Z}, \mathbf{z}$	Latent representations for all nodes and one node $\mathcal{V}$ in G, respectively.

- Extensive experiments are conducted on real-world datasets to demonstrate the advantages of CFAD over several state-of-the-art baselines, which verifies our motivation that providing causality in anomaly detection can help us learn general and robust models.

II. Preliminaries

We now formulate the graph anomaly detection problem and give the definitions for three types of subgraphs, which will serve as background in our CFAD framework.

A. Problem Formulation

Following the commonly used notations, we use calligraphic fonts, bold lowercase letters, and bold uppercase letters to denote sets (e.g., $\mathcal{V}$), vectors (e.g., $\mathbf{x}$), and matrices (e.g., $\mathbf{X}$), respectively. Notations are listed in Table I for reference. In general, an attributed network can be represented by $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ denotes the set of nodes, $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ denotes the set of edges, and $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times h}$ denotes the set of node attributes. A binary adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is employed to denote the structure information of the attributed network, where $\mathbf{A}_{i,j} = 1$ if there is a link between nodes v_i and v_j; otherwise $\mathbf{A}_{i,j} = 0$. Since the information of $\mathcal{V}$ and $\mathcal{E}$ is both contained by $\mathbf{A}$, an attributed network can also be denoted as $G = (\mathbf{A}, \mathbf{X})$. Accordingly, the anomaly detection problem on attributed networks is given as follows:

Problem 1 (Anomaly Detection on Attributed Networks). Given an attributed network $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, we aim to learn an anomaly score function f to calculate the anomaly score $s_i = f(v_i)$ for each node in $\mathcal{V}$. Anomaly score s_i represents the degree of abnormality of node v_i. By ranking all the nodes with their anomaly scores, the anomaly nodes can be detected according to their positions.

B. Term Definitions

Next, we define three types of subgraphs used for counterfactual graph learning, i.e., local subgraphs, core subgraphs and environment subgraphs.

Definition 1 (Local Subgraphs). For a given node v, its local subgraph is defined as $G^l = (\mathcal{V}^l, \mathcal{E}^l, \mathbf{X}^l)$ where $\mathcal{V}^l$ is the set of nodes within l-hop neighbors of v. $\mathcal{E}^l$ and $\mathbf{X}^l$ are the corresponding edge set and attribute matrix associated with $\mathcal{V}^l$ individually.

Definition 2 (Core Subgraphs). For a given node v, its core subgraph can be denoted as $G^c = (\mathcal{V}^c, \mathcal{E}^c, \mathbf{X}^c)$ where $\mathcal{V}^c$ is the set of the most relevant nodes for the prediction (outcome) of a given classification model. $\mathcal{E}^c$ and $\mathbf{X}^c$ are the corresponding edge set and attribute matrix associated with $\mathcal{V}^c$, respectively.

Definition 3 (Environment Subgraphs). For a given node v, its environment subgraph is denoted as $G^e = (\mathcal{V}^e, \mathcal{E}^e, \mathbf{X}^e)$ where $\mathcal{V}^e$ is the difference set between $\mathcal{V}^l$ and $\mathcal{V}^c$, i.e., $\mathcal{V}^e = \mathcal{V}^l - \mathcal{V}^c$ where $\mathcal{V}^l$ and $\mathcal{V}^c$ are the node sets of the local subgraph and core subgraph. $\mathcal{E}^e$ and $\mathbf{X}^e$ are the corresponding edge set and attribute matrix associated with $\mathcal{V}^e$ individually.

Here, l-hop neighbors of v refer to the nodes whose shortest path length to v is l. And the most relevant nodes mean the most important nodes which can significantly influence the predictive outcome for a given classification model. Since these three types of subgraphs are extracted from G, their vertex sets ($\mathcal{V}^l$, $\mathcal{V}^c$ and $\mathcal{V}^e$) and edge sets ($\mathcal{E}^l$, $\mathcal{E}^c$ and $\mathcal{E}^e$) are the subsets of $\mathcal{V}$ and $\mathcal{E}$ in G, individually.

In GNN-based representation learning, a node's representation is obtained by structurally aggregating the representations from its neighbors in the graph. Since it is not necessary to take the full size of a node's neighborhood [18], we extract a subset of neighbors to form the local subgraph. For a given node, its representation based on the full graph is almost the same to the one on its local subgraph. Now we have defined the anomaly detection problem on attributed networks and introduced three types of subgraphs for counterfactual graph learning. In the next section, we provide model-free evidence that both theoretically and empirically explains our motivation for learning causal relations in anomaly detection.

III. Model-Free Evidence

In this section, we investigate the process of data-generation and the role of environment subgraphs, from a causal perspective.

A. A Causal View of Data-Generation Process

According to the definitions in Section II, the node representation is determined by its local subgraph. Similar to the interpretation of image generation process [10], we consider the causal generative process of local subgraphs is composed by two autonomous functions, i.e., we can modify the environment subgraph while keeping the core subgraph unchanged. This demand coincides with the concept of structural causal models (SCMs) and independent mechanisms [15]. An SCM is defined as a collection of d (structural) assignments:

$$G_j := f_j(\mathbf{PA}_j, U_j), j = 1, \ldots, d \tag{1}$$

where each variable G_j is a function of its parents $\mathbf{PA}_j \subseteq \{G_1, \ldots, G_d\} \backslash \{G_j\}$ and a noise variable U_j. The noise variables $U_1, \ldots, U_d$ are jointly independent. In SCMs, *intervention* is formalized as operations that modify a subset of assignments in (1), e.g., changing U_j or changing the functional form of f_j.

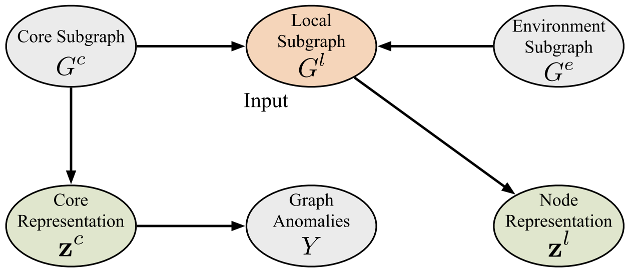

Core Subgraph G^c
Local Subgraph G^l
Environment Subgraph G^e
Input
Core Representation $\mathbf{z}^c$
Graph Anomalies Y
Node Representation $\mathbf{z}^l$

Fig. 1. Structural causal graph for anomaly detection task.

For anomaly detection task, Fig. 1 presents a causal graph that illustrates a data-generation process. The local subgraph G^l is caused by both the core subgraph G^c and environment subgraph G^e, as shown by the two incoming arrows to G^l. The node representation $\mathbf{z}^c$ is extracted based on the core subgraph G^c. The arrow from $\mathbf{z}^c$ to Y indicates that the ground-truth Y is conditioned on the representation $\mathbf{z}^c$. The representation $\mathbf{z}^l$ extracted from local subgraph should keep consistent semantics with $\mathbf{z}^c$ as long as the core subgraph is unchanged. Changing the environment subgraph G^e can be seen as an intervention on local subgraphs: for each observed sample $f(G_i^l, G_i^e)$, there is a set of unobserved counterfactual samples $f(G_i^l, \widetilde{G}_j^e)$ where $\widetilde{G}_j^e$ is any unobserved environment subgraph.

Assume the distributions of observed and counterfactual samples are denoted as $P^{\mathrm{F}}(G_i^l, G_i^e)$ and $P^{\mathrm{CF}}(G_i^l, \widetilde{G}_j^e)$, individually. Then we have $P^{\mathrm{F}}(G_i^l, G_i^e) = P(G_i^l) \cdot P(G_i^e | G_i^l)$ and $P^{\mathrm{CF}}(G_i^l, \widetilde{G}_j^e) = P(G_i^l) \cdot P(\widetilde{G}_j^e | G_i^l)$. Regarding $G^{e'}$ as any intervention (e.g., G_i^e or $\widetilde{G}_j^e$), the difference between the observed and counterfactual samples lies precisely in the intervention assignment mechanism, $P(G^{e'} | G_i^l)$ [21]. Here $G^{e'}$ and G_i^l are not independent according to the causal graph. As a result, the distribution of counterfactual samples, P^{CF}, is generally different from the distribution of observed samples P^{F}.

B. Role of Environment Subgraphs

To investigate whether it is necessary to manufacture environment subgraphs for enhancing anomaly detection models, we here analyze the impact of the environment subgraphs on anomaly node detection results. In particular, we analyze the changes of predicted anomaly scores when removing environment subgraphs. We define the notion of *impact degree* for environment subgraphs as the difference between predicted anomaly scores with and without the use of the environment subgraphs.

In this controlled experiment, for each local subgraph we select $q\%$ of the least important nodes as the environment subgraph, and the rest $(1-q)\%$ nodes as the core subgraph. The anomaly scores are computed by the GNN-based autoencoder method [22]. Fig. 2 reports the results on three benchmark datasets, where x-axis represents the ratio of removed nodes ($q\%$), and the y-axis represents the impact degree. Before computing the impact degree, we normalize the anomaly scores in range from 0 to 1. We can observe that the impact degree increases as the rise of the number of removed nodes. If we remove fewer nodes (e.g., less than 40%), the predicted

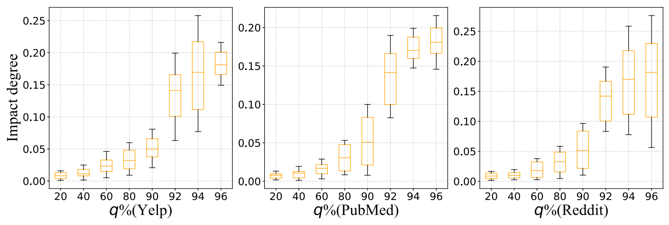


Fig. 2. The impact of environment subgraphs.

anomaly scores are almost unaffected. One possible explanation is that learning node representations with many neighboring nodes is redundant [18]. However, removing a large number of neighboring nodes dramatically influences the predicted scores. For example, the median impact degree reaches ∼14% when removing 92% of nodes for Yelp dataset. This result indicates that unless a large number of nodes are selected as core nodes, the environment subgraphs do have an effect on the predicted anomaly scores. In addition, we analyze the anomaly detection accuracy when using different number of core nodes on three datasets, and the results further prove our observation that neighboring nodes place an impact on detection performance, which coincides well with the works [18], [23]. This motivates us to exert interventions on environment subgraphs to boost the anomaly detection performance.

