



Defense against membership inference attack in graph neural networks through graph perturbation

Kai Wang¹ · Jinxia Wu¹ · Tianqing Zhu¹ · Wei Ren¹ · Ying Hong²

Published online: 16 December 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH, DE 2022

Abstract

Graph neural networks have demonstrated remarkable performance in learning node or graph representations for various graph-related tasks. However, learning with graph data or its embedded representations may induce privacy issues when the node representations contain sensitive or private user information. Although many machine learning models or techniques have been proposed for privacy preservation of traditional non-graph structured data, there is limited work to address graph privacy concerns. In this paper, we investigate the privacy problem of embedding representations of nodes, in which an adversary can infer the user's privacy by designing an inference attack algorithm. To address this problem, we develop a defense algorithm against white-box membership inference attacks, based on perturbation injection on the graph. In particular, we employ a graph reconstruction model and inject a certain size of noise into the intermediate output of the model, i.e., the latent representations of the nodes. The experimental results obtained on real-world datasets, along with reasonable usability and privacy metrics, demonstrate that our proposed approach can effectively resist membership inference attacks. Meanwhile, based on our method, the trade-off between usability and privacy brought by defense measures can be observed intuitively, which provides a reference for subsequent research in the field of graph privacy protection.

Keywords Graph neural network · Graph privacy-preserving · Membership inference attack · Perturbation injection

1 Introduction

A large number of real-world relationships can be represented as graphs, such as online social networks [1, 2], citation networks [3], financial networks [4], and biomedical datasets [5], which are graph structured data. One of

the main sources of graph data is social networks [6], where people/mobile devices are abstracted as nodes and links represent relationships between nodes. Graph neural networks (GNNs) [7], which utilize the neural network to combine graph structure and node feature information together, have shown state-of-the-art performance on wide range of complex tasks like node classification [3, 8, 9], link prediction [10, 11], graph classification [12], etc. The key pre-processing step is to design node embedding algorithms [13] which transform graph structured data from non-Euclidean spaces to a low-dimension representation for various downstream tasks. However, in addition to being important for GNN downstream tasks, node embedding may lead to privacy concerns [14].

In many realistic scenarios and applications, the embedding will be released for further utilization without sufficient privacy-oriented consideration. While achieving high performance, GNN models require significant resources and time during data collection and training. Therefore, a well-trained GNN model is of great value. At the same time, to meet the demand, many AI platforms provide model-related selling services for model owners. This commercial behavior has

✉ Ying Hong
yhong@wtu.edu.cn
Kai Wang
wk_2020@cug.edu.cn
Jinxia Wu
wujinxia@cug.edu.cn
Tianqing Zhu
tianqing.zhu@ieee.org
Wei Ren
weirencs@cug.edu.cn

¹ School of Computer Science, China University of Geosciences, No.388 Lumo Road, Wuhan 430074, People's Republic of China

² School of Computer Science and Artificial Intelligence, Wuhan Textile University, No.1 Sunshine Avenue, Wuhan 430200, People's Republic of China

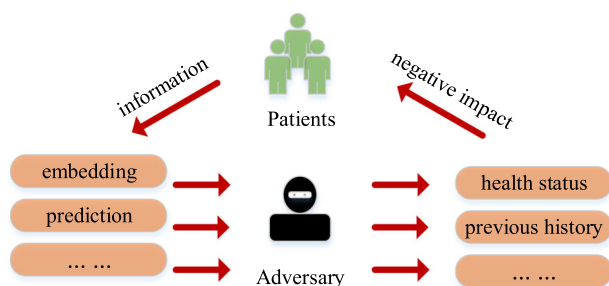


Fig. 1 An example illustrates privacy issue on GNNs. An adversary can use certain information derived from the GNNs to determine whether a person is part of a training dataset to infer the person's physical condition

drawn people's great attention on model security and graph data security, especially when models are trained with potentially private and sensitive data [15–19]. In previous work, node embedding (white-box attacks) combined with clustering algorithms have been used to implement membership inference attacks (MIA) [14]. In MIA, the adversary aims to infer whether one node is part of the graph data used to train the model or not.

This paper presents the use of graph reconstruction technique to defense against membership inference attack. At present, researchers have not been able to attach sufficient importance to the harm that such privacy attacks can produce, and have not yet proposed a complete corresponding defense method. Consider a very realistic scenario, which is the prediction of the spread of the 2019-nCoV. Nodes are used to represent users, edges represent propagation paths, and node features represent clinical symptoms. Symptom prediction and propagation prediction are performed, respectively, through node classification and link prediction. An attacker can infer the physical condition of a particular user by identifying whether the user is part of the training dataset, as shown in Fig. 1. Although infected people are not contagious after recovery, the disclosure of their privacy can still bring them different degrees of negative social impact or trouble.

In the context of machine learning (ML), privacy violation occurs when an adversary deduces certain information about the record of a particular user's data in a training set. The success of various inference attacks is used to quantify this information leakage. Privacy risks in graph-based ML models have not been sufficiently explored and quantified. Membership inference attack is a common type of machine learning attack that can infer a specific individual record with limited background knowledge. Correspondingly, the defense method for inference attacks is worthy of in-depth analysis and research.

Our main contributions are summarized as follows:

- We introduce a realistic membership inference attack on graph neural networks in white-box setting.

- We propose a defense measure against the above-mentioned attack, which significantly reduces the accuracy of inference attack while preserving the utility of the graph.
- We carry out relevant experiments to prove the availability and reliability of the defense method.

The paper structure will be organized as follows. Section 2 gives background on GNNs and node embedding algorithm as well as common attacks on graph are introduced. Section 3 gives a complete description about our defense methodology. Section 4 describes experimental setup. Section 5 gives analysis of our experiments on utility-privacy trade-off. We will conclude our work in Sect. 6.

2 Preliminaries

2.1 Notations

We consider the defense algorithm targeting membership inference attacks on graph neural networks. Given a graph denoted as $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_i\}_{i=1}^{|\mathcal{V}|}$ and $\mathcal{E} = \{e_j\}_{j=1}^{|\mathcal{E}|}$ represent the sets of nodes and edges, respectively. An adjacency matrix $A \in \{0, 1\}^{|\mathcal{V}| \times |\mathcal{V}|}$ describes the relationships between nodes. Here $A_{ij} = 1$ means node pair v_i and v_j are linked, 0 otherwise. Node features are represented as a matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$, where d is the dimension of node feature and N is the number of nodes (i.e., $N = |\mathcal{V}|$). Thus, the node features of node v_i are described by the i -th row of \mathbf{X} and an attributed graph can be represented as $G = \{\mathbf{A}, \mathbf{X}\}$.

2.2 Graph neural networks

The classical machine learning algorithms and models do not have a satisfying performance on graph structured data, which are comfortable with individually represented data, but are less efficient when dealing with more complex structured relational data. To resolve this issue, inspired by the success of Convolutional Neural Networks (CNN) [20, 21], a new series of deep learning algorithms are proposed, namely Graph Neural Networks (i.e., GNN). GNN generalized by the graph convolutional network (GCN) and its variants extends the convolution operations to irregular graph data, and has achieved excellent performance in various tasks and fields [22, 23]. Generally, GNN contains a few graph convolutional layers, each of which obtains an initial vector for a particular node, and aggregates information of node v 's neighbors at previous layer to learn a hidden state of node v , which corresponding to the AGGREGATE operation in Equation (1). GNN models then update the representation of node v following the nonlinear transformation after each aggregation step, which corresponding to the UPDATE operation in

Equation (2). Formally, the aggregation and update function can be formulated as follows:

$$z'_v = \text{AGGREGATE}_l\left(\left\{h_u^{l-1}, \forall u \in \mathcal{N}(v)\right\}\right), \tag{1}$$

$$h'_v = \text{UPDATE}_l\left(z'_v\right), \tag{2}$$

where $\mathcal{N}(v)$ is the neighborhood of v (i.e., v itself and the set of nodes which have an edge connected to v), and l represents the l -th layer of the GNN. z'_u and h'_u denote the hidden state and the representation (embedding) vector of node u at layer l , respectively. First of all, we initialize v 's feature x_v as its representation as well as the input of layer l_0 .

