Received 16 June 2023, accepted 17 July 2023, date of publication 20 July 2023, date of current version 26 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3297437

RESEARCH ARTICLE

Differentially Private Social Graph Publishing With Nearest Neighbor Structure Preservation

XINJIAN ZHAO[1], FEI XIA[1], GUOQUAN YUAN[1], SEN ZHANG[2], SHI CHEN[1], AND WEIWEI NI[2]
[1]Information and Telecommunication Branch, State Grid Jiangsu Electric Power Company Ltd., Nanjing 210024, China
[2]School of Computer Science and Engineering, Southeast University, Nanjing 211189, China

Corresponding author: Sen Zhang (zhangsen98k@163.com)

This work was supported by the Science and Technology Project of State Grid Jiangsu Electric Power Company Ltd., under Grant J2022109.

ABSTRACT The goal of privacy-preserving social graph publishing is to protect individual privacy while preserving graph utility. Node nearest neighbor structure is a crucial social graph utility as it is the basis for many graph analysis tasks. Most existing methods with differential privacy focus on preserving degree distribution yet neglect the maintenance of connections between nodes' neighbors. Moreover, they require massive noise added to mask the change of a single edge, thereby rendering poor utility on the neighbor structure. As a result, it is tough to preserve high utility on the neighbor structure under differential privacy. To tackle these problems, we propose Priv-NNS, a private graph publishing algorithm to preserve the neighbor structure while guaranteeing individual privacy. This algorithm first decomposes a graph into subgraphs via extracting a nearest neighbor structure around each node. Then to yield node vectors satisfying differential privacy while preserving the neighbor structure, we design a private graph encoding approach with structure-awareness, which learns topological features of the subgraph by maximizing the co-occurrence probability among nodes. During this process, a novel objective perturbation approach with a random term, only requiring a scalar noise rather than a vector noise, is devised to balance the neighbor structure retained against the noise injected. Through formal privacy analysis, we prove that the yielded synthetic graph obeys ε-edge differential privacy. Experimental results demonstrate that Priv-NNS preserves high utility on the neighbor structure.

INDEX TERMS Differential privacy, graph publishing, nearest neighbor structure, graph encoding, objective perturbation.

I. INTRODUCTION

Social graphs model relevant social interactions of individuals in a community and are widely used in social network analysis. An important data utility of such graphs is represented as node nearest neighbor structure, which is the cornerstone in many important graph analysis applications, such as community detection [1] and influence maximization [2]. While the potential benefits of the neighbor structure are indisputable, direct publication of social graphs potentially results in individual privacy (e.g., node identities) being revealed from personalized degree values. Therefore, it is an imperative issue that preserves the neighbor structure without leaking individual privacy in privacy-preserving graph publishing.

Differential privacy [3] is an extensively used statistical model of privacy as it provides a rigorous mathematical framework for preserving privacy. Previously, a variety of works have been proposed for graph publishing under differential privacy. The dK-2 [4] and HRG [5] are the two most prominent ones. Particularly, dK-2 extracts the structure of a graph into joint degree sequences, and generates a synthetic graph using the perturbed sequence values. An alternative solution [6] is presented to further improve the dK-2 by considering global sensitivity rather than local sensitivity. Compared with dK-2 based solutions, with a statistical hierarchical random graph model [7], the HRG method stores a cluster of nodes via privately estimating the

The associate editor coordinating the review of this manuscript and approving it for publication was Aasia Khanum.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

connection probabilities among nodes. Despite their success, two fundamental problems in differentially private graph publishing remain unsolved. First, whether preserving the degree sequences or cluster of nodes, most existing methods essentially maintain the utility with regard to the distribution in the statistical sense. However, the distribution ignores the maintenance of connections between nodes' neighbors. Second, the existing methods require massive noise added to mask the change of a single edge. As a consequence, how to preserve the neighbor structure while obeying differential privacy inevitably brings new challenges.

Towards this end, we present Priv-NNS, a differential privacy-based graph publishing algorithm to preserve high utility on node nearest neighbor structure without compromising individual privacy. More precisely, we first decompose a graph into subgraphs via extracting a nearest neighbor structure around each node. After that, aiming to generate the node vectors satisfying differential privacy where the neighbor structure is preserved, we design a private graph encoding approach with structure-awareness, which learns topological features of the subgraph via maximizing the co-occurrence probability among nodes. To reduce the damage caused by noise in the graph encoding process, a novel objective perturbation approach with a random term, which only requires a scalar noise rather than a vector noise, is devised to balance the neighbor structure retained against the noise injected. Finally, a synthetic graph can be reconstructed by calculating distances between nodes in noisy vectors. Through formal privacy analysis, we prove that the generated synthetic graph satisfies ε-edge differential privacy. Extensive encouraging experimental results exhibit the superiority and effectiveness of our proposed private graph publishing method.

The main contributions of our paper can be summarized as follows:

- We propose Priv-NNS, a privacy-preserving graph publishing algorithm. It can preserve high data utility on node nearest neighbor structure and maintain ε-edge differential privacy simultaneously.
- We design a private graph encoding approach with structure-awareness to learn topological features of the decomposed subgraph by maximizing the co-occurrence probability among nodes.
- We devise a novel objective perturbation approach with a random term to reduce the damage of noise in the graph encoding process. This approach only requires a scalar noise rather than a vector one and thus balances the neighbor structure retained against the noise injected.
- Empirically, over real datasets, we demonstrate that differentially private synthetic graphs generated by Priv-NNS achieve a significant improvement with respect to their counterparts, in terms of their ability to maintain the neighbor structure of the original graphs.

The remainder of this paper is below. Section II reviews related works. The problem statement and preliminaries are presented in Sections III and IV, respectively. Our proposed Priv-NNS is described in Section V, followed by experimental results in Section VI. Finally, the conclusion is drawn in Section VII.

II. RELATED WORK

When applying differential privacy to graph data [8], [9], two variants of differential privacy are introduced [10]: in node-differential privacy, two graphs are said to be neighbors if one can be obtained from the other by adding or deleting a node and its adjacent edges; in edge-differential privacy, two graphs are said to be edge neighbors if one can be obtained from the other by adding or removing a link connecting two nodes. Obviously, satisfying node-differential privacy is much harder than satisfying edge-differential privacy, since removing one node may cause the removal of $|\mathcal{V}| - 1$ edges where $\mathcal{V}$ is the set of nodes. In comparison, edge-differential privacy is easy to implement and has a good balance with utility preservation. Due to this reason, most of existing methods consider edge-differential privacy. In the following sections, we review previous works with edge-differential privacy on two directions, namely private graph statistics publishing and private graph publishing, and discuss how our work differs from existing works.

A. GRAPH STATISTICS PUBLISHING WITH DIFFERENTIAL PRIVACY

Degree distribution and subgraph counting queries are two kinds of statistics that are continually studied in the literature. Hay et al. [10] consider releasing the degree distribution of a graph under differential privacy using a constrained inference technique. The idea is to impose an ordering constraint on the pre-noise degree sequence, add Laplace noise to the sorted sequence then post-process the noisy sequence to restore the order constraint. Subgraph counting queries ask how many "edge-induced isomorphic copies" of a query graph Q (e.g., triangle), are present in a graph. Zhang et al. [11] define the Ladder framework for producing accurate differential privacy estimates of subgraph counting queries, including triangles and k-stars. The Ladder framework combines the concept of "local sensitivity at distance t" [12] with the exponential mechanism [13]. Shen and Yu [14] consider finding frequently occurring subgraphs under differential privacy. Recently, Ahmed et al. [15] present an eigenspectrum publishing method regarding online social graphs, which achieves both space efficiency and utility preservation by reducing adjacency matrix sizes.

Nevertheless, differential privacy requires one to define a privacy budget in advance, which determines the amount of perturbation that will be applied to the outputs of algorithms. Therefore, private graph statistics publishing needs to limit in advance the number of queries that will be answered or provides answers of increasingly lower quality. In contrast, private graph publishing can devote the entire privacy budget to the capture of the original graph properties, without further degradation of the privacy of the sampled graphs. For this reason, we concentrate on differentially private publication of synthetic graphs in this paper.

