Neurocomputing 521 (2023) 113–125

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Differential privacy preservation for graph auto-encoders: A novel anonymous graph publishing model

Xiaolin Li [a,b], Li Xu [a,b,*], Hongyan Zhang [a,b], Qikui Xu [c]

[a] College of Computer and Cyber Security, Fujian Normal University, Fuzhou, Fujian, China
[b] Fujian Provincial Key Laboratory of Network Security and Cryptology, Fujian Normal University, Fuzhou, Fujian, China
[c] School of Life Sciences, Fudan University, Shanghai, China

ARTICLE INFO

Article history:
Received 27 February 2022
Revised 21 October 2022
Accepted 27 November 2022
Available online 5 December 2022

Keywords:
Graph neural network
Differential privacy
Multi-task learning
Anonymous graph publishing

ABSTRACT

In recent years, the number of users in social networks has grown substantially, and more data-intensive applications have been developed. This creates a demand for the ability to mine large-scale graph data more efficiently, so that the information mined can be maximized (e.g., mining social relationships between people). However, the direct publication of the original graphs leads to potential leakage of users' privacy. Therefore, graph anonymization techniques are often utilized to process the original graphs. A key challenge of it lies in the balance between anonymity and usability. In this paper, we introduced the idea of graph auto-encoder, a fundamental element in graph neural networks, and proposed the Differential Privacy Deep Graph Auto-Encoder (DP-DGAE). Our main idea is to convert the anonymous graph publishing problem into a privacy-preserving problem for generative models, and optimize the models in terms of both privacy and usability using a multi-task learning approach. Theoretical analysis and experimental evaluations show that the DP-DGAE achieves anonymity while ensuring usability.

1. Introduction

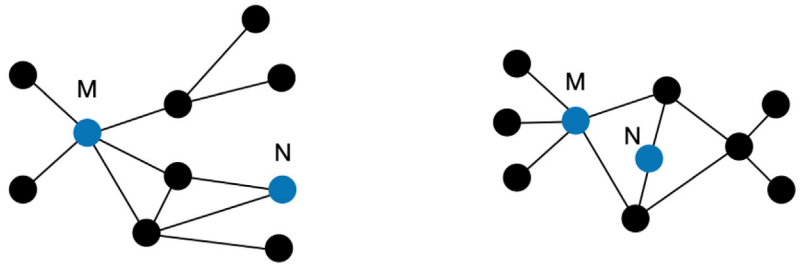
Nowadays, by modeling user relationships in social networks in the format of graphs, a large pool of structured data has been generated. Such open graph (network) data provide the best materials for data mining and data analysis [1,2]. Data companies derive enormous potential value from data by releasing and sharing graph data with research facilities and enterprise partners, which can assist decision models in applications including social, business and transportation, etc., through improved advertising, recommendations, etc. [3,4].

However, since graph data contain a wealth of information, it can lead to leakage of users' privacy, as well as their relationships. What is more, even if the graph is anonymized when it is published, it is difficult to guarantee the safety of users' privacy under collaborative attacks [5,6]. For such reasons, traditional privacy protection methods such as differential privacy and other techniques are often used [7]. However, these schemes do not make full use of the existing empirical knowledge in graphs to obtain potential information, while deep learning uses multiple nonlinear layers of transformation to perform representation learning, and is hence more powerful in data abstraction [8]. Recently, Graph Neural Networks (GNNs) have extended deep neural networks to graph-structured data by jointly encoding network structure and node features, demonstrating a powerful ability to learn node representations. They have been proven to have powerful capabilities in tasks including link prediction [9], node classification [10], and graph classification [11,12]. However, due to overfitting in deep learning, it makes the model implicitly remember the details of training data, and there is a risk of privacy leakage. Therefore, many methods that incorporate differential privacy into deep learning have recently been proposed. These schemes hide or alter sensitive features of the training dataset while maintaining the usability of further learning in the deep learning model. However, a pressing problem is how to perform privacy protection in the graph generation neural network model so that the published graphs have high usability while protecting the privacy of the links.

For example, in Fig. 1, a social networking service provider encouraged researchers to study the relationship structure of users in its network. As shown in Fig. 1(a), the data publisher first removes the users' tags and then releases the network to the public. However, when the attacker has the background knowledge of the current social network, such as the number of friends of the target user and the social information of the friends, the attacker

* Corresponding author at: College of Computer and Cyber Security, Fujian Normal University, Fuzhou, Fujian, China.
E-mail address: xuli@fjnu.edu.cn (L. Xu).

https://doi.org/10.1016/j.neucom.2022.11.083
0925-2312/



(a) Anonymized original graph (b) DP-DGAE gerenated graph

Fig. 1. A pair of anonymized and generated graphs.

can use the degree distribution of the nodes and the neighbor information of the target node to accurately identify the target node in the published graph (e.g., user M as the only node with degree 5 and the degree of its neighbor users are 1,1,2,3 and 4, respectively, and the user N as the only node with degree 2 and the degree of its neighbor user are 2 and 4, respectively.). As a result, an attacker can identify M and N and obtain connection information or mutual friend information between them, which seriously compromises users' privacy and poses a potential fraud risk.

In this paper, we implement the ϵ-differential privacy preserving in the graph generation neural network model. When the deep graph auto-encoder is used for a binomial classification task, the generated graph can satisfy the ϵ-differential privacy. We proposed a novel ϵ-**D**ifferential **P**rivacy **D**eep **G**raph **A**uto-**E**ncoder (DP-DGAE) that uses the functional mechanism (FM) [13] to perturb the objective function of the graph auto-encoders to preserve the ϵ-differential privacy. At the same time, multi-task learning is used to make the model achieve a balance between privacy and usability. While meeting privacy requirements, it retains the structural characteristics of the published graph and its good performance in link prediction. As shown in Fig. 1(b), our model trains on top of the original graph and publishes the generated graph. In order for a downstream data mining task to produce results similar to the original graph, we require the generated graph to be similar to the original graph in terms of global structural availability, which can be measured by common graph properties (e.g., graph (b) has the same degree distribution as (a)). Also, in the original graph, based on the number of common neighbors, users M and N have a large probability of being connected in the future, which can also be reflected in the generated graph. For privacy protection, although an attacker may be able to re-identify the target node by degree in the generated network, the structural information of the target node as well as its neighbors has been changed, so the attacker cannot accurately obtain the link information between the target node and its neighbor nodes. To evaluate its performance, we apply the DP-DGAE in real social networks. Experimental results show that the DP-DGAE model achieves a good balance between usability and anonymity.

The main contributions are summarized as follows.

- We apply differential privacy to the output layer of the graph generation model and transfer an anonymous graph publishing problem into a privacy-preserving problem of the deep graph generation model.
- We use multi-task learning to simultaneously learn the perturbed objective function and the original objective function and cast the multi-task learning problem as a multi-objective optimization problem with the overall objective of finding a Pareto optimal solution so that the model can strike a balance between privacy and usability.
- Aiming at the zero mean and high variance of the differential privacy noise distribution, we design a gradient modification method to improve the problem of ignoring a certain objective function when the traditional multi-task learning method is applied to deep learning with differential privacy.

2. Related work

2.1. Differential privacy

Differential privacy [7] has become a prevalent privacy model because of its good mathematical properties. It is used to quantify the notion of "indistinguishability" of neighboring databases. There have been extensive researches on enforcing differential privacy to particular analysis tasks in social networks, e.g., graph data publishing [14,15], track privacy protection [16], collaborative recommendation [17], and transmitting information [18] in social network analysis. Recently, the privacy issues exposed by the rapid development of deep learning have led to the application of differential privacy to deep learning as well, e.g., [19,20].

2.2. Graph neural network

Graph neural networks efficiently obtain a representation of nodes by aggregating network structures as well as features [21,22]. In this way, the feature information of nodes is propagated through the network topology, and the information obtained by node aggregation generates node embeddings, which are then used for the next tasks such as node and graph classification [11,12], link prediction [9], and recommendation [23,24].

2.3. Differential privacy deep learning mechanism

Deep learning is widely used because of its powerful data abstraction capability, but there is a risk of privacy leakage because of the existence of over-fitting, so the model implicitly memorizes some details of the training data [25]. Therefore, there is a growing trend to include differential privacy in deep learning models to protect the privacy of the models as well as the data [19]. Existing differential privacy deep learning schemes can be classified into three categories based on where differential privacy is deployed: input layer [26], hidden layer [20,27], and output layer [28].