Finally, at the last convolutional layer (i.e., L), a softmax function will be applied to the node representations for downstream tasks, like node classification, can be formulated as:

$$y_i \leftarrow \text{softmax}\left(z_i^{(L)} \mathbf{W}\right), \tag{3}$$

where $y_i \in \mathbb{R}^c$, c means the number of classes of node set and \mathbf{W} is a learnable weight matrix. $y_i(j)$ corresponds to the predicted probability that node i belongs to class j .

One of the structures that is commonly used as an update function, is Multilayer Perceptron (MLP). In the meantime, different GNN models may apply different aggregation functions, and with the development of GNNs, more and more aggregation functions have been proposed, with considerable results have been achieved. In this paper, we mainly focus on GraphSAGE [24]), one of the representative GNN architectures.

2.3 Node embedding algorithm

In order to facilitate space and computational overhead of graphs' further processing, graph embedding algorithms provide an effective method to embed graph structured data into low-dimensional representation. Specifically, mark an embedding algorithm as $\Psi : V \rightarrow \mathcal{R}^h$, where V comes from graph G . Each node v_i in set V can be transformed from discrete representation to a h -dimension vector which capture the attributes of the original graph, like the relationship between each node pair. Deep learning-based node embeddings generate low-dimensional representation for each node with both adjacency matrix \mathbf{A} and feature matrix \mathbf{X} . With this technique, designing parallel or distributed algorithms directly on graph data is no longer a challenge. Meanwhile, downstream applications such as node classification, link prediction and clustering can also be performed on standard ML algorithms. For GNN models, the output of the intermediate layer corresponds to the low-dimensional embedding representation of the node set. As well as graph embedding algorithms have been researched, which proposed to display

the whole graph with a single vector captures the vital properties of original one. Our work focus on the application of node embedding algorithms in this paper, we consider the defense against membership inference attacks based on the white-box background that adversary has access to target model's node embedding.

2.4 Membership inference attack

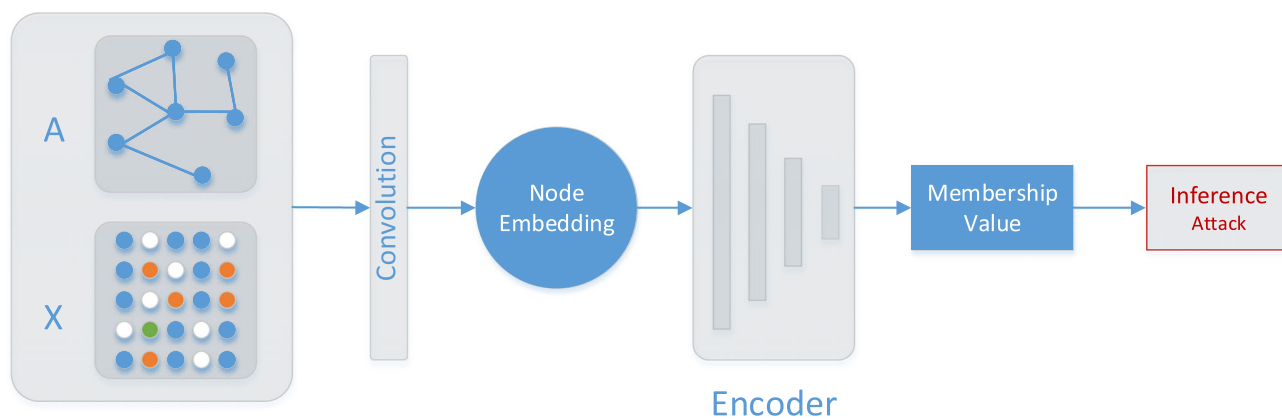
In machine learning, there are several representative attacks, including membership inference [25], attribute inference [26], model inversion [27] and model stealing [28]. The target of the first three attacks is the training set of the model, and model stealing is to refactor the model parameters. Researchers have conducted in-depth studies on these attacks, and the analysis of privacy risks brought by these attacks against ML model is being gradually proposed [29].

Membership inference attack [15] is one of the security threats faced by machine learning algorithms. It is used to infer whether specific data exists in the training set of machine learning models, which brings great security risks to users and challenges the security of machine learning models. In case of MIA on graph structured data, the adversary aims to infer whether one node is part of the graph data used to train the model or not. [30] proposed three threat models for node-level membership inference attacks based on adversary background knowledge, with classifying the adversary's background knowledge to three dimensions, i.e., shadow dataset, shadow model, and node topology.

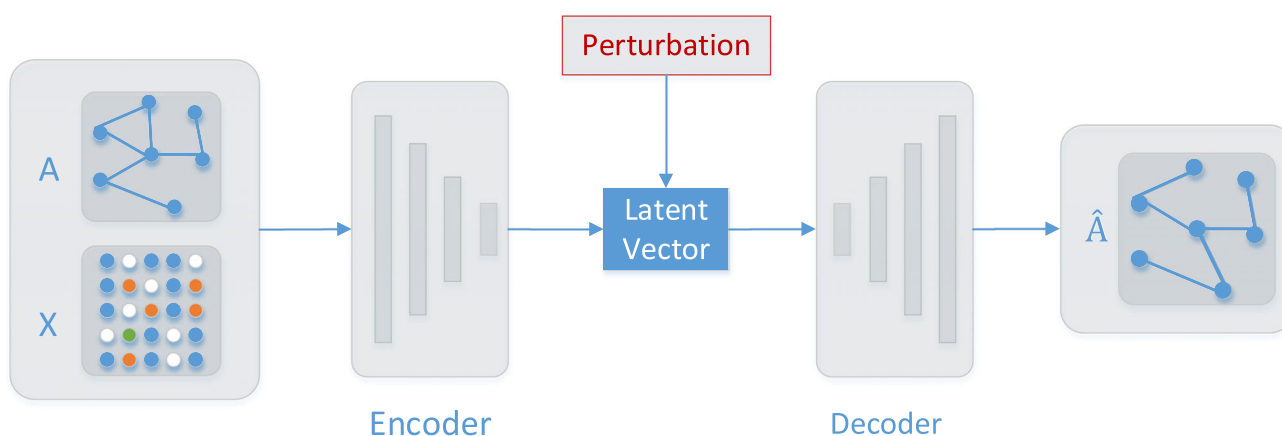
The focus of our research is Membership Inference Attack(MIA), we propose a reliable method to defense MIA on graph data, in order to guard the sensitive information on the basis of obtaining its normal utility.

3 Perturbation defense methodology for GNN

The architecture of defense methodology we elaborate in this paper is shown in Fig. 2. In this figure, (a) illustrates the membership inference attack, and (b) is our defense strategy. An adversary who performs a membership inference attack aims to infer whether a node comes from the model's training set. In the white-box setting, the adversary has access to the intermediate layer's output of GNN model, which corresponds to the embedding vector of each node. The output predictions and parameters of GNN model are agnostic to the adversary. The adversary trains an encoder-decoder network to map embedding vector to a single membership value, which is a regular pre-processing operation. Finally, based on the distribution difference of membership value between training nodes and testing nodes, the adversary can determine whether a node is a member of the training set. The



(a) Membership Inference Attack



(b) Perturbation Defense Model

Fig. 2 Perturbation defense methodology. **a** depicts the process of membership inference attack, and **b** describes the principle of the perturbation defense method. When the perturbation defense method

generates the adjacency matrix with noise, it is used to replace the original input adjacency matrix of the graph convolutional network algorithms to obtain the node embedding with noise

membership inference attack is detailed in Section 3.1. The implementation of the perturbation defense method is based on the graph reconstruction model, which mainly consists of two parts, an encoder and a decoder. In order to reduce the attack success rate, we limit the prior knowledge of the attacker to be privacy-free. So we first obtain the noisy reconstructed adjacency matrix by adding perturbations during the training process of the graph reconstruction model. Then we replace the original input adjacency matrix of GNNs with the reconstructed adjacency matrix to obtain the noisy node embedding representations. If the embedding vectors mastered by the attacker are noisy (i.e., privacy-free), then the defense effect can be achieved. Our proposed defense methodology is detailed in Section 3.2. We also depict the process of adding perturbations as shown in Fig. 3.

