Federated Graph Anomaly Detection Through Contrastive Learning with Global Negative Pairs

Nannan Wu, Yazheng Zhao, Hongdou Dong, Keao Xi, Wei Yu, Wenjun Wang

College of Intelligence and Computing, Tianjin University, Tianjin, China
{nannan.wu, yazhengzhao, hongdong_dou, xikeao_99, wjwang}@tju.edu.cn, weiyu@zyufl.edu.cn

Abstract

Anomaly detection on attributed graphs has applications in various domains such as finance and email spam detection, thus gaining substantial attention. Distributed scenarios can also involve issues related to anomaly detection in attribute graphs, such as in medical scenarios. However, most of the existing anomaly detection methods are designed for centralized scenarios, and directly applying them to distributed settings may lead to reduced performance. One possible reason for this issue is that, when graph data are distributed across multiple clients, federated graph learning may struggle to fully exploit the potential of the dispersed data, leading to suboptimal performance. Building on this insight, we propose FedCLGN, a federated graph anomaly detection framework that leverages contrastive self-supervised learning. First, we put forward an augmentation method to maintain global negative pairs on the server. This involves identifying anomalous nodes using pseudo-labels, extracting embedding representations of the negative pairs corresponding to these anomalous nodes from clients, and uploading them to the server. Then, we adopt graph diffusion to enhance the feature representation of nodes, capturing the global structure and local connection patterns. This strategy can strengthen the differentiation between positive and negative instance pairs. Finally, the effectiveness of our approach is verified by experimental results on four real graph datasets.

Introduction

The main objective of anomaly detection on attribute graphs is to identify nodes that deviate from normal patterns.(Chandola, Banerjee, and Kumar 2009). Traditional anomaly detection methods primarily focus on simple data that lack complex linkage relationships, such as a set of discrete data points. However, as intricate networks such as social and biological networks continue to emerge, the detection of anomalies in attribute graphs has gained significant importance(Ma et al. 2021). In contrast to structured data, attribute graphs contain more intricate information, including topological structures and node-specific attributes. Therefore, identifying anomalous nodes within such a complex environment poses considerable challenges for anomaly detection in attribute graphs.

Conventional centralized methods for anomaly detection often require direct access to sensitive user information, which raises substantial privacy concerns(Pazho et al. 2023). In scenarios with limited information resources, federated learning has become a robust solution, leveraging its ability to facilitate distributed collaborative training(Wang et al. 2024a). Recent years have witnessed a surge of work focused on detecting anomalies in attribute graphs within distributed environments. Nevertheless, performing anomaly detection tasks directly in distributed settings can lead to performance degradation in federated graph learning, which primarily dues to restricted data access during the training process(Wang et al. 2024c).

In this paper, we propose FedCLGN, a framework dedicated to federated graph anomaly detection. In the face of the primary challenge of data privacy protection, FedCLGN strictly follows the principles of federated learning by not directly accessing client data but instead performing parameter aggregation on the server side. Addressing the challenge of scarce anomaly information in a federated environment, we propose a groundbreaking innovation. FedCLGN shares structural information of the global graph (excluding node features) on the server and uploads the embedding representations of negative pairs corresponding to anomalous nodes. By leveraging graph diffusion techniques, it facilitates the exchange of negative pair information among clients, thereby maintaining a global negative pair pool. Using negative pairs sampled from the global negative pair pool in subsequent contrastive learning on clients makes the difference between positive and negative sample pairs in the contrastive learning more pronounced, thereby improving the accuracy of anomaly detection. It is noteworthy that FedCLGN does not directly share node features, but instead shares the embedding representations of the negative pairs corresponding to the nodes. More crucially, FedCLGN only uploads the embedding representations of negative pairs corresponding to anomalous nodes to the server. These two key designs jointly ensure data privacy and security.

In this paper, we propose FedCLGN.The primary contributions of this paper can be summarized as follows:

- We propose FedCLGN, a novel framework for federated graph anomaly detection that addresses the challenge of detecting anomalies in attribute graphs within a federated environment lacking critical information. With distinc-

tive strategies, FedCLGN achieves high performance in anomaly detection while ensuring privacy protection.

- We propose a new global negative pairs strategy. This strategy disseminates negative pairs information corresponding to anomalous nodes from various clients throughout the entire global graph and maintains a global negative pair information pool. Consequently, it amplifies the discernible differences between positive and negative instance pairs in contrast learning, ultimately enhancing the accuracy and robustness of anomaly detection tasks based on contrastive learning.
- We evaluated the proposed FedCLGN on four datasets. The experimental results not only verify the effectiveness of our algorithm compared to baseline methods but also demonstrate the robustness of our approach.

The remainder of this paper is organized as follows. We provide an overview of the related work in section 2 . We provide a detailed definition of anomaly detection for attribute network nodes in a distributed environment in section 3 . A comprehensive explanation of the technical details of FedCLGN is provided in section 4 . The experimental results are presented and analyzed in section 5 . We draw a conclusion of our work in section 6 .

Related Work

In this section, we review the most relevant work, including anomaly detection on attributed graphs, contrastive self-supervised learning and federated graph learning.

Anomaly Detection on Attributed Graphs

Given its significant potential in fields like finance and chemistry, anomaly detection in attribute graphs has garnered growing attention(Pang et al. 2021). In the early stages, researchers applied some shallow methods relying on feature engineering for anomaly detection in attribute graphs. AMEN (Perozzi and Akoglu 2016) proposes and utilizes normality as a metric to measure anomalies in the neighborhood of nodes. Radar (Li et al. 2017) Radar learns the residuals of node attribute information to identify anomalies that are distinctly different from most nodes. While early shallow methods have been successful, their heavy dependence on manual feature engineering results in high costs and reduced performance as the network structure and attributes grow more complex (Ruff et al. 2021).