The existing methods of adding differential privacy to deep generative models [28] are primarily designed to process data with regular(Euclidean) structures such as images and text. However, unlike images and other grid-based data, graphs have flexible structures and arbitrary node orders, and traditional deep generation models do not capture the structural information of graphs well. Therefore, the existing method of adding differential privacy to deep generative models [28] cannot be directly used for graph data. Meanwhile, for existing differential privacy graph generation

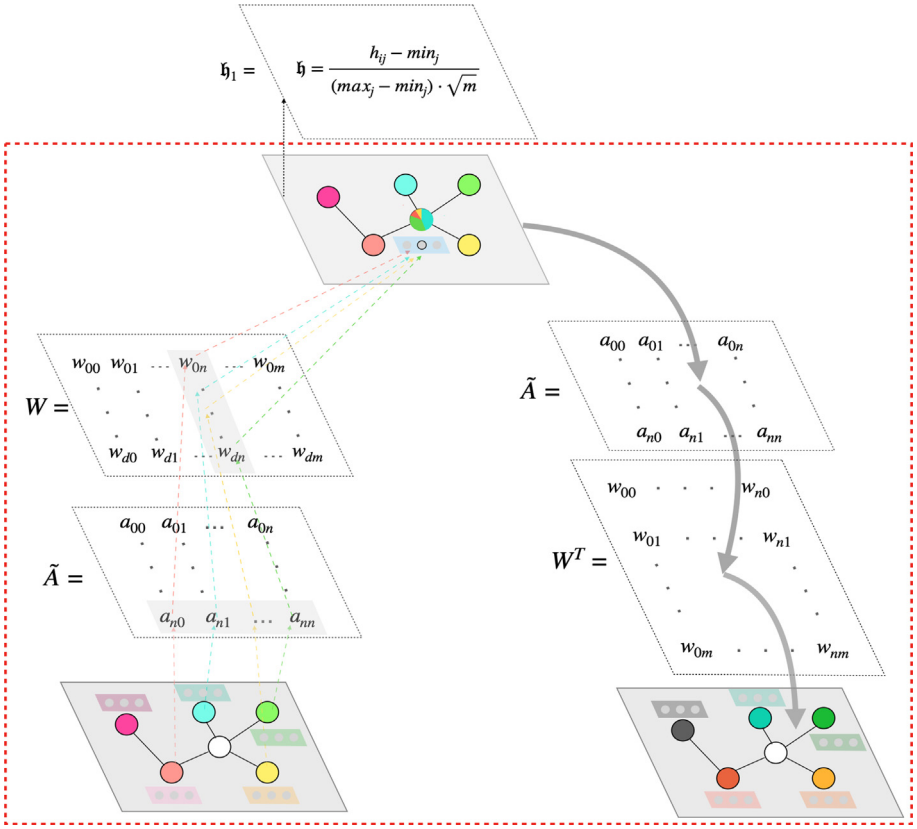


Fig. 2. The structure of Graph Auto-Encoder(in the red box) and Differential Privacy Graph Auto-Encoder.

models such as DPGGAN model [27] which does not make sufficient use of the attribute information in the graph, while adding noise to the gradient because of the small privacy budget will limit the number of training epochs, leading to lower utility and the model is limited by the number of training epochs.

3. Preliminaries

3.1. Notations

We focus on semi-supervised graph link generation with differential privacy properties in an attributed graph $G = (A, X)$, where $A \in \mathbb{R}^{n \times n}$ is the symmetric adjacency matrix with n nodes and $X \in \mathbb{R}^{n \times d}$ is the node feature matrix, where d is the dimension of node features. Specifically, $A_{ij} = 1$ represents there is an edge between node i and j, otherwise, $A_{ij} = 0$.

3.2. Differential privacy

Definition 1 (ϵ-Differential Privacy [7]). A randomized algorithm $\mathscr{A}$ is ϵ-differentially private if and only if for any two databases D_1 and D_2 differing at most one tuple, and for any output O $\in$ Range($\mathscr{A}$),

$$Pr[\mathscr{A}(D_1) \in O] \leqslant e^{\epsilon} \times Pr[\mathscr{A}(D_2) \in O] \tag{1}$$

This method hides the difference of any tuple from an adversary. We can control the information leakage caused by the difference between two neighboring data sets by controlling the privacy budget ϵ. The smaller the ϵ, the higher the privacy requirements. We can achieve differential privacy through *the Laplace mechanism* [7]. In Laplace mechanism, for any two neighboring databases D_1 and D_2, the sensitivity of a function f is defined as $\Delta f = max_{D_1,D_2} ||f(D_1) - f(D_2)||$. Given a particular function f and calculating its global sensitivity, the noise η can be drawn from a Laplace distribution with $p(x|\lambda) = \frac{1}{2\lambda} e^{-|x|/\lambda}$ probability density function, where $\lambda = \Delta f/\epsilon$, according to different privacy budget ϵ. By injecting noise η into the output of $f(D)$, it can ensure the ϵ-differential privacy.

3.3. Functional mechanism

The functional mechanism [13] achieves ϵ-differential privacy by releasing parameters $\overline{\omega}$ that minimizes the objective function $f_X(\omega)$ after perturbation. For the objective function $f_X(\omega)$, assume that the model parameter ω is a vector containing d-dimensional values $\omega_1, \ldots, \omega_d$. We use the product of these d-dimensional vectors to construct an nth degree polynomial $\Phi_n = \{\omega_1^{m_1} \cdot \omega_2^{m_2} \cdots \omega_n^{m_n} | \sum_{l=1}^{d} m_l = n\}$. According to Stone-Weierstrass Theorem [29], we can rewrite $f_X(\omega)$ using the polynomial form of ω, for some $N \in [0, \infty]$, i.e., $f(t_i, \omega) = \sum_{n=0}^{N} \sum_{\phi \in \Phi_n} \lambda_{\phi t_i} \phi(\omega)$ where $\lambda_{\phi t_i} \in \mathbb{R}$ denotes the coefficient of $\phi(\omega)$ in the polynomial. Zhang et al. [13] developed an approximation polynomial form based on Taylor expansion [30], which is used to solve the calculation problem of the polynomial form of the objective function including the term of unbounded degree. For instance, given the cost function $f(t_i, w)$, assume that there exist $2m$ functions $f_1, \ldots, f_m$ and $g_1, \ldots, g_m$ such that $f(t_i, \omega) = \sum_{l=1}^{\infty} f_l(g_l(t_i, \omega))$, and each $g_l(t_i, \omega)$ is a polynomial function of ω, where $m \in N$ is the number of functions f and g. By analyzing the above decomposition of $f(t_i, \omega)$, we can know it is feasible to apply Taylor expansion to each $f_l(\cdot)$. Thus, the obtain the polynomial form if the objective function $f(X, \omega)$ i.e.,

$$\tilde{f}_X(\omega) = \sum_{i=1}^{|X|} \sum_{l=1}^{m} \sum_{R=0}^{\infty} \frac{f_l^{(R)}(z_l)}{R!} (g_l(t_i, \omega) - z_l)^R \tag{2}$$

where each z_l is a real number.

We perturb $\tilde{f}_X(\omega)$ by injecting Laplace noise $Lap(\frac{\Delta}{\epsilon})$ into its coefficients λ_ϕ, and then derive the model parameter $\overline{\omega}$ that minimizes the perturbed function $\overline{f}_X(\omega)$, where the global sensitivity Δ of $\tilde{f}_X(\omega)$ satisfies the following inequality:$\Delta \leqslant 2 \max_t \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} \|\lambda_{\phi t}\|_1$.

3.4. Graph auto encoder

By training the graph auto-encoder [31](Fig. 2), the input feature matrix X and adjacency matrix A can be encoded into some intermediate representation $h \in \mathbb{R}^{n \times m}$, where m is a constant, and the reconstruction of the adjacency matrix A is completed by decoding this intermediate representation. Given the reconstruction matrix $\widehat{A}$, the negative log-likelihood of the reconstruction process is,

$$\begin{aligned} RE(A_i, W) &= -logP\left(A_i | \widehat{A}_i, W\right) \\ &= -\sum_{j=1}^{d} \left(A_{ij} log \widehat{A}_{ij} + (1 - A_{ij}) log\left(1 - \widehat{A}_{ij}\right)\right) \end{aligned} \tag{3}$$

where W is a weight matrix, We demonstrate this model using a *graph convolution network* (GCN) [10] as encoder and decoder.

$$h = \sigma\left(\widetilde{A}XW\right), \quad \widehat{A} = \sigma\left(\widetilde{A}hW^T\right), \tag{4}$$

$\sigma(\cdot)$ is the sigmoid function and $\widetilde{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}$ is the symmetrically normalized adjacency matrix. We can sum all the node topology reconstruction losses $RE(A_i, W)$ in the graph to get the loss function on the whole graph A:

$$\begin{aligned} RE(A, W) &= \sum_{i=1}^{n} RE(A_i, W) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{d} \left[A_{ij} log\left(1 + e^{-\widetilde{A}h_i W_j}\right) + (1 - A_{ij}) log\left(1 + e^{\widetilde{A}h_i W_j}\right)\right] \end{aligned} \tag{5}$$

we can stack multiple graph auto-encoder to produce a deep graph auto-encoder. The output layer of the deep graph auto-encoder includes a single binomial variable to predict A. The reconstruction matrix $\widehat{A}$ is fully linked to the hidden layer $h_{(k)}$ through the weight matrix $W_{(k)} \in \mathbb{R}^{m \times n}$, where k is the number of hidden layers in the deep graph auto-encoder. We use the sigmoid function as an activation function of $\widehat{A}$, i.e., $\widehat{A} = \sigma\left(\widetilde{A}h_{(k)}W_{(k)}\right)$. let A_T be the adjacency matrix used to train the model, the cross-entropy error function is given by

$$C(A_T, \theta) = -\sum_{i=1}^{n} \sum_{j=1}^{n} (a_{ij} log \hat{a}_{ij} + (1 - a_{ij}) log(1 - \hat{a}_{ij})) \tag{6}$$

where n is the number of nodes in A_T, a and $\hat{a}$ are the variables in A_T and $\widehat{A}$ respectively.