B. GRAPH PUBLISHING WITH DIFFERENTIAL PRIVACY

A few research efforts have been made towards differentially private publication of social graph data. For instance, Mir and Wright [16] focus on generating representative synthetic graphs via the Kronecker graph model. Their approach estimates the model parameter from the input graph under differential privacy. Sala et al.'s Pygmalion model uses the dK-series of the input graph, which records the distribution of the pairs of degrees observed on edges [4]. Subsequent work combined the dK-series model that constructs a synthetic graph via smooth sensitivity [6]. Xiao et al. encode the structure of a graph via private edge counting queries, under the hierarchical random graph model, and claim better results than the dK-series approach [5]. Chen et al. [17] use the exponential mechanism to sample an adjacency matrix after clustering the input graph. Proserpio et al. [18] suggest non-uniformly down-weighting the edges of a graph in order to reduce high global sensitivity due to the possibility of very high degree nodes. They demonstrate the approach, combined with MCMC-based sampling, to generate private synthetic graphs. Although these methods deal with differentially private publishing of social graph data, they cannot preserve the utility on node nearest neighbor structure. This is because the previous works reconstruct synthetic graphs by privately extracting important statistics (e.g., degree sequences), and thus retain the utility on the distribution in the statistical sense. However, statistics ignore the maintenance of connections between node neighbors. Moreover, differential privacy prefers to maintain statistical distribution. However, the neighbor structure does not completely belong to the statistical distribution, and it is also characterized as a structural feature. Therefore, it is a challenging issue to preserve the neighbor structure simultaneously meeting differential privacy. Very recently, Gao and Li [19] propose a private scheme to preserve both the adjacency matrix and persistent homology, where the persistent structures are in the form of holes. Yet, a high-dimensional hole exists only if the low-dimensional surface is complete, which potentially limits this scheme's ability to maintain high data utility.

In recent years, there has been widespread use of spectral methods based on matrix eigenvectors for community detection. In light of this, certain private graph publishing algorithms [20], [21] have been developed to maintain the spectrum of the original graph, thereby retaining the community structure. In a similar vein to [20] and [21], Elias et al. [22] devise a method to capture the community structure by privately approximating all cuts of the input graph. While these proposed algorithms can preserve the neighbor structure to some extent, they only achieve (ε, δ)-differential privacy, which is a less stringent version of ε-differential privacy. Our work, on the other hand, differs from these approaches by focusing specifically on preserving the node nearest neighbor structure while satisfying ε-differential privacy.

TABLE 1. Frequently used symbols.

Symbol	Description
ε	Differential privacy parameter
$\mathcal{G}, \widetilde{\mathcal{G}}$	Original graph and synthetic graph
$\mathcal{G}, \mathcal{G}'$	Any two neighboring graph datasets
$\mathcal{A}$	A randomized algorithm
$\mathcal{V}$	Set of nodes of $\mathcal{G}$
$\mathcal{E}$	Set of edges of $\mathcal{G}$
$\lvert\mathcal{V}\rvert$	Number of nodes in $\mathcal{G}$
$\mathbf{x}, \mathbf{y}$	Lowercase letters denote vectors
$\mathbf{X}, \mathbf{Y}$	Bold capital letters denote matrices
$\mathbf{x}_i^{nns}$	Nearest neighbor structure encoding of node i
$\lVert\mathbf{x}\rVert_2$	The ℓ_2-norm of vector $\mathbf{x}$
$\lVert\mathbf{X}\rVert_2$	The spectral norm of matrix $\mathbf{X}$
$\lVert\mathbf{X}\rVert_F$	The Frobenius norm of matrix $\mathbf{X}$
$\lVert A\rVert_{1,1}$	The elementwise norm-1 of A
$\mathcal{L}(\cdot), \widetilde{\mathcal{L}}(\cdot)$	The loss functions
η	Learning rate of Skip-gram
K	Negative sampling number of Skip-gram

As a summary, how to maximize the utility on node nearest neighbor structure while protecting individuals with edge-differential privacy in published graphs remains open in the literature.

III. PROBLEM STATEMENT

Throughout this work, we investigate the following setup. A data holder wishes to publicly release an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with no additional labels on nodes and edges. Meanwhile, the data holder, wishing to prevent the individual privacy leakage caused by personalized degree values in the released graph as well as preserve high utility on nearest neighbor structure, seeks a graph release mechanism that obeys the definition of ε-edge differential privacy. To tackle this problem, we first decompose a graph into subgraphs via extracting a nearest neighbor structure around each node, and then design a private graph encoding approach with structure-awareness to learn topological features of the subgraph. Along the way, a novel objective perturbation approach is designed to achieve differential privacy. For clarity, the process of privately releasing graphs is shown in Figure 1. After deriving the processed graph, the analyst can perform graph analysis tasks, including but not limited to the analysis on node nearest neighbor structure.

IV. PRELIMINARIES

In this section, we briefly review the concept and some important properties of differential privacy and Skip-gram model for word representation learning, which are important for understanding the proposed Priv-NNS method. Table 1 summarizes the mathematical notations frequently used throughout this paper.

A. DIFFERENTIAL PRIVACY

Differential privacy has gradually emerged as the de-facto principle for individual privacy protection in data release. It provides a strong privacy guarantee to ensure that the output

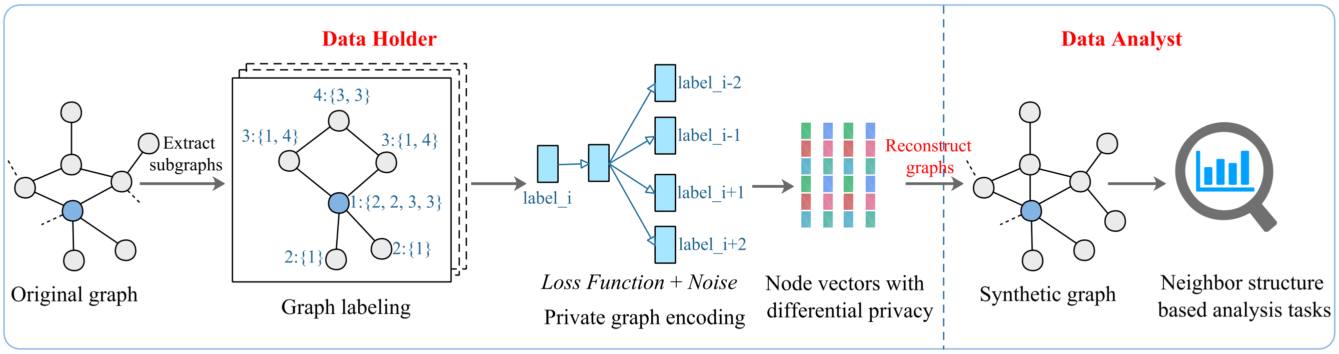


FIGURE 1. Pipeline of proposed Priv-NNS method.

of a computation is insensitive to the changes in any individual's record in the dataset. Thus, the privacy can be protected for the individuals contained in the database. Differential privacy is based on the notion of neighboring datasets.

Definition 1: (edge-differential privacy [10]) A graph analysis mechanism $\mathcal{A}$ maintains ε-edge differential privacy for any input pair of neighboring graphs $\mathcal{G}$ and $\mathcal{G}'$, where $\mathcal{G}$ and $\mathcal{G}'$ differ by at most one edge, and for any possible output set, Ω, if $\mathcal{A}$ satisfies:

$$\mathbb{P}[\mathcal{A}(\mathcal{G}) = \Omega] \leq \exp(\varepsilon) \times \mathbb{P}[\mathcal{A}(\mathcal{G}') = \Omega],$$

where the parameter ε denotes the privacy level and smaller values indicate a stronger privacy guarantee.

Laplace Mechanism: One way to satisfy differential privacy is to add noise to the output of a query. In the Laplace mechanism [3], to publish $f(\mathcal{G})$ where $f : \mathcal{G} \rightarrow \mathbb{R}^d$ while satisfying ε-edge differential privacy, one publishes

$$\mathcal{A}(\mathcal{G}) = f(\mathcal{G}) + Lap(\frac{S_f(\mathcal{G})}{\varepsilon})^d,$$

where $S_f(\mathcal{G}) = \max_{\mathcal{G},\mathcal{G}'} \|f(\mathcal{G}) - f(\mathcal{G}')\|_1$ is the global sensitivity of the quality function. $\mathbb{P}[Lap(\beta) = x] = \frac{1}{2\beta} e^{-|x|/\beta}$, in which $\beta = S_f(\mathcal{G})/\varepsilon$.

In addition, differential privacy also features the following main properties that are utilized in implementing Priv-NNS.