Recently, Graph Neural Networks (GNNs) have shown their remarkable effectiveness in domains like Natural Language Processing (NLP) and recommendation systems. (Zhou et al. 2022). Naturally, GNNs have also been utilized for anomaly detection tasks in attribute graphs. DOMINANT (Ding et al. 2019)detects anomalies by leveraging the differences between the reconstructed attribute and adjacency matrices and their original matrix. SpecAE (Li et al. 2019) enhances the distinction between abnormal and normal node representations in the embedding space.

Contrastive Self-Supervised Learning

Contrastive self-supervised learning has gained widespread attention because of its ability to obtain supervised information directly in the data through clever design (Liu et al. 2021a). Naturally, contrastive self-supervised learning is also applied to the field of graph learning. DGI Veličković et al. (2018) utilizes node attribute information to enhance the graph and construct pairs of node-graph contrastive instances to learn node embeddings. GCC (Qiu et al. 2020) facilitates subgraph-level contrastive learning through pretraining tasks and InfoNCE loss.

Although there are many graph learning methods based on contrastive self-supervised learning, most of them are not directly designed for anomaly detection in attribute graphs. CoLA (Liu et al. 2021b) is the first work to utilize contrastive self-supervised learning for anomaly detection, and its designed node-subgraph contrastive pairs can better facilitate the calculation of anomaly scores. Despite the great success of CoLA in centralized environments, anomaly detection in attribute graphs within a distributed setting continues to pose challenges.

Federated Graph Learning

Federated learning is a distributed learning paradigm proposed to safeguard data privacy (Wang et al. 2024b). Federated graph learning is an important branch of federated learning, specifically designed as a federated paradigm for distributed graph data scenarios such as medical application scenarios. FedCG (Caldarola et al. 2021) addresses data heterogeneity across clients through clustering methods and then uses Graph Convolutional Networks (GCNs) to share knowledge across clients. Feddy (Jiang et al. 2020) devises a decryption mechanism for aggregating model parameters in a privacy-preserving context.

Problem Statement

Definition 1 (Attributed Graphs) An attributed graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ represents the set of nodes, $\mathcal{E}$ represents the set of edges, and $\mathbf{X} \in \mathbf{R}^{n \times d}$ represents the attribute matrix. The i^{th} row vector $\mathbf{x}_i \in \mathbf{R}^d$ of the attribute matrix represents the attribute vector of the i^{th} node. The structure of the attribute graph is represented as a binary adjacency matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$, where the existence of a link between nodes v_i and v_j is represented by $\mathbf{A}_{i,j} = 1$, otherwise $\mathbf{A}_{i,j} = 0$.

Definition 2 (Graph Diffusion) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is an undirected graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$. The generalized graph diffusion via the diffusion matrix can be represented as $\mathbf{S} = \sum_{k=0}^{\infty} \Theta_k \mathbf{T}^k$, where Θ_k are the coefficients and $\mathbf{T}^k \in \mathbf{R}^{n \times n}$ defines the transfer matrix.

Definition 3 (Subgraph Federated Learning) Assume there is a server and M clients in the FL system. $\mathcal{G}_{C_i} = (\mathcal{V}_{C_i}, \mathcal{E}_{C_i}, \mathbf{X}_{C_i})$ is a subgraph of client C_i, for $C_i \in [M]$. Note that there are certain differences in the number of nodes and graph structure of each client, and no overlapping nodes between different clients.

Node Anomaly Detection in Distributed Systems. Assuming a distributed system comprising M clients, where $\mathcal{G}_{Ck}$ represents the data held by client C_k, our objective is to train an anomaly score function f in this decentralized environment. In federated scenarios, different clients have different

