Graph Embedding Matrix Sharing With Differential Privacy

SEN ZHANG AND WEIWEI NI
School of Computer Science and Engineering, Southeast University, Nanjing 211189, China
Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, Nanjing 211189, China
Corresponding author: Weiwei Ni (wni@seu.edu.cn)
This work was supported by the National Natural Science Foundation of China under Grant 61772131.

ABSTRACT Graph embedding maps a graph into low-dimensional vectors, i.e., embedding matrix, while preserving the graph structure, solving the high computation and space cost for graph analysis. Matrix factorization (MF) is an effective means to achieve graph embedding since maintaining the utility of the graph structure. The personalized graph structure features implied in the embedding matrix can identify the individual, which potentially breaches individual sensitive information in the original graph. Currently, protecting individual privacy without compromising the utility is the key to sharing the embedding matrix. Differential privacy is a gold standard for publishing sensitive information while protecting privacy. The existing methods on differentially private MF, however, cannot be directly incorporated onto MF-based graph embedding as they undergo either high global sensitivity or iterative noise error accumulation, potentially rendering poor utility of MF-based graph embedding. To address the deficiency, this study proposes PPGD, a differentially private perturbed gradient descent method for MF-based graph embedding matrix sharing. Specifically, a Lipschitz condition on the objective function of the MF and a gradient clipping strategy are devised for bounding global sensitivity. Along the way, a scalable solution to global sensitivity that is independent on the original dataset is proposed. Further, a composite noise added means in the gradient descent is designed to guarantee privacy while enhancing the utility. The theoretical analysis shows that PPGD can generate processed embedding matrix with the utility maximization while achieving ($\varepsilon, \delta$)-differential privacy. The experimental evaluations confirm the effectiveness and efficiency of PPGD.

INDEX TERMS Differential privacy, Matrix factorization, Embedding matrix, Gradient descent.

I. INTRODUCTION Nowadays graph embedding, aiming at mapping a graph into low-dimensional vectors, i.e., embedding matrix while preserving the graph structural properties, solves the high computation and space cost for graph analysis tasks. Matrix factorization (MF) is playing a critical role in achieving graph embedding, due to its ability to maintain the utility of the graph structure. Publishing embedding matrix enables a wide spectrum of graph analysis tasks such as graph reconstruction and link prediction, and further presents tremendous opportunities for collaboration (see Fig. 2) between data holders and data analysts (e.g., the machine learning community). Yet, improper publication can jeopardize individual privacy. Specifically, most of graph embedding methods use proximity-based node similarities to preserve the structural properties of the graph to drive embedding. Nevertheless, the personalized graph structure features implied in embedding matrix can identify the individual, which may lead to the leak of individual sensitive information in original graphs.

At present, protecting individual privacy while maintaining the utility of the graph structure has become a critical issue when one shares embedding matrix. Differential privacy [1] is widely-adopted to cope with publishing sensitive information while protecting privacy, since it provides strong and
provable guarantees of privacy against attackers with prior knowledge. Existing methods focusing on differentially private MF, however, cannot be directly incorporated onto MF-based graph embedding, and the limitation is two-fold. First, due to the high correlation of graph data, existing methods [2]–[4], if used for MF-based graph embedding, suffer from high global sensitivity. In particular, when the input dataset contains a large number of related data, existing methods require adding a prohibitive amount of noise, which renders the published embedding matrix next to useless, and cannot well maintain the utility of the graph structure. Second, according to the sequential composition theory of differential privacy, iterative computations potentially yield increasing error accumulation that leads to poor utility of MF-based graph embedding. For instance, Hua et al. [2] and Liu et al. [3] are two pioneering works that implement differentially private MF with Stochastic Gradient Descent (SGD) to alleviate accumulation noise. Unfortunately, the two methods, as well as the subsequent line of work [5], [6] cannot well balance privacy and utility. Further, inspired by [2], Xu et al. [7] propose MF-based graph embedding under differential privacy, which is the most related to our study, yet suffers from poor scalability due to complex sensitivity calculations and only protects one of two sub-matrices obtained by MF.

To tackle the deficiency of existing methods, this study puts forth PPGD, a differentially private perturbed gradient descent method for guaranteeing privacy throughout MF-based graph embedding. Particularly, a Lipschitz condition on the objective function of MF and a gradient clipping strategy are devised to bound lower global sensitivity. On this basis, a generic solution to global sensitivity that is independent on original dataset is presented. Further, a composite noise added means is designed for meeting differential privacy, which inherently drives PPGD to escape from local minima to prompt good utility guarantees. Theoretical analysis is provided to guarantee that PPGD can maximize the utility of the graph structure while achieving $(\epsilon, \delta)$-differential privacy.