Algorithm 1: **:Pseudo Code of a DP-DGAE model**

1: Construct the feature reconstruction function $RE(X, W)$
2: Derive polynomial approximation of features reconstruction function $RE(X, W)$, denoted as $\widehat{RE}(X, W)$
3: The function $\widehat{RE}(X, W)$ is perturbed by using *functional mechanism* (FM), the perturbed function is denoted as $\overline{RE}(X, W)$
4: Compute $\overline{W} = arg\, min_W \overline{RE}(X, W)$
5: Differential Privacy Graph Auto-Encoder stacking

a (*continued*)

Algorithm 1: **:Pseudo Code of a DP-DGAE model**

6: Derive and perturb the polynomial approximation of feature-to-structure reconstruction cross-entropy error $C(A_T, \theta)$, the perturbed function is denoted as $\overline{C}(A_T, \theta)$
7: Calculate the Pareto optimal solution α
8: Compute $\overline{\theta} = argmin_\theta \left[\alpha C(A_T, \theta) + (1 - \alpha)\overline{C}(A_T, \theta)\right]$
9: Return $\overline{\theta}$

4. Model:DP-DGAE

In this section, we will discuss the deep graph auto-encoder model with differential privacy preservation, namely the DP-DGAE model. Our algorithm for building DP-DGAE consists of eight steps(Algorithm 1). We use the functional mechanism to enforce ε-differential privacy in our DP-DGAE model. Therefore, in the first step, we construct the feature reconstruction function $RE(X, W)$. In the second step, we derive a polynomial approximation of features reconstruction function $RE(X, W)$, denoted as $\widehat{RE}(X, W)$. In the third step, we first calculate the global sensitivity for the graph auto-encoder, and then use the functional mechanism to perturb the polynomial approximation of the feature reconstruction function $\widehat{RE}(X, W)$. The feature reconstruction function after the perturb is denoted as $\overline{RE}(X, W)$. In the fourth step, we pre-train the graph auto-encoder to obtain the locally optimal perturbed parameters $\overline{W}$. That results in differential privacy graph auto-encoder (DP-GAE)(Fig. 2). In the fifth step, we stack DP-GAE to construct the DP-DGAE(Fig. 3). Before each stacking operation, we need to normalize the representation of the current layer, so we introduce a normalization layer,denoted as $\mathfrak{h}$ (Fig. 3). In the sixth step, in order to complete the reconstruction of the graph topology, we need to derive and perturb the polynomial approximation of the

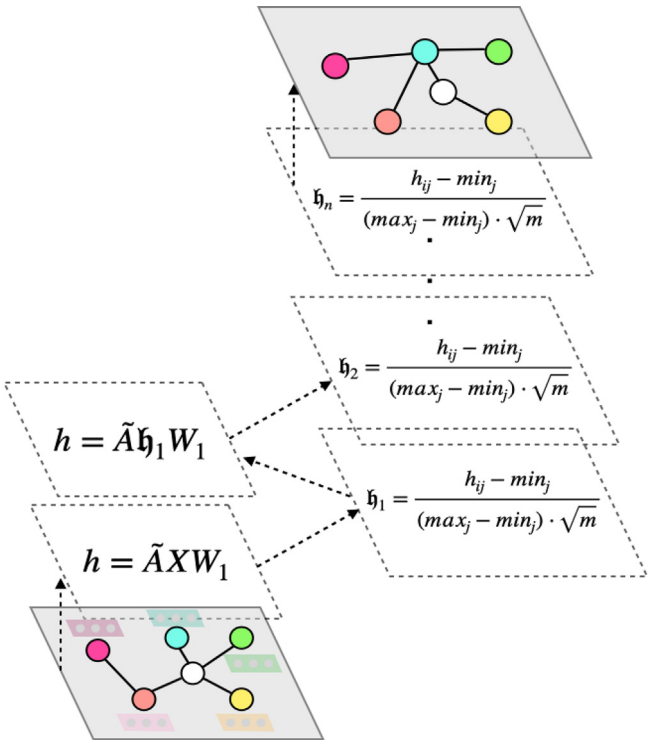


Fig. 3. The structure of Differential Privacy Deep Graph Auto-Encoder.

cross-entropy error $C(A_T,\theta)$, denoted as $\overline{C}(A_T,\theta)$. In the seventh step, we use multi-task learning to consider both $C(A_T,\theta)$ and $\overline{C}(A_T,\theta)$ loss functions and transfer a multi-task learning problem into a multi-objective optimization problem to obtain a Pareto optimal solution α that allows both objective functions to simultaneously decrease in the minimize direction. In the eighth step, the back-propagation algorithm acts on all the parameters of DP-DGAE. In our framework, the parameters and outputs of the DP-DGAE model are preserved by ϵ-differential privacy. Since the GAE underlying our model is a graph generation model, we can make the graph generated by the DP-DGAE model satisfy differential privacy. First, in order to add noise to the features in the pre-training phase and complete the reconstruction of the features, let us construct the feature reconstruction function as follows.

4.1. Feature reconstruction

For the feature reconstruction function $RE(X,W)$, referring to Eq. (3), we use the same inputs A and X. We use the input feature matrix X to replace A in the reconstructed loss function to evaluate the reconstruction process of the feature matrix.

$$\begin{aligned} RE(X_i,W) &= -logP\left(X_i|\widehat{X}_i,W\right) \\ &= -\sum_{j=1}^{d}\left(X_{ij}\,log\widehat{X}_{ij}+(1-X_{ij})\,log\left(1-\widehat{X}_{ij}\right)\right) \end{aligned} \tag{7}$$

We use GCN as encoder and decoder.

$$h=\sigma\left(\widetilde{A}XW^T\right),\quad \widehat{X}=\sigma\left(\widetilde{A}hW^T\right) \tag{8}$$

We can sum all the node feature reconstruction losses $RE(X_i,W)$ in the graph to get the loss function on the whole graph A:

$$\begin{aligned} RE(X,W) &= \sum_{i=1}^{n}RE(X_i,W) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{d}\left[X_{ij}\,log\left(1+e^{-\widetilde{A}h_iW_j}\right)+(1-X_{ij})\,log\left(1+e^{\widetilde{A}h_iW_j}\right)\right] \end{aligned} \tag{9}$$

After obtaining the feature reconstruction loss function, we use the Functional mechanism to add differential privacy to the feature reconstruction loss function, adding noise to the features of the nodes can be achieved by using the feature reconstruction loss function as the loss of the model in the pre-training phase, and retain the learned affine matrix $\overline{W}$ to initialize the weight matrix in the topological reconstruction phase.

4.2. Polynomial approximation

For the feature reconstruction function $RE(X,W)$, referring to Eq. (9), we apply the Taylor expansion to it. $\forall j\in\{1,\ldots,d\}$, let f_{1j},f_{2j}, and g_j, be three functions defined as follows:

$$\begin{aligned} &g_j(X_i,W_j)=\widetilde{A}h_iW_j; \\ &f_{1j}(z_j)=X_{ij}log(1+e^{-z_j}); \\ &f_{2j}(z_j)=(1-X_{ij})log(1+e^{z_j}); \end{aligned} \tag{10}$$

Then, we have

$$RE(X_i,W)=\sum_{j=1}^{d}\left(f_{1j}(g_j(X_i,W_j))+f_{2j}(g_j(X_i,W_j))\right) \tag{11}$$

By Eq. (2) and setting z_{ij} = 0, we have:

$$\widetilde{RE}(X,W)=\sum_{i=1}^{|X|}\sum_{j=1}^{d}\sum_{l=1}^{2}\sum_{R=0}^{\infty}\frac{f_{lj}^{(R)}(0)}{R!}\left(\widetilde{A}h_iW_j\right)^R \tag{12}$$

Because using Taylor expansion will introduce infinite sums, we simplify the private data reconstruction process by proposing an approximation that reduces the degree of summation by truncating all polynomial terms with order larger than 2 in the Taylor series of Eq. (12) to obtain an objective function that contains only low-order polynomials:

$$\begin{aligned} \widehat{RE}(X,W)=\sum_{i=1}^{|X|}\sum_{j=1}^{d}&\left(\sum_{l=1}^{2}f_{lj}^{(0)}(0)+\left(\sum_{l=1}^{2}f_{lj}^{(1)}(0)\right)\widetilde{A}h_iW_j\right. \\ &\left.+\left(\sum_{l=1}^{2}\frac{f_{lj}^{(2)}(0)}{2!}\right)\left(\widetilde{A}h_iW_j\right)^2\right) \end{aligned} \tag{13}$$

We truncate the Taylor series with an order larger than 2 so that the approximate loss function is in quadratic form and is a convex function, so the reconstruction process becomes a convex optimization problem with a theoretical global minimum, avoiding the risk of falling into a local minimum, and also ensuring that the approximate loss function can choose to use second-order derivatives to assist the optimization algorithm in accelerating the optimization.

4.3. Perturbation objective function

In this section, we add the Laplace noise determined by the functional mechanism to the polynomial coefficients of the objective function $\widehat{RE}(X,W)$. The perturbed function is denoted as $\overline{RE}(X,W)$. Then we derive the model parameter $\overline{W}$ that minimizes the perturbed function $\overline{RE}(X,W)$. Before that, we need to know the scale of Laplace noise distribution, so we need to explore the sensitivity, denoted as Δ, of $\widehat{RE}$ on X. According to the definition of Differential privacy's sensitivity, we can obtain the following lemma.

Lemma 1. *Suppose X and X' be any two neighboring datasets, $\widehat{RE}(X,W)$ and $\widehat{RE}(X',W)$ are the reconstruction objective functions of the graph auto-encoder on these two neighboring datasets, respectively, then the global sensitivity of $\widehat{RE}$ on any two neighboring datasets is as follows:*

$$\Delta=2\max_{x}\sum_{j=1}^{d}\sum_{R=0}^{2}||\lambda_{jx}^{(R)}||\leqslant d\left(z+\frac{1}{4}z^2\right) \tag{14}$$

where z is the dimension of the hidden layer.