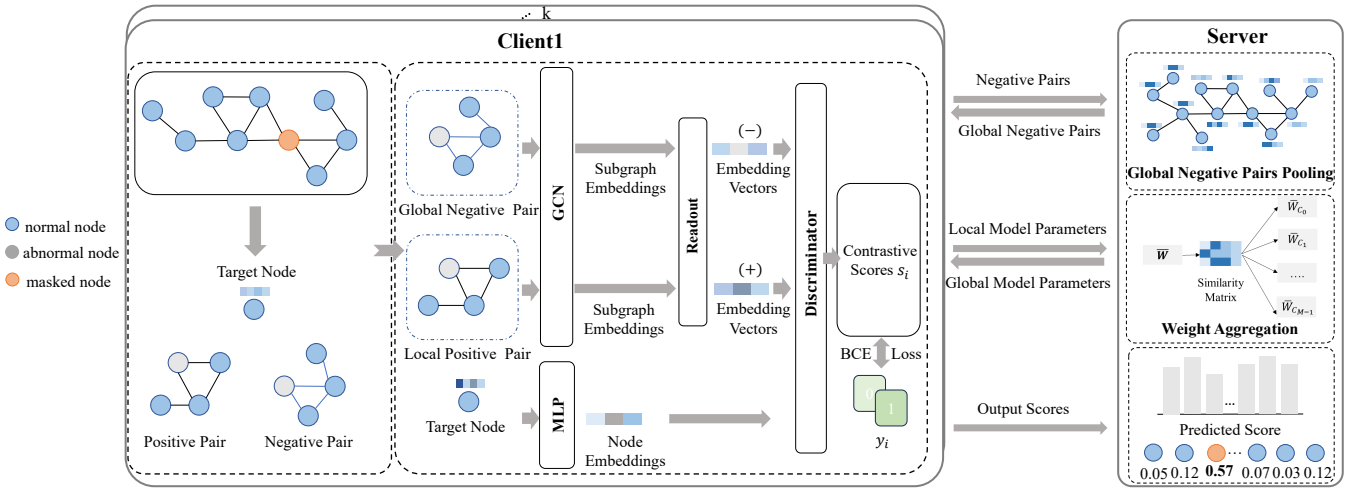

Figure 1: Overview of the FedCLGN framework. The left box represents the contrastive learning-based anomaly detection performed on a client, and the right box represents the global negative pair maintenance, parameter aggregation, and output abnormal nodes performed on the server. Note that the local training process discovers the pseudo-label $\bar{Y}$ based on the confidence threshold λ and uploads embedding representation of negative pairs corresponding to anomalous node $\mathbf{U}$ to the server. The server maintains the embedding representation of global negative pairs and sends the global negative pair features obtained by diffusion to each client. Finally, the client predicts anomaly nodes by calculating the positive and negative pairwise scores corresponding to the nodes and sends the results to the server.

definitions of anomalies, and anomalies are invisible to each other. Traditional federated learning frameworks such as FedAvg (McMahan et al. 2017), which simply aggregate and update parameters, can lead to suboptimal models. The simple aggregation update can be expressed as:

$$\mathcal{W}^t \leftarrow \frac{N_k}{N} \sum_{k=1}^{K} \mathcal{W}_k^t \tag{1}$$

where $\mathcal{W}$ represents the parameters of the global anomaly score function in the t-th round, $\mathcal{W}_k^t$ denotes the parameters of the anomaly score function on the k-th client in the t-th round, and K represents the number of clients. And, N_k represents the total number of nodes for the k-th client, and N represents the total number of nodes for all clients.

However, the global model ultimately obtained through such a simple parameter update may not be applicable to anomaly detection tasks on all clients. To address this, we enhance the model's performance on all clients by updating the parameters with the assistance of global negative pairs. Inspired by FedPub(Baek et al. 2023), we adopt a personalized approach to parameter aggregation and utilize global negative pairs to guide the calculation of similarity. Specifically, as follows:

$$\mathcal{W}_i^t \leftarrow \sum_{j=1}^{K} \alpha_{ij} \mathcal{W}_j^t \tag{2}$$

$$\alpha_{ij} = \frac{exp(S(i,j;\mathbf{P}^t)}{\sum_{k=1}^{K} exp(S(i,k;\mathbf{P}^t)} \tag{3}$$

where α_{ij} represents the similarity between client i and cilent j, $\mathbf{P}^t$ denotes the set of global negative pairs in the t-th round, and similarity calculation function S is influenced by global negative pairs.

Methodology

In this section, we describe the overall framework of FedCLGN, as shown in Figure 1. It specifically includes three parts: local contrastive model training, maintenance of global negative pairs, and aggregation of parameter updates. The contrastive learning model is first trained on the client to capture the inconsistencies between node and its local neighbors. To address the problem of limited data on a single client side, we utilize the classic federated learning algorithm FedPub to implement parameter aggregation and personalized updates between each client side. Thirdly, the server aggregates the negative pair embeddings corresponding to the anomaly nodes uploaded by the clients, obtains global negative pairs through graph diffusion, and sends them to each client to complete the training process. And the overall procedure of FedCLGN is depicted in Algorithm 1.

Global Negative Pairs

Previous work on graph anomaly detection based on contrast learning has been for centralized data, where positive and negative pairs are sampled on full graph data. However, with distributed data, we encounter two challenges. Firstly, each client's data should be linked relationships, and there may be missing edges between them that are not captured by a single data owner, but these linking relationships may carry important information, thus reducing the effectiveness of anomaly detection. Concurrently, each client's data capacity is intrinsically constrained, and the sparsity of the

graph data leads to the fact that the negative pairs sampled in contrastive learning tend to coincide with the positive pair nodes, resulting in diminishing their differentiation, and ultimately also resulting in a lower performance of model.

To improve the performance of the model, we share the entire graph structure on the server and maintain a global node feature embedding, which is used for negative pair sampling across clients. Next, we will detail the generation and update of the global negative pair.

Generation. Negative pairs play a key role in contrastive learning approach, which makes the model more focused on the characteristic anomalies of the nodes. Inspired by this, we set up public negative pairs on the server side for individual clients to use, which allows a certain degree of information exchange between different clients and compensates for the lack of information. We sample pseudo-labeling methods to obtain anomalous nodes for different clients. Specifically, we extract these prediction scores from the final anomalous node score S. If the predicted score for the k^{th} node in S_k exceeds the threshold, it is considered a pseudo label and the formula is expressed as:

$$\bar{Y}_{C_i}^k = \begin{cases} 1, & \text{if } (S_k > \lambda) \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where λ between $(-1, 0)$, serves as the confidence threshold to determine the pseudo label. And $\bar{Y}_{C_i}^k$ represents the pseudo label for the k^{th} target node of client C_i.

Then, we extract the negative pairs embedding representations $\mathbf{U}_{C_i}$ corresponding to all nodes with $\bar{Y}_{C_i}^k$ = 1 and upload them to the server. This approach effectively safeguards data privacy by ensuring that only the negative pairs representations associated with anomalous nodes are uploaded. Secondly, given the small proportion of anomalous nodes, uploading only their corresponding negative pairs representation vectors helps reduce the communication overhead.