Empirical experiments comprehensively evaluate the efficiency and effectiveness of PPGD on several large graphs in two graph analysis tasks.

The remainder of our work is organized as follows. In the next section, we provide a literature review of related work on graph data release and matrix factorization. The problem statement and differential privacy are introduced in Section III. Section IV describes the proposed framework with differential privacy. The detailed experimental results are reported in Section V, followed by conclusion in Section VI.

II. RELATED WORK

In this section, we review prior works on differential privacy mechanisms applied to graph data publication and matrix factorization. In addition, we briefly introduce MF-based graph embedding method.

A. DIFFERENTIAL PRIVACY IN GRAPH DATA PUBLISHING

In recent years, releasing private graph data has adopted two types of approaches. First is to release the whole graphs, yet guarantee that the output graph is an anonymous version of the original graph. Second is to release some intermediate and sanitized forms of the original graph. Towards these ends, the target of approaches is to achieve edge privacy or node privacy. In this study, we propose a differentially private perturbed gradient descent that preserves edge-privacy and publishes the embedding matrix of the input graph.

Most previous works [8]–[13] focus on edge privacy and publish processed form of the original graphs, which can be used to extract crucial statistics about the original graph. For instance, based on the $dk$-graph model, Sala et al. [9] introduce a differentially private scheme to share meaningful graph datasets. Wang and Wu [10] propose using $dk$-series to summarize a graph into a distribution of degree correlations. Blum et al. [11] release the covariance matrix of the original data perturbed by random noise. While these studies cope with differentially private publishing of social graph data,
none of them address the utility of preserving graph embedding matrix, the central theme of this study.

**B. DIFFERENTIAL PRIVACY IN MATRIX FACTORIZATION**

MF is playing a critical role in graph embedding, which has received increasing research attention. Recently, plenty of differentially private MF methods have been proposed. In this study, we mainly focus on differentially private MF methods using gradient descent as a base optimization technique, and discuss the differences between our study and existing work.

Hua et al. [2] and Liu et al. [3] perform two pioneering works to deal with MF. More concretely, Hua et al. utilize objective perturbation technique derived from [14] to ensure that low-dimensional data to be released obeys differential privacy. Essentially, Liu et al. sample two differentially private matrices by utilizing the exponential mechanism. Zhang et al. [5] achieve a differential privacy MF via employing objective perturbation and working together with k-coRating. Jain et al. [6] use the Frank-Wolfe method [15] with differential privacy as the building block of MF to tackle matrix completion. Recently, motivated by [2], Xu et al. [7] present differentially private MF-based graph embedding via objective function perturbation, which is the most related to our study. However, for bounding global sensitivity of the target non-private function, they suffer from prohibitively complex analytic calculations, which results in poor scalability. In addition, they can only protect one of two sub-matrices obtained by MF, which potentially increases the risk of privacy breaches. Further, [2], [3], [5], [7] use SGD as a base optimization technique, but [16] indicates that standard SGD used for non-convex MF may get stuck at local minima, which inherently limits their utility.

However, ours presented in section IV can cope with the above shortcomings. In Table 1, we extend our qualitative comparison to some other methods that utilize similar architectures.

---

**C. GRAPH EMBEDDING METHOD**

Generally, graph embedding aims to represent a graph in a low-dimensional space and effectively preserve as much graph structural property information as possible. Commonly used graph embedding methods can be divided into five categories [17]: matrix factorization, deep learning, edge reconstruction, graph kernel, and generative model. In this study, we mainly focus on MF-based graph embedding with the following objective function, as described in [18].

\[
\min f(U) = \frac{1}{2} \| M - UU^T \|_F^2
\]

where \( M \in \mathbb{R}^{N \times N} \) stands for node similarity matrix that preserves the graph structural properties, which can be obtained by a variety of measures such as Katz Index [19] and rooted pagerank [20]. \( U, U' \in \mathbb{R}^{N \times k}, N \) is the number of the nodes in the graph, and \( k(k \ll N) \) is the embedding dimension. After reaping \( U \) and \( U' \), one can choose \( \tilde{U} = [U, U'] \) as embedding matrix.

---

**III. PROBLEM STATEMENT AND DIFFERENTIAL PRIVACY**

**A. PROBLEM DEFINITION**

Suppose we have a graph \( G \) with \( N \) nodes. We utilize \( A \) to denote the adjacency matrix of \( G \). Throughout this study, we focus on undirected graphs, so the adjacency matrix \( A \) and node similarity matrix \( M \) are symmetric matrices. According to Eq. (1), the objective function can be restated as:

\[
\min f(U) = \frac{1}{2} \| M - UU^T \|_F^2
\]

Then, the formal definition of graph embedding matrix is as follows:

**Definition 1 (Graph embedding matrix):** Given an \( N \)-node graph \( G \), graph embedding algorithms based on matrix factorization decompose the learned \( N \times N \) node similarity matrix \( M \) to yield low-dimensional graph embedding matrix that preserves the inherent properties and structures of the graph.