Proof 1. *Assume that X and X' differ in the last tuple. Let x_n (x'_n) be the last tuple in X (X'). Then, $\Delta=\sum_{j=1}^{d}\sum_{R=0}^{2}\left\|\sum_{x_i\in X}\lambda_{jx_i}^{(R)}-\sum_{x'_i\in X'}\lambda_{jx'_i}^{(R)}\right\|=\sum_{j=1}^{d}\sum_{R=0}^{2}\left\|\lambda_{jx_n}^{(R)}-\lambda_{jx'_n}^{(R)}\right\|$. We can show that $\lambda_{jx_n}^{(0)}=\sum_{l=1}^{2}f_{lj}^{(0)}(0)=x_{nj}log2+(1-x_{nj})log2=log2$. Similarity, we can show that $\lambda_{jx_n}^{(0)}=log2$. As a result, $\lambda_{jx_n}^{(0)}=\lambda_{jx'_n}^{(0)}$. Therefore*

$$\Delta=\sum_{j=1}^{d}\sum_{R=0}^{2}\left\|\lambda_{jx_n}^{(R)}-\lambda_{jx'_n}^{(R)}\right\|=\sum_{j=1}^{d}\sum_{R=1}^{2}\left\|\lambda_{jx_n}^{(R)}-\lambda_{jx'_n}^{(R)}\right\|\leqslant\sum_{j=1}^{d}\sum_{R=1}^{2}\left(\left\|\lambda_{jx_n}^{(R)}\right\|+\left\|\lambda_{jx'_n}^{(R)}\right\|\right)\leqslant 2\max_{x}\sum_{j=1}^{d}\sum_{R=1}^{2}\left\|\lambda_{jx}^{(R)}\right\|\leq 2\max_{x}\left[\sum_{j=1}^{d}\left(\frac{1}{2}-x_j\right)\sum_{e=1}^{m}\widetilde{A}h_{xe}+\sum_{j=1}^{d}\left(\frac{1}{8}\sum_{e,q}\widetilde{A}h_{xe}\widetilde{A}h_{xq}\right)\right]\leq 2\left(\frac{1}{2}d\times z+\frac{1}{8}d\times z^2\right)=d\left(z+\frac{1}{4}z^2\right)$$

where h_{xe} is the state of e-th hidden variable derived from the tuple x, i.e., $h=\sigma\left(\widetilde{A}xW^T\right)$.

4.4. Perturbation sigmoid layer

We use the sigmoid function as the activation function for the output layer, focusing on a binomial prediction problem. This layer

contains a prediction matrix in which the probability values $\hat{a}_{ij}$ represents the probability of connecting edges between nodes i and j. Therefore, we choose the cross-entropy loss function given in Eq. (6) to deal with this task. The ϵ-differential privacy of the sigmoid layer can be guaranteed by obtaining a polynomial approximation of the cross-entropy loss function and then perturbing it using the functional mechanism. Eq. (6) can be approximated as

$$\widehat{C}(A_T,\theta)=\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1}^{2}\sum_{R=0}^{2}\frac{f_l^{(R)}(0)}{R!}\left(A\mathfrak{h}_{i(k)}W_{j(k)}\right)^R \tag{15}$$

where $W_{(k)}$ is the weight matrix of the k-th layer and $\mathfrak{h}_{(k)}$ is the hidden variables of the k-th layer input, n is the number of elements in the adjacency matrix.

In the following lemma, we will give the global sensitivity of the topology reconstruction function $\widehat{C}(A_T,\theta)$ over the target topology A_T.

Lemma 2. *Suppose A_T and A_T' be any two neighboring graphs (network). Assuming that topological reconstruction using graph auto-encoder on neighboring datasets A_T and A_T' corresponds to the objective functions $\widehat{C}(A_T,\theta)$ and $\widehat{C}(A_T',\theta)$, respectively, then the global sensitivity of the objective $\widehat{C}$ over the target topology A_T is as:*

$$\Delta_C=n\left(|\mathfrak{h}_{(k)}|+\frac{1}{4}|\mathfrak{h}_{(k)}|^2\right) \tag{16}$$

where $|\mathfrak{h}_{(k)}|$ is the dimension of the hidden variable in $\mathfrak{h}_{(k)}$.

Proof 2. *By applying Lemma 1, we can compute the sensitivity Δ_C of $\widehat{C}(A_T,\theta)$ on the set of graph (network) edges A_T as follows: $\Delta_C\leqslant 2max_a\left[\sum_{j=1}^{n}(\frac{1}{2}-a_j)\sum_{e=1}^{|\mathfrak{h}_{(k)}|}\widetilde{A}\mathfrak{h}_{ae(k)}+\sum_{j=1}^{n}\left(\frac{1}{8}\sum_{e,q}\mathfrak{h}_{ae(k)}\mathfrak{h}_{aq(k)}\right)\right]\leqslant n\left(|\mathfrak{h}_{(k)}|+\frac{1}{4}|\mathfrak{h}_{(k)}|^2\right)$, where $\mathfrak{h}_{ae(k)}$ is the state of the e-th hidden variable at the k-th normalization layer. $|\mathfrak{h}_{(k)}|$ is the number of hidden variables in $\mathfrak{h}_{(k)}$.*

After computing the sensitivity Δ_C, we can derive the perturbed objective function denoted as $\overline{C}(A_T,\theta)$. The back-propagation algorithm is used to train all the parameters in the Differential Privacy deep Graph Auto-Encoder (DP-DGAE), aiming to optimize the function $\overline{C}(A_T,\theta)$ to ensure that the output (graph) of the DP-DGAE satisfies Differential Privacy.

4.5. Approximation error bounds

We give the error due to our approximation approaches $\widetilde{RE}(X,W)$ and $\widehat{C}(A_T,\theta)$ in the following theorem. The error of approximation is influenced by the objective function as well as the dimensionality of the datasets. It can be learned from the following theorem that the average error due to approximation is acceptable.

Theorem 1. *Given the reconstruction functions $\widetilde{RE}(X,W)$ and $\widetilde{C}(A_T,\theta)$ and their approximate forms $\widetilde{RE}(X,W)$ and $\widehat{C}(A_T,\theta)$, the average error of the approximations is given as follows:*

$$|\widetilde{RE}\left(X,\widehat{W}\right)-\widetilde{RE}\left(X,\widetilde{W}\right)|\leqslant\frac{2(e^2-e)}{3(1+e)^3}\times d \tag{17}$$

$$|\widetilde{C}\left(A_T,\widehat{\theta}\right)-\widetilde{C}\left(A_T,\widetilde{\theta}\right)|\leqslant\frac{2(e^2-e)}{3(1+e)^3}\times n \tag{18}$$

where $\widetilde{W}=argmin_W\widetilde{RE}(X,W)$, $\widehat{W}=argmin_W\widehat{RE}(X,W)$, $\widetilde{\theta}=argmin_\theta\widetilde{C}(A_T\theta)$, $\widehat{\theta}=argmin_\theta\widehat{C}(A_T,\theta)$.

Proof 3. *Let $\widetilde{W}=\arg min_W\widetilde{RE}(X,W)$, $\widehat{W}=\arg min_W\widehat{RE}(X,W)$, $U=max_W\left(\widetilde{RE}(X,W)-\widehat{RE}(X,W)\right)$ and $S=min_W\left(\widetilde{RE}(X,W)-\widehat{RE}(X,W)\right)$. We have that:*

$U\geqslant\widetilde{RE}\left(X,\widehat{W}\right)-\widehat{RE}\left(X,\widehat{W}\right)$ and $\forall W^*:S\leqslant\widetilde{RE}(X,W^*)-\widehat{RE}(X,W^*)$. Therefore, we have that:

$\widetilde{RE}\left(X,\widehat{W}\right)-\widetilde{RE}(X,W^*)\leqslant U-S+\left(\widehat{RE}\left(X,\widehat{W}\right)-\widehat{RE}(X,W^*)\right)$.

In addition, $\widehat{RE}\left(X,\widehat{W}\right)-\widehat{RE}(X,W^*)\leqslant 0$, so:

$\widetilde{RE}\left(X,\widehat{W}\right)-\widetilde{RE}(X,W^*)\leqslant U-S$.

If $U\geqslant 0$ and $S\leqslant 0$ then we have:

$|\widetilde{RE}\left(X,\widehat{W}\right)-\widetilde{RE}(X,W^*)|\leqslant U-S$

The error depends on the maximum and minimum value of $\widetilde{RE}(X,W)-\widehat{RE}(X,W)$. To quantify the magnitude of the error, we first rewrite $\widetilde{RE}(X,W)-\widehat{RE}(X,W)$ as: $\widetilde{RE}(X,W)-\widehat{RE}(X,W)=\sum_{j=1}^{d}\left(\sum_{i=1}^{|X|}\sum_{l=1}^{2}\sum_{R=3}^{\infty}\frac{f_{lj}^{(R)}(Z_{lj})}{R!}\left(g_{lj}(x_i,W_j)-z_{lj}\right)^R\right)$.