Update. After the server receives $\mathbf{U}$ from each client, the global graph's negative pairs characteristics are described as:

$$\mathbf{X_G} = \bigcup_i \mathbf{U}_{C_i} \tag{5}$$

We address the issue of cross-client missing information by updating global negative pairs embeddings. To further improve this, we apply graph diffusion on the server to refine the global negative pairs pooling. Here, we adopt a particular approach to graph diffusion, personalized PageRank (PPR) to enhance the node embedding representation.

$$\mathbf{S}_{t+1}^{\mathrm{PPR}} = \alpha \mathbf{v} + (1-\alpha)\mathbf{A}\mathbf{S}_t^{\mathrm{PPR}} \tag{6}$$

where α is the transfer probability, t is the diffusion time.

These enhanced global negative pairs representations further improve our anomaly detection in federated learning, especially in coping with the problem of missing information across clients. Clients can then utilize this global negative pairs pooling for contrastive learning.

Local Anomaly Detection Module

To capture anomaly-aware supervised signals from raw graph data in an unsupervised manner, and inspired by CoLA (Liu et al. 2021b), we incorporate an anomaly detection module. It consists of three components: **instance pair sampling**, **a GNN-based contrastive learning model**, and **anomaly score calculation**.

Instance Pair Sampling. The success of contrastive learning largely depends on the pairs of data instances to be compared. In graph contrastive learning, data instance pairs that have been verified to be feasible can be divided into three categories: "node-node", "node-subgraph", and "subgraph-subgraph". To model the local distribution pattern of nodes in a network, we use "target node-local subgraph" as the contrastive instance pair. The first element can be any node in the network as the target node. The second element is a local subgraph sampled from an initial node. If the subgraph is sampled from the target node, it is a positive instance pair. Starting from the sub-sampling nodes other than the target, to obtain a negative instance pair.

A GNN-based Contrastive Learning Model. The sampled instance pair P_i can be denoted as :

$$P_i = (v_i, \mathcal{G}_i, y_i) \tag{7}$$

where v_i is the target node with an attribute vector $\mathbf{x}_{v_i}$, and $\mathcal{G}_i$ represents the local subgraph, which can be denoted as:

$$\mathcal{G}_i = (\mathbf{A}_i, \mathbf{X}_i) \tag{8}$$

and y_i is the label of P_i which can be denoted as:

$$y_i = \begin{cases} 1, P_i \text{ is a positive instance pair} \\ 0, P_i \text{ is a negative instance pair.} \end{cases} \tag{9}$$

First, the local subgraph $\mathcal{G}_i$ is fed into the GCN layer to obtain neighbor subgraph embedding. A layer of GCN can be expressed as:

$$\mathbf{H}_i^{(\ell+1)} = \phi\left(\widetilde{\mathbf{D}}_i^{-\frac{1}{2}}\widetilde{\mathbf{A}}_i\widetilde{\mathbf{D}}_i^{-\frac{1}{2}}\mathbf{H}_i^{(\ell)}\mathbf{W}^{(\ell)}\right) \tag{10}$$

where $\mathbf{H}_i^{\ell}$ is the input for the convolution layer ℓ, and $\mathbf{H}_i^{(\ell+1)}$ is the output after the convolution layer. $\widetilde{\mathbf{A}}_i = \mathbf{A}_i + \mathbf{I}$ is the subgraph adjacency matrix with selfloop, $\widetilde{\mathbf{D}}_i$ is the degree matrix of local subgraph, $\mathbf{W}^{(\ell)} \in \mathbf{R}^{d^{(\ell)} \times d^{(\ell+1)}}$ is the weight matrix of the (ℓ)-th layer. Note that $\phi(\cdot)$ is a non-linear activation function, such as $Relu(x) = max(0, x)$. With a *L*-layer GCN, the GCN model can be written as:

$$\mathbf{E}_i = \phi\left(\widetilde{\mathbf{D}}_i^{-\frac{1}{2}}\widetilde{\mathbf{A}}_i\widetilde{\mathbf{D}}_i^{-\frac{1}{2}}\mathbf{H}_i^{(\ell)}\mathbf{W}^{(\ell)}\right) \tag{11}$$

where $\mathbf{E}_i$ is the embeddings of subgraph nodes. Then, the embeddings of nodes in subgraph $\mathbf{E}_i$ are transformed into a local subgraph embedding vector $\mathbf{e}_i^{lg}$ by an average pooling function. Specifically, the readout function is written as:

$$\mathbf{e}_i^{\mathrm{lg}} = \sum_{k=0}^{n_i-1} \frac{(\mathbf{E}_i)_k}{n_i} \tag{12}$$

Algorithm 1: The Proposed FedCLGN Algorithm

Input: $\{\mathcal{G}_{C_k} = (\mathcal{V}_{C_k}, \mathcal{E}_{C_k}, \mathbf{X}_{C_k})\}_{C_k=1}^{K}$.
Output: A set of anomalous nodes L.