Throughout this study, we assume the data holder has the ability to obtain node similarity matrix \( M \). The data holder wishes to publicly release embedding matrix...
\( \mathbf{U} = \{ \mathbf{x}_1, \ldots, \mathbf{x}_N \} \) containing \( N \) individuals with dimension \( k \), with each data point in \( \mathbf{x}_i (i \in N) \) representing the similarity weight with other individuals. From a privacy standpoint, the personalized graph structure features implied in embedding matrix can identify the individual, which renders the leak of individual sensitive information in the original graph. The data holder wishes to prevent identification of the individuals in embedding matrix, seeking a matrix release mechanism that obeys the definition of \((\varepsilon, \delta)\)-differential privacy, with \( \varepsilon > 0 \) and \( \delta \geq 0 \) given. Meanwhile, the data holder wishes to retain the utility of the graph structure implied in embedding matrix. The main purpose of releasing embedding matrix is to allow the data analyst to perform graph analysis (i.e. graph reconstruction and link prediction), and the cooperation between data holders and data analysts is illustrated in Fig. 2.

### B. DIFFERENTIAL PRIVACY

Differential privacy has become a gold standard in private data analysis. In general, it requires that the distribution of algorithm outputs has a negligible effect in changing one record in underlying database.

**Definition 2 (Differential privacy [1]):** For \( \varepsilon > 0 \), \( \delta \geq 0 \), a randomized algorithm \( \mathcal{A} \) obeys \((\varepsilon, \delta)\)-differential privacy if and only if for any neighboring databases \( D_1 \) and \( D_2 \) (differing by at most one record), for any possible output \( S \in \text{Range}(\mathcal{A}) \),

\[
P[\mathcal{A}(D_1) = S] \leq \exp(\varepsilon) \times P[\mathcal{A}(D_2) = S] + \delta \quad (3)
\]

where \( \varepsilon \) is the privacy budget, which controls the algorithm to reveal the amount of sensitive information about individuals. A positive \( \delta \) allows the algorithm to yield an unlikely output that reveals more information, while only with probability bounded by \( \delta \). This concept is sometimes referred to as approximate differential privacy. An \((\varepsilon, 0)\)-differentially private algorithm is simply said to be \( \varepsilon \)-differentially private.

For making the algorithm outputs differentially private, two standard random perturbation mechanisms, such as Laplace and Gaussian mechanisms, are frequently used. Since the perturbation needs to mask the contribution of each individual entry of the database \( D \), the proportion of the added noise is closely related to the concept of sensitivity, measuring the maximum change in the output when a single record is added or removed. The formal definition of sensitivity is as follows.

**Definition 3 (L₁ and L₂ sensitivity [1]):** For any two neighboring databases \( D_1 \) and \( D_2 \), the \( L_1 \) and \( L_2 \) sensitivity of a function \( J : D \rightarrow \mathbb{R}^d \) is

\[
\Delta_1(J) = \max_{D_1, D_2} \|J(D_1) - J(D_2)\|_1 \quad (4)
\]

\[
\Delta_2(J) = \max_{D_1, D_2} \|J(D_1) - J(D_2)\|_2 \quad (5)
\]

**Theorem 1 (Laplace mechanism [1]):** For any function \( J : D \rightarrow \mathbb{R}^d \) with sensitivity \( \Delta J \), the mechanism \( \mathcal{A} \),

\[
\mathcal{A}(D) = J(D) + \text{Lap}(\Delta J)/\varepsilon \quad (6)
\]

obeys \( \varepsilon \)-differentially private.

**Theorem 2 (Gaussian mechanism [21]):** Let \( J \) be any function with \( L_2 \) sensitivity \( \Delta_2(J) \). The Gaussian output perturbation mechanism, which returns \( J(D) + \mathcal{N}(0, \sigma^2) \), with

\[
\sigma \geq \frac{\Delta_2(J)}{\varepsilon} \sqrt{2 \ln(1.25/\delta)} \quad (7)
\]

is \((\varepsilon, \delta)\)-differentially private.

For a sequence of differentially private algorithms, the following theorem, which including the case of \( \varepsilon \)-differential privacy by setting \( \delta = 0 \), is a generalized composition theorem to guarantee the overall privacy.

**Theorem 3 (Generalized sequential composition [21]):** Let \( \mathcal{A}_1, \ldots, \mathcal{A}_p \) be \( p \) calculations, each obeys \((\varepsilon, \delta)\)-differential privacy. A sequence of \( \mathcal{A}_i(D) \) over database \( D \) obeys \((\sum_{i=1}^p \varepsilon_i, \sum_{i=1}^p \delta_i)\)-differential privacy.