Theorem 1: (Composition rule [23]) If $\mathcal{A}_1$ is ε_1-differentially private, and $\mathcal{A}_2$ is ε_2-differentially private, then $(\mathcal{A}_1 \circ \mathcal{A}_2)$ is $(\varepsilon_1 + \varepsilon_2)$-differentially private.

Theorem 2: (Post-processing [23]) If $\mathcal{A}$ is an ε-differential privacy algorithm and $\mathcal{B}$ is an arbitrary data-independent mapping, then $\mathcal{B} \circ \mathcal{A}$ is also ε-differentially private.

B. LEARNING REPRESENTATIONS

Word2Vec [24] is one of the most well-known word representation learning methods, the target of which is to convert words into a low-dimensional continuous vector space in which semantically similar words are mapped to nearby points. Furthermore, word vectors can be utilized for answering analogy questions via leveraging simple algebra operators, e.g., the closest vector to "China" + "capital" is found to be "Beijing".

Word2Vec leverages a simple and elegant model to learn word representations, called Skip-gram. In order to learn a word's representation, Skip-gram first employs a sliding window on the input text stream, in which the center word is called the target word and its surrounding words are called context words. Then, it utilizes the representation vector of the target word to predict the representation vector of each individual context word. More to the point, given a sequence of words $w_1, w_2, \ldots, w_T$, the target word w_t whose representation needs to be learned, and the window size C, the training objective of Skip-gram is to maximize the log probability of predicting the context words $w_{t-C}, \ldots, w_{t+C}$ which appear nearby the target word w_t:

$$\max \sum_{t=1}^{T} \log \mathbb{P}\left(w_{t-C}, \ldots, w_{t+C} \mid w_t\right). \tag{1}$$

To make the optimization problem easy to handle, it is assumed that the context words and the target word are independent (i.e., the likelihood of observing a context word is independent of observing any other context words given the target word). The probability $\mathbb{P}\left(w_{t-C}, \ldots, w_{t+C} \mid w_t\right)$ is calculated as:

$$\mathbb{P}\left(w_{t-C}, \ldots, w_{t+C} \mid w_t\right) = \prod_{-C \leq c \leq C, c \neq 0} \mathbb{P}\left(w_{t+c} \mid w_t\right).$$

With the above assumption, the objective in Equation (1) can be simplified:

$$\max \sum_{t=1}^{T} \sum_{-C \leq c \leq C, c \neq 0} \log \mathbb{P}\left(w_{t+c} \mid w_t\right).$$

In addition, $\mathbb{P}\left(w_{t+c} \mid w_t\right)$ is defined via softmax function:

$$\mathbb{P}\left(w_{t+c} \mid w_t\right) = \frac{\exp\left(\mathbf{w}_{t+c} \cdot \mathbf{w}_t\right)}{\sum_{i=1}^{\mathscr{V}} \exp\left(\mathbf{w}_i \cdot \mathbf{w}_t\right)}, \tag{2}$$

in which $\mathbf{w}_{t+c}$ and $\mathbf{w}_t$ denote representation vectors of words w_{t+c} and w_t, and $\mathscr{V}$ denotes the number of words in the vocabulary (i.e., the vocabulary size).

Negative Sampling: Equation (2) is computationally very expensive provided that the vocabulary contains plenty of words. Negative sampling [24] is an efficient solution to handle this issue. Rather than leveraging all words in the vocabulary, negative sampling randomly chooses a relatively few words that are not in the context of the target word (these words are referred to as negative samples). Then, negative sampling attempts to distinguish the target word from the negative samples via utilizing a binary classification objective of logistic regression. After that, negative sampling guarantees that if a word w appears in the context of another word w', then the representation vector of w is closer to that of w' compared with those of negative samples.

V. THE PROPOSED PRIV-NNS SOLUTION

In this section, we elaborate the design of Priv-NNS, a graph publishing algorithm with edge-differential privacy to retain high utility on node nearest neighbor structure and ensure individual privacy simultaneously.

A. OVERVIEW OF PRIV-NNS

Before presenting the details, we first give an overview of our method. The target of our work is to release a sanitized graph $\widetilde{\mathcal{G}}$ that approximates the true node nearest neighbor structure of the original graph $\mathcal{G}$ as closely as possible while satisfying ε-edge differential privacy.

The steps of Priv-NNS may be summarized as:

(1) **Subgraph labeling:** The first step extracts a subgraph around each target node as its neighborhood and then encodes this neighborhood based on the Weisfeiler-Lehman algorithm [25], [26]. The goal of the encoding is to assign subgraphs of similar structure with similar numerical vectors as representation.

(2) **Private graph encoding:** The second step aims to privately map nodes into vectors while preserving the neighbor structure. Specifically, given one neighborhood, we consider its node labels in each iteration as a sequence. Then, considering all neighborhoods in $\mathcal{G}$, we include all sequences together to form a "corpus", where each label is like a "word" and each sequence is like a "sentence". With this analogy in mind, we develop private Word2Vec model by perturbing the objective function and then utilize this model on the corpus to obtain node vector of each label. Finally, neighborhood encoding can be obtained by averaging the label vectors in the neighborhood.

(3) **Synthetic graph generation:** Finally, we reconstruct a synthetic graph via calculating distances between nodes in the generated noisy vectors.

In the rest of this section we discuss and justify each step of Priv-NNS, the pseudocode of which is given in Algorithm 1.

Algorithm 1 Priv-NNS Algorithm

```
Require: graph $\mathcal{G}$; number of iterations $iter$; privacy budget
    $\varepsilon$; number of epochs to train the neural network $\mathfrak{e}$; learning rate $\eta$; dimension $d$; negative sampling number $K$.
Ensure: synthetic graph $\widetilde{\mathcal{G}}$ with $\varepsilon$-edge differential privacy.
 1: // Subgraph labeling
 2: for each target node $u$ in $\mathcal{G}$ do
 3:     Extract neighborhood $\mathcal{G}_u$ from $\mathcal{G}$ for $u$
 4:     for each node $v$ in $\mathcal{G}_u$ do
 5:         Label $v$ with initialization, that is $label^{(0)}[v]$, and
            form a multiset $mul^{(0)}[v] = \{label^{(0)}[w] | (w, v) \in \mathcal{E}\}$
 6:     end for
 7:     for $i$ iterates from 1 to $iter$ do
 8:         for each node $v$ in $\mathcal{G}_u$ do
 9:             Assign $v$ with a new label $sig^{(i)}[v]$ that reflects its
                previous label $label^{(i-1)}[v]$ and the multiset of its
                neighbors $mul^{(i-1)}[v]$
10:         end for
11:         Sort nodes in ascending order of $sig^{(i)}[v]$
12:         Update $label^{(i)}[v]$ and $mul^{(i)}[v]$
13:     end for
14: end for
15: // Private graph encoding
16: Initialize $\mathbf{X} = [x_{ij}]_{|\mathcal{L}|\times d}$ with $x_{ij} \sim \mathcal{N}(0, 0.01)$
17: for $i$ iterates from 1 to $\mathfrak{e}$ do
18:     for label in corpus do
19:         for context in context-window do
20:             // Objective perturbation
21:             $O(\mathbf{X}) = -\log \mathbb{P}(v|u) + Lap(\frac{S_f(\mathcal{G})}{\varepsilon})$ // see
                Algorithm 2 for details
22:             $\mathbf{X}^{new} = \mathbf{X}^{old} - \eta \frac{\partial O(\mathbf{X})}{\partial \mathbf{X}}$
23:         end for
24:     end for
25: end for
26: // Neighbor structure encoding
27: for each target node $u$ in $\mathcal{G}$ do
28:     Generate $\mathbf{x}_u^{nns}$ via averaging the label vectors in the
        nearest neighbor structure
29:     Add $\mathbf{x}_u^{nns}$ into $\mathbf{M}$
30: end for
31: // Synthetic graph generation
32: Generate a synthetic graph $\widetilde{\mathcal{G}}$ via calculating distances
    between nodes in $\mathbf{M}$ // see Section V-D for details
33:
34: return $\widetilde{\mathcal{G}}$
```

B. SUBGRAPH LABELING

To learn structural role of target individual u, we first introduce neighborhood of u, i.e., the subgraph that encloses u.