$\mathcal{W}_k^0 \leftarrow \mathcal{W}^0, k = 1, \cdots, K$;
$B \leftarrow \mathcal{B}$;
$T_c \leftarrow \mathcal{T}_c$;
$\mathbf{X_G}, \mathrm{L}_s \leftarrow \phi$;
for t ∈ 1,2,$\cdots$,T_c **do**
  // For each Client C_k:
  // Train stage.
  $(P_1^{(+)}, \cdots, P_B^{(+)})_{v_i} \sim \mathcal{N}(v_i), \forall v_i \in \mathcal{V}_{C_k}$;
  if t=1 **then**
    $(P_1^{(-)}, \cdots, P_B^{(-)})_{v_i} \sim \mathcal{N}(v_j), \forall v_i \neq v_j \in \mathcal{V}_{C_k}$;
  else
    //Accept global negative pairs from the server.
    $(P_1^{(-)}, \cdots, P_B^{(-)})_{v_i} \sim \mathcal{N}^*(v_i), \forall v_i \in \mathcal{V}_{C_k}$;
  end if
  $s_{v_i} \leftarrow$ Equation (14), $\forall v_i \in \mathcal{V}_{C_k}$;
  $\mathcal{L}_{\mathrm{cml}} \leftarrow$ Equation (15);
  // Inference stage.
  $(P_1^{(+)}, \cdots, P_B^{(+)})_{v_i} \sim \mathcal{N}(v_i), \forall v_i \in \mathcal{V}_{C_k}$;
  $(P_1^{(-)}, \cdots, P_B^{(-)})_{v_i} \sim \mathcal{N}(v_j), \forall v_i \neq v_j \in \mathcal{V}_{C_k}$;
  $s_{v_i} \leftarrow$ Equation (14), $\forall v_i \in \mathcal{V}_{C_k}$;
  $S_{v_i} \leftarrow$ Equation (16), $\forall v_i \in \mathcal{V}_{C_k}$;
  $\bar{Y}_{C_k} \leftarrow$ Equation (4);
  Upload $\mathbf{U}_{C_k}$ and L_{C_k} to the Server by $\bar{Y}_{C_i}$;
  // In Server:
  $\mathbf{X_G} \leftarrow \bigcup_k \mathbf{U}_{C_k}$;
  $\mathrm{L}_s \leftarrow \bigcup_k \mathrm{L}_{C_k}$;
  Update global negative pairs via Equation (6);
  $\mathcal{W}_k^t \leftarrow$ Equation (2), $k = 1, \cdots, K$;
end for
$\mathrm{L} \leftarrow \mathrm{L}_s$;

where $(\mathbf{E}_i)_k$ is the k-th row of $\mathbf{E}_i$, and n_i is the number of nodes of the local subgraph $\mathcal{G}_i$. And the target node also be mapped to the same embedding space:

$$\mathbf{e}_i^{\mathrm{tn}} = \phi(\mathbf{x}_{v_i}\mathbf{W}^{(0)}) \tag{13}$$

where $\mathbf{x}_{v_i}$ is the attribute vector of target node, and $\mathbf{W}^0$ is the weight matrix shared with GCN.

Here, we apply a bilinear function as a discriminator to calculate the prediction score:

$$s_i = Discriminator(\mathbf{e}_i^{lg}, \mathbf{e}_i^{tn}) = \sigma\left(\mathbf{e}_i^{lg}\mathbf{W}^{(d)}\mathbf{e}_i^{tn\top}\right) \tag{14}$$

where $\mathbf{W}^d$ is the weight matrix of discriminator, and $\sigma(\cdot)$ is the logistic sigmoid function.

As the objective function, we adopt a standard binary crossy-entropy (BCE) loss, as follows:

$$\mathcal{L}_{\mathrm{cml}} = -\sum_{i=1}^{B} y_i \log(s_i) + (1 - y_i)\log(1 - s_i). \tag{15}$$

Anomaly Score Calculation. After training, contrastive learning models tend to learn the patterns of normal nodes as they account for the vast majority of the training data. Ideally, a normal node should have a predicted score of its positive pair $s^{(+)}$ that is close to 1, while its negative pair $s^{(-)}$ should have a score near 0. In contrast, for an anomalous node, the predicted scores for both positive and negative pairs would be less distinct (around 0.5), as the model struggles to differentiate its matching pattern effectively. Given these properties, we can define the anomaly score for each node as the difference between its negative and positive scores. For normal nodes, this score is negative, while for abnormal nodes, it approaches zero. Therefore, a higher anomaly score indicates a higher likelihood of the node being anomalous. The anomaly score for node v_i is given as:

$$f(v_i) = s_i^{(-)} - s_i^{(+)} \tag{16}$$

However, the local subgraph of sampling provides only a partial view of the neighboring structures of the target node and fails to capture the entire neighbor distribution. This incomplete observation can lead to a lack of understanding of the anomaly, affecting the anomaly detection performance. If the anomaly is estimated with one-shot sampling, sampling the normal neighbor as a local subgraph may cause the anomaly to be ignored. To solve this problem, we propose to generate the anomaly score using the prediction score from multiple rounds of positive and negative sampling. The final anomaly score for each node is written as follows:

$$f(v_i) = \frac{\sum_{r=1}^{R}(s_{i,r}^{(-)} - s_{i,r}^{(+)})}{R} \tag{17}$$

where R is the sampling round.