The following theorem, the proof of which is offered in [21], claims that a differentially private result of a query still remains differentially private after post-processing.

**Theorem 4 (Post-Processing [21]):** If \( \mathcal{A} \) is an \((\varepsilon, \delta)\)-differentially private algorithm, then its sequential composition \( \mathcal{B} \circ \mathcal{A} \) with any other algorithm \( \mathcal{B} \) that does not have access to the private database \( D \) is also \((\varepsilon, \delta)\)-differentially private.

### C. PRIVACY DEFINITION

In this study, we concern about preventing identification of the individuals using associations between individuals and some possible background information such as the personalized graph structure features. To this end, we assume that \( G_1 \) and \( G_2 \) be neighboring graphs differing by one edge \( e \) (i.e., \( G_1 = G_2 + e \) or vice versa). Let \( \mathbf{M}_1 \) and \( \mathbf{M}_2 \) be two graph embedding matrices derived from \( G_1 \) and \( G_2 \) following Katz Index [19]. We formally define differentially private embedding matrix as follows.

**Definition 4 (Differentially private embedding matrix):** Released graph embedding matrix, making no statistical difference on the output in case of the addition or removal of an edge between nodes from an original network graph, maintains differential privacy.

Through the analyses mentioned above, we manage to facilitate the graph embedding process using differential privacy. An overall flowchart of this proposed method is demonstrated in Fig. 3.

### IV. THE PROPOSED SOLUTION

In this section we present PPGD, a solution that releases differentially private embedding matrix via MF optimized by the perturbed gradient descent. Table 1 summarizes the main notation and symbols used in this study. Suppose the node similarity matrix \( \mathbf{M} \in \mathbb{R}^{N \times N} \) of the graph \( G \) has been obtained by Katz Index [19]. The steps of PPGD may be summarized as:

1. **Optimization for matrix factorization:** The first step designs the objective function of MF that meets Lipschitz condition.
A. STEP 1: OPTIMIZATION FOR MATRIX FACTORIZATION
Recall that the objective function (2) of this study, which is a challenging non-convex optimization problem. Dauphin et al. [16] state that standard analysis of gradient update for non-convex optimization oftentimes get stuck at local minimal. To this end, a perturbed stochastic gradient descent method [22] (see Algorithm 1) with regularity properties (gradient and Hessian Lipschitz) is employed as a base optimization technique of MF.

Algorithm 1 Perturbed Gradient Descent (PGD)
Input: \( x_0 \), step size \( \eta \), perturbation radius \( r \), total iterations \( T \)
1 for \( t = 0, 1, \ldots \), do
2 \( x_{t+1} \leftarrow x_t - \eta (\nabla f(x_t) + \xi_t) \),
\( \cdot \xi_t \sim \mathcal{N}(0, (r^2/d)I) \)
3 return \( x_T \)

Definition 5: A differentiable function \( f(\cdot) \) is \( \ell \)-smooth or \( \ell \)-gradient Lipschitz if for any \( x, y : \| \nabla f(x) - \nabla f(y) \|_2 \leq \ell \| x - y \|_2 \).

Definition 6: A twice-differentiable function \( f(\cdot) \) is \( \rho \)-Hessian Lipschitz if for any \( x, y : \| \nabla^2 f(x) - \nabla^2 f(y) \|_2 \leq \rho \| x - y \|_2 \).

Theorem 5 ([22]): For any \( \epsilon, \zeta > 0 \), if function \( f \) meets Definitions 5 and 6, and one runs PGD (see Algorithm 1) with parameter \((\eta, r)\) selected as:
\[
\eta = \frac{1}{\ell r} \cdot \frac{\epsilon}{\epsilon \zeta}, \quad r = \epsilon \cdot \frac{\sqrt{T}}{\log T}
\]
where \( \Omega \) and log factor \( r \) are defined as follows:
\[
\Omega = 1 + \frac{\ell}{\sqrt{p \epsilon}}, \quad r = c \cdot \log \left( \frac{d \Delta f \Omega}{\rho \epsilon} \right)
\]
where \( c \) is a sufficiently large absolute constant and \( \Delta f = f(x_0) - f^* \). Then, PGD will find an \( \epsilon \)-approximate local minimum at least once, with at least probability \( 1 - \zeta \), in iterations:
\[
O \left( \frac{\ell (f(x_0) - f^*)}{\epsilon^2} \cdot \Omega \right)
\]
where \( x_0 \) is the initial point and \( f^* \) is the optimal value of \( f \). It is worth noting that [22] asserts that the perturbation \( \xi_t \) (Line 2) in Algorithm 1 can escape local minima efficiently, which is an essential property used in our study.