IV. Method

In this section, we introduce the details of the proposed framework, CFAD. Specifically, we first present how to extract the core subgraph of a given node, and then illustrate how to train a generator for synthetic representations. Next, we combine synthetic environment representations with core representations to manufacture counterfactual subgraphs, and finally propose the counterfactual subgraph-based anomaly detection model.

A. *Extracting Core Subgraphs*

We now illustrate how to extract the core subgraph of a given node based on causal explanations for GNNs. Causal explanations aim to find which fraction of the input graph is most influential to the model's decision [24]. Based on this idea, we introduce causal explanation techniques to identify the nodes' causal contributions in a graph, and distill the top-K most relevant nodes as the core nodes constituting the core subgraph. Existing causal explanation methods mainly focus on annotated samples [17], [20]. Since obtaining the annotated samples for anomaly detection is cost prohibitive, we design a Granger causality-based causal explanation method to determine the core subgraph without relying on annotation information.

Specifically, inspired by the work [17], we introduce the notion of Granger causality [19] to extract core subgraphs. For a given node v_t and its local subgraph G^l, we use δ_{G^l} to denote the model error of the GNN when considering the local subgraph G^l, while $\delta_{G^l\backslash\{v_j\}}$ represents the model error excluding the information from node $v_j \in G^l$. Following the idea of Granger causality, we can quantify the causal contribution of node v_j to the output of the GNN. The causal contribution of node v_j is defined as the decrease in the model error, formulated as

$$\Delta_{\delta,v_j} = \delta_{G^l\backslash\{v_j\}} - \delta_{G^l}. \tag{2}$$

When coping with unlabeled nodes, we use the reconstruction loss of the deep graph convolutional autoencoder [22] to calculate δ_{G^l} and $\delta_{G^l\backslash\{v_j\}}$. The reconstruction loss in the autoencoder can be adopted to accurately evaluate the abnormality of nodes [25], [26]. Hence it can be regarded as the model error for anomaly detection tasks. For this purpose, based on the autoencoder architecture, we first compute the reconstructed adjacency matrix and the attributes corresponding to the local subgraph G^l and the one excluding node v_j, $G^l\backslash\{v_j\}$. For simplicity, the functions for reconstructing the adjacency matrices and attributes in the autoencoder are denoted as $f_{\mathbf{A}}(\cdot)$ and $f_{\mathbf{X}}(\cdot)$, respectively. The associated outputs can be formulated as follows:

$$\widetilde{\mathbf{A}}_{G^l} = f_{\mathbf{A}}(G^l), \quad \widetilde{\mathbf{X}}_{G^l} = f_{\mathbf{X}}(G^l), \tag{3}$$

$$\widetilde{\mathbf{A}}_{G^l\backslash\{v_j\}} = f_{\mathbf{A}}\left(G^l\backslash\{v_j\}\right), \quad \widetilde{\mathbf{X}}_{G^l\backslash\{v_j\}} = f_{\mathbf{X}}\left(G^l\backslash\{v_j\}\right). \tag{4}$$

Then we compare the reconstructed adjacency matrices and attributes, i.e., $\widetilde{\mathbf{A}}_{G^l}$, $\widetilde{\mathbf{X}}_{G^l}$, $\widetilde{\mathbf{A}}_{G^l\backslash\{v_j\}}$ and $\widetilde{\mathbf{X}}_{G^l\backslash\{v_j\}}$, with the original ones, $\mathbf{A}_{G^l}$ and $\mathbf{X}_{G^l}$. As a result, the model error is measured by the reconstruction loss of the autoencoder (denoted as $\mathcal{L}$):

$$\delta_{G^l} = \mathcal{L}\left(\mathbf{A}_{G^l}, \mathbf{X}_{G^l}, \widetilde{\mathbf{A}}_{G^l}, \widetilde{\mathbf{X}}_{G^l}\right), \tag{5}$$

$$\delta_{G^l\backslash\{v_j\}} = \mathcal{L}\left(\mathbf{A}_{G^l}, \mathbf{X}_{G^l}, \widetilde{\mathbf{A}}_{G^l\backslash\{v_j\}}, \widetilde{\mathbf{X}}_{G^l\backslash\{v_j\}}\right). \tag{6}$$

Now, the causal contribution of node v_j is measured by the loss difference Δ_{δ,v_j} associated with the original local subgraph and the one removing node v_j.

Given causal contributions of nodes in a local graph, we can sort the nodes accordingly and distill the top-K most relevant nodes and corresponding edges as the core subgraph. Here, the parameter K can be tuned by performing a grid search on the values near the average vertex degree.

B. *Training Representation Generator*

In this subsection, we present the representation generator which takes random noises as input and outputs synthetic representations for producing counterfactual subgraphs. For this purpose, we first train an Autoencoder (AE) to embed a local subgraph into a group of node representations. Based on obtained node representations, we introduce a generative adversarial network (GAN) to produce synthetic representations, each of which is corresponding to a node in the local graph. Fig. 3 illustrates the two-step training process for the representation generator.

1) Autoencoder Training: The first step is to train an AE with the encoder $\mathbf{Enc}(\cdot)$ and decoder $\mathbf{Dec}(\cdot)$. The encoder takes the local subgraph G^l as input and embeds it into a group of latent representations, i.e., $\mathbf{Z} = \mathbf{Enc}(G^l) \sim q(\mathbf{Z}|G^l)$, while the

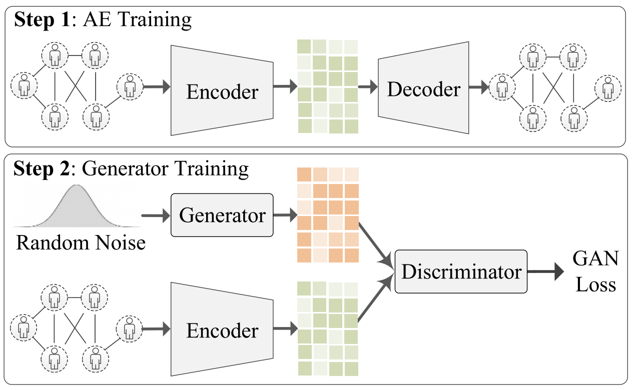


Fig. 3. Two-steps training process for the synthetic representation generator.

decoder learns to reconstruct the local subgraph from the latent representations, i.e., $\widetilde{G}^l = \mathbf{Dec}(\mathbf{Z}) \sim p(G^l|\mathbf{Z})$.

As for the encoder, it is built with multiple GNN layers that aggregate each node to a low-dimensional latent representation. Following the neighborhood message-passing mechanism, GNN layers compute the node representations by aggregating node features $\mathbf{X}$ from local neighborhoods in an iterative manner. Hence, the encoder $\mathbf{Eec}(\cdot)$ with L layers can be expressed as:

$$\mathbf{H}^1 = \text{GNN}^1(\mathbf{A}, \mathbf{X}), \quad \ldots, \quad \mathbf{Z} = \text{GNN}^L(\mathbf{A}, \mathbf{H}^{L-1}), \tag{7}$$

where GNN^L is the last layer and $\mathbf{Z}$ is the desired node representations learned from the encoder. Note that in CFAD the model encoder is compatible with any arbitrary GNN-based architectures [27], [28], [29]. For efficiency, we select simple graph convolution (SGC) [30] in our implementation.

After obtaining the latent representation $\mathbf{Z}$, the decoder aims to reconstruct the attributed network. Specifically, the decoder takes $\mathbf{Z}$ as input and then calculates the inner product to obtain the rebuilt adjacency matrix $\widetilde{\mathbf{A}}$:

$$\widetilde{\mathbf{A}} = \text{sigmoid}(\mathbf{Z} \cdot \mathbf{Z}^T). \tag{8}$$

To approximate the original node attributes from the encoded representation, we leverage a simple fully-connected layer to reconstruct the attribute information as follows:

$$\widetilde{\mathbf{X}} = f_{\text{ReLU}}(\mathbf{Z} \cdot \mathbf{W} + \mathbf{b}), \tag{9}$$

where $\mathbf{W}$ is the weight matrix, and $\mathbf{b}$ is the corresponding bias term. Then, considering the reconstruction errors, the objective function of the AE can be formulated as:

$$\mathcal{L}_{\text{AE}} = (1-\alpha)\left\|\mathbf{A} - \widetilde{\mathbf{A}}\right\|_F^2 + \alpha\left\|\mathbf{X} - \widetilde{\mathbf{X}}\right\|_F^2, \tag{10}$$

where $\|\cdot\|_F^2$ refers to the Frobenius norm and α is an important controlling parameter which balances the trade-off between structure reconstruction and attribute reconstruction.