Time Complexity Analysis

We analyze the time complexity of our framework in two main parts: the local anomaly detection module and the client-server communication. For the client training process, the time complexity of implementing subgraph sequence sampling using RWR (Tong, Faloutsos, and Pan 2006) is $O(c\delta)$, where c is the number of sampled neighbor nodes and δ is the mean degree of the graph. Sampling each node in R rounds, the total time complexity is $O(cn\delta R)$, where n is the number of client subgraph nodes. the time complexity of the GCN is $O(c^2nR)$, and hence the time complexity of the client training process is $O(cnR(c + \delta))$. In terms of communication, the time complexity for the server to aggregate the parameters of each client is $O(M)$. The number of communication rounds between the client and the server is T_{sc}. and the time complexity of the FL-based update of anomaly information between the clients is mainly generated by the local anomaly node embedding update. To reduce the communication complexity, we only upload the embedding vectors of negative pairs corresponding to anomalous nodes and do not upload the normal node's, which results in a large amount of savings in the communication overhead. Therefore the total time complexity of FedCLGN is $O(T_{sc}(cnR(c + \delta) + M^2))$.

# Dataset	# nodes	# edges	# attributes	# anomalies
Cora	2,708	5,429	1,433	150
Citeseer	3,327	4,723	3,703	150
BlogCatalog	5,196	171,743	8,189	300
Flickr	7,575	239,738	12,407	450

Table 1: The statistics of the datasets. The upper two datasets are citation networks, and the remainder are social networks.

Experiments

Datasets

We employ four datasets that are commonly used to detect anomalies in attributed graphs. They are two citation networks, i.e., Cora, Citeseer (Liu et al. 2021b), two social networks, i.e., BlogCatalog, Flickr (Liu et al. 2021b; Zheng et al. 2021). We divide the dataset into a certain number of participants to construct distributed subgraphs, which are obtained using the METIS graph partitioning algorithm (Karypis 1997). The METIS algorithm differs from the Louvain algorithm (Blondel et al. 2008), used in (Zhang et al. 2021), in that it allows the number of subsets to be specified, which results in a more rational experimental setup.

Since there are no ground-truth anomalies in the aforementioned datasets, We refer to the node attribute anomalies injection method used in the previous research (Song et al. 2007; Ding et al. 2019; Liu et al. 2021b;) to generate node attribute exceptions for each client dataset. We randomly pick a node v_i as the target node. Then, we select another n nodes $V = (v_1, v_2, \ldots, v_n)$ as a candidate set. Here, we set $n = 50$. For each $v_j \in V$, between the attribute vectors of x_i and x_j we compute their Euclidean distance. Then, we pick the node v_j with the largest Euclidean distance from the node v_i as the anomaly node. We change x_i to x_j. The total number of anomalies is given in the last column of Table 1.

Experimental Settings

Baselines. We compare our proposed FedCLGN framework with the prevalent strategies for anomaly detection:

- **DOMINANT** (Ding et al. 2019) is a deep graph autoencoder method for detecting anomalies in attribute graphs. It utilizes graph neural network-based architectures to process graph structure and features.
- **AnomalyDAE** (Fan, Zhang, and Li 2020) use two autoencoders to capture the complex interactions between network structure and node attribute.
- **MGAD** (Kim et al. 2024) utilizes metapaths and GCN layers to efficiently leverage few labeled anomalies and propagate context information between anomalous and normal nodes, improving detection accuracy.
- **CoLA** (Liu et al. 2021b) is an anomaly node detection method based on contrastive self-supervised learning. It can capture the relationship between node and neighbor subgraphs in an unsupervised manner.
- **FedAvg-CoLA** combines CoLA with the federated framework FedAvg for direct use in distributed anomaly node detection in attribute graph. FedAvg-CoLA aggregates model parameters based on weighted average to combine local models with global models.
- **FedPub-CoLA** combines CoLA with the federated framework FedPub for direct use in distributed anomaly node detection in attribute graph. It uses functional embeddings and random graphs to calculate similarities for weighted server-side aggregation. Each client learns a personalized sparse mask to update only relevant parameters. This is an ablation version of FedCLGN, which can be used to verify the validity of pseudo-label discovery and global negative-sample diffusion.

Evaluation Metrics. To measure the performance of our proposed framework and baselines, we employ the ROC-AUC for evaluation. The ROC-AUC has been widely used in previous work to evaluate anomaly detection performance. The ROC curve depicts the relationship between the true positive rate (TPR) and the false positive rate (FPR) of the method, which can be used with the ground truth labels and the anomaly detection results to compute the TPR and FPR. The AUC (Area Under the Curve) is the area below the ROC curve and is used to quantify the overall performance of the method. The value of AUC ranges from 0 to 1, with values closer to 1 indicating better performance of the method.

Parameter Settings. We set the number of experimental rounds $\mathcal{T}_c$ to 50. We train the model for Cora and Citeseer datasets with 15 epochs and train on BlogCatalog, and Flickr with 100 epochs. For Cora and Citeseer, BlogCatalog, and Flickr, we set the learning rate to be 0.001, and 0.003, respectively. we fixed the size c of the sampled graph (number of nodes in the subgraph) to 4, the embedding dimension to be 64, and the batch size to be $\mathcal{B}$ = 300.