Lipschitz Condition on Matrix Factorization: We make the following assumptions on the parameter space, which promotes the function \( f(U) \) to be smooth. The restricted region \( \{ U \| U \|_2^2 \leq \| M \|_2^2 \} \) is preconditions for Lemma 1 and Lemma 2, which tells that under the assumption, we always have a good spectral property over \( U \).

Lemma 1: Within the region \( D \in \{ U \| U \|_2^2 \leq \| M \|_2^2 \} \), the function \( f(U) = \frac{1}{2} \| M - UU^T \|_F^2 \) satisfies:
\[
\| \nabla f(U_1) - \nabla f(U_2) \|_F \leq \ell \| U_1 - U_2 \|_F
\]
where gradient Lipschitz parameter \( \ell = \frac{8 \| M \|_2}{\| M \|_2} \).

proof: For any \( U_1, U_2 \in D \), we have:
\[
\| \nabla f(U_1) - \nabla f(U_2) \|_F
\leq \| 2(U_1 U_1^T - M)U_1 - 2(U_2 U_2^T - M)U_2 \|_F
\leq \| 2(U_1 U_1^T - U_2 U_2^T) + 2(M(U_1 - U_2)) \|_F
\leq 6 \max(\| U_1 \|_2^2, \| U_2 \|_2^2) \| U_1 - U_2 \|_F
+ 2 \| M \|_2 \| U_1 - U_2 \|_F
\leq 8 \| M \|_2 \| U_1 - U_2 \|_F
\]
The second inequality is because of following decomposition and triangle inequality:

\[
U_1^T U_1 - U_2^T U_2 = U_1^T (U_1 - U_2) + U_1(U_1 - U_2)^T U_2 + (U_1 - U_2)^T U_2^T U_2
\]

**Lemma 2:** Within the region \(\mathcal{D} \in \{U||U||_2^2 \leq ||M||_2^2\},\) the function \(f(U) = \frac{1}{2}||M - UU^T||_F^2\) satisfies:

\[
||\nabla^2 f(U_1) - \nabla^2 f(U_2)||_F \leq \rho ||U_1 - U_2||_F
\]

where Hessian Lipschitz parameter \(\rho = 12\sqrt{||M||_2^2}\).

**Proof:** For any \(U_1, U_2 \in \mathcal{D}\), we have:

\[
||\nabla^2 f(U_1) - \nabla^2 f(U_2)||_F = 2||U_1^T U_1 - 3U_2 U_2^T||_F \\
\leq 12\sqrt{||M||_2^2} ||U_1 - U_2||_F
\]

The last line is based on similar technics as in the proof of Lemma 1, via broadening

\[
U_1^T U_1 - U_2^T U_2 = U_1(U_1 - U_2)^T + (U_1 - U_2)U_2^T
\]

Notice that the main result Theorem 5 supports the function \(f(\cdot)\) whose input \(x\) is a vector. In order for applying Theorem 5, we always vectorize the input matrix \(U \in \mathbb{R}^{N \times k}\) to be a vector in \(\mathbb{R}^{Nk}\). Thus, the dimension \(d\) is henceforth equal to \(Nk\). However, for simplicity of presentation, instead of explicit vectorization, we still write everything in matrix form, while the reader should always remember that the operations are the same as first vectorizing everything.

**B. STEP 2: GLOBAL SENSITIVITY BOUND**

In this section, the \(L_1\) sensitivity of step size \(\eta\) and \(L_2\) sensitivity of gradient \(\nabla f(U)\) are bounded. Along the way, a composite noise added scheme is designed to ensure differential privacy throughout MF-based graph embedding.

**Step 2a: Sensitivity Computation for Step Size:** From the Section IV-A, we can find that \(||M||_2\) that is the maximum singular value of \(M\) with sensitive information is the building block of the step size. According to the post-processing Theorem 4 of differential privacy, if satisfying differential privacy, \(||M||_2\) can ensure that the step size is differentially private.

Next, we show how to bound global sensitivity of \(||M||_2\). A common method is implemented based on the definition (3), that is, \(\Delta_1(||M||_2) = \max ||M\text{grad}(G_1)||_2 - ||M\text{grad}(G_2)||_2\). Nevertheless, bounding such sensitivity is an overly complex analytical computation. Without loss of generality, we try to bound this sensitivity via looking for a lower upper bound. Formally, bounding global sensitivity is as follows.

**Theorem 6 (\cite{23}):** For any \(M \in \mathbb{R}^{m \times n}\), there exists a unit 2-norm n-vector \(z\) such that \(M^T M z = \mu^2 z\) where \(\mu = ||M||_2\).