2) Generator Training: In the second step, we propose to learn a representation generator g with the help of the pre-trained AE in an adversarial manner. The generator g takes a noise vector sampled from a prior distribution $p(\tilde{\mathbf{z}})$ as input. Given the encoder $\mathbf{Enc}(\cdot)$ from AE, we train the generator g to produce node representations. The distribution of generated representations resembles the distribution of latent embeddings from $\mathbf{Enc}(\cdot)$ as similar as possible, such that the discriminator D in GAN cannot reliably distinguish them. Based on the concept of Least Squares GAN [31], the mini-max game is formalized as follows:

$$\min_g \max_D \quad \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(D\left(\mathbf{z}\right)\right)\right] + \mathbb{E}_{\tilde{\mathbf{z}}\sim p(\tilde{\mathbf{z}})}\left[\log\left(1 - D\left(g\left(\tilde{\mathbf{z}}\right)\right)\right)\right], \tag{11}$$

where $p(\tilde{\mathbf{z}})$ is the prior distribution. Previous research [32] and our preliminary experiments both show that Gaussian prior is a robust option across different datasets. The parameters in $\mathbf{Enc}(\cdot)$ are kept frozen when training the generator.

Once the training process of generator g is converged, we use g to produce a set of synthetic representations from a random noise. Each synthetic representation is corresponding to a node in the local subgraph. Meanwhile, the synthetic representation distribution should be similar to the one of the real node. Next we illustrate how to use synthetic representations to construct counterfactual subgraphs.

C. Generating Counterfactual Subgraphs

To obtain qualified counterfactual subgraphs, we first extract high-quality synthetic representations, and then produce counterfactual subgraphs based on extracted representations.

1) Synthetic Representation Extraction: Since some of the generated synthetic representations tend to be noisy and distorted due to the unstable nature of GANs, we first remove synthetic representations with low quality before producing counterfactual subgraphs. Specifically, the quality score is calculated by the similarity degree between synthetic and real representations. For a given node, we compute the quality score of a group of synthetic representations by combining two values: one is the similarity degree between the synthetic representations and the real node representations of its local subgraph; and another is the the similarity degrees across all nodes. The higher the similar degree, the higher the quality score.

Formally, for a given node v_j, we denote its local subgraph's representation as $\mathbf{Z}_j^l = [\mathbf{z}_1^c, \ldots, \mathbf{z}_m^c, \mathbf{z}_1^e, \ldots, \mathbf{z}_n^e]$, which can be split into two groups: one for the core nodes $\mathbf{Z}_j^c = [\mathbf{z}_1^c, \ldots, \mathbf{z}_m^c]$ and another for the environment nodes $\mathbf{Z}_j^e = [\mathbf{z}_1^e, \ldots, \mathbf{z}_n^e]$. For synthetic representation $\widetilde{\mathbf{Z}}_s^l = [\tilde{\mathbf{z}}_1^c, \ldots, \tilde{\mathbf{z}}_m^c, \tilde{\mathbf{z}}_1^e, \ldots, \tilde{\mathbf{z}}_n^e]$, the similarity degree with its local subgraph is calculated by the Euclidean distance between $\widetilde{\mathbf{Z}}_s^l$ and $\mathbf{Z}_j^l$:

$$d_{\text{Euc}}(\widetilde{\mathbf{Z}}_s^l, \mathbf{Z}_j^l) = \frac{1}{m}\sum_{t=1}^{m}\|\tilde{\mathbf{z}}_t^c - \mathbf{z}_t^c\|^2 + \frac{1}{n}\sum_{t=1}^{n}\|\tilde{\mathbf{z}}_t^e - \mathbf{z}_t^e\|^2. \tag{12}$$

The similarity degree with all nodes is computed as:

$$d_{\text{Euc}}\left(\widetilde{\mathbf{Z}}_s^l, \mathbf{Z}_1^l, \ldots, \mathbf{Z}_a^l\right) = \frac{1}{a}\sum_{t=1}^{a} d_{\text{Euc}}\left(\widetilde{\mathbf{Z}}_s^l, \mathbf{Z}_t^l\right), \tag{13}$$

where a is the number of nodes. Correspondingly, the quality score of synthetic representation $\widetilde{\mathbf{Z}}_s^l$ for node v_j is defined as:

$$Q\left(\widetilde{\mathbf{Z}}_s^l|v_j\right) = \beta \cdot d_{\text{Euc}}\left(\widetilde{\mathbf{Z}}_s^l, \mathbf{Z}_j^l\right) \tag{14}$$

$$+ (1-\beta) \cdot d_{\text{Euc}}\left(\widetilde{\mathbf{Z}}_s^l, \mathbf{Z}_1^l, \ldots, \mathbf{Z}_a^l\right), \tag{15}$$

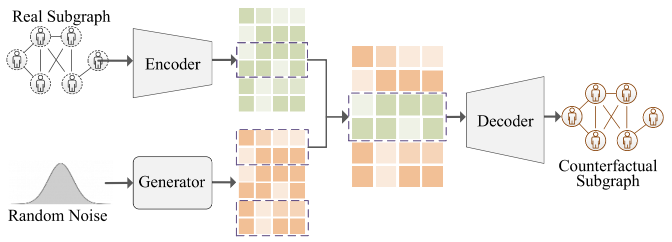


Fig. 4. The process of generating counterfactual subgraphs. During this process, the representations of core nodes in the synthetic representation matrix (orange block) are replaced with the real representations in the encoded representation matrix (green block). Then the mixed representation matrix is decoded into the counterfactual subgraph.

where β is a parameter to adjust the weights of two terms.

Based on the quality score, we extract the top P generated representations:

$$f_{\text{select}} = \begin{cases} 1, & Q(\widetilde{\mathbf{Z}}_s^l | v_j) > \epsilon, \\ 0, & \text{otherwise}, \end{cases} \tag{16}$$

where f_{select} denotes the select condition, and ϵ is a threshold determined by the quality score ranking result based on the number P of desired representations.

2) Counterfactual Subgraphs: Here we show how to adopt the synthetic representations to generate counterfactual subgraphs. The main idea is to replace the synthetic representations associated with the core nodes with the real representations and then utilize the mixed representations to construct the counterfactual subgraph. The generation process is illustrated in Fig. 4.

In particular, for node v_t, the node set of its local subgraph is denoted as $\mathcal{V}^l = \{v_1^c, \ldots, v_m^c, v_1^e, \ldots, v_n^e\}$, where $\mathcal{V}^c = \{v_1^c, \ldots, v_m^c, \}$ and $\mathcal{V}^e = \{v_1^e, \ldots, v_n^e\}$ are the node sets of its core subgraph and its environment subgraph, respectively. During the generation process: (1) the encoder first transforms the nodes in $\mathcal{V}^l$ into node representations $\mathbf{Z}^l = [\mathbf{z}_1^c, \ldots, \mathbf{z}_m^c, \mathbf{z}_1^e, \ldots, \mathbf{z}_n^e]$; (2) then the generator takes a random noise as input and generates another group of node representations $\widetilde{\mathbf{Z}}^l = [\tilde{\mathbf{z}}_1^c, \ldots, \tilde{\mathbf{z}}_m^c, \tilde{\mathbf{z}}_1^e, \ldots, \tilde{\mathbf{z}}_n^e]$, each of which is corresponding to a node in $\mathcal{V}^l$; (3) among generated representations $\widetilde{\mathbf{Z}}_l$, the representations $\widetilde{\mathbf{Z}}^c$ associated with the core nodes are replaced by $\mathbf{Z}^c$ to form the mixed representations $\mathbf{Z}^{\text{mix}} = [\mathbf{z}_1^c, \ldots, \mathbf{z}_m^c, \tilde{\mathbf{z}}_1^e, \ldots, \tilde{\mathbf{z}}_n^e]$; (4) finally, the mixed representations $\mathbf{Z}^{\text{mix}}$ is fed into the decoder to produce the counterfactual subgraph. Following these four steps, each node can obtain multiple counterfactual graphs, which will be adopted to train a robust model for anomaly detection.

D. Anomaly Detection Model

This section presents the counterfactual subgraph-based anomaly detection model CFAD, which distinguishes normal and abnormal nodes according to the computed anomaly scores using a few labeled anomalies. Fig. 5 shows CFAD's architecture. In this model, (1) the encoder first learns node representations of real subgraphs and counterfactual subgraphs, which are passed to the abnormality valuator $f_{\theta_s}(\cdot)$ for estimating the anomaly score s_i; (2) then the Gaussian prior-based reference score R_s is computed to guide the learning process of anomaly scores; (3) lastly, s_i and R_s are inputs to the loss functions, including our designed counterfactual subgraph-based contrastive loss and consistency regularization term, to guide the optimization.

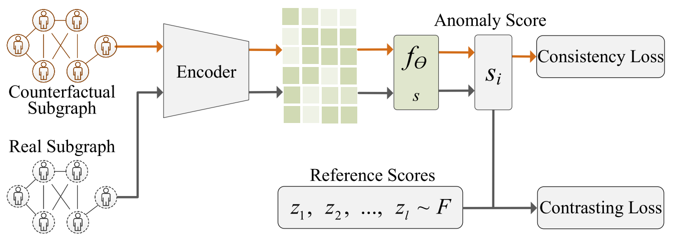


Fig. 5. The model architecture for the anomaly detection with counterfactual subgraphs.

Concretely, for the abnormality valuator $f_{\theta_s}(\cdot)$, it is built with the feed-forward layers that transform the intermediate node representations to a scalar anomaly score:

$$s_i = f_{\theta_s}(v_i) = \mathbf{u}_s^T \cdot \text{ReLU}(\mathbf{W}_s \cdot \mathbf{z}_i + \mathbf{b}_s^1) + b_s^2, \tag{17}$$

where s_i is the anomaly score of node v_i and $\mathbf{z}_i$ is the representation of node v_i. $\mathbf{W}_s$ and $\mathbf{u}_s$ are the learnable weight matrix and weight vector, respectively. $\mathbf{b}_s^1$ and b_s^2 are the corresponding bias terms.