Anomaly Detection Results

The AUC scores of the four benchmark datasets are shown in Table 2. We repeated the experiment five times to ensure that the proposed framework is authentic and reliable. Our findings lead to the following conclusions:

- On all four datasets, our proposed FedCLGN achieves the best anomaly detection performance. In particular, compared with the best results of the baselines, our framework obtains an improvement of on AUC averagely. The main reason is that FedCLGN utilizes the global negative pairs to address the challenge of missing key information in a distributed environment. It improves the contrastive learning performance and further improves the FedCLGN performance.
- Through joint learning, multiple clients are able to train models collaboratively, thus significantly scaling the size of the training data. This approach not only increases the diversity of the data, but also enables the model to capture a wider distribution of features by integrating anomaly node embeddings from different clients. In this collaborative learning environment, the model is able to recognize more complex patterns and anomalies, and the final anomaly detection model obtained shows a signifi-

Methods	Cora			Citeseer		
	5 Clients	10 Clients	15 Clients	5 Clients	10 Clients	15 Clients
DOMINANT	0.790±0.009	0.780±0.013	0.773±0.017	0.767±0.011	0.814±0.013	0.724±0.011
AnomalyDAE	0.687±0.032	0.723±0.006	0.712±0.191	0.717±0.004	0.708±0.004	0.719±0.005
MGAD	0.793±0.004	0.796±0.015	0.789±0.011	0.762±0.007	0.777±0.004	0.738±0.003
CoLA	0.728±0.010	0.672±0.016	0.625±0.025	0.663±0.013	0.685±0.014	0.587±0.015
FedAvg-CoLA	0.803±0.017	0.748±0.017	0.679±0.014	0.814±0.014	0.794±0.009	0.711±0.014
FedPub-CoLA	0.806±0.011	0.751±0.009	0.655±0.021	0.796±0.011	0.795±0.003	0.712±0.005
FedCLGN	**0.861±0.009**	**0.807±0.010**	**0.800±0.002**	**0.840±0.008**	**0.826±0.008**	**0.812±0.007**

Methods	BlogCatalog			Flickr		
	5 Clients	10 Clients	15 Clients	5 Clients	10 Clients	15 Clients
DOMINANT	0.760±0.000	0.746±0.001	0.735±0.001	0.716±0.004	0.749±0.015	0.792±0.003
AnomalyDAE	0.814±0.006	0.792±0.010	0.792±0.001	0.693±0.004	0.676±0.002	0.694±0.006
MGAD	0.760±0.003	0.745±0.000	0.734±0.001	0.721±0.002	0.758±0.005	0.806±0.002
CoLA	0.748±0.008	0.717±0.004	0.595±0.018	0.759±0.012	0.669±0.017	0.700±0.004
FedAvg-CoLA	0.825±0.012	0.800±0.008	0.788±0.010	0.756±0.009	0.692±0.020	0.656±0.012
FedPub-CoLA	0.823±0.004	0.793±0.008	0.805±0.007	0.775±0.006	0.722±0.015	0.683±0.017
FedCLGN	**0.846±0.004**	**0.823±0.005**	**0.840±0.004**	**0.876±0.001**	**0.858±0.002**	**0.849±0.009**

Table 2: AUC values comparison on four benchmark datasets. Although not federated frameworks, the initial four baselines represent graph anomaly detection methodologies. We experimented on these methods for each client and computed the results to get the mean value. The best-performing method in each experiment is in bold.

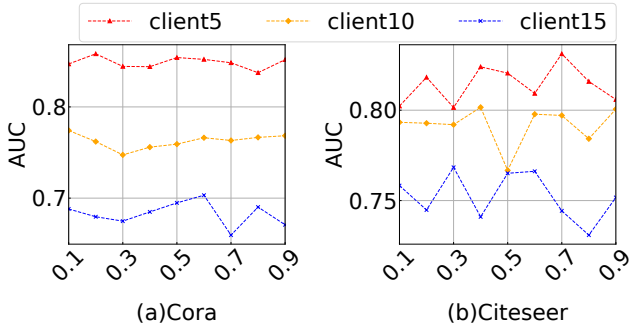

Figure 2: Analysis of transfer probability α.

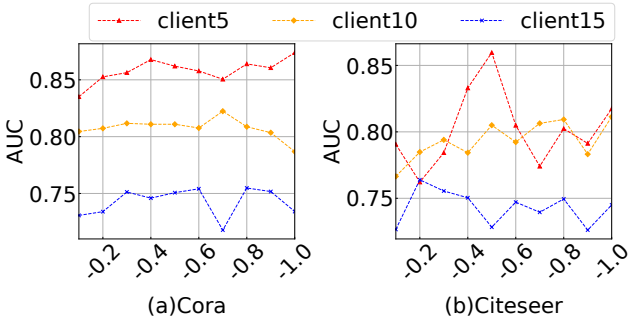

Figure 3: Analysis of confidence threshold λ.

cant improvement in performance compared to a dataset trained independently by a single client.

Parameter Study

In this section, we will explore the impact of two key parameters on the performance of the proposed framework : λ(confidence threshold), α(transfer probability). We conducted experiments on two datasets, and set the number of experimental rounds to 15 based on the experimental parameter settings mentioned earlier.

Analysis of α. We analyze the effect of transfer probability α (0.1-0.9) on graph diffusion and model performance. Figure 2 shows stable performance changes, suggesting the current α balances local and global information propagation well for the task.

Analysis of λ. We further test the impact of confidence threshold λ varying from -0.1 to -1.0. The results are shown in Figure 3. As indicated from the chart, optimal λ values for different datasets can enhance the eventual AUC value and [-0.3, -0.5] appears to be an ideal range.

Conclusion

In this paper, we propose a framework for federated graph anomaly detection based on contrastive learning that addresses the problem of joint training under distributed graph data. A global negative pair updating strategy is implemented by a global graph structure with the provision of data privacy, and graph diffusion is utilized to enhance the effect of contrastive learning. In addition, we will further optimize and extend FedCLGN to show adaptability in larger and more heterogeneous distributed data environments.