3.1 Membership inference attack on graph

Under the white-box setting [14], the information possessed by the attacker is the embedding of each node, i.e., the output of intermediate graph convolutional layer of target GNN model. This is practical in the real situation. Some resource-based websites will provide this information for other scientific researchers to use. Adversary may take advantage of these resources to infer the privacy information of the data.

During the attack, the graph convolutional layer is responsible for calculating the low-dimensional embedding representation of the input graph data. The parameters of GNNs are updated in each training iteration and specifically adjusted for a higher performance of the training data, so as to generate a distinguishable footprint (i.e., two-dimensional projection)

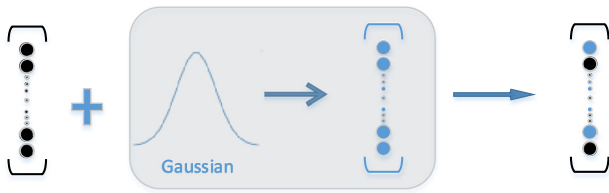


Fig. 3 The left side of the plus sign is the original node embedding representation, and the right side is the perturbed representation. We randomly sample noise from the Gaussian function with the same dimension as the node embedding representation, and sum them up to achieve the perturbation effect. The original embedding vector is shown in black, the noisy vector is shown in blue, and the final vector contains two colors indicating that it has been perturbed

between the embedding of the training data points and the test data points. With the 2D-TSNE algorithm [31], the distinguishable footprints can be visually illustrated.

The attack method is unsupervised, which means the adversary does not have a supervised label to form a mapping between the intermediate embedding and the node’s belonging or latent value. Figure 2(a) illustrates this attack’s methodology. The adversary designs an encoder–decoder network, which trained in an unsupervised manner to map the low-dimensional embedding value to a single membership value. Take a node v ’s embedding $\Psi(v)$ as input, encoder generates a latent value(i.e., single membership value). And next, the latent value is passed to a decoder as its input for obtaining the reconstruction of embedding $\Psi(v)$. The encoder–decoder network aims to minimize the reconstruction loss, formulated as,

$$\|\Psi(v) - f_d(f_e(\Psi(v)))\|_2^2, \tag{4}$$

where we denote the encoder as f_e , and decoder as f_d .

After the encoder–decoder network is trained, the adversary can obtain the single membership value corresponding to the embedded value of the target graph node. According to the previous analysis, we could know that the membership values of the training node and the test node will show two cluster distributions. Therefore, the clustering algorithms can be used to classify the membership values into two clusters, such as K-Means. At the same time, considering that the adversary does not have the ability to distinguish between the two clusters which represents the training set node. Therefore, the adversary needs to know the prior distribution characteristics of a small number of members and non-members in advance. After that, for any new nodes, the adversary classifies them into clusters and can directly determine whether they are members or not.

We utilize GraphSAGE as the target model from which the adversary get the node embedding representation of the target graph to mount MIA. GraphSAGE [24]) is the first to extend

the original graph convolutional network to an inductive setting with a redesigned aggregation function. On account of the number of neighbors of a node can be a wide range of values, it is inefficient to obtain complete node neighbors. GraphSAGE makes use of a sampling method to obtain a fixed number of neighbors for each node. In this paper, we follow GraphSAGE’s way to perform graph convolutions, which can be defined as follows:

$$\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k \left(\left\{ \mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v) \right\} \right), \tag{5}$$

$$\mathbf{h}_v^k \leftarrow \sigma \left(\mathbf{W}^k \cdot \text{CONCAT} \left(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k \right) \right), \tag{6}$$

where CONCAT is the concatenation operation and the aggregation function should keep the arrangement of the node ordering unchanged, such as the mean, sum, or maximum function.

Algorithm 1: Our Defense on MIA

- 1 **Input:** $G = \{\mathbf{A}, \mathbf{X}\}$, a graph with adjacency matrix and features; $Epoch$, the number of epochs; f_e and f_d , encoder and decoder architecture; $latent$, the intermediate output of encoder-decoder network; $noise \sim N(\mu, \sigma^2)$, adding to $latent$ matrix.
- 2 **Output:** reconstructed adjacency matrix $\hat{\mathbf{A}}$, with original feature matrix remained.
- 3 **for** $epoch = 1$ to $Epoch$ **do**
- 4 $latent \leftarrow f_e(\mathbf{A}, \mathbf{X})$;
- 5 Sample $noise$ from Gaussian distribution in the same dimensions as $latent$ matrix;
- 6 $latent \leftarrow latent + noise$;
- 7 $\hat{\mathbf{A}} \leftarrow f_d(latent)$;
- 8 $loss \leftarrow \sum_{v \in V} \|\hat{\mathbf{A}} - \mathbf{A}\|^2$;
- 9 Optimize $loss$ value;
- 10 **Return** reconstructed adjacency matrix $\hat{\mathbf{A}}$;
- 11 Take the reconstructed adjacency matrix $\hat{\mathbf{A}}$ as GNN models’ input, and release the corresponding node embeddings to the public.

3.2 Perturbation defense method

According to what we learned in the previous section, the adversary has obtained the node embedding vector of the target graph data, so that it can launch membership inference attacks. In this paper, we modify the node embedding vectors in an indirect way. We take advantage of a graph reconstruction algorithm, to add subtle perturbation to graph’s representation of the hidden space during the graph reconstruction process. In other words, we add perturbation to the embedding output of the encoder in the graph reconstruction model, and then input that noisy embedding to the decoder, resulting in a noisy graph reconstruction adjacency matrix. This noisy adjacency matrix is then put into a regular graph convolutional network to get the output of its first convolutional layer, and release it to the public or platform. As a result, the adversary gets the perturbed knowledge and this method achieves the corresponding defense

effect. The reconstruction algorithm we select is graph auto-encoder(GAE) [32], and we achieve subtle perturbation by adding Gaussian noise. We present the training process of defense method we proposed in Algorithm 1.

GAE, based on the variational auto-encoder(VAE) [33], works as a framework for unsupervised learning on graph structured data. The model uses latent variables and can learn interpretable latent representations of undirected graphs. This model is implemented by a graph convolutional network (GCN) encoder and a simple decoder for inner product. The encoder of GAE is made up of two graph convolutional layers, which is defined as,

$$\mathbf{Z} = \text{enc}(\mathbf{X}, \mathbf{A}) = \text{conv}(f(\text{conv}(\mathbf{A}, \mathbf{X}; \mathbf{W}_1)); \mathbf{W}_2), \quad (7)$$

where \mathbf{Z} denotes the representation matrix of a graph in latent space, $f()$ is activation function, and \mathbf{W} is the weight matrix. The decoder of GAE is designed to decode the node relationship information from their latent matrix through reconstructing adjacency matrix of the graph, which takes the form,

$$\hat{\mathbf{A}} = \text{dec}(\mathbf{z}_v, \mathbf{z}_u) = \sigma(\mathbf{z}_v^T \mathbf{z}_u), \quad (8)$$

where \mathbf{z}_v is the latent vector of node v , and $\sigma()$ denotes the activation function. GAE is trained by minimizing the negative cross entropy between the original adjacency matrix \mathbf{A} and corresponding reconstructed adjacency matrix $\hat{\mathbf{A}}$.

In the process of training the encoder–decoder, in each epoch, we add a certain amount of Gaussian noise to the latent vector of each node. Compare to differential privacy technique [34–36], which also make use of random noise, but from the same data distribution (i.e., using the same set of parameters), we propose to observe the trade-off between graph privacy and usability caused by the noise from different data distributions. Therefore, we design multiple sets of parameters to randomly sample noise from Gaussian functions without applying differential privacy. For the subsequent analysis, we need to observe the correlated effects of the gradual increase in noise on the privacy and usability of the graphical data. The size of the added Gaussian noise needs to be finely controlled to make a trade-off between privacy and utility, where the mean remains unchanged at 0, and the variance range is [0, 0.3] and takes a value per 0.05. On the premise of ensuring the availability of the reconstructed graph, it can also cause a certain defensive effect on the membership inference attacks. After acquiring the reconstruction graph, we put the reconstruction one into the GNN model, extract the output of the intermediate layer of it after the training process completed, and release the node embeddings to the public that harbors the adversary finally.