Definition 2: (Nearest neighborhood) For an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, given a target individual u, the neighborhood of u is the subgraph $\mathcal{G}_u^h$ induced from $\mathcal{G}$ by the set of nodes $\mathcal{V}_u = \{v \mid v \in \mathcal{V}, d(v, u) \leq h\}$ where $d(v, u)$ is the shortest distance from v to u, and h is a user-defined threshold. The purpose of using different labels to mark nodes' different roles in an enclosing subgraph is to assign subgraphs of similar structure with similar numerical vectors as representation. A simple way to represent topological structure is to leverage graph isomorphism test, that is, isomorphic neighborhoods should be tightly encoded together.

Nevertheless, because graph isomorphism is proved to be NP-hard, the Weisfeiler-Lehman algorithm is applied, whose core is to decompose a neighborhood into its subtree patterns, and neighborhoods with similar patterns could be assigned with similar representations. For this, it iteratively labels a neighborhood leveraging the following node coloring process.

(1) Initially, the algorithm colors target node u and other nodes $\mathcal{G}_u$ according to their shortest path distances to u, i.e., $label^{(0)}[v] = dist(v, u)$. Then, for each node in $\mathcal{G}_u$, it collects a multi-set $mul^{(0)}[v]$ that consists of its neighbors' labels,

$$mul^{(0)}[v] = \left\{label^{(0)}[w] \mid (w, v) \in \mathcal{E}\right\}, \tag{3}$$

in which the labels in $mul^{(0)}[v]$ are sorted in ascending order.

(2) In the i-th iteration, the algorithm assigns every node v with a new label, which reflects its previous label and the multiset of its neighbors. Particularly, it defines signature of v by concatenating v's label $label^{(i-1)}[v]$ with $mul^{(i-1)}[v]$, i.e.,

$$sig^{(i)}[v] = \left\{label^{(i-1)}[v] \mid mul^{(i-1)}[v]\right\}. \tag{4}$$

It utilizes set comparison to sort the nodes based on their signatures: $sig\,[v_1] < sig\,[v_2]$ if and only if $label\,[v_1] < label\,[v_2]$ or $\exists l, mul\,[v_1]\,[l] < mul\,[v_2]\,[l], \forall j < l, mul\,[v_1]\,[j] = mul\,[v_2]\,[j]$. After sorting, it assigns a new label $label^{(i)}[v]$ to each node in which nodes with the same signature are assigned the same label, and updates $mul^{(i)}[v]$ accordingly.

(3) Provided that the node labels are stable or reach a predefined number of iterations, the algorithm terminates.

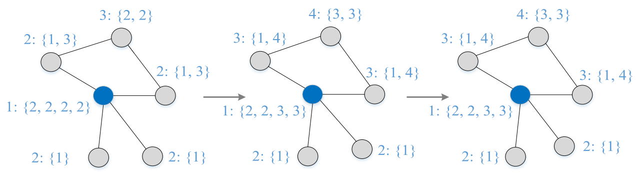
As a summary, the Weisfeiler-Lehman based subgraph labeling iterates over each node and its neighbors in the cause of creating a multiset label. The resulting multiset is given a new label, and then it is utilized for the next iteration. Therefore, multiset labels that belong a given iteration *iter* can be regarded as co-occurred for purpose of partially preserving a notion of similarity.

C. PRIVATE GRAPH ENCODING

In this section, we aim to generate the node vectors that meet edge-differential privacy while preserving high utility on node nearest neighbor structure. In particular, we privately encode u's nearest neighbor structure based on its labeling results and produce a numerical vector $\mathbf{x}_u$ satisfying differential privacy as its representation. For clarity, in what follows, we first leverage an example to display how we extend the word representation learning to graph structure learning, and then present a novel objective function perturbation.

1) REPRESENTATION LEARNING-BASED ENCODING

Given one neighborhood, we regard its node labels in each iteration as a sequence. Then, considering all neighborhoods in $\mathcal{G}$, we include all sequences together to form a "corpus", in which each label is like a "word" and each sequence is like a "sentence". With this analogy in mind, we develop the Word2Vec model with objective perturbation and then utilize this model on the corpus to obtain node vector of each label. Finally, neighborhood encoding can be reaped by averaging the label vectors in the neighborhood.



(a) Initialization (b) First iteration (c) Second iteration

FIGURE 2. Illustration of neighborhood labeling.

Example 1: Figure 2 shows an example of the algorithm. It initializes node labels according to their distances to target individual u (the blue node), and collects multisets of their neighbor labels. Labeling-based encoding, two sequences $\langle 1\_1, 1\_2, 1\_2, 1\_3, 1\_3, 1\_4\rangle$ and $\langle 2\_1, 2\_2, 2\_2, 2\_3, 2\_3, 2\_4\rangle$ are generated, where each label, say 2_1, represents that a node is labeled with 1 in the second iteration. Then, it trains label representations based on the sequences to maximize preserving the connection probabilities between nodes.

Formally, given u_i, the probability of predicting v_j is defined by a softmax function, that is:

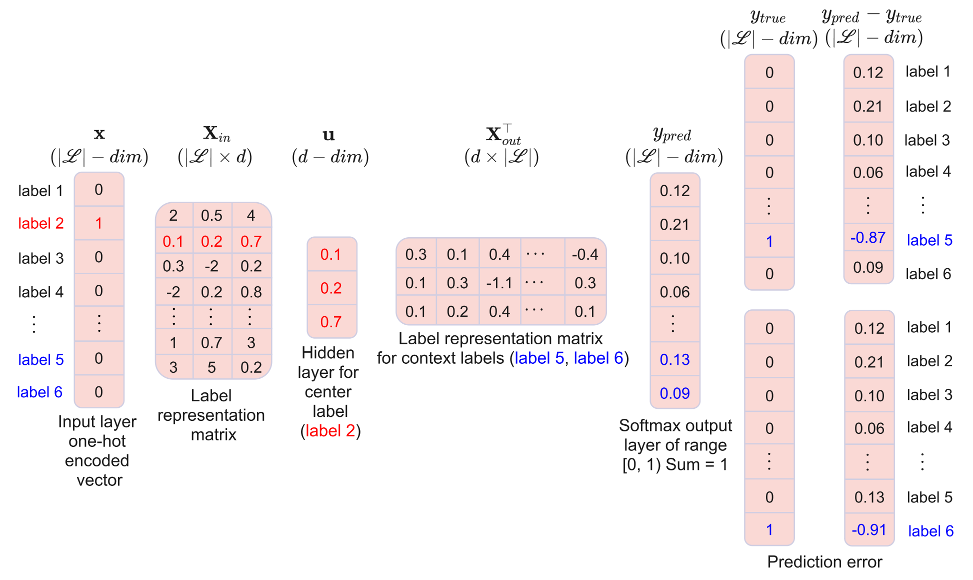
$$\mathbb{P}\left(v_j \mid u_i\right) = \frac{\exp\left(\mathbf{v}_j \cdot \mathbf{u}_i\right)}{\sum_{k=1}^{\mathscr{V}} \exp\left(\mathbf{v}_k \cdot \mathbf{u}_i\right)}, \tag{5}$$

where $\mathbf{v}_j$ and $\mathbf{u}_i$ are representation vectors of the nodes v_j and u_i, and the vocabulary size $\mathscr{V}$ is simply the number of all unique vocabs in the corpus (i.e., $\mathscr{V} = |\mathscr{L}|$). $\mathbf{v}_j$ and $\mathbf{u}_i$ are illustrated in Figure 3.