Acknowledgments

This work is supported by the National Key R&D Program of China (No.2022YFB3102100, No.31400), and Key R&D and Transformation Plan of Qinghai Province (No.2022-QY-218). The corresponding authors are Wei Yu, Hongdou Dong and Wenjun Wang.

References

Baek, J.; Jeong, W.; Jin, J.; Yoon, J.; and Hwang, S. J. 2023. Personalized subgraph federated learning. In *International conference on machine learning*, 1396–1415. PMLR.

Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10): P10008.

Caldarola, D.; Mancini, M.; Galasso, F.; Ciccone, M.; Rodolà, E.; and Caputo, B. 2021. Cluster-driven graph federated learning over multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2749–2758.

Chandola, V.; Banerjee, A.; and Kumar, V. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3): 1–58.

Ding, K.; Li, J.; Bhanushali, R.; and Liu, H. 2019. Deep anomaly detection on attributed networks. In *Proceedings of the 2019 SIAM international conference on data mining*, 594–602. SIAM.

Fan, H.; Zhang, F.; and Li, Z. 2020. Anomalydae: Dual autoencoder for anomaly detection on attributed networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5685–5689. IEEE.

Jiang, M.; Jung, T.; Karl, R.; and Zhao, T. 2020. Federated dynamic gnn with secure aggregation. *arXiv preprint arXiv:2009.07351*.

Karypis, G. 1997. METIS: Unstructured graph partitioning and sparse matrix ordering system. *Technical report*.

Kim, H.; Kim, J.; Lee, B. S.; and Lim, S. 2024. Label-based Graph Augmentation with Metapath for Graph Anomaly Detection. arXiv:2308.10918.

Li, J.; Dani, H.; Hu, X.; and Liu, H. 2017. Radar: Residual analysis for anomaly detection in attributed networks. In *IJCAI*, volume 17, 2152–2158.

Li, Y.; Huang, X.; Li, J.; Du, M.; and Zou, N. 2019. Specae: Spectral autoencoder for anomaly detection in attributed networks. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 2233–2236.

Liu, X.; Zhang, F.; Hou, Z.; Mian, L.; Wang, Z.; Zhang, J.; and Tang, J. 2021a. Self-supervised learning: Generative or contrastive. *IEEE transactions on knowledge and data engineering*, 35(1): 857–876.

Liu, Y.; Li, Z.; Pan, S.; Gong, C.; Zhou, C.; and Karypis, G. 2021b. Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE transactions on neural networks and learning systems*, 33(6): 2378–2392.

Ma, X.; Wu, J.; Xue, S.; Yang, J.; Zhou, C.; Sheng, Q. Z.; Xiong, H.; and Akoglu, L. 2021. A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(12): 12012–12038.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Pang, G.; Shen, C.; Cao, L.; and Hengel, A. V. D. 2021. Deep learning for anomaly detection: A review. *ACM computing surveys (CSUR)*, 54(2): 1–38.

Pazho, A. D.; Noghre, G. A.; Purkayastha, A. A.; Vempati, J.; Martin, O.; and Tabkhi, H. 2023. A survey of graph-based deep learning for anomaly detection in distributed systems. *IEEE Transactions on Knowledge and Data Engineering*, 36(1): 1–20.

Perozzi, B.; and Akoglu, L. 2016. Scalable anomaly ranking of attributed neighborhoods. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, 207–215. SIAM.

Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; and Tang, J. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 1150–1160.

Ruff, L.; Kauffmann, J. R.; Vandermeulen, R. A.; Montavon, G.; Samek, W.; Kloft, M.; Dietterich, T. G.; and Müller, K.-R. 2021. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5): 756–795.

Song, X.; Wu, M.; Jermaine, C.; and Ranka, S. 2007. Conditional anomaly detection. *IEEE Transactions on knowledge and Data Engineering*, 19(5): 631–645.

Tong, H.; Faloutsos, C.; and Pan, J.-Y. 2006. Fast random walk with restart and its applications. In *Sixth international conference on data mining (ICDM'06)*, 613–622. IEEE.

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2018. Deep graph infomax. *arXiv preprint arXiv:1809.10341*.

Wang, H.; Jia, Y.; Zhang, M.; Hu, Q.; Ren, H.; Sun, P.; Wen, Y.; and Zhang, T. 2024a. FedDSE: Distribution-aware Submodel Extraction for Federated Learning over Resource-constrained Devices. In *Proceedings of the ACM on Web Conference 2024*, 2902–2913.

Wang, H.; Xu, H.; Li, Y.; Xu, Y.; Li, R.; and Zhang, T. 2024b. FedCDA: Federated Learning with Cross-rounds Divergence-aware Aggregation. In *The Twelfth International Conference on Learning Representations*.

Wang, H.; Zheng, P.; Han, X.; Xu, W.; Li, R.; and Zhang, T. 2024c. FedNLR: Federated Learning with Neuron-wise Learning Rates. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3069–3080.

Zhang, K.; Yang, C.; Li, X.; Sun, L.; and Yiu, S. M. 2021. Subgraph federated learning with missing neighbor generation. *Advances in Neural Information Processing Systems*, 34: 6671–6682.

Zheng, Y.; Jin, M.; Liu, Y.; Chi, L.; Phan, K. T.; and Chen, Y.-P. P. 2021. Generative and contrastive self-supervised learning for graph anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(12): 12220–12233.

Zhou, Y.; Zheng, H.; Huang, X.; Hao, S.; Li, D.; and Zhao, J. 2022. Graph neural networks: Taxonomy, advances, and trends. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(1): 1–54.