**Theorem 7:** The \(L_1\) sensitivity of \(||M||_2\) is \(\Delta_1(||M||_2) \leq 1.\)

**Proof:** Based on Theorem 6, if \(z \neq 0\) is such that \(M^T M z = \mu^2 z\) with \(\mu = ||M||_2\), then \(\mu^2 ||z||_1 = M^T M z_1 = ||M||_\infty ||M||_1 ||z||_1 = ||M||_\infty ||z||_1\). Further, in this study we normalize \(M\) by \(\frac{M}{\text{sum}(M)}\) where \(\text{sum}(M)\) denotes the sum of all elements in \(M\), which results in each element \(0 \leq M(x, y) \leq 1\) for any \(x, y \in N\). Within the region of normalization, \(||M||_\infty \leq 1\) and \(||M||_1 \leq 1.\) As a consequence, \(\Delta_1(||M||_2) \leq 1.\)

It is worth noting that \(||M||_2 \geq 0\), but the perturbation noise may lead to \(||M||_2\) becoming the negative number. To this end, we take the absolute value over the perturbed \(||M||_2\) (Line 2 in Algorithm 2).

**Step 2b: Sensitivity Computation for Gradient:** As shown in Algorithm 1, though the gradient \(\nabla f(U)\) has indirect access to private node similarity matrix \(M\), it can still breach individual privacy. We now present our private gradient update technique described in Algorithm 2. Motivated by the gradient clipping technique \([24]\), our solution provides differential privacy for training data via clipping the norm of the gradient update for each record, and then perturbing these clipped gradients with Gaussian mechanism.

As presented in Algorithm 2, we clip each gradient in \(\ell_2\) norm, i.e., the gradient vector \(\nabla f(U)\) is replaced by \(\frac{\nabla f(U)}{\max\{1, \frac{\nabla f(U)}{C_{gra}}\}}\) where \(C_{gra}\) is a clipping threshold. This clipping guarantees that if \(\|\nabla f(U)\|_2 < C_{gra}\), then \(\nabla f(U)\) is preserved. In contrast, if \(\|\nabla f(U)\|_2 > C_{gra}\), it gets scaled down to be of norm \(C_{gra}\). More concretely, for maintaining the property of Algorithm 1 as much as possible, we clip the \(L_2\) norm of the gradient of the function \(f(\cdot)\) to have a norm at most \(C_{gra} = \frac{r}{\sigma}\) where \(r\) is an undisturbed value (Line 4). Then, we add Gaussian noise \(\mathcal{N}(0, \sigma^2I)\) to these clipped gradient updates (Line 5). Finally, we carry out the descent step (Line 6).

Notice that while similar work has been done in \([24], [25], [26]\), for reducing the impact of noise, they generally set a small clipping threshold, which potentially breaches the gradient update scale to affect the utility of the original algorithm. In contrast, ours can tackle this problem and the detailed analysis is introduced in Theorem 8.

**C. STEP 3: PRIVATE EMBEDDING MATRIX GENERATION**

In this section, we first present privacy analysis of PPGD. Through the analyses mentioned above, we have the following theorems about Algorithm 2.

---

**Algorithm 2 PGGD With Differential Privacy (PPGD)**

Input: \(U_0\), step size \(\eta^t\), perturbation radius \(r\), total iterations \(T\).

1. for \(t = 0, 1, \ldots, T\) do
   2. \(g(M) = ||M||_2 + \text{Lap}(\frac{\Delta_1(||M||_2)}{\epsilon})\),
   3. \(\eta^t \leftarrow \frac{1}{\frac{r^2}{\sigma^2} + \epsilon} \cdot g(M)\) as the base of \(t, \ell, \eta\), \(r\)
   4. \(\nabla f(U_t) \leftarrow \frac{\nabla f(U_t)}{\max\{1, \frac{\|\nabla f(U_t)\|_2}{C_{gra}}\}}\)
   5. \(\tilde{f}(U_t) \leftarrow \nabla f(U_t) + \xi_t\),
      \(\xi_t \sim \mathcal{N}(0, (r^2/d)I)\)
   6. \(U_{t+1} \leftarrow U_t - \eta^t \tilde{f}(U_t)\)
7. return \(U_T\)
Theorem 8: The gradient update is differentially private for near-free.