For the reference score, we define it as the mean value of the anomaly scores of a set of randomly selected normal nodes, which serves as the reference to quantify how much the scores of anomalies deviate from those of normal nodes. Since Gaussian distribution is commonly a robust choice to fit the abnormality scores for a wide range of datasets [5]. We first sample a set of k anomaly scores from the Gaussian prior distribution, i.e., $\{r_1, r_2, \ldots, r_k\} \sim \mathcal{N}(\mu, \sigma^2)$, each of which denotes the abnormality of a random normal node. The reference score is then computed as the mean value of all the sampled scores:

$$R_s = \frac{1}{k} \sum_{i=1}^{k} r_i. \tag{18}$$

With the reference score in hand, the deviation between the anomaly score of node v_i and the reference score can be defined in the form of the standard score: $\text{dev}(v_i) = (s_i - R_s)/\delta_r$, where δ_r is the standard deviation of the set of sampled anomaly scores $\{r_1, r_2, \ldots, r_k\}$.

Then, we introduce contrasting loss [33] as the objective function by replacing the distance function with the deviation. For labeled node v_i^l and its corresponding counterfactual node $\widetilde{v}_i^l$, their losses are defined as:

$$\mathcal{L}_{lr} = (1 - y_i) \cdot \left|\text{dev}\left(v_i^l\right)\right| + y_i \cdot \max\left(0, m - \text{dev}\left(v_i^l\right)\right), \tag{19}$$

$$\mathcal{L}_{cf} = (1 - y_i) \cdot \left|\text{dev}\left(\widetilde{v}_i^l\right)\right| + y_i \cdot \max\left(0, m - \text{dev}\left(\widetilde{v}_i^l\right)\right), \tag{20}$$

where y_i is the ground-truth of input node v_i^l. If node v_i^l is an abnormal node then $y_i = 1$; otherwise $y_i = 0$. Here the confidence margin m is defined as a radius around the deviation.

For unlabeled node v_i^u and its corresponding counterfactual node $\widetilde{v}_i^u$, since they should have the same anomaly score, we introduce a consistency regularization term [34] as the objective function:

$$\mathcal{L}_{cr} = \left\| \text{dev}\,(v_i^u) - \text{dev}(\widetilde{v}_i^u) \right\|^2 . \tag{21}$$

This loss function penalizes the inconsistent anomaly scores of real unlabeled node v_i^u and its corresponding counterfactual node $\widetilde{v}_i^u$ to enhance robustness of the model. As a result, the model is optimized by a combination of these three losses:

$$L_{\text{final}} = \mathcal{L}_{lr} + \lambda_1 \cdot \mathcal{L}_{cf} + \lambda_2 \cdot \mathcal{L}_{cr}, \tag{22}$$

where λ_1 and λ_2 are two hype-parameters to adjust loss weights. In this loss, the contrasting loss $\mathcal{L}_{cf}$ and the consistency regularization term $\mathcal{L}_{cr}$ are based on the counterfactual subgraphs. Therefore, it can learn robust causal representations for nodes in graph during model training. The attained representations reveal the causal correlations between the target node and its subgraph and allow us to better differentiate abnormal nodes from others in the latent space. Consequently, the proposed model improves the anomaly detection performance which we will detail in the next section.

E. Training Process of the Proposed Model

Here we summarize the overall training procedures of the synthetic representation generator and anomaly detection model.

The training procedure of the generator is summarized in Algorithm 1. The autoencoder is first trained based on the attributed network G using the loss in (10) (Line 1-5), and the encoder is saved as $\mathbf{Enc}$ for training generator g (Line 6). Next, to train generator g, the local subgraph of each node in G is extracted and stacked into the subgraph set $\mathcal{G}$ (Line 8-11). Finally, the generator g and discriminator D are alternately updated using representation matrices produced by the encoder (Line 12-18). The trained generator g will be used to produce synthetic representations for producing counterfactual subgraphs.

The training procedure of the anomaly detection model is summarized in Algorithm 2. In an epoch of the training phase, we split the node set into several mini-batches. Then in each iteration, for labeled node v_i^l, its representation $\mathbf{z}_i$ is first computed based on its local subgraph G_i^l by (7) (Line 5). After that, the local subgraph-based contrasting loss $\mathcal{L}_{lr}$ in (19) is calculated based on its representation $\mathbf{z}_i$ (Line 6). Similarly, its counterfactual representation $\widetilde{\mathbf{z}}_i$ is computed based on its counterfactual subgraph $\widetilde{G}_i^l$ by (7), and the counterfactual subgraph-based contrasting loss $\mathcal{L}_{cf}$ in (19) is calculated based on $\widetilde{\mathbf{z}}_i$ (Line 7-8). Next, for unlabeled node v_j^u, its representation $\mathbf{z}_j$ and counterfactual representation $\widetilde{\mathbf{z}}_j$ are computed based on its local subgraph G_j^l and counterfactual subgraph $\widetilde{G}_j^l$, individually (Line 9). Based on these two representations, the consistent regularization loss $\mathcal{L}_{cr}$ in (21) is calculated (Line 10). Finally, the combination of these three losses, $\mathcal{L}_{lr}$, $\mathcal{L}_{cf}$ and $\mathcal{L}_{cr}$, forms the final loss L_{final}, and the parameters of the model are optimized by a back-propagation with a gradient descent algorithm. After the training phase, anomaly scores of nodes are computed using (17).

Algorithm 1: Training Process of the Synthetic Representation Generator.

```
Input: Attributed network $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$; Number of
  training epochs: $T_1$ and $T_2$.
Output: The synthetic representation generator $g$.
1: Initialize parameters of the encoder and decoder;
2: while $t < T_1$ do
3:   Sample a batch of nodes in $\mathcal{V}$;
4:   Update parameters for training the encoder and
     decoder using the loss in (10);
5: end while
6: Save the encoder as $\mathbf{Enc}$;
7: Initialize parameters of generator $g$ and discriminator $D$;
8: for each node $v_i$ in $\mathcal{V}$ do
9:   Extract the local subgraph $G_i^l$ of $v_i$;
10:  Put $G_i^l$ into the subgraph set $\mathcal{G}$;
11: end for
12: while $t < T_2$ do
13:  Sample a batch of noises from noise prior $p(\widetilde{\mathbf{z}})$;
14:  Sample a batch of subgraphs in $\mathcal{G}$, and encode each
     subgraph into a representation matrix by $\mathbf{Enc}$;
15:  Update the parameters of discriminator $D$;
16:  Sample a batch of noises from noise prior $p(\widetilde{\mathbf{z}})$;
17:  Update the parameters of generator $g$.
18: end while
```

Algorithm 2: Anomaly Detection Model Training

```
Input: Five parts: (1) labeled nodes with their local
  subgraphs and labels: $\{(v_i^l, G_i^l, y_i)\}_{i=1}^I$; (2) labeled nodes
  with their counterfactual subgraphs and labels:
  $\{(v_i^l, \widetilde{G}_i^l, y_i)\}_{i=1}^I$; (3) unlabeled nodes with their local
  subgraphs: $\{(v_j^u, G_j^l)\}_{j=1}^J$; (4) unlabeled nodes with their
  counterfactual subgraphs: $\{(v_j^u, \widetilde{G}_j^l)\}_{j=1}^J$; and (5) number
  of training epochs $T$ and hyper-parameters $m$, $\lambda_1$ and $\lambda_2$.
Output: Anomaly scores $s$ of unlabeled nodes.
1: Initialize parameters of the model;
2: while $t < T$ do
3:   Randomly split nodes into batches;
4:   for each mini-batch do
5:     Compute representation $\mathbf{z}_i$ of labeled node $v_i^l$ using
       its local subgraph $G_i^l$ by (7);
6:     Calculate $\mathcal{L}_{lr}$ in (19) via $\mathbf{z}_i$;
7:     Compute counterfactual representation $\widetilde{\mathbf{z}}_i$ of labeled
       node $v_i^l$ using its counterfactual subgraph $\widetilde{G}_i^l$ by (7);
8:     Calculate $\mathcal{L}_{cf}$ in (20) via $\widetilde{\mathbf{z}}_i$;
9:     Compute representation $\mathbf{z}_j$ and counterfactual
       representation $\widetilde{\mathbf{z}}_j$ of unlabeled node $v_j^u$ using its
       local subgraph $G_j^l$ and counterfactual subgraph $\widetilde{G}_j^l$,
       individually, by (7);
10:    Calculate $\mathcal{L}_{cr}$ in (21) via $\mathbf{z}_j$ and $\widetilde{\mathbf{z}}_j$;
11:    $L_{\text{final}} \leftarrow \mathcal{L}_{lr} + \lambda_1 \cdot \mathcal{L}_{cf} + \lambda_2 \cdot \mathcal{L}_{cr}$;
12:    Take gradient steps and update the parameters;
13:  end for
14: end while
15: Compute anomaly scores of nodes using (17).
```

F. Theoretical Analysis

Generative interventions can be helpful to eliminate spurious correlations, resulting in better generalization across different detection environments. It is theoretically impractical to generate perfect interventions and eliminate all the spurious correlations. However, we can estimate the lower and upper bounds for the causal effects.