Calculating Equation (5) often results in unaffordable time and memory consumption since $\mathscr{V}$ is normally very large. To solve this problem, we approximate it using negative sampling proposed in Skip-gram (see Section IV-B). The idea is that instead of $\mathscr{V}$ times for every label in the vocabulary, we randomly select a relatively small number of vocabs in the corpus that are not contained in the target graph whose representation vector needs to be learned, to optimize the following objective function:

$$\min_{\mathbf{u},\mathbf{v}} \mathcal{L}(\mathbf{X}) = -\log\sigma\left(\mathbf{v}_{pos}\cdot\mathbf{u}\right) - \sum_{\mathbf{v}_{neg}\in\mathbf{X}_{neg}} \log\sigma\left(-\mathbf{v}_{neg}\cdot\mathbf{u}\right), \tag{6}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is a sigmoid function, $\mathbf{X}_{neg} = \{\mathbf{v}_{neg,j} | j = 1, \cdots, K\}$ is the negative sample collection, $\mathbf{v}_{neg}$ is a negative sample drawn from $\mathbf{X}_{neg}$ for K times, and $\mathbf{v}_j$ is the vector representation of v_j. We optimize Equation (6) using stochastic gradient descent where the gradients are derived



$\mathbf{x}$ $(|\mathscr{L}| - dim)$
$\mathbf{X}_{in}$ $(|\mathscr{L}| \times d)$
$\mathbf{u}$ $(d - dim)$
$\mathbf{X}_{out}^{\top}$ $(d \times |\mathscr{L}|)$
y_{pred} $(|\mathscr{L}| - dim)$
y_{true} $(|\mathscr{L}| - dim)$
$y_{pred} - y_{true}$ $(|\mathscr{L}| - dim)$
label 1
label 2
label 3
label 4
label 5
label 6
Input layer one-hot encoded vector
Label representation matrix
Hidden layer for center label (label 2)
Label representation matrix for context labels (label 5, label 6)
Softmax output layer of range [0, 1) Sum = 1
Prediction error

FIGURE 3. Neural network structure of Skip-gram. Current center label is "label 2".

as follows:

$$\frac{\partial \mathcal{L}(\mathbf{X})}{\partial \mathbf{v}_j} = \left(\sigma\left(\mathbf{v}_j \cdot \mathbf{u}\right) - t_j\right) \cdot \mathbf{u},$$
$$\frac{\partial \mathcal{L}(\mathbf{X})}{\partial \mathbf{u}} = \sum_{\mathbf{v}_j \in \{\mathbf{v}_{pos}\} \cup \mathbf{X}_{neg}} \left(\sigma\left(\mathbf{v}_j \cdot \mathbf{u}\right) - t_j\right) \cdot \mathbf{v}_j, \quad (7)$$

where $t_j = 1$ for positive labels ($\mathbf{v}_j = \mathbf{v}_{pos}$) and $t_j = 0$ for negative labels ($\mathbf{v}_j = \mathbf{v}_{neg} \in \mathbf{X}_{neg}$). $\mathbf{v}_j$ is the j-th label vector in the output matrix ($\mathbf{v}_j \in \mathbf{X}_{out}$).

After the learning process is completed, the representation vectors of two nodes u_i and u_j are close to each other if they are composed by similar subgraphs.

Remark 1: In Figure 3, a natural question is, why do we predict context labels? It is worth noting, here, that the ultimate goal of Skip-gram model is not to predict context labels, but to learn intelligent vector representation of labels. It just happens that predicting context labels inevitably results in good vector representations of labels, because of the neural network structure of Skip-gram. Neural network at its essence is just optimizing the weight matrix $\mathbf{X} = [\mathbf{X}_{in}\ \mathbf{X}_{out}]$ to correctly predict output. In Word2Vec Skip-gram, the weight matrix is, in fact, the vector representations of labels. Therefore, optimizing the weight matrix is equivalent to good vector representations of labels.

In fact, both $\mathbf{X}_{in}$ and $\mathbf{X}_{out}$ can be used as label vectors. However, $\mathbf{X}_{in}$ is usually used, because the function of $\mathbf{X}_{out}$ can be replaced by other network structures when used for some other downstream tasks, in which case, $\mathbf{X}_{out}$ does not exist. For scalability, in this paper, we utilize $\mathbf{X}_{in}$ as label vectors. After that, we encode u's nearest neighbor structure via averaging the label vectors in the neighbor structure, and then produce a numerical vector $\mathbf{x}_i^{nns}$ as its representation. For simplicity, the generated final node vectors $\{\mathbf{x}_i^{nns}\}_{i=1}^{|\mathcal{V}|}$ is represented as $\mathbf{M}$. In what follows, from a privacy perspective, we introduce in detail how to make $\mathbf{M}$ satisfy ε-edge differential privacy.

2) OBJECTIVE FUNCTION PERTURBATION

In this section, we aim to ensure that the generated $\mathbf{M}$ satisfies ε-edge differential privacy. It is worth mentioning, here, that the Theorem 2, a proof of which can be found in [23], allows us to move the burden of differential privacy to the $\mathbf{X}$; the differential privacy of the $\mathbf{M}$ will follow by the theorem.

To reflect our contribution, we first introduce two first-cut approaches to make $\mathbf{X}_{in}$ satisfy ε-differential privacy. The main ideas are to utilize random noise vectors drawn from Laplace distribution to perturb the gradient and the objective function. After that, we introduce our proposed method in details.

a: FIRST-CUT APPROACHES

To ensure that $\mathbf{X}_{in}$ satisfies differential privacy, a natural but naive approach is to inject noise into each gradient, that is

$$\mathbf{u} = \mathbf{u} - \eta \left[\sum \left(\sigma\left(\mathbf{v}_j \cdot \mathbf{u}\right) - t_j\right) \mathbf{v}_j + \mathscr{N}\right], \quad (8)$$

where $\mathscr{N} \sim Lap(\frac{S_f(\mathcal{G})}{\varepsilon/iternum})^d$ is a vector noise. In particular, for a fixed ε, as the number of iteration, *iternum*, increases, the privacy budget at each iteration decreases, and hence, the variation in a noisy gradient becomes larger, which makes the gradient optimization algorithm difficult to converge. Thus, this algorithm requires a large number of iterations to search the output results with high utility, but may lead to unpredictable error accumulation.

Some alternative solutions from [27], [28], and [29], which uses the objective perturbation to make sure that the final

Algorithm 2 Objective Perturbation

```
1:  for context in context-window do
2:      e = 0
3:      for each $v_j \in \{v_{pos}, v_{neg,1}, \cdots, v_{neg,K}\}$ do
4:          $q = \sigma\left(\mathbf{v}_j \cdot \mathbf{u}\right)$
5:          $g = \eta\left(q - t_j\right)$
6:          // add scalar Laplace noise
7:          $\mathbf{e} = \mathbf{e} + (g + \eta\mathscr{N})\mathbf{v}_j$
8:          $\mathbf{v}_j = \mathbf{v}_j - g\mathbf{u}$
9:      end for
10:     $\mathbf{u} = \mathbf{u} - \mathbf{e}$
11: end for
```

output satisfies differential privacy, can mitigate the shortcoming of gradient perturbation to a certain extent. Along the way, the differentially private objective function is shown as Equation (9).

$$\min_{\mathbf{u},\mathbf{v}} \widetilde{\mathcal{L}}(\mathbf{X}) = -\left[\log\sigma\left(\mathbf{v}_{pos}\cdot\mathbf{u}\right) + \mathscr{N}\cdot\mathbf{u}\right] - \sum_{\mathbf{v}_{neg}\in\mathbf{X}_{neg}} \log\sigma\left(-\mathbf{v}_{neg}\cdot\mathbf{u}\right), \tag{9}$$

where $\mathscr{N}$ is the Laplace noise vector corresponding to the vector $\mathbf{u}$. Then, the update rule about $\mathbf{u}$ is shown as Equation (10).

$$\mathbf{u} = \mathbf{u} - \eta\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j\right)\mathbf{v}_j + \mathscr{N}\right], \tag{10}$$

where $\mathscr{N} \sim Lap(\frac{S_f(\mathcal{G})}{\varepsilon})^d$. Although this method does not require partition for ε, it still puts a vector of noise into the learning process of $\mathbf{v}_j$, potentially resulting in poor utility.

b: OUR APPROACH

To cope with these issues, we change the Equation (9) to the following one for each edge (i, j) by injecting the Laplace noise.

$$\widetilde{\mathcal{L}}(\mathbf{X}) = -\left[\log\sigma\left(\mathbf{v}_{pos}\cdot\mathbf{u}\right) + \mathscr{N}\mathbf{v}_{pos}\cdot\mathbf{u}\right] - \sum_{\mathbf{v}_{neg}\in\mathbf{X}_{neg}} \log\sigma\left(-\mathbf{v}_{neg}\cdot\mathbf{u}\right), \tag{11}$$

where $\mathscr{N}$ is a scalar noise drawn from Laplace distribution, that is $\mathscr{N} \sim Lap(S_f(\mathcal{G})/\varepsilon)$.

The proposed objective perturbation has two benefits: (1) regardless of the number of iterations, the overall error is controllable, thereby avoiding the unpredictable error accumulation, and (2) instead of the vector noise injection, it only requires a scalar noise and thus can balance the related features retained against the noise injected. The optimization process with differential privacy is presented in Algorithm 2.