To derive the minimum and maximum values of the function above, we look into the remainder of Taylor Expansion for each j. Let $z_j\in[z_{lj}-1,z_{lj}+1]$. According to well-known result (Apostol 1967), $\frac{1}{|X|}\left(\widetilde{RE}(X,W_j)-\widehat{RE}(X,W_j)\right)$ must be in the interval:

$$\left[\sum_l\frac{\min_{z_j}f_{lj}^{(3)}(z_j)(z_j-z_{lj})^3}{6},\sum_l\frac{\max_{z_j}f_{lj}^{(3)}(z_j)(z_j-z_{lj})^3}{6}\right].$$

If $\sum_l\frac{\max_{z_j}f_{lj}^{(3)}(z_j)(z_j-z_{lj})^3}{6}\geqslant 0$ and $\sum_l\frac{\min_{z_j}f_{lj}^{(3)}(z_j)(z_j-z_{lj})^3}{6}\leqslant 0$, then we have that:

$$\frac{1}{|X|}\left(\widetilde{RE}(X,W_j)-\widehat{RE}(X,W_j)\right)\leqslant\sum_{j=1}^{d}\sum_l\frac{\max_{z_j}f_{lj}^{(3)}(z_j)(z_j-z_{lj})^3-\min_{z_j}f_{lj}^{(3)}(z_j)(z_j-z_{lj})^3}{6}$$

This analysis applies to the case of graph auto-encoder as follows. First, for the functions $f_{1j}(z_j)=x_{ij}log(1+e^{-z_j})$ and $f_{2j}(z_j)=(1-x_{ij})log(1+e^{z_j})$, we have:

$$f_{1j}^{(3)}(z_j)=\frac{x_{ij}\left(e^{z_j}-e^{2z_j}\right)}{\left(1+e_j^z\right)^3},\ f_{2j}^{(3)}(z_j)=(1-x_{ij})\frac{e^{-z_j}\left(e^{-z_j}-1\right)}{\left(1+e^{-z_j}\right)^3}.$$

It can be verified that:

$min_{z_j}f_{1j}^{(3)}(z_j)=\frac{e-e^2}{(1+e)^3}<0$, $max_{z_j}f_{1j}^{(3)}(z_j)=\frac{e^2-e}{(1+e)^3}>0$, $min_{z_j}f_{2j}^{(3)}(z_j)=\frac{e-e^2}{(1+e)^3}<0$, $max_{z_j}f_{2j}^{(3)}(z_j)=\frac{e^2-e}{(1+e)^3}>0$.

Thus, the average error of the approximation is at most

$$|\widetilde{RE}\left(X,\widehat{W}\right)-\widetilde{RE}\left(X,\widetilde{W}\right)|\leqslant\left[2\times\left(\frac{e^2-e}{(1+e)^3}-\frac{e-e^2}{(1+e)^3}\right)\right]\times\frac{d}{6}=\frac{2(e^2-e)}{3(1+e)^3}\times d.$$

$$|\widetilde{C}\left(A_T,\widehat{\theta}\right)-\widetilde{C}\left(A_T,\widetilde{\theta}\right)|\leqslant\left[2\times\left(\frac{e^2-e}{(1+e)^3}-\frac{e-e^2}{(1+e)^3}\right)\right]\times\frac{n}{6}=\frac{2(e^2-e)}{3(1+e)^3}\times n.$$

Theorem 1 shows that when reconstructing edges using node features and topological information, the error in Eq. (18) is almost negligible compared to the number of edges in the graph. This provides a guarantee that our model can be applied on large graph datasets. And we inject Laplace noise into the coefficients of the quadratic polynomial, which may lead to unbounded objective function after approximation, we will discuss the method of avoiding unbounded noisy objective function.

4.6. Avoiding unbounded noisy objective functions

We use the Functional mechanism to add Laplace noise to the coefficients of the objective function, which may lead to an

unbounded objective function. We constructed a convex objective function by truncating the terms of order greater than 2 in the Taylor series during the previous polynomial approximation, but the addition of noise may make the current objective function without any optimal solution. We change the quadratic polynomial into the matrix form $\widehat{C}(A_T,\theta)=\theta^T M\theta+\alpha\theta+\beta$ and we denote the form after adding noise to the coefficients as $\overline{C}(A_T,\theta)=\theta^T M^*\theta+\alpha^*\theta+\beta^*$. We can then ensure that $\widehat{C}(A_T,\theta)$ remains bounded after adding noise by making M symmetric and positive definite. [13].

We can achieve the symmetry of M^* by adding noise to the lower triangular part of the matrix and copying the lower triangular part to the upper triangular part of the matrix. For the positive definite let $\overline{C}(A_T,\theta)=\omega^T M^*\theta+\alpha^*\theta+\beta^*$ be the symmetrized noisy objective function. In addition, $\overline{C}(\omega)$ is unbounded at least one eigenvalue of M^* is not positive [32]. Therefore we need to eliminate the non-positive eigenvalues in M^*.

Let $V^T\Lambda V$ be the eigen-decomposition of M^*. Accordingly,

$$\overline{C}(A_T,\theta)=\theta^T\left(V^T\Lambda V\right)\theta+\alpha^*\left(V^T V\right)\theta+\beta^* \tag{19}$$

If the i-th diagonal element in Λ is non-positive, then we will remove this element from Λ. Also, to ensure that the matrix dimension match, we will remove the element of the i-th row in V, which we denoted as V'. The noisy objective function then becomes

$$\overline{C}(A_T,\theta)=\theta^T\left(V'^T\Lambda' V'\right)\theta+\alpha^*\left(V'^T V'\right)\theta+\beta^* \tag{20}$$

Adding Laplace noise to the coefficients results in non-positive elements in Λ. However, removing these non-positive elements from Λ does not have a significant impact on the learning process and the ability of the model to extract useful information. [13] Thus, the objective function in Eq. (20), obtained by removing the non-positive elements from Λ, avoids the problem of unboundedness while still leading to accurate model parameters.

4.7. Multi-task learning

In this model, we learn both tasks $C(A_T,\theta)$ and $\overline{C}(A_T,\theta)$. By optimizing the former we can make the output of our model have a similar topology to the original graph while having better performance in link prediction, and optimizing the latter ensures that the output of our model satisfies the ϵ-differential privacy. However, these two objective functions are mutually constrained and we cannot use the traditional approach of using a linear weighted combination to optimize both objective functions at the same time. Sener et al. [33] cast multi-task learning as multi-objective optimization, with the overall objective of finding a Pareto optimal solution. Because the method proposed by Sener et al. [33] does not consider the variance of the gradient distribution, but the differential privacy will maintain the mean of the original distribution when performing privacy protection, which will have a greater impact on the variance of the original distribution. Therefore, the problem that needs to be solved in this solution, so we propose a gradient distribution magnitude regularization coefficient λ for differential privacy.

Theorem 2. *Given two empirical loss $C(A_T,\theta)$ and $\overline{C}(A_T,\theta)$. We can derive the gradients $\nabla_\theta C$ and $\nabla_\theta\overline{C}$ of the shared parameters θ corresponding to the empirical loss of task 1 and task 2. We first calculate the coefficient λ, which is used to regularize the two gradient distributions to the same order of magnitude:*

$$\lambda=\sigma(\nabla_\theta C)\ /\sigma\left(\nabla_\theta\overline{C}\right) \tag{21}$$

where σ represents the standard deviation of the distribution.

Then we use the λ to regularize the gradient distribution $\nabla_\theta C=\nabla_\theta C/\lambda$ the optimization problem can be defined as $min_{\alpha\in[0,1]}||\alpha\nabla_\theta C(\theta)+(1-\alpha)\nabla_\theta\overline{C}(\theta)||_2^2$, which is a one dimensional quadratic function of α with an analytical solution:

$$\hat{\alpha}=\left[\frac{\left(\nabla_\theta\overline{C}(\theta)-\nabla_\theta C(\theta)\right)^T\nabla_\theta\overline{C}(\theta)}{||\nabla_\theta C(\theta)-\nabla_\theta\overline{C}(\theta)||_2^2}\right]_{+\frac{1}{T}}\ /\ \lambda \tag{22}$$

where $[\cdot]_{+\frac{1}{T}}$ represents clipping to $[0,1]$ as $[a]_{+\frac{1}{T}}=max(min(a,1),0),\nabla_\theta$ denotes the gradient of the shared parameters θ.

After computing the α, we can use the gradient descent on the shared parameters $\theta=\theta-\eta\left(\alpha\nabla_\theta C(A_T,\theta)+(1-\alpha)\nabla_\theta\overline{C}(A_T,\theta)\right)$ to optimize both objective functions.

5. Experiment

In this section, we will measure the effectiveness of the DP-DGAE model in real social networks. We will start by introducing the datasets used in our experiments. Then we will detail the experimental settings of the model. Finally, we will present the usability of DP-DGAE under different privacy budgets ϵ, as well as the comparison with the state of art models in terms of link prediction accuracy.

5.1. Datasets

We use three public datasets for experiments and the statistics of the datasets are shown in Table 1. We choose the same dataset splits as in kipf et al. [31] and zhang et al. [34], randomly sample 5% of the total number of edges as the validation and 10% for testing.

5.2. Experimental settings

In our model, we use three-layer GCNs as the encoder. Kipf et al. [31] experimentally concluded that adding input features to the model can significantly improve the predictive performance of the model across datasets, while they noted that using the inner product decoder may push the embedding away from the zero-center. So we use four-layer GCNs as the decoder for link prediction to get a more powerful decoding performance. The introduction of differential privacy in deep learning alleviates the problem of overfitting [35]. For Differential Privacy related hyper-parameters, we use Eq. (14) and Eq. (16) to calculate the global sensitivity of the objective function, while we observe the availability of the released graph for different privacy budgets by making ϵ vary from 0.1 to 10. We use the initial learning rate of 0.1 and decay every 2000 epochs to half the original. In addition, the weight decay $\in\{1e-8,1e-10,1e-9\}$ and clipping hyper-parameter $\in[0.8,12]$.

All experiments are done with one GeForce RTX 3090 GPU and a 24-core 2.4 GHz CPU. To measure the complexity of our proposed method, analogous to the operation of Chen et al. [36] we use our method DP-DGAE to compare with the graph generation model GAE proposed by Kipf et al. [31] in terms of training speed as well

Table 1
Dataset statistics.