We formalize the proposed framework with structural causal models (SCMs) [15]. Assume that $P(s|do(G^l))$ denotes the causality from local subgraph G^l to anomaly score s, which is the treatment effect of an input subgraph G^l on anomaly score s. We present the bound theorem as follows.

Theorem 1. Given the observed joint distribution, the lower and upper bound of $P(s|do(G^l))$ is $P(G^l, s) \leq P(s|do(G^l)) \leq P(G^l, s) + 1 - P(G^l)$.

Proof. Presume that all the unobserved interventions as ui.

$$P(s|do(G^l)) = \sum_{ui} P(s|G^l, ui) \cdot P(ui) \tag{23}$$

$$= \sum_{ui} P(s|G^l, ui) \cdot \big(P(ui, G^l) - P(G^l, ui) + P(ui)\big) \tag{24}$$

$$= \sum_{ui} P(s, G^l, ui) + \sum_{ui} P(s|G^l, ui) \cdot (P(ui) - P(ui, G^l)). \tag{25}$$

Because of $P(s|G^l, ui) \in [0, 1]$, $P(s|do(G^l))$ should be bigger than $\sum_{ui} P(s, G^l, ui)$, i.e.,

$$P(s|do(G^l)) \geq P(G^l, s). \tag{26}$$

On the other hand, $P(s|do(G^l))$ should be smaller than $\sum_{ui} P(s|G^l, ui) \cdot (P(ui) - P(ui, G^l))$, i.e.,

$$P(s|do(G^l)) \leq P(G^l, s) + 1 - P(G^l). \tag{27}$$

Hence $P(s|do(G^l))$ has a bound between $P(G^l, s)$ and $P(G^l, s) + 1 - P(G^l)$. □

We try to exert interventions on environment subgraphs and generate new local (counterfactual) subgraphs to explore the causality of G^l on anomaly score s, i.e., $P(s|do(G^l))$. Here, $P(s|do(G^l))$ is different from $P(s|G^l)$. For $P(s|G^l)$ it refers to the observational distribution and a change in G^l can indicate a change in unobserved confounding bias under the causal graph. Correspondingly, the observed change in G^l should vary in confounding variables. Hence, the observed $P(s|G^l)$ should be differentiated from $P(s|do(G^l))$. To identify the causal effect of $P(s|do(G^l))$, we make the assumption that iv denotes the intervention variable. It measures the difference between real environment subgraph and synthetic environment subgraph. We notice the following theorem.

Theorem 2. The causal effect from local subgraph G^l to anomaly score s can be identified by observing variable iv.

Proof. Presume that $\mathbf{PA}_j$ denotes the parent node of variable G^l. Then:

$$P(s|do(G^l)) = \sum_{\mathbf{PA}_j} P(s|G^l, \mathbf{PA}_j) \cdot P(\mathbf{PA}_j)$$

$$= \sum_{\mathbf{PA}_j, iv} P(s|G^l, \mathbf{PA}_j, iv) \cdot P(iv|G^l, \mathbf{PA}_j) \cdot P(\mathbf{PA}_j)$$

$$= \sum_{\mathbf{PA}_j} P(s|G^l, iv) \cdot P(iv|\mathbf{PA}_j) \cdot P(\mathbf{PA}_j)$$

$$= \sum_{iv} P(s|G^l, iv) \cdot P(iv). \tag{28}$$

Hence, we can identify the causal effect of G^l on s by observing iv. □

TABLE II
DATASET STATISTICS. HERE r_1 (RESP. r_2) DENOTES THE RATIOS OF LABELED ANOMALIES TO ALL ANOMALIES (RESP. NUMBER OF NODES)

Datasets	# node	# edge	feat.	anomal.	r_1	r_2
Yelp	4,872	43,728	10,000	223	4.48%	0.21%
PubMed	3,675	8,895	500	201	4.97%	0.27%
Reddit	15,860	136,781	602	796	1.26%	0.063%

V. EXPERIMENTAL EVALUATION

In this section, we perform empirical evaluations to demonstrate the effectiveness of our model in terms of anomaly detection performance, data efficiency, ablation study and sensitive analysis.

A. Experimental Settings

We employ three real-world attributed networks – Yelp, PubMed, and Reddit [27] for performance comparison, which are publicly available and have been widely used in previous research [27], [35]. For the Yelp dataset, the labels of the nodes are collected from Yelp.com. While, for the PubMed and Reddit datasets, we do not have labels for abnormal nodes. Thus we follow previous works [22], [33] to adopt the perturbation scheme to inject several anomalies (e.g., structural and contextual anomalies). The statistics of our three datasets are shown in Table II. For performance evaluation, we use three standard metrics: AUC-ROC, AUC-PR, and Precision@K.

We compare the proposed framework CFAD with nine unsupervised and semi-supervised anomaly detection methods: *LOT* [36] is a feature-based approach which detects outliers at the contextual level; *Autoencoder* [37] is a feature-based unsupervised deep model which introduces an anomaly regularizing penalty based upon L1 or L2 norms; *SCAN* [38] is an efficient algorithm for detecting network anomalies based on a structural similarity measure; *ConOut* [39] identifies network anomalies according to the corresponding subgraph and the relevant subset of attributes in the local context; *Radar* [40] is an unsupervised method that detects anomalies on attributed network by characterizing the residuals of attribute information and its coherence with network structure; *DOMINANT* [22] is a GCN-based autoencoder which computes anomaly scores using the reconstruction errors from both network structure and node attributes; *SemiGNN* [41] is a semi-supervised GNN model which leverages the hierarchical attention mechanism to correlate neighbors and views at different levels; *Deep-SAD* [42] is a state-of-the-art semi-supervised anomaly detection model, where the node attributes are leveraged as input features; *GDN* [33] is a novel GNN-based model which is capable of leveraging limited labeled anomalies to enforce statistically

TABLE III
Performance Comparison Between CFAD and Baselines on Three Datasets

Method	Yelp					PubMed					Reddit				
	A-ROC	A-PR	P@50	P@100	P@200	A-ROC	A-PR	P@50	P@100	P@200	A-ROC	A-PR	P@50	P@100	P@200
LOF	0.375	0.042	5.013	3.864	2.046	0.575	0.187	20.014	17.863	13.981	0.518	0.071	7.639	5.982	1.981
Autoencoder	0.365	0.041	3.987	3.076	0.968	0.584	0.236	23.268	23.679	15.079	0.722	0.347	34.982	7.183	23.528
SCAN	0.397	0.046	5.084	4.912	2.897	0.421	0.048	6.983	5.691	3.948	0.298	0.048	5.921	5.168	2.071
ConOut	0.402	0.041	5.071	5.023	3.067	0.511	0.093	12.561	10.054	8.642	0.551	0.085	11.936	10.716	7.294
Radar	0.415	0.045	5.196	5.076	2.961	0.573	0.244	27.941	24.169	18.457	0.721	0.281	30.721	27.068	20.928
DOMINANT	0.578	0.109	10.174	8.761	4.897	0.636	0.337	35.128	30.296	23.697	0.735	0.357	35.918	31.928	24.264
DeepSAD	0.460	0.062	5.791	6.897	3.678	0.528	0.115	13.964	13.761	9.827	0.503	0.066	8.964	6.726	2.284
SemiGNN	0.497	0.058	5.726	7.594	4.986	0.523	0.065	7.863	7.614	2.982	0.610	0.134	15.927	14.683	11.926
GDN	0.678	0.132	15.087	12.564	7.945	0.736	0.438	48.834	45.069	38.972	0.811	0.379	38.724	35.187	28.617
CFAD	**0.724**	**0.175**	**17.865**	**15.689**	**9.628**	**0.761**	**0.485**	**50.672**	**45.327**	**40.089**	**0.842**	**0.395**	**40.761**	**38.169**	**29.371**

significant deviations between abnormal and normal nodes on attributed networks.

For core subgraph extraction, we adopt the anomaly detection model introduced in [22] for both efficiency and simplicity. For the encoder architecture, we use the SGC model [30] to build the network encoder with two layers. The abnormality valuator in (17) employs two fully-connected layers with unit numbers of 512 and 1, respectively. The confidence margin m (in (19) and (20)) is set to 5. The reference score R_s is computed by (18) with $k = 5,000$ scores sampled from a standard Gaussian prior distribution. Unless otherwise specified, for each dataset, we randomly access 10 labeled abnormal nodes for each run of the experiment. We train our model at most 1,000 epochs with a batch size of 16. The dataset is split into 40-20-40% for training, validating and testing, respectively. The baseline methods adopt the same dataset, and the hyper-parameters are tuned based on the validation set, and the results are reported based on the test set. The experiments are performed by a Windows desktop with an Intel Xeon E5-1603 CPU and NVIDIA GeForce RTX 3090 GPU.

B. Anomaly Detection Results

We now report the anomaly detection performance between our proposed model and baselines in terms of AUC-ROC, AUC-PR, and Precision@K in Table III. We have the following observations: (1) For all three datasets, our proposed CFAD model has the best anomaly detection performance. Specifically, in terms of AUC-ROC, our model acquires improvements of 6.8%, 3.4%, and 3.8% on Yelp, PubMed, and Reddit datasets, respectively, compare to the best baseline. Meanwhile, the results of Precision@50, Precision@100, and Precision@200 also suggest that CFAD can better rank abnormal nodes on higher positions than baselines. For example, CFAD obtains an remarkable 32.5% improvement on Yelp in terms of Percision@50 compared to the best baseline GDN, demonstrating the advantages of CFAD and its potentials in real-world applications for detecting anomalies. This is primarily attributed to the generated counterfactual graphs that simulate the manipulations to the environment change and enhance model robustness and detection performance; (2) Notably, as an unsupervised baseline, DOMINANT [22] outperforms two supervised methods (DeepSAD and SemiGNN). One potential explanation is that DeepSAD fails to capture the adequate graph network characteristics, and SemiGNN relies on a relatively large number of labeled data to obtain the expected performance; (3) GDN, a specially designed method for few-shot anomaly detection, outperforms unsupervised method DOMINANT. This result demonstrates that leveraging supervised knowledge of labeled anomalies is an effective way for anomaly detection. Meanwhile, by learning causal relations, our proposed model CFAD is capable of obtaining accurate node representations under different environments and thus surpass GDN by a large margin. We also investigate the runtime of the models, and the experimental results suggest that CFAD is competitive compared to the deep learning-based baseline methods in terms of the runtime on these three datasets.