There is another graph reconstruction model, VGAE [32], which is a variational version of GAE. VGAE utilizes Kullback–Leibler divergence to learn the data distribution and complete the optimization of the reconstructed adjacency matrix, formulated as following,

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{Z}|\mathbf{X}, \mathbf{A})}[\log p(\mathbf{A} | \mathbf{Z})] - \text{KL}[q(\mathbf{Z} | \mathbf{X}, \mathbf{A}) \| p(\mathbf{Z})], \quad (9)$$

where \mathbf{Z} denotes the representation matrix of a graph in latent space and $p()$ is the Gaussian prior distribution and $q()$ is the empirical distribution.

4 Experiments

4.1 Datasets

We consider three standard datasets: Pubmed [37], Citeseer [38], and Cora [38]. The statistics of the above datasets are shown in Table 1.

- *Pubmed* The Pubmed Diabetes dataset contains 19, 717 scientific publications related to diabetes from the Pubmed database, which can be divided into three classes. The citation network consists of 44, 338 links. Each publication in this dataset is described, by a TF/IDF weighted word vector from a dictionary of 500 unique words. The dataset is used in the form of 60 training samples, 500 validation samples, and 500 test samples.
- *Citeseer* The CiteSeer dataset contains 3312 scientific publications, which can be divided into six classes. The citation network consists of 4732 links. The Citeseer dataset contains a dictionary of 3703 unique words, therefore features are 3703 dimensions, with 0 and 1 describing whether each word exists in the paper. It is assigned into 120 training samples, 500 validation samples, and 1000 test samples.
- *Cora* Cora dataset consists of 2708 machine learning papers and is divided into seven classes, which is a popular dataset for graph deep learning in recent years. The citation network consists of 5429 links. The Cora dataset contains a dictionary of 1433 unique words, therefore features are 1433 dimensions, with 0 and 1 describing whether each word exists in the paper. It is assigned into 140 training samples, 300 validation samples, and 1000 test samples.

It is required to note that the above allocation of the three datasets is for GraphSAGE. The defense method proposed in this paper allocates datasets in two proportions, 6 : 1 : 3 and 3 : 1 : 6, in which the validation set accounts for 10 percent, and the training set and test set account for 30 percent or 60 percent, respectively.

Table 1 Dataset statistics

Dataset	Nodes	Edges	Classes	Features	Train./Val./Test
Pubmed	19,717	44,338	3	500	60/500/1000
Citeseer	3,327	4,732	6	3,703	120/500/1000
Cora	2,708	5,429	7	1,433	140/500/1000

4.2 Metrics

Considering the large visual difference between graph structured data and image data, it is difficult for us to use traditional machine learning metrics to measure graph privacy and utility, in order to evaluate our proposed method. Hence, the followings are to elaborate on privacy and utility metrics in this paper.

- *Utility* These models are not trained on the full version of these datasets, and some of the edges have been deleted, but all the node features are retained. From the previously deleted edges, we construct validation sets and test sets, in which the same number of unconnected node pairs were randomly sampled. We report AP (average precision) scores and AUC (area under the ROC curve) to evaluate the utility of reconstructed graphs. Average precision computes the ratio of true positives and calculates the percentage of predicted samples actually exist in the original graph. The ROC curve plots the false positive rate and true positive rate on the x-axis and y-axis, respectively. The AUC score calculates the area under the ROC curve, where the larger the area, the greater the graph has been reconstructed. For the binary classification problem of the adjacency matrix obtained by the graph reconstruction model, the accuracy of random guess is 50%, and any higher accuracy indicates that the target graph reconstructed by the model has certain reliability.
- *Privacy* The membership inference attack is a binary classification problem: the attacker needs to determine whether a node is part of the training set of the target model or not. As a result, when inference accuracy gets any higher than random guess method, it indicates that a privacy leakage of target model's sensitive training data has happened. The goal of our proposed defense is to reduce the accuracy of membership inference attacks as more as possible. Once our defense has achieved some reduction in original inference accuracy, it succeeds in protecting the privacy of graph training data. The more the inference accuracy drops, the greater the protection of privacy. While the availability of graph data needs to be fully considered at the same time. Therefore, it is necessary to make a suitable trade-off between privacy and utility.

4.3 Models

In our experiment, we consider that the node embeddings obtained by the attacker comes from the case where the target model is a two-layer graph convolutional network (that is, GraphSAGE [24]). The node embeddings are the output of the first layer of the target model with a dimension of 16. After acquiring the node embeddings, the adversary inputs the node embedding into the encoder to obtain the membership value of dimension 1. The perturbation we add to the graph reconstruction process is sampled in a Gaussian distribution, where the mean remains unchanged at 0, and the variance range is [0, 0.3] and takes a value per 0.05. We keep the mean constant and adjust only the variance to be better able to control the noise size. Considering the embedding nature of the latent vector, excessive noise will lead to great deviation of the reconstructed adjacency matrix. Therefore, we initially tried to set the variance as 0.05, and found that this level of noise could largely preserve the availability of the reconstructed adjacency matrix and play a certain defense effect. We then chose 0.05 as the minimum variance and increased it at intervals of 0.05 until it reached 0.3. When the variance is 0.3, the availability of the reconstructed adjacency matrix continues to decrease, but the attack effect is little improved. We use the Adam optimizer with a learning rate of 0.01 and training epochs of 500.

4.4 Evaluation

We first show the impact of the added perturbations on the utility of the reconstructed adjacency matrix for the case of graph reconstruction modeled as GAE as depicted in Fig. 4. At this point, we reconstruct the adjacency matrix with 60 percent of the input from the original. We can observe that pubmed has the highest reconstruction success rate in all three datasets, both in terms of ROC and average precision. While citeseer and cora have similar reconstruction performance. As the Gaussian perturbation increases, the reconstruction availability of the three datasets shows a slow and linear decrease while maintaining the ranking.

Figure 5 illustrates the success rate of membership inference attacks against node embeddings obtained from the original adjacency matrix and those obtained from the reconstructed adjacency matrix. We can notice that the success rate of the MIA is intimately related to the availability of the

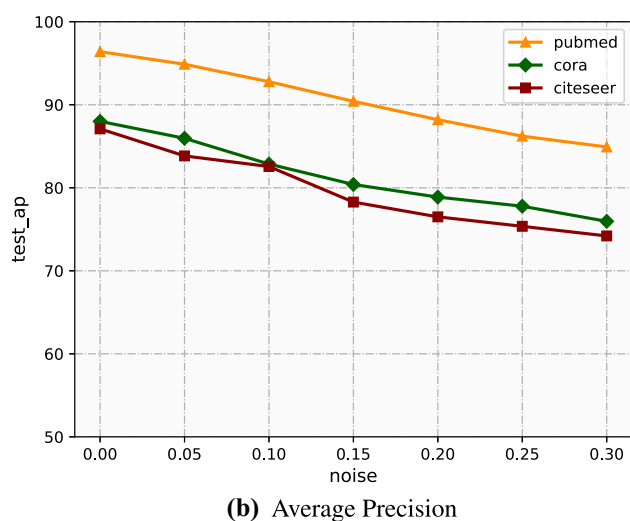
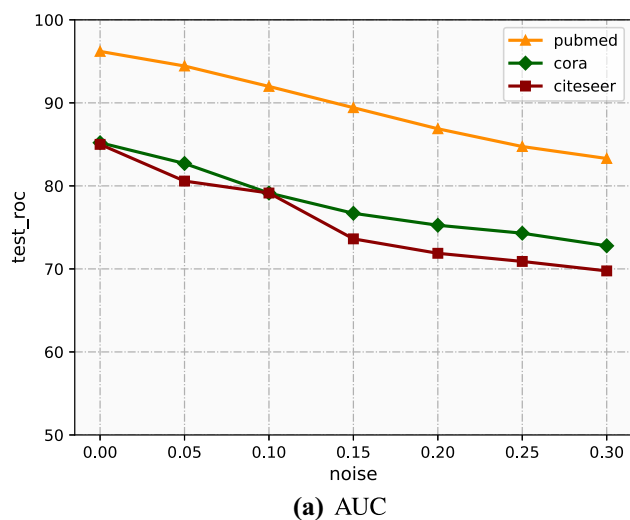


Fig. 4 Utility. We use area under the ROC curve (AUC) and average precision (AP) scores for each dataset to illustrate the utility of reconstructed adjacency matrix

reconstructed adjacency matrix. Pubmed, consistently has the highest reconstruction availability in all three datasets (even with the injection of perturbation), and correspondingly, the success rate of MIA shows a near-linear decrease with as the noise gradually increases.