Dataset	Nodes	edges	Attributes
Cora	2708	5278	1443
Facebook	1034	26794	576
IMDB	17577	287075	19

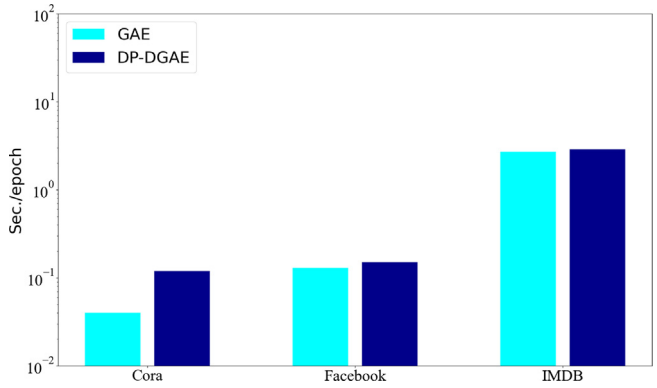


Fig. 4. Per-epoch training time in seconds for DP-DGAE and GAE on different datasets.

Table 2
Memory consumption of DP-DGAE and GAE during training.

	Cora	Facebook	IMDB
GAE	1.68Gib	1.46Gib	14.56Gib
DP-DGAE	1.96Gib	1.51Gib	22.63Gib

as memory consumption. See Fig. 4, the bar heights indicate the per-epoch training time, in the log scale. As detailed in Fig. 4, it can be seen that our scheme is mostly in the same order of magnitude in terms of training speed compared to non-private graph generation schemes, thus, it can be shown that our method can maintain a better efficiency in terms of privacy graph generation. Although space complexity is not our primary concern, we still record the memory consumption of DP-DGAE and GAE during training, as shown in Table 2. Compared to GAE, DP-DGAE has a limited increase in memory consumption in exchange for privacy guarantees for the GAE at an acceptable cost.

5.3. Preserving global structures

We compare the ability of DP-DGAE to retain the global network structure under different privacy budgets. The trained model is used to generate a new graph and the differences between the generated graph and the original graph are observed in terms of the statistical values of the graph structure.

In Table 3, we observe the variation of the structural usability metrics AVD(average degree), ACC(average cluster), and APL(average shortest path) under different privacy budgets. We can see that with the reduction of the privacy budget, most of the metrics are not much different from the original values. It can be seen that adding differential privacy to the output layer of the graph neural network for privacy graph generation can better preserve the global availability of the graph. We add noise to the coefficients of the loss function, which can be approximately interpreted as adding noise to the input of the model in the optimization process. The Laplace noise we use has the property of zero mean, and the gap between the structural usability metrics of the privacy graph and the original graph may increase as the privacy budget decreases, but the mean of the gap will stabilize as the number of experiments increases. In Table 4, we derived the overlap between the anonymized graph and the original graph in terms of the EC(the number of original edges), Top-10(nodes in the top 10% of degree values), Top-1(nodes in the top 1% of degree values) under different privacy budgets. In Table 4, for the metrics EC and Top-10 and Top-1, it can be observed that as the privacy budget decreases, the overlap percentage shows a downward trend. The reduction of the overlap percentage of original edges and the influential nodes is in line with our expectation of increasing privacy requirements. It can be seen that the differential privacy has less change on the global graph structure usability, which makes the structure usability of the anonymized graph well preserved, and at the same time, the perturbation of individual nodes is larger, which makes it more difficult for an attacker to recover the target nodes or the relationship between nodes in the anonymized graph using the already acquired background knowledge.

Beyond the single value statistics, we also compare the generated graph regarding degree distribution. For degree distribution as shown in Figs. 5–7, we sort the nodes according to their degrees in the original graph from largest to smallest, and then plot two degree distribution curves, all-node and origin, based on the degrees of the nodes in the original graph as well as in the divided training set. The yellow dash is the degree distribution in our anonymized graph according to the same node ids as in the pre-anonymized graph. It can be observed that the overall trend of the degree distribution of the graph after anonymization is the same as before anonymization, but due to the introduction of differential privacy, a random positive and negative perturbation is added to the degree values of the nodes, and at the same time, this perturbation shows a tendency to increase as the privacy budget decreases.

Table 3
Performance evaluation of a set of important graph global structure statistical metrics. The Original rows include the values of original networks, while the rest rows are the value corresponding to the network generated by the model under different privacy budgets.

Privacy	Facebook Networks			Cora Networks			IMDB Networks		
budgets	AVD	ACC	APL	AVD	ACC	APL	AVD	ACC	APL
original	44.25	0.43	3.03	5.63	0.18	6.76	28.07	0.28	5.07
$\epsilon = 10$	48.81	0.50	2.71	5.79	0.11	4.51	30.65	0.30	5.14
$\epsilon = 1$	49.11	0.53	2.77	5.71	0.08	4.58	30.61	0.28	5.05
$\epsilon = 0.1$	48.97	0.54	2.81	5.77	0.05	4.51	30.24	0.20	5.06

Table 4
Percentage of original edges retention and overlap of influence nodes.

privacy	Facebook Networks			Cora Networks			IMDB Networks		
budgets	EC	Top-10	Top-1	EC	Top-10	Top-1	EC	Top-10	Top-1
$\epsilon = 10$	67.55	89.75	75.00	48.39	57.46	83.26	37.05	77.18	78.82
$\epsilon = 1$	64.24	87.00	72.50	43.38	53.93	78.91	30.76	58.59	70.59
$\epsilon = 0.1$	60.64	84.33	70.00	27.97	38.76	65.87	16.97	34.41	64.12

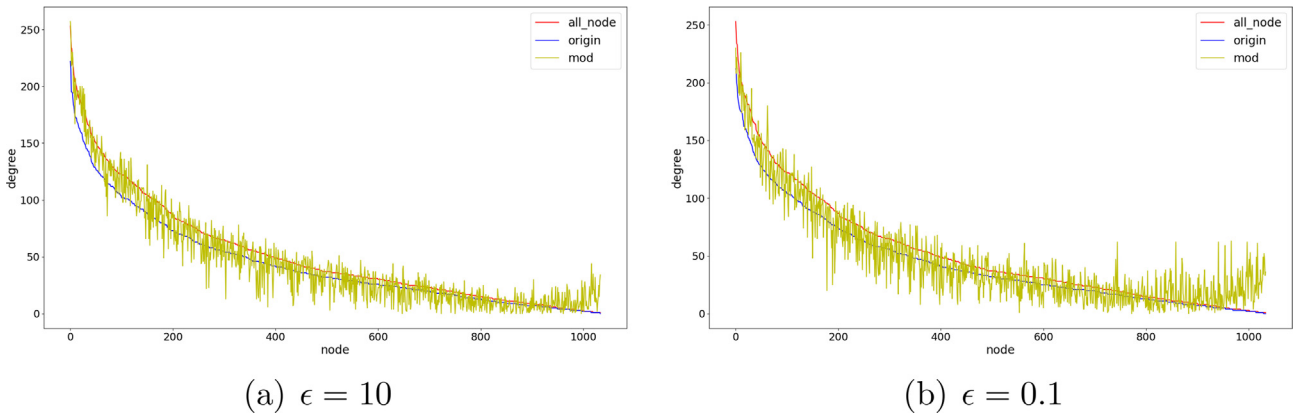


Fig. 5. Facebook degree change.

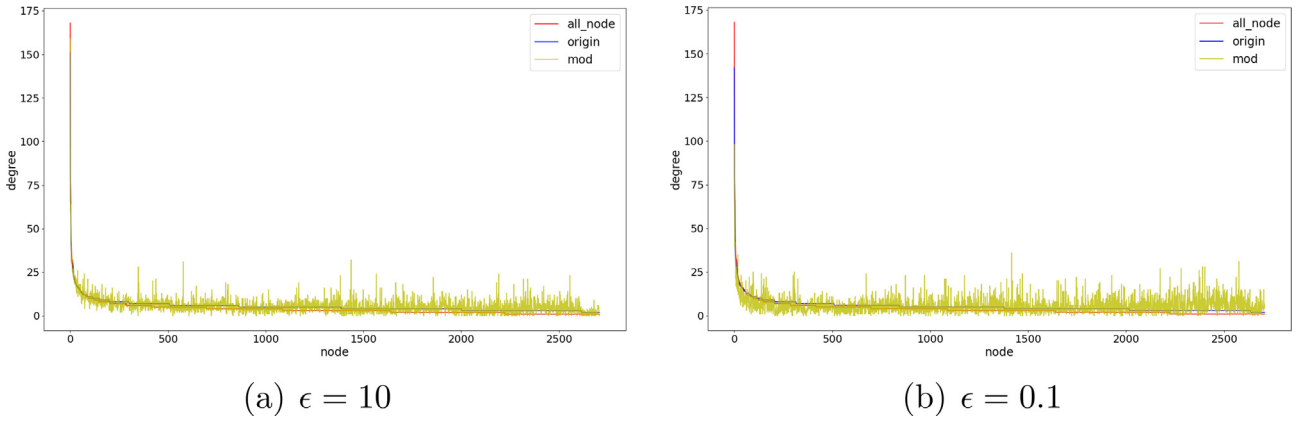


Fig. 6. Cora degree change.