C. Ablation Study

We now explore the effectiveness of three important components in CFAD. We design three variants: (1) *CFAD-lr*, which removes the counterfactual graphs from the input data; (2) *CFAD-cf*, which removes the consistency regularization loss; and (3) *CFAD-cr*, which removes the contrasting loss based on the counterfactual graphs. We also ablate the encoder network of CFAD by replacing SGC with graph convolutional network (GCN) [30].

Fig. 6 reports the performance of CFAD and its three variants with different network architectures (i.e., SGC and GCN). We have the following three observations. First, by incorporating the counterfactual graphs, *CFAD-cf* largely outperforms *CFAD-lr* in anomaly detection on three datasets. For instance, *CFAD-cf* achieves 1.8% performance improvement over *CFAD-lr* on Reddit dataset in terms of AUC-ROC. When considering the counterfactual graph-based consistency regularization loss, *CFAD-cr* achieves significant improvements over *CFAD-lr*, which demonstrates that the generated counterfactual graphs can effectively benefit model training and representation learning for nodes in graphs. Second, the full model combining the three components consistently performs better than the model removing one of the components on all three datasets. For example, the performances

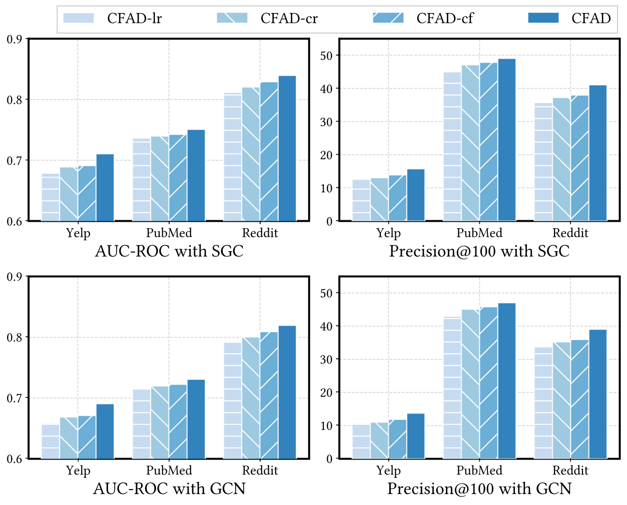


Fig. 6. Ablation study: Variants of CFAD.

of *CFAD* are 3.2%, 2.0%, and 2.2% higher than *CFAD-lr*, *CFAD-cf*, and *CFAD-cr*, respectively, on Yelp dataset. This verifies the effectiveness of our designed components based on counterfactual graphs. Finally, our proposed CFAD still obtained the best results when using the GCN architecture, which indicates that our method is model-agnostic and can be applied to other powerful GNN-architectures.

D. Data Efficiency

We now examine the labeled data efficiency of our model and baselines, since collecting a large number of labeled anomalies is very difficult in real-world anomaly detection applications. We vary the number of available labeled anomalies from 2 to 10, and show the results in Fig. 7, which covers the performance of our model and three semi-supervised baselines (SemiGNN, DeepSAD, and GDN), in terms of AUC-ROC and Precision@100. Similar results can also be observed in AUC-PR. We omit the comparison for unsupervised baselines since their performances are insensitive to labeled data.

We can see that the performances of all methods increased when more labeled data were available. The performance improvements of DeepSAD and SemiGNN are steadily increased with more labeled data. Compared to semi-supervised approaches, DeepSAD and SemiGNN, the two few-shot learning-based methods (GDN and CFAD) have better detection performance, which indicates the effectiveness of few-shot learning models on anomaly detection. While among them, CFAD is the most data-efficient method, which achieves the best average performance w.r.t. the different numbers of labeled anomalies and the fastest increasing rate. Impressively, the performance of CFAD with only six labeled anomalies exceeded the result of GDN with ten labeled anomalies. The superiority of CFAD is that it can generate a lot of counterfactual samples to benefit the model learning process.

E. Sensitivity Analysis

In this section, we analyze the sensitivity of the two critical hyper-parameters in CFAD. First, for a target node, K most

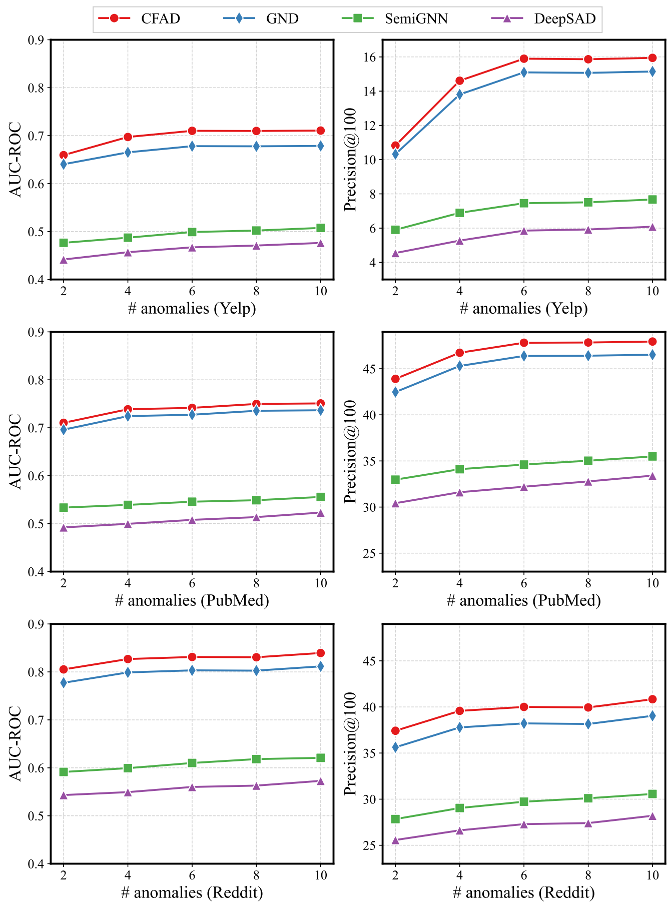


Fig. 7. Data efficiency on number of labeled anomalies.

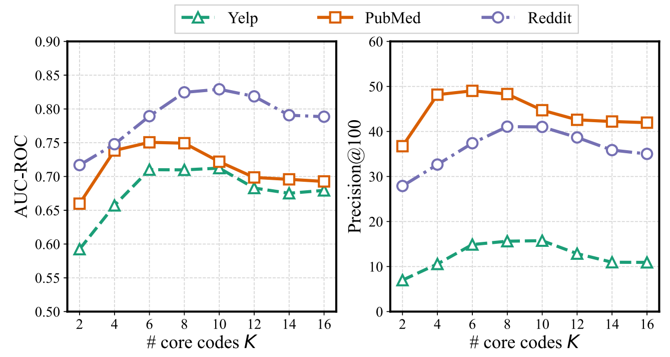


Fig. 8. Influence of core node number K.

relevant nodes are selected to form the core subgraph. We vary the number of K from 2 to 16, and present the analysis result in Fig. 8. We can see that CFAD initially achieves better performance with more nodes in the core subgraph, then the improvement approaches saturation (e.g., K ∼4-8 for PubMed dataset) and starts to decrease. We speculate that too few nodes cannot cover all the real core nodes of the target node, and many of the real core nodes are incorrectly classified into the environment subgraph. Consequently, the counterfactual subgraph cannot generate a similar or identical representation for the target node with the real core subgraph. Since synthetic nodes replace several real core nodes in the counterfactual subgraph, they

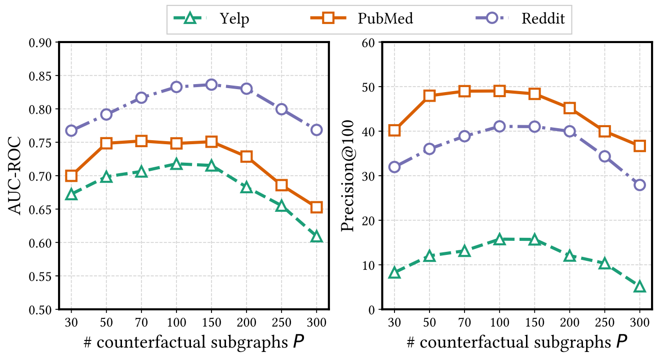


Fig. 9. Influence of counterfactual subgraph number P.