Proof: To bound the $L_2$ sensitivity of the gradient, we clip the $L_2$ norm of the gradient of the function $f(\cdot)$ to have a norm at most $C_{gra} = \frac{t}{\gamma}$ (Line 4). Along this way, we maintain that $\sqrt{2 \ln(1.25/\delta)/\varepsilon}$ is always equal to $\sqrt{d}$. Then, we get that $\delta$ is equal to $\frac{1}{10 \sqrt{e^{d^2}/2}}$, and recall the dimension $d$ is equal to $\frac{\ln N}{\ln \gamma}$. For the step size $\eta$, where in general the dimension $N$ of the social network is large, which renders the obtained $\delta$ to be very small and achieves strong privacy protection. Following Eq. 7, we can get that the standard deviation of Gaussian noise is $\sigma = \frac{t}{\gamma}$, achieving $(\varepsilon, \delta)$-differential privacy for the gradient update. Indeed, the gradient update is differentially private for near-free. More precisely, with the above settings, the total standard deviation of Gaussian noise in Algorithm 2 is equal to $\sqrt{d}$. Recall that Theorem 5, such $\sigma$ has two benefits: to escape from local minima while preserve the privacy of the gradient update. In addition, from Theorem 5, the $c$ is a sufficiently large value, which ensures that the sensitivity $C_{gra}$ can be bounded by a large value to maintain the inherently property of Algorithm 1.

Theorem 9: Algorithm 2 satisfies $(\varepsilon, \delta)$-differential privacy.

Proof: For the gradient update (Line 5), Theorem 8 has shown that by setting the gradient clipping threshold to $\frac{t}{\gamma}$, we can obtain $L_2$ sensitivity of $\|M\|_2$ to have a norm at most $C_{gra} = \frac{t}{\gamma}$ (Line 4). Along this way, we maintain that $\sqrt{2 \ln(1.25/\delta)/\varepsilon}$ is always equal to $\sqrt{d}$. Then, we get that $\delta$ is equal to $\frac{1}{10 \sqrt{e^{d^2}/2}}$, and recall the dimension $d$ is equal to $\frac{\ln N}{\ln \gamma}$.

Parameter Settings: In the experiments, a series of values for the embedding size $k = \{50, 100, 150, 200, 250, 300\}$ are selected to perform graph reconstruction and link prediction. The total privacy budget $\varepsilon$ is allocated to $\varepsilon_1 = 0.2\varepsilon$ and $\varepsilon_2 = 0.8\varepsilon$. The standard deviation $\sigma$ can be set by $\varepsilon$ under satisfying Theorem 8. The parameter $c$ is set to $10^6$. The parameters $\varepsilon$ and $\zeta$ are set to 0.9 and 0.01, respectively. The trial is repeated five times and the average results are reported.

Evaluation Metrics: To evaluate the utility of the proposed method on graph reconstruction and link prediction, we use MeanAveragePrecision (MAP) as the evaluation metric to estimate the precision of each node and calculates the average over all nodes, as follows:

$$MAP = \frac{\sum_i AP(i)}{|V|}$$

where $V$ is the node set in the graph, $AP(i) = \sum_{\text{Precision} \@ \hat{k}(i)} \frac{|E_{pred}(\hat{k}) \cap E_{obs}]}{|E_{pred}(\hat{k})|}$, and $E_{obs}$ and $E_{pred}$ are the predicted and observed edges for node $i$ respectively. For graph reconstruction, $E_{obs}$ is equal to $E$ where $E$ is the set of edges in the graph, and for link prediction, $E_{obs}$ is the set of hidden edges.

A. GRAPH RECONSTRUCTION TASK

Reconstructing the graph is one primal objective for graph embedding [30]. We train embedding vectors included in embedding matrix and rank pairs of nodes according to their similarity values, i.e., the inner product of two embedding vectors. The highest ranking pairs of nodes are used for graph reconstruction. As the number of possible pairs of nodes ($N(N-1)$) can be very large for graphs with a large number of nodes, we randomly sample about 10% pairs of nodes for evaluation.
The privacy budget $\varepsilon$ is used as a key parameter for determining the privacy level. Fig. 4 reports the average precision of graph reconstruction of different schemes under varying embedding dimensions. PPGD is demonstrated over three $\varepsilon$ values ($\varepsilon = 0.1, 0.5, 1$). As a comparison algorithm, MDPNE is reported by setting $\varepsilon = 1$. According to [31], $\varepsilon = 0.1$ and $\varepsilon = 1$ correspond to high and medium privacy guarantees, respectively. As shown from Fig. 4, the horizontal axis represents the dimension. Under the same dimension, the average precision of PPGD is improved as $\varepsilon$ increases. The reason is that, as $\varepsilon$ increases, a lower magnitude of noise is injected and a lower degree of privacy is guaranteed. Furthermore, Fig. 4 also demonstrates the effect of varying embedding dimensions on graph reconstruction. As the number of dimensions increases, the MAP value increases. This is intuitive since a higher number of dimensions is capable of preserving more graph structural information.