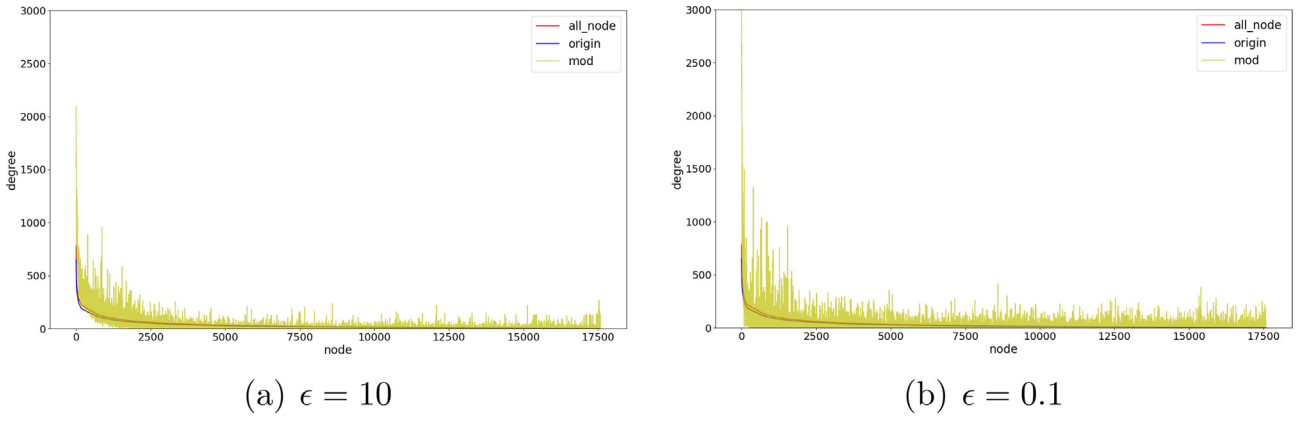


Fig. 7. IMDB degree change.

As shown in Fig. 8, we compared the cosine similarity of the degree distribution before and after anonymity with DPGGAN and related methods [27]. The higher the better. At the same level of privacy budget, the similarity between the degree distribution of the anonymous graph published by DP-DGAE and the degree distribution of the original graph is the highest. It can be seen that DP-DGAE has better performance in statistical characteristics such as degree distribution, which shows that DP-DGAE has a good ability to preserve the graph structure.

The above information shows that it is difficult for the attacker to recover the target node through the anonymized graph if the attacker has the relationship information of the target node with other nodes, the final output of the model can protect the relationship privacy of the user due to the introduction of ϵ-differential

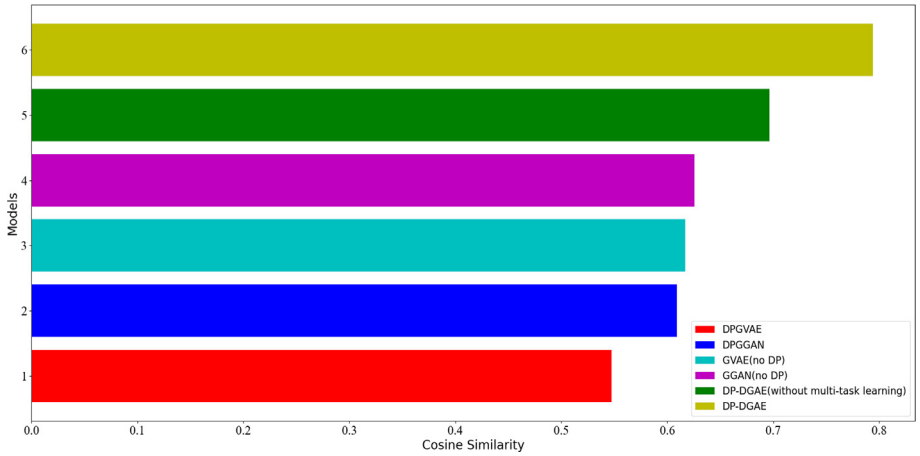


Fig. 8. The cosine similarity of degree distribution on networks generated by DP-DGAE with the same privacy budget $\epsilon = 1$ compared with various models proposed in DPGGAN. Higer similarity means better global data utility.

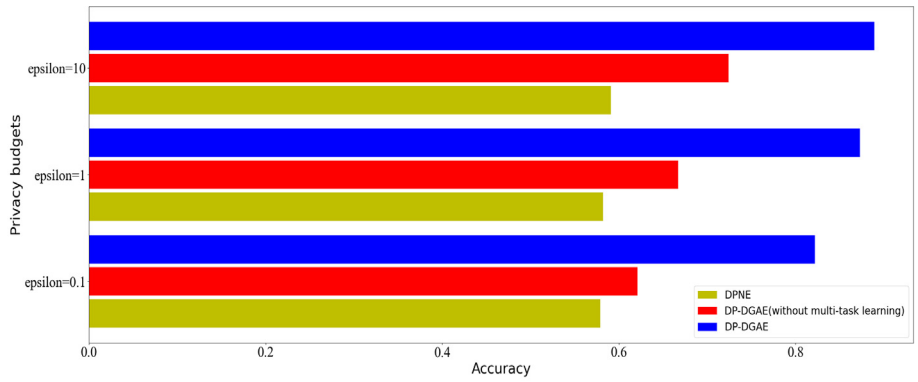
privacy, and at the same time, the overlap rate of the edges in the anonymized graph with those in the original graph has been reduced by a large percentage ($32.35\% - 35.44\%, 51.60\% - 72.02\%$ and $62.94\% - 83.02\%$ overlap rate drops on Facebook, Cora and IMDB, respectively), and it can also be seen that the probability of re-identifying the target node in the anonymized graph by the attacker using the relationship information of the target user is greatly reduced.

5.4. Link prediction performance

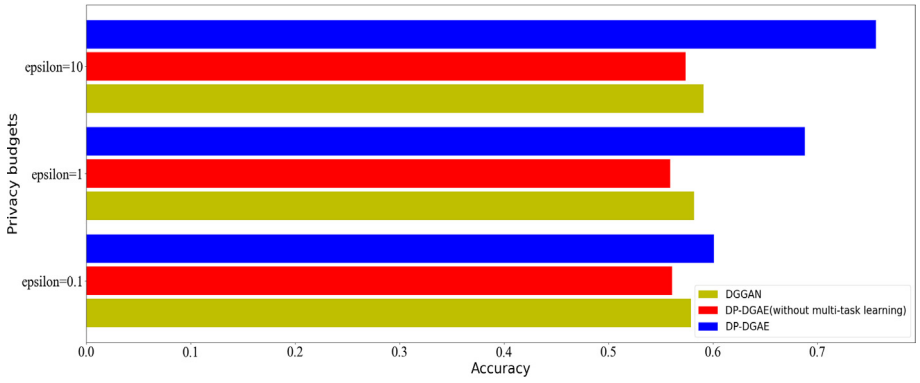
In order to show that the output of DP-DGAE maintains privacy while maintaining the usability of further learning under the machine learning model, we use the probability graph obtained from the DP-DGAE model for link prediction. Results for the link prediction task in Facebook, Cora, and IMDB networks are summarized in Table 5. We report AUC(area under the ROC curve) and AP(average precision) scores of the model in the test set. Numbers show mean results for 10 runs with random initializations on fixed dataset splits. We use the Probability graph obtained from the DP-DGAEs model for link prediction, and we can see that after adding differential privacy, the predictive performance decreases accordingly compared to that without adding differential privacy ($3.99\% - 8.60\%, 9.98\% - 17.17\%$ and $5.27\% - 37.64\%$ predictive performance drops on Facebook, Cora, and IMDB, respectively). This means that the probabilistic graphs published by our model preserve the privacy of the relational information as well as the structural information of the users in the original graph, while retaining their ability to be further learned under the deep learning model to obtain valid link prediction results. In Fig. 9, for the link prediction accuracy, we compare it with DPNE [37] on the Cora network and DPGGAN on the IMDB network. When DP-DGAE does not use multi-task learning (only $\overline{C}(A_T, \theta)$ is used as the objective function), its link prediction accuracy is similar to the DPNE and DPGGAN. When we use the results obtained by DP-DGAE (using multi-task learning) for link prediction, we can find that compared with other schemes, we have obtained the best results in link prediction accuracy. Since the introduction of differential privacy changes the connection information of the original graph, it may make the prediction probability of the model for negative sampling greater than that for positive sampling, which will have an impact on the link prediction reliability of the model. To measure the ability of our model to rank positive and negative samples during link prediction, in Fig. 10, we compare the *area under the ROC curve* (AUC) with DPLP [38] on the Facebook network at the maximum level of privacy interference $\epsilon = 0.1$. The results show that DP-DGAE significantly outperforms all variants of the comparison scheme in terms of link prediction performance after using multi-task learning. The above shows that while DP-DGAE protects privacy, it also retains its ability to be applied to machine learning models for further learning.

Table 5
Accuracy of links prediction.

privacy	Facebook Networks		Cora Networks		IMDB Networks	
budgets	AUC	AP	AUC	AP	AUC	AP
original	98.64	98.17	88.35	89.37	92.47	93.99
$\epsilon = 10$	94.64	94.43	79.37	79.39	87.64	88.71
$\epsilon = 1$	93.00	93.15	77.04	77.25	73.37	73.98
$\epsilon = 0.1$	89.57	89.81	72.14	72.20	57.90	56.34



(a) Cora network



(b) IMDB network

Fig. 9. Accuracy of links predicted on networks generated by DP-DGAE and DP-DGAE(without multi-task learning) with varying ϵ values compared with DPNE and DPGGAN. Higher accuracy means better link prediction capability.