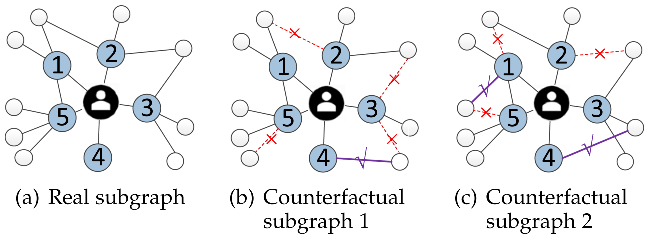


Fig. 10. Structures of the real and counterfactual subgraphs. The dotted lines denote the removed edges and the purple bold lines refer to the newly added edges. (a) Real subgraph. (b) Counterfactual subgraph 1. (c) Counterfactual subgraph 2.

eventually harm the model training process as well as detection performance. On the other hand, a larger K (i.e., too many core nodes) cannot help generate effective counterfactual subgraphs nor improve the detection performance. Within more nodes in the core subgraph, CFAD degenerates to the model without the counterfactual subgraph. In addition, the number of nodes K required to reach the performance peak is different across datasets. For example, it only needs 6 nodes for PubMed dataset but 10 nodes for Reddit dataset to have the best performance. This difference is due to the various number of nodes/edges in the graphs.

Next, we observe the influence of the number of selected synthetic subgraphs P (cf. (16)). The number of P determines how many counterfactual subgraphs are generated for model training, which is vital for obtaining a robust model. The AUC-ROC and Precision@100 with different P values are illustrated in Fig. 9. We can see that the model achieves better performance when P changes ranging from 70 to 150 for Yelp and PubMed datasets and from 100 to 200 for Reddit dataset. Too many synthetic subgraphs will decrease the anomaly detection performance. The reason behind that is when we select too many synthetic subgraphs, they often contain noised and distorted data that harm the model training and degrade the detection performance. Since there is a relatively large range of P values for obtaining the expected performance, it would not be hard for searching the proper values of P.

F. Counterfactual Subgraphs Visualization

Here, we conduct visual analysis on real subgraphs and provide the counterfactual subgraphs. We take a node from PubMed dataset as an example to conduct inspection. We depict the network structures of the real local subgraph and two generated counterfactual subgraphs in Fig. 10. The dotted line denotes that this edge exists in the real subgraph but not in the counterfactual subgraph, and the purple bold line refers to the new added edges. The central black circle represents the target node, the blue circles refer to core nodes, and the small white circles denote environment nodes. From these figures, we can see that the core subgraphs, which consist of the target node (black circle) and core nodes (blue circles), keep unchanged for both the real and counterfactual subgraphs. Instead, both counterfactual subgraphs consisting of the environment nodes (white circles), shown in Fig. 10(b) and (c), are obviously different from the real one. At the same time, the two counterfactual environment subgraphs are also different, i.e., they contains different links (edges), since the environment subgraphs are generated based on different random noises.

The attribute values of the nodes in Fig. 10 are presented in Table IV, where we only show a small number of attributes and nodes since others exhibit similar trends. This table shows that for the core nodes, although some of the attribute values in the counterfactual subgraphs (CS-1 and CS-2) are different from the ones in the real subgraphs (Real), their differences are quite small. For example, for core node 1, the average difference of the attribute values of the real and counterfactual subgraphs is only 0.0025. On the contrary, for the environment nodes, the average difference of attribute values of the real and counterfactual subgraphs is relatively large. For example, the difference of the environment node 1, 0.0117, is approximately four times bigger than the core node 1.

The above results suggest that the structures and attributes of the counterfactual subgraphs are quite similar to the real subgraph. For nodes in environment subgraphs, both structures and attributes of the counterfactual subgraphs become different from the real ones. Hence, the generated counterfactual subgraphs are differentiated from real subgraphs, but their core subgraphs possess similar structures and attributes with the real core subgraph, i.e., they can preserve the most information of the real subgraph. Therefore, the counterfactual subgraphs can be adopted to learn causal representations by dealing with environment changes and further enhance the model performance.

VI. Related Work

In this section, we review the literature in terms of network anomaly detection and causal representation learning.

A. Network Anomaly Detection

With the vast developments of communication technologies, the Internet of Things, user devices, and many other network-based services, the security capability is becoming an indispensable part of a reliable and robust network system against cyber adversaries.

Anomaly detection, especially on attributed networks, has drawn increasing research attention in the community partly due to the challenges of (1) how to handling network topologies

TABLE IV
NODE ATTRIBUTES IN REAL AND COUNTERFACTUAL SUBGRAPHS (CS: COUNTERFACTUAL SUBGRAPHS)

Attribute No.	Core node 1			Core node 2			Environment node 1			Environment node 2		
	Real	CS-1	CS-2	Real	CS-1	CS-2	Real	CS-1	CS-2	Real	CS-1	CS-2
1	0.0612	0.0589	0.0622	0.1225	0.1117	0.1196	0.0661	0.0322	0.0223	0.0509	0.0221	0.0319
2	0	0	0	0	0	0	0	0.0041	0.0038	0	0.0019	0.0016
3	0.0383	0.0417	0.0365	0.0383	0.0357	0.0419	0.0138	0.0068	0.0066	0	0.0036	0.0031
4	0	0	0	0	0	0	0	0.0109	0.0099	0	0.0068	0.0056
5	0.2059	0.1975	0.2192	0	0.0005	0	0	0	0	0	0	0
6	0	0.0004	0.0002	0	0	0	0	0.0105	0.0096	0	0.0053	0.0044

and node features in non-Euclidean space and (2) how to detect anomaly nodes when considering complex node interactions with rich node features. Many recent researches adopt the graph neural network (GNN) architectures to solve the network anomaly detection problem due to its strong representation learning power [6], [43] for aggregating neighboring nodes in graphs. Supervised methods are prevailing in node classification and link prediction tasks. However, since it is challenging to collect massive labeled data for network anomaly detection (e.g., for a newly launched system or when the network encounters new types of attacks), semi-supervised and unsupervised models are widely used for label-limited anomaly detection in networks. These models focus on designing new mechanisms that connect node features with local structures for measuring the node abnormality.

Among them, one group of detection models adopts a *mesoscopic* view and studies the node feature and structure similarities in local community or ego-network. Another group learns anomaly nodes in the feature subspace, e.g., user preference is incorporated into the processes of clusters extraction and outliers detection [44]. Besides, many detection models are proposed to address specific issues associated with network anomaly detection such as data sparsity, scalability, relational dependencies, interpretability, and training complexity [23].

Specifically, DOMINANT [22] builds a deep autoencoder architecture on top of GCN, achieving superior performance compared to other shallow models. GAS [45] is another GCN-based large-scale anti-spam method for detecting spam advertisements. ComGA [46] presents a community-aware attributed graph anomaly detection model that incorporates a tailored deep GCN. ResGCN [47] is a residual-based GCN aiming to address the problems of sparsity and nonlinearity capturing, residual modeling, and network smoothing for efficient network anomaly detection. Contrastive learning and self-supervised learning are also introduced to detect the anomalies in attributed networks [25], [48].

Other studies focus on designing semi-supervised detection model that assumes a few labeled samples are available for model training. For instance, DevNet [5] fulfills an end-to-end framework of computing anomaly scores with a neural deviation learning, in which a few labeled anomalies are adopted to enforce statistically significant deviations of the anomaly scores of anomalies from that of normal data objects in the upper tail. SemiGNN [41] is a semi-supervised GNN that adopts hierarchical attention to model the multi-view graph for fraud detection. A cross-domain anomaly detection model COMMANDER is proposed to improve the detection performance for unlabeled target graph by leveraging another labeled source graph. A multi-hypersphere graph learning approach MHGL is designed to effectively leverage existing labels by learning fine-grained normal patterns to distinguish both seen and unseen anomalies. Meta-learning, few-shot learning, and one-class classification algorithms are also used to address the problem of labeled data shortage and are incorporated into network anomaly detection models for improving the label-efficiency, given the fact that anomaly samples are hard to obtain in practical situations. For example, Meta-PN [49] is a GNN-based meta-learning algorithm that uses a decoupled network architecture to infer high-quality pseudo labels on unlabeled nodes via meta-label propagation strategy. Few-shot learning on network anomaly detection is studied in [33], where the authors propose to incorporate prior knowledge to learn anomaly-informed models, reducing the cost of collecting valuable labeled anomalies. Specifically, a graph deviation network (GDN) first assigns anomaly scores to nodes in the graph using GNNs. Then it defines the anomaly mean based on prior probability to guide the subsequent anomaly score learning. Readers can refer [23] for a comprehensive review on deep learning-based graph anomaly detection, including the above mentioned models.

B. Causal Representation Learning

Causal representation learning aims to learn a representation exposing the causal relations that are invariant under different interventions [14]. Generating counterfactual samples is an effective way to remove spurious correlations and help learn causal representations, and has attracted considerable attention in visual and language learning [10]. For example, GANs are commonly used to produce counterfactual images, which learn causal representations for enhancing model performance. In [12], the authors consider changing domains as interventions on images under the data-generation process and steer the generative model to produce counterfactual features, which helps the model learn causal relations across different domains. Image generation process is decomposed into independent causal mechanisms in [10]. Different proper generators are used to produce image shape, texture, and background. Then the produced elements are combined to infer counterfactual images. Steer generative models are proposed in [11] to manufacture interventions on features caused by confounding factors, such as viewpoint or background, for learning causal representations. Researchers also design a model to generate counterfactuals by incorporating a structural causal model in an improved variant of Adversarially Learned Inference for image classification [50].

Due to the data structure differences, these methods of generating counterfactual images cannot be applied to graph-structured data, which motivates us to design a novel Granger causality-based causal explanation method to extract the core subgraphs. Keeping core subgraphs unchanged allows us to steer the generative model to perform interventions on environment subgraphs for generating the counterfactual subgraphs. Meanwhile, by incorporating the newly designed counterfactual graph-based contrasting loss and consistency regularization term, our model achieves significant improvements on network anomaly detection.