Remark 2: In the line 5 of Algorithm 2, $\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j$ is called a prediction error. Recall that negative sampling attempts to maximize the probability of observing positive pairs $\mathbb{P}(v_{pos}|u) \to 1$ while minimizing the probability of observing negative pairs ı$(v_{neg}|u) \to 0$. If good label vectors are learned, $\sigma(\mathbf{v}_{pos}\cdot\mathbf{u}) \approx 1$ for positive pairs, and $\sigma(\mathbf{v}_{neg}\cdot\mathbf{u}) \approx 0$ for negative pairs. The prediction error will gradually approach zero $\sigma\left(\mathbf{v}_{pos}\cdot\mathbf{u}\right) - t_j \approx 0$, as the model iterates through the training samples (positive pairs) and optimizes the weights. It is easy to observe that missing one edge will influence the t_j that is directly related to the calculation of global sensitivity. Thus, we can derive the following theorem.

Theorem 3: If each $\mathscr{N} \sim Lap(\frac{S_f(\mathcal{G})}{\varepsilon})$, then the obtained $\mathbf{X}_{in}$ obeys ε-edge differential privacy.

Proof: Let $\mathcal{G}$ and $\mathcal{G}'$ be two neighboring graphs which differ in a single edge.

$$\begin{aligned}
&\frac{\mathbb{P}\left[\mathbf{X}_{in} = \flat \mid \mathcal{G}\right]}{\mathbb{P}\left[\mathbf{X}_{in} = \flat \mid \mathcal{G}'\right]} \\
&= \frac{\mathbb{P}\left[\mathbf{u}_1 = \flat_1, \mathbf{u}_2 = \flat_2, \ldots, \mathbf{u}_{|\mathscr{L}|} = \flat_{|\mathscr{L}|} \mid \mathcal{G}\right]}{\mathbb{P}\left[\mathbf{u}_1 = \flat_1, \mathbf{u}_2 = \flat_2, \ldots, \mathbf{x}_{|\mathscr{L}|} = \flat_{|\mathscr{L}|} \mid \mathcal{G}'\right]} \\
&= \frac{\prod_{i=1}^{|\mathscr{L}|}\mathbb{P}\left[\mathbf{u}_i = \flat_i \mid \mathcal{G}\right]}{\prod_{i=1}^{|\mathscr{L}|}\mathbb{P}\left[\mathbf{u}_i = \flat_i \mid \mathcal{G}'\right]} = \frac{\mathbb{P}\left[\mathbf{u}_x = \flat_x \mid \mathcal{G}\right]}{\mathbb{P}\left[\mathbf{u}_x = \flat_x \mid \mathcal{G}'\right]} \\
&= \frac{\mathbb{P}\left[\mathbf{u}_x - \eta\cdot\sum\left[\left(\sigma(\mathbf{v}_j\cdot\mathbf{u}) - t_j\right)\cdot\mathbf{v}_j + \mathscr{N}\mathbf{v}_j\right] = \flat_x \mid \mathcal{G}\right]}{\mathbb{P}\left[\mathbf{u}_x - \eta\cdot\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j'\right)\cdot\mathbf{v}_j + \mathscr{N}\mathbf{v}_j\right] = \flat_x \mid \mathcal{G}'\right]} \\
&= \frac{\mathbb{P}[\sum(\eta\mathscr{N}\mathbf{v}_j) = \mathbf{u}_x - \flat_x - \eta\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j\right)\cdot\mathbf{v}_j\right] \mid \mathcal{G}]}{\mathbb{P}[\sum(\eta\mathscr{N}\mathbf{v}_j) = \mathbf{u}_x - \flat_x - \eta\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j'\right)\cdot\mathbf{v}_j\right] \mid \mathcal{G}']} \\
&\overset{(1)}{=} \frac{\mathbb{P}[\mathscr{N} = \overbrace{\frac{(\mathbf{u}_x - \flat_x)\,\eta^{-1}}{\sum\mathbf{v}_j} - \frac{\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j\right)\cdot\mathbf{v}_j\right]}{\sum\mathbf{v}_j}}^{f(\mathcal{G})} \mid \mathcal{G}]}{\mathbb{P}[\mathscr{N} = \underbrace{\frac{(\mathbf{u}_x - \flat_x)\,\eta^{-1}}{\sum\mathbf{v}_j} - \frac{\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j'\right)\cdot\mathbf{v}_j\right]}{\sum\mathbf{v}_j}}_{f(\mathcal{G}')} \mid \mathcal{G}']} \\
&\le e^{\frac{\varepsilon|f(\mathcal{G}) - f(\mathcal{G}')|}{S_f(\mathcal{G})}} \le e^{\varepsilon},
\end{aligned}$$

where step (1) holds because the noise $\mathscr{N}$ is taken as the same value for each iteration (lines 3-9 in Algorithm 2).

In what follows, we show the upper bound of $S_f(\mathcal{G}) = \max|f(\mathcal{G}) - f(\mathcal{G}')|$, which will be utilized for injecting noise into the objective function.

Lemma 1: The global sensitivity of the utility function f, $S_f(\mathcal{G})$, is $\max|f(\mathcal{G}) - f(\mathcal{G}')| \le 1$.

Proof: $S_f(\mathcal{G})$ is the maximum change in $f(\cdot)$ in the output space if one edge is missing. We have:

$$\begin{aligned}
&|f(\mathcal{G}) - f(\mathcal{G}')| \\
&= \left\|\frac{\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j\right)\cdot\mathbf{v}_j\right]}{\sum\mathbf{v}_j} - \frac{\sum\left[\left(\sigma\left(\mathbf{v}_j\cdot\mathbf{u}\right) - t_j'\right)\cdot\mathbf{v}_j\right]}{\sum\mathbf{v}_j}\right\|_2 \\
&\le 1,
\end{aligned}$$

where the last inequality follows since $t_j' - t_j \le 1$.

D. SYNTHETIC GRAPH GENERATION

Recall that the $\mathbf{M} = \{\mathbf{x}_i^{nns}\}_{i=1}^{|\mathcal{V}|}$ is able to preserve node nearest neighbor structure because $\mathbf{x}_i^{nns}$ is reaped by averaging the label vectors in the neighbor structure. Moreover, if two nodes share many common neighbors in social graphs, they tend to be similar. This assumption has proved to be reasonable in many fields [30], [31]. Thus, we are able to capture the degree of correlation between nodes based on $\mathbf{M}$. In this paper, a unified model for learning a graph in [32] is employed to synthesize social graphs, that reads as follows:

$$\min_{\mathbf{W}\in\mathcal{W}} \|\mathbf{W}\circ\mathbf{Z}\|_{1,1} - \alpha\mathbf{1}^\top \log(\mathbf{W}\mathbf{1}) + \frac{\beta}{2}\|\mathbf{W}\|_F^2, \tag{12}$$

where $\mathbf{Z}_{ij} = \left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2$, $\|A\|_{1,1}$ is the elementwise norm-1 of A, $\circ$ denotes the Hadamard product. The first term of Equation (12) is equal to $\operatorname{tr}\left(\mathbf{M}^\top\mathbf{L}\mathbf{M}\right)$, where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian, $\mathbf{D} = \operatorname{diag}(\mathbf{W}\mathbf{1})$, $\mathbf{1} = [1, \ldots, 1]^\top$, and $\mathbf{M} \in \mathbb{R}^{|\mathcal{V}|\times d} = [\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{V}|}]^\top$. The optimization is over the set $\mathcal{W}$ of valid adjacency matrices (non-negative, symmetric, with zero diagonal).

Remark 3: The smoothness term is a weighted ℓ-1 norm of $\mathbf{W}$, encoding weighted sparsity, that penalizes edges connecting distant rows of $\mathbf{M}$. The interpretation is that when the given distances come from a smooth manifold, the corresponding graph has a sparse set of edges, preferring only the ones associated to small distances in $\mathbf{Z}$. In fact, as stated in [32], the separate rows of $\mathbf{M}$ do not have to be smooth signals. Two non-smooth signals $\mathbf{x}_i$, $\mathbf{x}_j$ can have a small distance between them, and therefore a small entry $\mathbf{Z}_{ij}$.
According to [32], the problem can be rewritten as a sum of three functions in order to fit it to primal dual algorithms reviewed by [33]. The general objective form is

$$\min_{\mathbf{w}} f_1(\mathbf{w}) + f_2(\xi\mathbf{w}) + f_3(\mathbf{w}), \tag{13}$$

with

$$\begin{aligned} f_1(\mathbf{w}) &= \mathbb{I}\{\mathbf{w} \geq 0\} + 2\mathbf{w}^\top\mathbf{z}, \\ f_2(\mathbf{v}) &= -\alpha\mathbf{1}^\top \log(\mathbf{v}), \\ f_3(\mathbf{w}) &= \beta\|\mathbf{w}\|^2, \textit{with } \zeta = 2\beta, \end{aligned}$$

in which $\mathbb{I}\{condition\} = 0$ if condition holds, ∞ otherwise, f_2 is defined on the dual variable $\mathbf{v} = \xi\mathbf{w}$ where ξ is the linear operator that satisfies $\mathbf{W}1 = \xi\mathbf{w}$, and ζ is the Lipschitz constant of f_3.