Further, for citeseer and cora, we can note that MIA has a relatively high success rate when only the adjacency matrix is reconstructed without perturbation injection. However, when only Gaussian noise with a variance value of 0.05 is injected during the reconstruction process of the adjacency matrix, the success rate of MIA shows a dramatic decrease. And with the progressive increase in noise, the attack effect of MIA on both datasets stays above and below 55%, accompanied by a very slight fluctuation. This can, to a certain extent, indicate that adding additional perturbations even though slight, citeseer, and cora make a large degree of damage to their adjacency

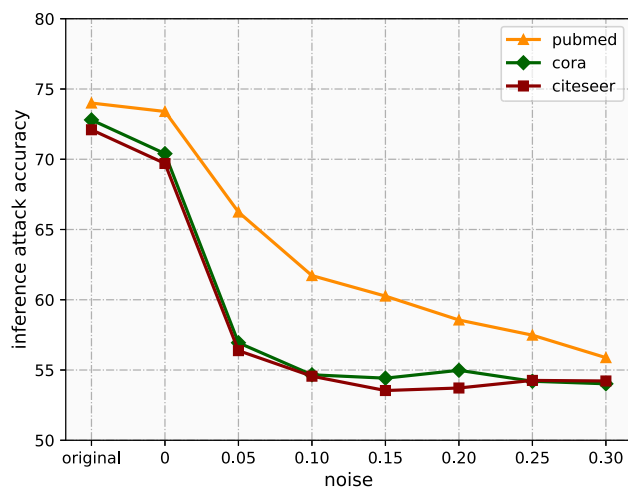


Fig. 5 Privacy. We use the membership inference attack accuracy to measure the privacy-preserving effect of our defense method

matrix structure in the case of relatively low availability of the originally reconstructed adjacency matrix. Thus, the success rate of MIA is not much different from random guess.

For the sake of experimental completeness, and given the somewhat random nature of the noise sampled from the Gaussian distribution, for each dimension of the noise and for each of the three datasets, we generate five samples of the reconstructed adjacency matrix accordingly. The consequences seen so far, for both usability and privacy measures, are handled by averaging (i.e., summing and dividing by 5).

Due to the fact that the reconstruction of the adjacency matrix does not require the entire adjacency matrix to be input into the graph reconstruction model, we experiment for the ratio of data utilized in the model accordingly, as shown in Fig. 6. It is conceivable that the higher the percentage of data put into the graph reconstruction model, the stronger the usability of the reconstruction. Also, the dataset pubmed has the relatively highest reconstruction availability under the same conditions even if only 30% of the data is used. Compared to using 60% of the data, there is a relatively close usability (difference of no more than 4%). In terms of privacy protection, the dataset pubmed reconstructed using 60% of the data generally has a higher success rate of MIA attacks on it than the one reconstructed with 30% data, regardless of the size of perturbation. The reason for this is that those reconstructed with 60% data contain more available sensitive information. For cora and citeseer, it still presents small fluctuations with no obvious regularity because of the originally low availability of reconstruction for these two datasets, and also, the relatively strong influence of the injected noise on the reconstructed data.

We also validate the reliability of our proposed defense method further using another graph reconstruction model,

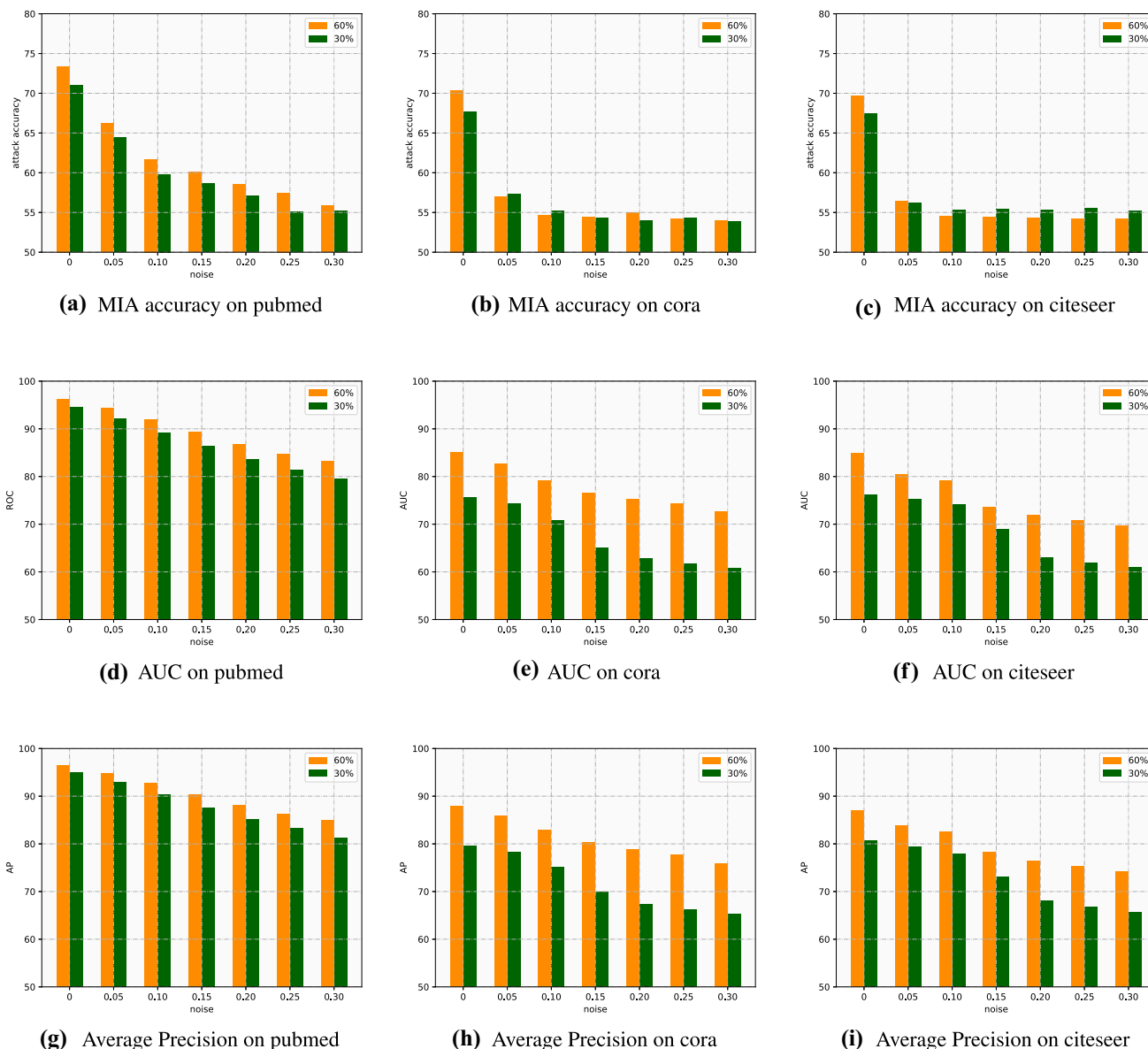


Fig. 6 Utility and Privacy with training data ratio 30% and 60% on three datasets. We use the membership inference attack accuracy to measure the privacy-preserving effect, and the same AUC and AP to measure the

availability of the reconstructed adjacency matrix. The two neighboring bars indicate training data ratio 30% and 60%, respectively, and show the experimental effects on the three datasets

VGAE, as shown in Fig. 7. VGAE and GAE have slightly different performance on different datasets. In terms of usability, GAE reconstructs the pubmed outperforms VGAE, but performs the opposite on the other two datasets. On the other hand, in terms of privacy, VGAE and GAE have roughly the same defense effect for MIA, with only slight differences in performance. In the meantime, we can find that the performance of VGAE and GAE in data utility gradually converge with the gradual increase in noise.