VII. Conclusions

In this article, we proposed a counterfactual graph-based anomaly detection model CFAD, which can conduct anomaly detection with varying environments based on a few labeled anomaly nodes. We designed a novel intervention mechanism by steering the generator to manufacture the synthetic environment features. These features are combined with the real representations of the core nodes to generate counterfactual subgraphs, which help the detection model learn transferable and causal relations across different environments without relying on many labels. Extensive experiments show the proposed approach achieves state-of-the-art performance on three public real-world attributed network datasets.

For future work, we plan to explore the node dependency and network structure to further enhance the designed model. Also, we will apply the method of generating counterfactual subgraphs to other applications, such as node classification, graph classification and link prediction. We are interested in investigating whether this approach can be helpful to eliminate spurious correlations during model training and enhance the performance for these tasks.

References

[1] L. Ruff et al., "A unifying review of deep and shallow anomaly detection," *Proc. IEEE*, vol. 109, no. 5, pp. 756–795, May 2021.

[2] Y. Liu et al., "Anomaly detection in dynamic graphs via transformer," *IEEE Trans. Knowl. Data Eng.*, early access, Nov. 2, 2021, doi: 10.1109/TKDE.2021.3124061.

[3] 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.

[4] P. Jiao et al., "Generative evolutionary anomaly detection in dynamic networks," *IEEE Trans. Knowl. Data Eng.*, early access, Nov. 22, 2021, doi: 10.1109/TKDE.2021.3129057.

[5] G. Pang, C. Shen, and A. van denHengel, "Deep anomaly detection with deviation networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 353–362.

[6] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surveys*, vol. 54, no. 2, pp. 1–38, 2021.

[7] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 1521–1528.

[8] R. Geirhos et al., "Shortcut learning in deep neural networks," *Nature Mach. Intell.*, vol. 2, no. 11, pp. 665–673, 2020.

[9] D. Krueger et al., "Out-of-distribution generalization via risk extrapolation (REx)," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 5815–5826.

[10] A. Sauer and A. Geiger, "Counterfactual generative networks," in *Proc. Int. Conf. Learn. Representations*, 2021.

[11] C. Mao, A. Cha, A. Gupta, H. Wang, J. Yang, and C. Vondrick, "Generative interventions for causal learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3947–3956.

[12] X. Zhang et al., "Learning causal representation for training cross-domain pose estimator via generative interventions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 11 270–11 280.

[13] J. Peters, D. Janzing, and B. Schölkopf, *Elements of Causal Inference: Foundations and Learning Algorithms*. Cambridge, MA, USA: MIT Press, 2017.

[14] S. Yang, K. Yu, F. Cao, L. Liu, H. Wang, and J. Li, "Learning causal representations for robust domain adaptation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2750–2764, Mar. 2023.

[15] J. Pearl, *Causality*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[16] R. Suter, D. Miladinovic, B. Schölkopf, and S. Bauer, "Robustly disentangled causal mechanisms: Validating deep representations for interventional robustness," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6056–6065.

[17] W. Lin, H. Lan, and B. Li, "Generative causal explanations for graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6666–6679.

[18] 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, Jan. 2021.

[19] S. L. Bressler and A. K. Seth, "Wiener–Granger causality: A well established methodology," *Neuroimage*, vol. 58, no. 2, pp. 323–329, 2011.

[20] D. Luo et al., "Parameterized explainer for graph neural network," in *Proc. Conf. Neural Inf. Process. Syst.*, 2020, pp. 19 620–19 631.

[21] F. Johansson, U. Shalit, and D. Sontag, "Learning representations for counterfactual inference," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 3020–3029.

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

[23] X. Ma et al., "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Trans. Knowl. Data Eng.*, early access, Oct. 8, 2021, doi: 10.1109/TKDE.2021.3118815.

[24] X. Wang, Y. Wu, A. Zhang, X. He, and T.-S. Chua, "Towards multi-grained explainability for graph neural networks," in *Proc. Conf. Neural Inf. Process. Syst.*, 2021, pp. 18446–18458.

[25] 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.

[26] H. Qin, X. Zhan, and Y. Zheng, "CSCAD: Correlation structure-based collective anomaly detection in complex system," *IEEE Trans. Knowl. Data Eng.*, early access, Mar. 22, 2022, doi: 10.1109/TKDE.2022.3154166.

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

[28] Y. Ye and S. Ji, "Sparse graph attention networks," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 905–916, Jan. 2023.

[29] J. Li et al., "Higher-order attribute-enhancing heterogeneous graph neural networks," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 560–574, Jan. 2023.

[30] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6861–6871.

[31] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2794–2802.

[32] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," in *Proc. Int. Conf. Learn. Representations*, 2016.

[33] K. Ding, Q. Zhou, H. Tong, and H. Liu, "Few-shot network anomaly detection via cross-network meta-learning," in *Proc. Web Conf.*, 2021, pp. 2448–2456.

[34] A. Abuduweili, X. Li, H. Shi, C.-Z. Xu, and D. Dou, "Adaptive consistency regularization for semi-supervised transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6923–6932.

[35] Z. Li, X. Chen, J. Song, and J. Gao, "Adaptive label propagation for group anomaly detection in large-scale networks," *IEEE Trans. Knowl. Data Eng.*, early access, May 20, 2022, doi: 10.1109/TKDE.2022.3176478.

[36] M. M. Breunig, H. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 93–104.

[37] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 665–674.

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

[39] P. I. Sánchez, E. Müller, O. Irmler, and K. Böhm, "Local context selection for outlier ranking in graphs with multiple numeric node attributes," in *Proc. Sci. Statist. Database Manage. Conf.*, 2014, pp. 16:1–16: 12.

[40] J. Li, H. Dani, X. Hu, and H. Liu, "Radar: Residual analysis for anomaly detection in attributed networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 2152–2158.

[41] D. Wang et al., "A semi-supervised graph attentive network for financial fraud detection," in *Proc. IEEE Int. Conf. Data Mining*, 2019, pp. 598–607.

[42] L. Ruff et al., "Deep semi-supervised anomaly detection," in *Proc. Int. Conf. Learn. Representations*, 2020.

[43] L. Huang et al., "Hybrid-order anomaly detection on attributed networks," *IEEE Trans. Knowl. Data Eng.*, early access, Oct. 6, 2021, doi: 10.1109/TKDE.2021.3117842.

[44] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 1346–1355.

[45] A. Li, Z. Qin, R. Liu, Y. Yang, and D. Li, "Spam review detection with graph convolutional networks," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 2703–2711.

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

[47] Y. Pei, T. Huang, W. van Ipenburg, and M. Pechenizkiy, "ResGCN: Attention-based deep residual modeling for anomaly detection on attributed networks," *Mach. Learn.*, vol. 111, no. 2, pp. 519–541, 2022.

[48] Y. Zheng, M. Jin, Y. Liu, L. Chi, K. T. Phan, and Y.-P. P. Chen, "Generative and contrastive self-supervised learning for graph anomaly detection," *IEEE Trans. Knowl. Data Eng.*, early access, Oct. 12, 2021.

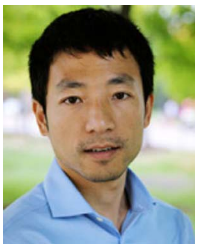
[49] K. Ding, J. Wang, J. Caverlee, and H. Liu, "Meta propagation networks for graph few-shot semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 6524–6531.

[50] S. Dash, V. N. Balasubramanian, and A. Sharma, "Evaluating and mitigating bias in image classifiers: A causal perspective using counterfactuals," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 915–924.

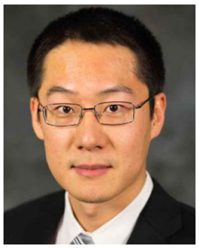
Chunjing Xiao received the PhD degree from the University of Electronic Science and Technology of China, Chengdu, China. He is currently an associate professor with the School of Computer and Information Engineering, Henan University, Kaifeng, China. He was a visiting scholar with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA. His current research interests include recommender systems, anomaly detection, and Internet of Things.

Xovee Xu (Graduate Student Member, IEEE) received the BS and MS degrees in software engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, China, in 2018 and 2021, respectively. He is currently working toward the PhD degree in computer science with UESTC. His research interests include social network data mining and knowledge discovery, primarily focuses on information diffusion in networks, human behavior understanding, and spatial-temporal data modeling.

Yue Lei received the BE degree from the School of Computer and Information Engineering, Henan University, Kaifeng, China, in 2019. He is currently working toward the MS degree with Henan University. His current research interests include Internet of Things, anomaly detection, and data analytics.



Kunpeng Zhang received the PhD degree in computer science from Northwestern University, USA. He is assistant professor with the Robert H. Smith School of Business, University of Maryland, College Park. He is interested in large-scale data analysis, with particular focuses on social data mining, image understanding via machine learning, social network analysis, and causal inference. He has published papers in the area of social media, artificial intelligence, network analysis, and information systems on various conferences and journals.



Siyuan Liu received the 1st PhD degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, and the 2nd PhD degree from the University of Chinese Academy of Sciences. He is an assistant professor with the Management Information Systems and Dr. John Coyle Early Career professor with the Department of Supply Chain & Information Systems, Smeal College of Business, Pennsylvania State University. His research interests include spatial and temporal data mining, social networks analytics, and data-driven behavior models.

Fan Zhou received the BS degree in computer science from Sichuan University, Chengdu, China, in 2003, and the MS and PhD degrees from the University of Electronic Science and Technology of China, Chengdu, in 2006 and 2012, respectively. He is currently a professor with the School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests include machine learning, neural networks, spatio-temporal data management, graph learning, recommender systems, and social network data mining.