E. PRIVACY AND COMPLEXITY ANALYSIS

1) PRIVACY ANALYSIS

In this section, we formally prove that the yielded synthetic graph obeys ε-edge differential privacy.

Theorem 4: Priv-NNS algorithm satisfies ε-edge differential privacy.

Proof: As we proved in Theorem 3, the node vectors represented as matrix $\mathbf{X}_{in}$ maintain bounded ε-edge differential privacy. Recall that the postprocessing property (see Theorem 2) of differential privacy states that any operation performed on the output of a differential privacy-based algorithm, without accessing the raw data, remains differential privacy with the same level of privacy. Therefore, Priv-NNS algorithm obeys ε-edge differential privacy.

2) COMPLEXITY ANALYSIS

Here we analyze the computational complexity of each step of Priv-NNS.

(1) *Subgraph labeling:* The time complexity of this algorithm is $\mathcal{O}(iter|\mathcal{V}|(m_h+n_h))$ where m_h and n_h are respectively edge number and node number in a neighborhood with parameter h and $iter$ is the number of iterations in neighborhood labeling.

(2) *Private graph encoding:* The time complexity of Skip-gram with negative sampling is $\mathcal{O}(K|\mathcal{V}|)$.

(3) *Synthetic graph generation:* In graph generation, all $\binom{|\mathcal{V}|}{2}$ possible edges between $|\mathcal{V}|$ nodes are considered, that results in a cost of at least $\mathcal{O}(|\mathcal{V}|^2)$ computations per iteration. After T iterations, the time complexity of synthesizing graphs is $\mathcal{O}(T|\mathcal{V}|^2)$.

Thus the total complexity is $\mathcal{O}(iter|\mathcal{V}|(m_h + n_h) + K|\mathcal{V}| + T|\mathcal{V}|^2)$.

VI. EXPERIMENTS

Because node nearest neighbor structure is the cornerstone in many graph analysis applications, such as community detection [1] and influence maximization [2]. Hence, in this section we investigate the neighbor structure preservation of synthetic graphs by two application utility metrics: community detection and influence maximization. Moreover, six basic characteristics in social graphs, namely average degree, power law exponent, triangle count, characteristic path length and clustering coefficient, are employed to further confirm Priv-NNS's performance.

A. EXPERIMENTAL SETUP

1) DATASETS

To comprehensively evaluate our proposed method, we conduct extensive experiments on the following six real graph datasets from different application domains:

- Arenas Email [34]: This dataset is a communication network with 1,133 nodes and 5,451 edges.
- Cora [35]: It is a typical paper citation network with 2,211 nodes and 5,001 edges.
- PPI [36]: It is a protein-protein interaction network with 3,890 nodes and 38,739 edges.
- Powergrid[1]: This dataset is an electrical grid of western US with 4,941 nodes and 6,594 edges.
- p2p-Gnutella[2]: This dataset is an internet peer-to-peer network with 6,301 nodes and 20,777 edges.
- DBLP [37]: This dataset is a collaboration network with 12,591 nodes and 49,743 edges.

2) COMPARED METHODS

We compare ours with the Original (non-private Priv-NNS), and five differentially private graph publishing algorithms, namely two well-known algorithms dK-2 [6], HRG [5], and three recently proposed algorithm PHDP [19],

[1]http://konect.uni-koblenz.de/
[2]http://snap.stanford.edu/data/index.html

DPGGAN [38] and DPGVAE [38]. It is worth mentioning, here, that the dK-2 proposed in [4] is based on local sensitivity, and thus does not provide differential privacy. Hence, we compare with an improved scheme [6] under a more relaxed privacy notion, that is, (ε, δ)-differential privacy. Such scheme further improves the work of Sala et al. [4] by considering global sensitivity instead of local sensitivity.

3) PARAMETER SETTINGS

For all datasets, we vary the privacy budget $\varepsilon \in \{0.1, 0.2, 0.4, 0.8, 1.6, 3.2\}$ for all the private methods. Regarding dK-2, it uses the smooth sensitivity [12] to reduce large noise injection. Adding noise based on smooth sensitivity requires two necessary parameters φ and ψ related to ε and δ in which $\varphi = \frac{\varepsilon}{2}$ and $\psi = \frac{\varepsilon}{4(d+\ln(2/\delta))}$. Following [6], we keep $\delta = 0.01$ for dK-2. With regard to HRG, it consumes privacy budgets in sampling the dendrogram and computing noisy connection probabilities. With a fixed privacy budget ε, we assign $\varepsilon_1 = 0.5\varepsilon$ for sampling the dendrogram and $\varepsilon_2 = 0.5\varepsilon$ for computing noisy connection probabilities, because this allocation has been experimentally demonstrated in [5], which can achieve high utility across different utility metrics. As for Priv-NNS, the value of parameter h is set as 2, the number of epochs to train the neural network $\mathfrak{e}$ is set as 1.5×10^3, and the negative sampling K is set as 5, which has been proved experimentally in [24] that this value is applicable to both small datasets and large datasets. Also, the dimension d of representations is set as 128 for all the datasets and the learning rate is $\eta = 1 \times 10^{-3}$. For graph reconstruction, we set the threshold $\alpha = \beta = 1$ where the settings theoretically find to work well in [32], and set maximum number of iterations $T = 4 \times 10^3$ for optimizing the objective function in Eq. (13). For DPGGAN and DPGVAE, their parameter settings follow those in the original models.

4) GRAPH STATISTICS

To demonstrate the connection between preserving nearest neighbor structure and graph utility, the performance of published graph under utility metrics is compared. As shown in Table 2, the evaluation includes two application utility metrics, the community detection and the influence maximization, and six graph utility metrics.

Here $\mathcal{C}$ is the community partition of $\mathcal{G}$. ℓ_C is the number of edges between the nodes in C and d_C is the sum of degrees of the nodes in C. Given a diffusion model $\mathcal{M}$ and a positive integer k, influence maximization selects a set S^* of k users from $\mathcal{V}$ as the seed set to maximize the influence spread $\sigma_{\mathcal{G},\mathcal{M}}(S^*)$, i.e., $\sigma_{\mathcal{G},\mathcal{M}}(S^*) = \arg\max_{S\subseteq\mathcal{V}\wedge|S|\leq k} \sigma_{\mathcal{G},\mathcal{M}}(S)$. $d_{(v)}$ indicates the degree of node v and $d_{\min}$ denotes the minimum degree in a graph. $\hat{d}$ is the sorted list of degrees in the graph. $n_\triangle$ is the number of triangles in $\mathcal{G}$ and n_W is the number of wedges (i.e., length two paths) in $\mathcal{G}$.

5) EVALUATION METRIC

We compare the performance of Priv-NNS to four baselines on eight utility metrics over all datasets. We measure the accuracy of each method by the mean relative error (MRE) [39], which is given as follows:

TABLE 2. Graph statistics used to measure graph properties.

Metric name	Computation
Community detection	$\sum_{C\in\mathcal{C}} \left(\frac{\ell_C}{m} - \left(\frac{d_C}{2m}\right)^2\right)$
Influence maximization	$\arg\max_{S\subseteq\mathcal{V}\wedge\|S\|\leq k} \sigma_{\mathcal{G},\mathcal{M}}(S)$
Average degree	$\frac{\sum_{v\in\mathcal{V}} d(v)}{\|\mathcal{V}\|}$
Power law exponent	$1 + n\left(\sum_{v\in\mathcal{V}} \log \frac{d(v)}{d_{\min}}\right)^{-1}$
Triangle count	$\frac{\|\{\{u,v,w\}\|\{(u,v),(v,w),(u,w)\}\subseteq\mathcal{E}\}\|}{6}$
Characteristic path length	$\frac{1}{\|\mathcal{V}\|(\|\mathcal{V}\|-1)} \sum_{u\neq v} d(u,v)$
Clustering coefficient	$\frac{3\times n_\triangle}{n_W}$

$$\text{MRE} = \frac{1}{|\mathcal{A}|} \sum_{\mathcal{A}_i\in\mathcal{A}} \left|\hat{\mathcal{A}}_i(\mathcal{G}) - \mathcal{A}_i(\mathcal{G})\right| / |\mathcal{A}_i(\mathcal{G})|,$$

where $\mathcal{A}_i(\mathcal{G})$ is the true aggregation result for one query, and $\hat{\mathcal{A}}_i(\mathcal{G})$ is the differentially private aggregation result. A low MRE indicates a low error and, thus, a better data utility. For each result reported, we repeat each experiment 50 times to get the average values.