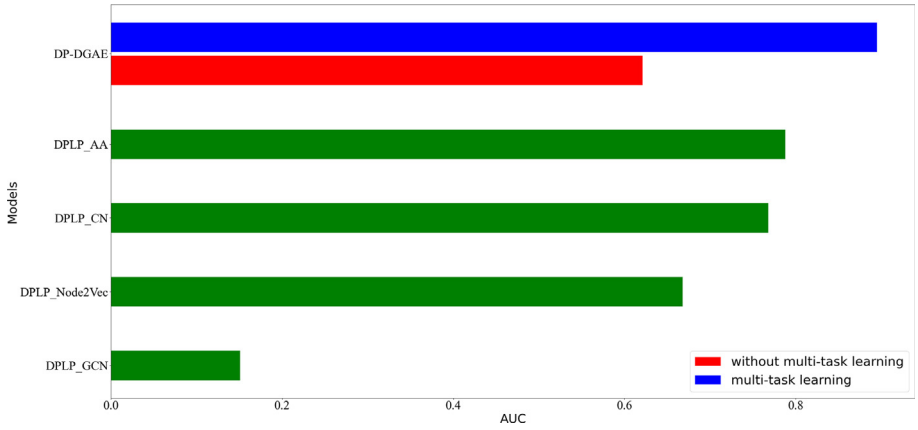


Fig. 10. Performance(AUC) of links predicted on networks generated by DP-DGAE and DP-DGAE(without multi-task learning) with same ϵ value compared with several variants of DPLP. Higher AUC means better link prediction capability.

6. Conclusion

The graph generation model will obtain a reconstruction graph based on the features of the nodes in the original graph and the structural information. Due to the prominence of privacy preservation issues in deep learning, many privacy preservation techniques in deep learning have emerged. We accomplish the publication of anonymous graphs by transfering the traditional graph data privacy preservation problem into a privacy preservation problem in a deep graph generation model using the existing theory of privacy preservation in deep learning, and by using multi-task learning to address a common problem in traditional graph data privacy preservation: the balance between usability and anonymity. Comprehensive experiments show that the privacy graphs published by our model protect the relational privacy of users while retaining the ability to obtain valid link prediction results by further analysis under the deep learning model.

CRediT authorship contribution statement

Xiaolin Li: Conceptualization, Methodology, Software. **Li Xu:** Supervision, Writing - review & editing. **Hongyan Zhang:** Formal analysis, Investigation. **Qikui Xu:** Data curation, Validation.

Data availability

Data will be made available on request.

Declaration of Competing Interest

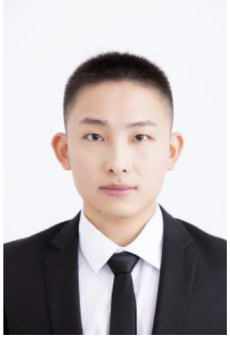
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank the National Natural Science Foundation of China (NOs. U1905211, 61771140, 62171132).

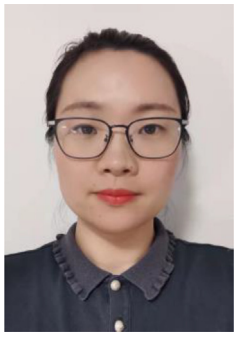
References

[1] L. Hu, T. Yang, L. Zhang, W. Zhong, D. Tang, C. Shi, N. Duan, M. Zhou, Compare to the knowledge: Graph neural fake news detection with external knowledge, in: ACL/IJCNLP, 2021. 10.18653/v1/2021.acl-long.62.
[2] X. Zhang, L. Xu, Z. Xu, Influence maximization based on network motifs in mobile social networks, IEEE Trans. Network Sci. Eng. 9 (2022) 2353–2363, https://doi.org/10.1109/TNSE.2022.3163203.
[3] M. Weber, J. Chen, T. Suzumura, A. Pareja, T. Ma, H. Kanezashi, T. Kaler, C.E. Leiserson, T.B. Schardl, Scalable graph learning for anti-money laundering: A first look, CoRR abs/1812.00076 (2018). arXiv:1812.00076.
[4] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, ICLR, in, 2018.
[5] Z. Cai, Z. He, X. Guan, Y. Li, Collective data-sanitization for preventing sensitive information inference attacks in social networks, TDSC (2018), https://doi.org/10.1109/TDSC.2016.2613521.
[6] S. Ji, W. Li, M. Srivatsa, J.S. He, R.A. Beyah, General graph data de-anonymization: From mobility traces to social networks, Trans. Inf. Syst. Secur. (2016), https://doi.org/10.1145/2894760.
[7] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, TCC, in, 2006.
[8] L.Y.B.Y., H. G, Deep learning, Nature 521 (2015) 436–444.
[9] M. Zhang, Y. Chen, Link prediction based on graph neural networks, NIPS, in, 2018.
[10] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, ICLR (2017).
[11] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, J. Pei, AM-GCN: adaptive multi-channel graph convolutional networks, KDD (2020), https://doi.org/10.1145/3394486.3403177.
[12] M. Zhang, Z. Cui, M. Neumann, Y. Chen, An end-to-end deep learning architecture for graph classification, AAAI, in, 2018.
[13] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, M. Winslett, Functional mechanism: Regression analysis under differential privacy, VLDB 10.14778/2350229.2350253 (2012).
[14] W.-Y. Day, N. Li, M. Lyu, Publishing graph degree distribution with node differential privacy, SIGMOD (2016), https://doi.org/10.1145/2882903.2926745.
[15] S. Zhang, W. Ni, N. Fu, Community preserved social graph publishing with node differential privacy, ICDM (2020), https://doi.org/10.1109/ICDM50108.2020.00184.
[16] T. Cunningham, G. Cormode, H. Ferhatosmanoglu, D. Srivastava, Real-world trajectory sharing with local differential privacy, VLDB (2021).
[17] F. McSherry, I. Mironov, Differentially private recommender systems: Building privacy into the netflix prize contenders, KDD (2009), https://doi.org/10.1145/1557019.1557090.
[18] Y. Wang, J. Lam, H. Lin, Differentially private average consensus with general directed graphs, Neurocomputing 458 (2021) 87–98, https://doi.org/10.1016/j.neucom.2021.06.016.
[19] R. Shokri, V. Shmatikov, Privacy-preserving deep learning, CCS (2015), https://doi.org/10.1145/2810103.2813687.
[20] M. Abadi, A. Chu, I.J. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, CCS (2016), https://doi.org/10.1145/2976749.2978318.
[21] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks?, ICLR, in, 2019
[22] W.L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, NIPS, in, 2017.
[23] R. Ying, R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, KDD (2018), https://doi.org/10.1145/3219819.3219890.
[24] L. Feng, Y. Cai, E. Wei, J. Li, Graph neural networks with global noise filtering for session-based recommendation, Neurocomputing 472 (2022) 113–123, https://doi.org/10.1016/j.neucom.2021.11.068.
[25] G. Ateniese, L.V. Mancini, A. Spognardi, A. Villani, D. Vitali, G. Felici, Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers, IJSN (2015), https://doi.org/10.1504/IJSN.2015.071829.
[26] M. Balog, I. Tolstikhin, B. Schölkopf, Differentially private database release via kernel mean embeddings, in: ICML, 2018.
[27] C. Yang, H. Wang, K. Zhang, L. Chen, L. Sun, Secure deep graph generation with link differential privacy, in: IJCAI, 2021. 10.24963/ijcai.2021/450, main Track.
[28] N. Phan, Y. Wang, X. Wu, D. Dou, Differential privacy preservation for deep auto-encoders: an application of human behavior prediction, AAAI (2016).
[29] W. Rudin, Principles of Mathematical Analysis, Third Edition., McGraw-Hill, 1976.
[30] G. Arfken, In Mathematical Methods for Physicists, Third Edition., Elsevier, 1985.
[31] T.N. Kipf, M. Welling, Variational graph auto-encoders, CoRR abs/1611.07308 (2016). arXiv:1611.07308.
[32] G. Strang, Introduction to Linear Algebra, Fifth Edition., Wellesley-Cambridge Press, 2016.
[33] O. Sener, V. Koltun, Multi-task learning as multi-objective optimization, NIPS (2018).
[34] M. Zhang, Y. Chen, Link prediction based on graph neural networks, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems, vol. 31, Curran Associates Inc, 2018.
[35] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, A. Roth, The reusable holdout: Preserving validity in adaptive data analysis, Science 349 (2015) 636–638, https://doi.org/10.1126/science.aaa9375, arXiv:https://www.science.org/doi/pdf/10.1126/science.aaa9375.
[36] J. Chen, T. Ma, C. Xiao, Fastgcn: Fast learning with graph convolutional networks via importance sampling, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 – May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018.https://openreview.net/forum?id=rytstxWAW.
[37] D. Xu, S. Yuan, X. Wu, H. Phan, DPNE: differentially private network embedding, PAKDD (2018), https://doi.org/10.1007/978-3-319-93037-4_19.
[38] A. De, S. Chakrabarti, Differentially private link prediction with protected connections, in: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2–9, 2021, AAAI Press, 2021, pp. 63–71.https://ojs.aaai.org/index.php/AAAI/article/view/16078.



Xiaolin Li received the B.S. degree from Shandong agricultural University in 2016. He is currently a M. S. candidate at Fujian Normal University, China. His research interests include cyber security, wireless networks and communication, deep learning.

Li Xu received the Ph.D. degree from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2004. He is currently a Professor and Doctoral Supervisor with the College of Computer and Cyber Security, Fujian Normal University, Fuzhou, China. He is also the Dean of the College of Computer and Cyber Security, Fujian Normal University. His research interests include network and information security, wireless networks and communication, intelligent information in communication networks. He has authored or coauthored more than 150 papers in international journals and conferences, including IEEE Transactions on Computer, IEEE Transactions on Reliability, IEEE Transactions on Parallel and Distributed Systems, Information Science.



Hongyan Zhang received the B.S. degree from Shandong University of Science and Technology in 2003, and the M.S. degree from Fujian Normal University in 2008. She is currently a Ph.D. candidate at Fujian Normal University, China. Her research interests include network security and wireless networks and communication.

QiKui Xu graduated from the University of Sydney, Australia in 2019 with a double bachelor's degree in Computer Science and biochemistry. He is currently a graduate student at Fudan University and West Lake University, and his research interests include complex network modeling, bioinformatics, and biochemistry.