5 Discussion

Utility With the evaluation metrics provided by the graph reconstruction models GAE and VGAE [32], we can find that the perturbation defense strategy proposed in this paper maintains the utility of the reconstructed graphs. And with the increase in defense strength, the graph availability shows a slow decrease close to linearity, which is acceptable and empirical. As the model uses more training data, the usability of the reconstructed graph increases while being less affected by the perturbation defense strategy.

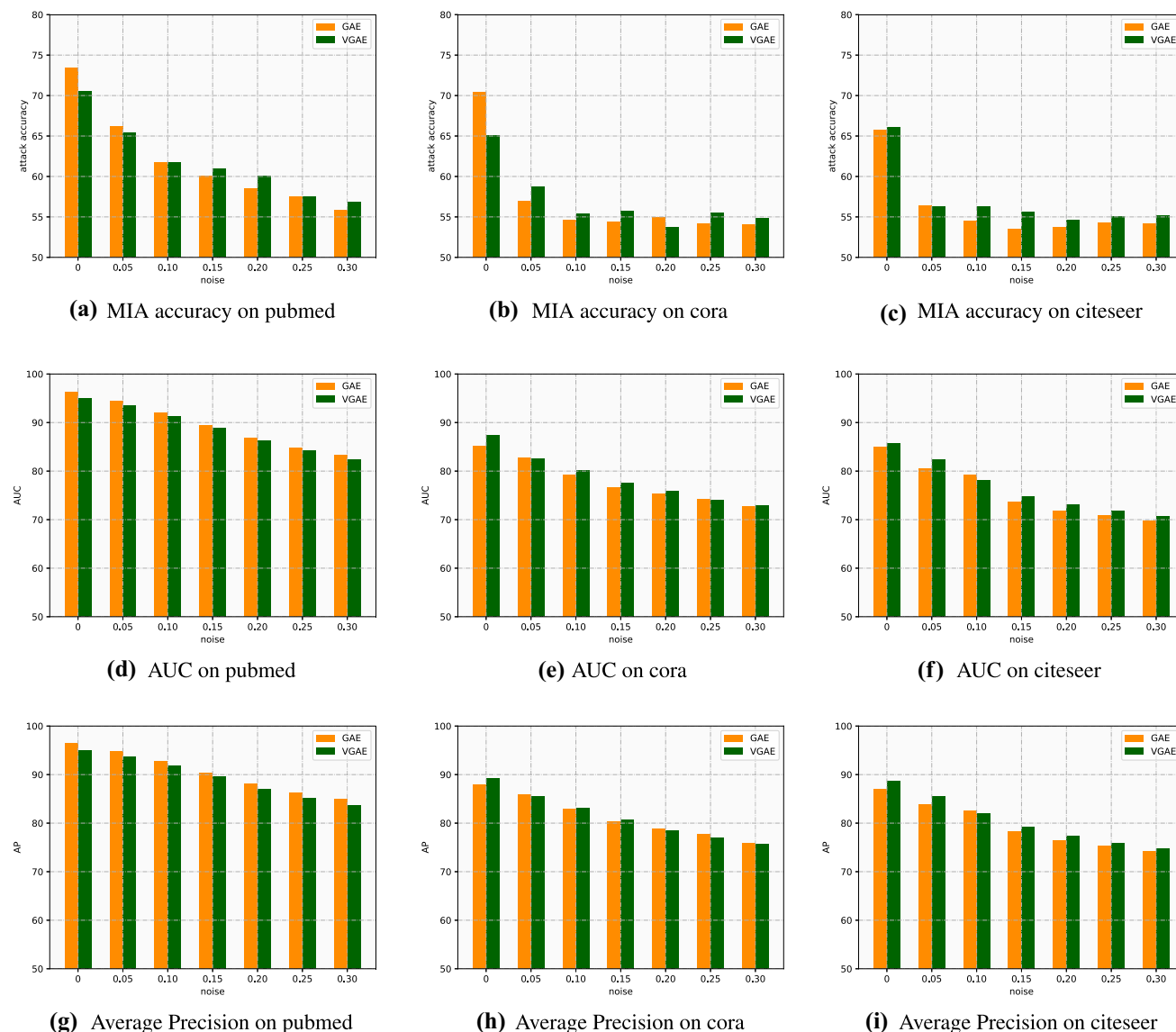


Fig. 7 Utility and Privacy with different graph reconstruction models on three datasets. We use the membership inference attack accuracy to measure the privacy-preserving effect, and the same AUC and AP to

measure the availability of the reconstructed adjacency matrix. The two neighboring bars indicate model GAE and VGAE, respectively, and show the experimental effects on the three datasets

Privacy When enough private information is available, the adversary can carry out an attack with a high success rate. Perturbation defense strategy reduces the private component of the information held by the attacker, thus reducing the success rate of member inference attacks. Different datasets have different tolerance for perturbations, such as pubmed, which retains more usable information even after perturbations are added to a certain amount, and then cora and citeseer, which are more affected even if the perturbations are controlled to a smaller range.

Computational complexity In the proposed defense model, the critical node embedding representation is obtained by aggregating the neighborhood features of the node. In other

words, the computational complexity depends on the number of edges and can be expressed as $\mathcal{O}(|\mathcal{E}|d)$, where $|\mathcal{E}|$ is the number of edges and d is the feature dimension. Because the number of edges is linear, the computational complexity is also linear.

Discussion on other defense Considering the privacy protection of graph data, defenders can make some changes to its node embedding vector before releasing it to the public. Commonly used modification methods can be classified into two categories: direct methods and indirect methods. Directness means that after obtaining the intermediate layer output from the GNN model, some modifications are made directly on the output. Indirectness refers to some modifications to the

input of the GNN model, such as modifications to the adjacency matrix or node feature matrix. With these modified data input to the GNN model, the perturbed model middle layer representation or output is then provided to the adversary. The defense method we propose in this paper achieves the defense effect from an indirect manner. From an intuitive point of view, the defense effect can be achieved by referring to the approach mentioned in [39]. Adding noise directly to the output [40] of the model is a common defense mechanism against inferential attacks. [39] chooses to inject noise directly to the embedded representation of the graph as it is oriented to graph-level tasks. The idea can also be applied to node-level tasks by adding noise to the node embedding and inference attacks in black-box manner by adding noise to the output of GNN models.

6 Related work

6.1 Previous attacks on graph

Research on privacy issues for graph neural networks is just in its infancy. It takes the first step toward a link stealing attack. Specifically, given black-box access to the target GNN model, [41] aims to predict whether a link exists between any pair of nodes in the training graph. [14] quantifies privacy leakage in node embedding for the first time through three major inference attacks targeting graph neural networks: membership inference attack, graph reconstruction attack, and attribute inference attack. Similarly, for the privacy discussion of graph embedding, [39] has recently done the relevant work. It has systematically studied the information leakage problem of graph embeddings by three inference attacks, namely, property inference attack, subgraph inference attack, and graph reconstruction attack. [42] proposes a model inversion attack on graphs, which reconstructs the structure of the graph from known information in order to steal the privacy of the training data in a white-box setting. [43] takes the first step to propose membership inference attacks for various graph-level target model. Not only two types of attacks, training-based and threshold-based, are developed, but also compared with attacks on node-level GNN. [44] conducts a comprehensive study of model extraction attacks against GNN models, which aim to construct a surrogate model behaves like the target one. This paper systematically classify the attack scenarios into seven categories based on the background knowledge of the adversary. Further advancing the work, [45] proposes a model extraction attack against inductive graph neural networks. This implicates that only a set of query graph is needed to successfully execute the attack even if it comes from a different domain than the target graph. These papers above, are more generalized proposed the basic attack method, which can be seen, more elaborate

attack on graph privacy need to be further research. Currently, there are still relatively few defense methods for graph privacy, and there is an enormous amount of work to be carried out.