Particularly, for PPI, PPGD($\varepsilon = 1$) obtains better precision than NonPriv-SVD in dimensions 100, 150, 200, 250 and 300, yet outperforms MDPNE($\varepsilon = 1$) through all dimensions. Moreover, PPGD($\varepsilon = 0.1$) and PPGD($\varepsilon = 0.5$) exceed NonPriv-SVD and MDPNE($\varepsilon = 1$) in dimensions 200, 250, and 300. For Route, with the increase of dimensions, PPGD($\varepsilon = 1$) can achieve results close to NonPriv-SVD, especially in dimensions 250 and 300. In addition, MDPNE($\varepsilon = 1$) has a competitive result with PPGD($\varepsilon = 1$) in dimension 50, and as the dimension increases, MDPNE($\varepsilon = 1$) becomes unstable. The reason is that the standard SGD used for non-convex MF oftentimes traps into local minima, which inherently limits the utility of MDPNE($\varepsilon = 1$) retention the graph structure. For Facebook, PPGD($\varepsilon = 1$) is close to NonPriv-SVD in dimensions 50 and 150. Further, PPGD($\varepsilon = 1$) can reap more competitive results than NonPriv-SVD in dimensions 250 and 300, but overcomes MDPNE($\varepsilon = 1$) in all dimensions. This indicates that during optimization process, the proposed PPGD is capable of escaping from local optimum to promote generating good utility of the graph structure for embedding matrix release while satisfy a suitable privacy preserving requirement.

**B. LINK PREDICTION TASK**

Link prediction, which aims at predicting which pairs of nodes are likely to constitute edges, is a typical application of graph embedding. We randomly hide 10% of the edges for testing and learn the embedding using the remaining 90% to predict the most likely edges that are not observed in the training data from the learned embedding. Then, we rank pairs of nodes in a similar way to graph reconstruction, evaluating the highest ranking pairs on the testing graph.

Fig. 5 shows the average results for link prediction with varying embedding dimensions over different privacy budgets such as $\varepsilon = 0.1, \varepsilon = 0.5,$ and $\varepsilon = 1$. Here we can
see that the prediction precision of PPGD($\varepsilon = 1$) is close to the non-private NonPriv-SVD, and exceeds MDPNE($\varepsilon = 1$) in all dimensions. In particular, for PPI, PPGD($\varepsilon = 1$) achieves better prediction precision with dimensions 200 and 250. In addition, from Fig. 5, PPGD($\varepsilon = 0.5$) can also obtain competitive results approaching to PPGD($\varepsilon = 1$) and NonPriv-SVD, and overcomes MDPNE($\varepsilon = 1$). Even for $\varepsilon = 0.1$, which means that more noise will be injected, PPGD can still yield competitive results, especially for dimensions 200, 250 and 300 in Route, and dimensions 250 and 300 in Facebook. This again demonstrates that PPGD is able to retain the utility of the graph structure in graph analysis, due to its ability to escape from local minima efficiently.

In summary, the above evaluations for graph reconstruction and link prediction demonstrate the effectiveness of PPGD on several aspects. First, even when compared to the non-private NonPriv-SVD scheme, it still retains a high precision when reconstructing the graph. Second, its performance is significantly improved when the privacy budget increases. A proper privacy budget can be chosen to achieve a better trade-off between privacy and utility. Finally, the utility loss may be negligible with an adequate privacy budget.

VI. CONCLUSION

In this study, we have investigated how to share graph embedding matrix with differential privacy. The underlying highlights lie in two aspects. i) We use a Lipschitz condition on the objective function of MF and a gradient clipping strategy to bound global sensitivity to propose a scalable solution that is independent on original graph data. ii) We design a composite noise added means to ensure differential privacy throughout the gradient descent solver of MF-based graph embedding problem. Meanwhile, the composite noise derives the gradient descent based optimization to escape from local minima facilitating good utility guarantees. Theoretical analysis is provided to guarantee that the proposed method can generate perturbed embedding matrix with the utility maximization while achieving ($\varepsilon$, $\delta$)-differential privacy. Simulations illustrate the effectiveness and efficiency of the proposed method. In spite of the bright sides, the future effort is to consider a scenario where embedding vectors are explored under differential privacy when the objective function of MF is unknown.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers whose valuable comments have greatly improved the presentation of the paper.

REFERENCES


**SEN ZHANG** is currently pursuing the Ph.D. degree with the Complex Data Management Laboratory, School of Computer Science and Engineering, Southeast University, Nanjing, Jiangsu, China. His research interests include data mining, data privacy protection, big data technology, and complex data management.

**WEIWEI NI** received the B.E. and Ph.D. degrees in computer science from Southeast University, China, in 2001 and 2005, respectively. He is currently a Professor with Southeast University. His research interests include data mining, data provenance, data privacy protection, big data technology, and complex data management.