6.2 Previous defenses on graph

The following progress has been made to protect the privacy of the graph, which still remains a great deal of work to be done. [46] presents a training framework, APGE, which combines disentangling and purging mechanisms. Because by learning the node representation, the adversary can infer sensitive information that the user originally did not intend to reveal. Therefore, APGE tries to remove the user's private information from the learned node representation by this framework. [47] takes into account the scenario where cloud servers use private user data to train models. Therefore, the LDP mechanism that can interfere with its features with minimal communication overhead is proposed to protect the privacy of node features. After this, the data with noise will be collected by the server. Moreover, it provides an aggregation-based denoising mechanism, which is inserted into GNNs as a separate but simple layer. [48] also proposes a vertically federated GNN learning paradigm motivated by the existing work on split-learning [49, 50], which is dedicated to protecting private information in node classification tasks. [39] presents a discussion of the corresponding defense methods after proposing membership inference attacks for graph-level GNN. It takes the form of adding noise to the target graph embedding and does the evaluation accordingly.

7 Conclusions

This work provides a defense against white-box membership inference attacks (i.e., requiring node embedding) via perturbation injection. Specifically, we inject a certain amount of noise in the latent space of the adjacency matrix to achieve the corresponding defense effect in an indirect way. This defense method achieves privacy protection while guaranteeing the availability of the reconstructed adjacency matrix. By designing to sample noise from Gaussian function with different data distributions, we can visually observe the trade-off between privacy and usability based on our proposed defense method in this paper. We apply two reconstruction models and three evaluation metrics to illustrate the reasonable performance of this defense method. Current defense methods against these three attacks have not been proposed in large numbers.

There are several representative types of privacy attacks on privacy leakage for graph neural networks, namely membership inference, model inversion, attribute inference, and model stealing. There are not yet a large number of defense

methods proposed for these attacks, which indicates that privacy protection of graphs has not yet received the attention it deserves. Simultaneously, the measurement of graph usability has not been well defined due to the visualization nature of graphs unlike pictures. All of them are worth exploring and studying in depth in the future. On the other hand, the generality of the defense method is also a key concern. We will also enhance the generality of this defense method in our future work.

Funding This study was not funded.

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest All authors declare that we have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

- Zhu, T., Li, J., Hu, X., Xiong, P., Zhou, W.: The dynamic privacy-preserving mechanisms for online dynamic social networks. *IEEE Trans. Knowl. Data Eng.* **34**(6), 2962–2974 (2022). <https://doi.org/10.1109/TKDE.2020.3015835>
- Breuer, A., Eilat, R., Weinsberg, U.: Friend or faux: graph-based early detection of fake accounts on social networks. In: Huang, Y., King, I., Liu, T., van Steen, M. (eds.) *WWW '20: The Web Conference 2020*, pp. 1287–1297. Taipei, (2020). [arxiv:2004.04834](https://arxiv.org/abs/2004.04834)
- Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks. In: 6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings, Vancouver, (2018). [arxiv:1710.10903](https://arxiv.org/abs/1710.10903)
- Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. *Data Min. Knowl. Discov.* **29**(3), 626–688 (2015). <https://doi.org/10.1007/s10618-014-0365-y>
- Gilmer J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: D. Precup, Y.W. Teh. (ed.) *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, vol. 70, pp. 1263–1272. Sydney, (2017). [arxiv:1704.01212](https://arxiv.org/abs/1704.01212)
- Sun, L., Wang, J., Yu, P.S., Li, B.: Adversarial attack and defense on graph data: a survey. *CoRR* abs/1812.10528. [arxiv:1812.10528](https://arxiv.org/abs/1812.10528)
- Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017 Conference Track Proceedings, Toulon, (2017). [arxiv:1609.02907](https://arxiv.org/abs/1609.02907)
- Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks?. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, (2019). [arxiv:1810.00826](https://arxiv.org/abs/1810.00826)
- Chiang, W., Liu, X., Si, S., Li, Y., Bengio, S., Hsieh, C.: ClusterGCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks. In: A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, G. Karypis. (eds.) *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2019*, pp. 257–266. Anchorage, (2019). [arxiv:1905.07953](https://arxiv.org/abs/1905.07953)
- Zhang, M., Chen, Y.: Link prediction based on graph neural networks. In: S. Bengio, H.M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett. (eds.) *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, Montréal, Canada, pp. 5171–5181. Canada, (2018). [arxiv:1802.09691](https://arxiv.org/abs/1802.09691)
- Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge graph embedding for link prediction: a comparative analysis. *ACM Trans. Knowl. Discov. Data* **15**(2), 1 (2021)
- Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: McIlraith, S.A., Weinberger, K.Q. (eds.) *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), pp. 4438–4445. New Orleans, (2018)
- Cai, H., Zheng, V.W., Chang, K.C.: A comprehensive survey of graph embedding: problems, techniques, and applications. *IEEE Trans. Knowl. Data Eng.* **30**(9), 1616–1637 (2018)
- Duddu, V., Boutet, A., Shejwalkar, V.: Quantifying privacy leakage in graph embedding. In: Mühlhäuser, M., Polyzos, G.C., Michahelles, F., Guinea, A.S., Wang, L. (eds.) *MobiQuitous '20: Computing, Networking and Services, Virtual*, pp. 76–85. Darmstadt, (2020). [arxiv:2010.00906](https://arxiv.org/abs/2010.00906)
- Shokri, R., Stronati, M., Song, C., Shmatikov, V.: Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on Security and Privacy, SP 2017, pp. 3–18. San Jose, (2017). [arxiv:1610.05820](https://arxiv.org/abs/1610.05820)
- Liu, B., Ding, M., Shaham, S., Rahayu, W., Farokhi, F., Lin, Z.: When machine learning meets privacy: a survey and outlook. *ACM Comput. Surv.* **54**(2), pp. 31:1–31:36 (2021). <https://doi.org/10.1145/3436755>
- Sun, H., Zhu, T., Zhang, Z., Jin, D., Xiong, P., Zhou, W.: Adversarial attacks against deep generative models on data: a survey. *IEEE Trans. Knowl. Data Eng.* (2021). [arxiv:2112.00247](https://arxiv.org/abs/2112.00247)
- Papernot, N., McDaniel, P.D., Goodfellow, I.J., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: R. Karri, O. Sinanoglu, A. Sadeghi, X. Yi. (eds.) *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017*, pp. 506–519. Abu Dhabi, (2017). [arxiv:1602.02697](https://arxiv.org/abs/1602.02697)
- Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction APIs. In: Holz, T., Savage, S. (eds.) *25th USENIX Security Symposium, USENIX Security 16*, pp. 601–618. Austin, (2016). [arxiv:1609.02943](https://arxiv.org/abs/1609.02943)
- Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 3431–3440. Boston, (2015). [arxiv:1411.4038](https://arxiv.org/abs/1411.4038)
- Kim, Y.: Convolutional neural networks for sentence classification. In: A. Moschitti, B. Pang, W. Daelemans. (eds.) *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 1746–1751. Doha, (2014). [arxiv:1408.5882](https://arxiv.org/abs/1408.5882)
- Wang, Z., Kuang, W., Xie, Y., Yao, L., Li, Y., Ding, B., Zhou, J.: FederatedScope-GNN: towards a Unified, comprehensive and efficient package for federated graph learning. In: A. Zhang, H. Rangwala. (eds.) *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, August 14 - 18, 2022, pp. 4110–4120. ACM, (2022). <https://doi.org/10.1145/3534678.3539112>

23. Zhang, K., Yang, C., Li, X., Sun, L., Yiu, S.: Subgraph federated learning with missing neighbor generation. In: M. Ranzato, A. Beygelzimer, Y.N. Dauphin, P. Liang, J.W. Vaughan Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6–14, 2021, virtual, pp. 6671–6682. (2021).[arxiv:2106.13430](https://arxiv.org/abs/2106.13430)
24. Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, R. Garnett. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, pp. 1024–1034. Long Beach, (2017).[arxiv:1706.02216](https://arxiv.org/abs/1706.02216)
25. Hu, H., Salcic, Z., Sun, L., Dobbie, G., Yu, P.S., Zhang, X.: Membership inference attacks on machine learning: a survey. *ACM Computing Surveys (CSUR)* (2021). <https://doi.org/10.1145/3523273>
26. Zhou, J., Chen, Y., Shen, C., Zhang, Y.: Property inference attacks against GANs. *CoRR abs/2111.07608* (2021).[arxiv:2111.07608](https://arxiv.org/abs/2111.07608)
27. Zhang, H., Wu, B., Yuan, X., Pan, S., Tong, H., Pei, J.: Trustworthy graph neural networks: aspects, methods and trends. *CoRR abs/2205.07424* (2022). <https://doi.org/10.48550/arXiv.2205.07424>
28. Y. Shen, X. He, Y. Han, Y. Zhang.: Model stealing attacks against inductive graph neural networks. In: 2022 IEEE Symposium on Security and Privacy (SP) pp. 1175, IEEE. (2022).[arxiv:2112.08331](https://arxiv.org/abs/2112.08331)
29. Liu, Y., Wen, R., He, X., Salem, A., Zhang, Z., Backes, M., De Cristofaro, E., Fritz, M., Zhang, Y.: *{ML – Doctor}*: Holistic risk assessment of inference attacks against machine learning models. In: 31st USENIX Security Symposium (USENIX Security 22) pp. 4525–4542. (2022).[arxiv:2102.02551](https://arxiv.org/abs/2102.02551)
30. He, X., Wen, R., Wu, Y., Backes, M., Shen, Y., Zhang, Y.: Node-level membership inference attacks against graph neural networks. *CoRR abs/2102.05429* (2021). [arXiv:2102.05429](https://arxiv.org/abs/2102.05429)
31. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(11), 535 (2008).<http://www.jmlr.org/papers/v9/vandermaaten08a.html>
32. Kipf, T.N., Welling, M.: Variational graph auto-encoders. *CoRR abs/1611.07308* (2016). [arxiv:1611.07308](https://arxiv.org/abs/1611.07308)
33. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Y. Bengio, Y. LeCun. (eds.) 2nd International Conference on Learning Representations, ICLR 2014, April 14–16, 2014, Conference Track Proceedings, Banff, (2014).[arxiv:1312.6114](https://arxiv.org/abs/1312.6114)
34. Zhu, T., Ye, D., Wang, W., Zhou, W., Yu, P.: More than privacy: applying differential privacy in key areas of artificial intelligence. *IEEE Trans. Knowl. Data Eng.* (2020). <https://doi.org/10.1109/TKDE.2020.3014246>
35. Ye, D., Shen, S., Zhu, T., Liu, B., Zhou, W.: One parameter defense—defending against data inference attacks via differential privacy. *IEEE Trans. Inf. Forensics Secur.* **17**, 1466 (2022).[arxiv:2203.06580](https://arxiv.org/abs/2203.06580)
36. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: privacy via distributed noise generation. In: S. Vaudenay. (ed.) *Advances in Cryptology - EUROCRYPT 2006*, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings, Lecture Notes in Computer Science, vol. 4004, pp. 486–503. Russia, (2006).https://doi.org/10.1007/11761679_29
37. Namata, G., London, B., Getoor, L., Huang, B., EDU, U.: Query-driven active surveying for collective classification. In: 10th International Workshop on Mining and Learning with Graphs. vol. 8, p. 1. (2012).<https://people.cs.vt.edu/~bhuang/papers/namata-mlg12.pdf>
38. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. *AI Magaz.* **29**(3), 93–93 (2008).<https://doi.org/10.1609/aimag.v29i3.2157>
39. Zhang, Z., Chen, M., Backes, M., Shen, Y., Zhang, Y.: Inference attacks against graph neural networks. *CoRR abs/2110.02631* (2021). [arxiv:2110.02631](https://arxiv.org/abs/2110.02631)
40. Zhang, Z., Wang, T., Li, N., Honorio, J., Backes, M., He, S., Chen, J., Zhang, Y., PrivSyn: Differentially Private Data Synthesis. In: M. Bailey, R. Greenstadt. (eds.) 30th USENIX Security Symposium, USENIX Security 2021 (August), pp. 11–13, pp. 929–946. (2021).<https://www.usenix.org/system/files/sec21fall-zhang-zhikun.pdf>
41. He, X., Jia, J., Backes, M., Gong, N.Z., Zhang, Y., Links, Stealing, from graph neural networks. In: M. Bailey, R. Greenstadt. (eds.) 30th USENIX Security Symposium, USENIX Security 2021 (August), pp. 11–13, pp. 2669–2686. (2021).[arxiv:2005.02131](https://arxiv.org/abs/2005.02131)
42. Zhang, Z., Liu, Q., Huang, Z., Wang, H., Lu, C., Liu, C., Chen, E.: GraphMI: extracting private graph data from graph neural networks. In: Z. Zhou. (ed.) *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / 19–27 August 2021*, pp. 3749–3755, Montreal, (2021).[arxiv:2106.02820](https://arxiv.org/abs/2106.02820)
43. Wu, B., Yang, X., Pan, S., Yuan, X.: Adapting membership inference attacks to GNN for graph classification: approaches and implications. In: Bailey, J., Miettinen, P., Koh, Y.S., Tao, D., Wu, X. (eds.) *IEEE International Conference on Data Mining, ICDM 2021, December 7–10, 2021*, pp. 1421–1426. New Zealand, (2021).[arxiv:2110.08760](https://arxiv.org/abs/2110.08760)
44. Wu, B., Yang, X., Pan, S., Yuan, X.: Model extraction attacks on graph neural networks: taxonomy and realization. *CoRR abs/2010.12751* (2020). [arxiv:2010.12751](https://arxiv.org/abs/2010.12751)
45. Shen, Y., He, X., Han, Y., Zhang, Y.: Model stealing attacks against inductive graph neural networks. *CoRR abs/2112.08331* (2021). [arxiv:2112.08331](https://arxiv.org/abs/2112.08331)
46. Li, K., Luo, G., Ye, Y., Li, W., Ji, S., Cai, Z.: Adversarial privacy-preserving graph embedding against inference attack. *IEEE Internet Things J.* **8**(8), 6904–6915 (2021).[arxiv:2008.13072](https://arxiv.org/abs/2008.13072)
47. Sajadmanesh, S., Gatica-Perez, D.: Locally private graph neural network. In: Y. Kim, J. Kim, G. Vigna, E. Shi. (eds.) *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19*. pp. 2130–2145. Republic of Korea, (2021).[arxiv:2006.05535](https://arxiv.org/abs/2006.05535)
48. Zhou, J., Chen, C., Zheng, L., Wu, H., Wu, J., Zheng, X., Wu, B., Liu, Z., Wang, L.: Vertically federated graph neural network for privacy-preserving node classification (2021).[arxiv:2005.11903](https://arxiv.org/abs/2005.11903)
49. Gu, Z., Huang, H., Zhang, J., Su, D., Lamba, A., Pendarakis, D., Molloy, I.M.: Securing input data of deep learning inference systems via partitioned enclave execution. *CoRR abs/1807.00969* (2018). [arxiv:1807.00969](https://arxiv.org/abs/1807.00969)
50. Osia, S.A., Shamsabadi, A.S., Sajadmanesh, S., Taheri, A., Katevas, K., Rabiee, H.R., Lane, N.D., Haddadi, H.: A hybrid deep learning architecture for privacy-preserving mobile analytics. *IEEE Internet Things J.* **7**(5), 4505–4518 (2020).[arxiv:1703.02952](https://arxiv.org/abs/1703.02952)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.