B. UTILITY ANALYSIS

1) COMMUNITY DETECTION

Community detection technology, which aims to identify subgraphs with dense internal connections compared to external connections, has received significant attention. In our experiments, we utilize the state-of-the-art community detection algorithm Louvain [40] to obtain the optimal partition. We then evaluate the similarity of the community structures found by Louvain in the original graph $\mathcal{G}$ and the perturbed graph $\mathcal{G}'$ using the *modularity* score. From the analysis of Figure 4, the key observation is that the graphs sampled from our Priv-NNS consistently exhibit the most similar community structure to their corresponding original graphs. This can be attributed to the fact that Priv-NNS preserves the neighbor structure by optimizing the utility-privacy tradeoff. In contrast, the synthetic graphs generated by all the compared methods for the six datasets suffer from a more pronounced degradation as the amount of added noise increases with the number of nodes.

2) INFLUENCE MAXIMIZATION

Influence maximization is a crucial algorithmic problem in social influence analysis, where a set of k users (known as seed set) is selected from a social network to maximize the expected number of influenced users (called influence spread). In our evaluation, we employ a cost-effective lazy forward algorithm [41] based on the independent cascade model [42] to select ten seed nodes with the highest information broadcasting ability. We then compare the percentage of influenced users across different anonymized graphs, all having the same propagation probability of 0.1. Figure 5 illustrates the Mean Relative Error (MRE) of the expected spread by seed set size. This experiment demonstrates that

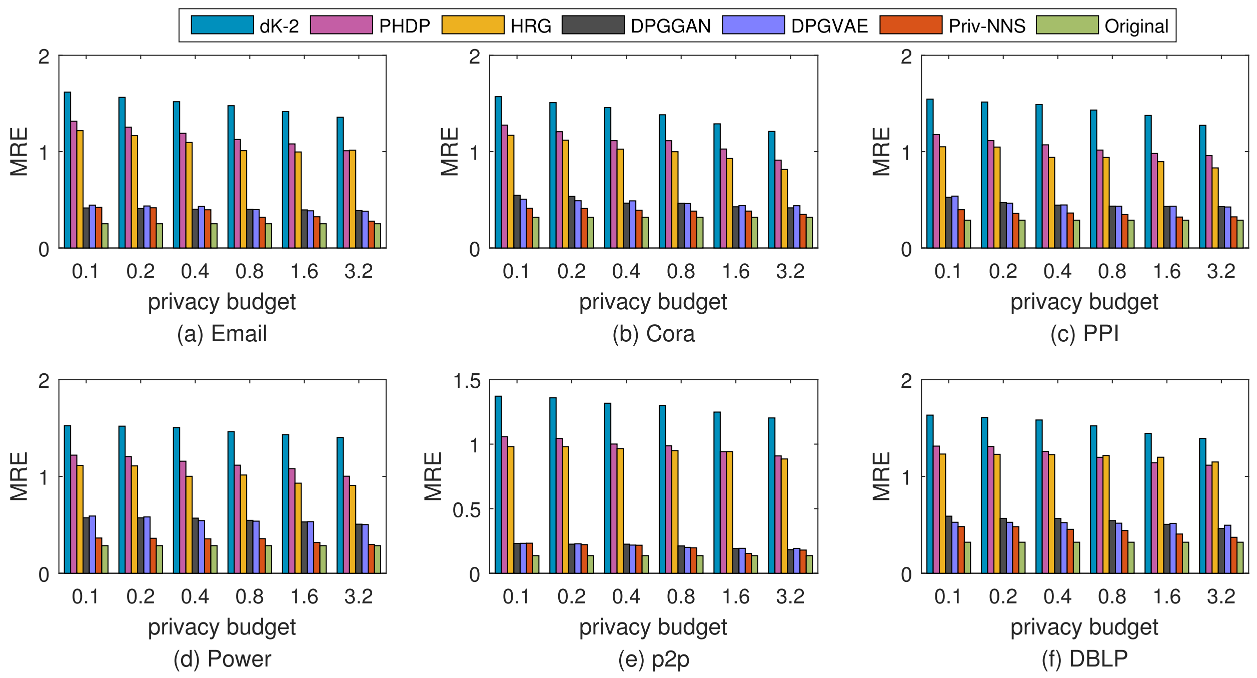


FIGURE 4. Comparison of differential privacy-based methods in terms of MRE of community detection.

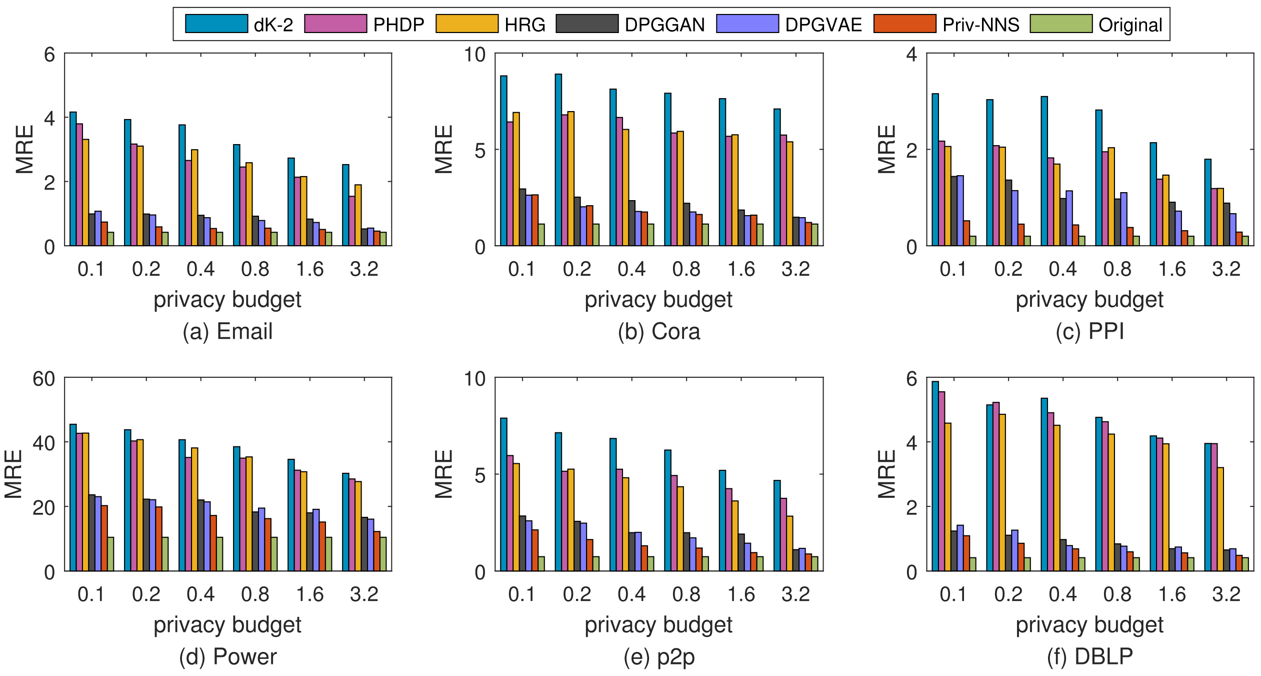


FIGURE 5. Comparison of differential privacy-based methods in terms of MRE of influence maximization.

compared to dK-2, PHDP, and HRG, the synthetic graphs generated by Priv-NNS, DPGGAN, and DPGVAE can more effectively simulate the information broadcasting capability of the original social graphs. Additionally, when compared to DPGGAN and DPGVAE, our proposed algorithm yields superior results as expected. Since the problem of influence maximization is closely related to recommendation and advertising applications, the synthetic graphs produced by Priv-NNS hold high practical value.

3) AVERAGE DEGREE

The degree of a node in the network refers to the number of edges incident to the node, which is one of the most fundamental characteristics of a graph. The MRE of average degree

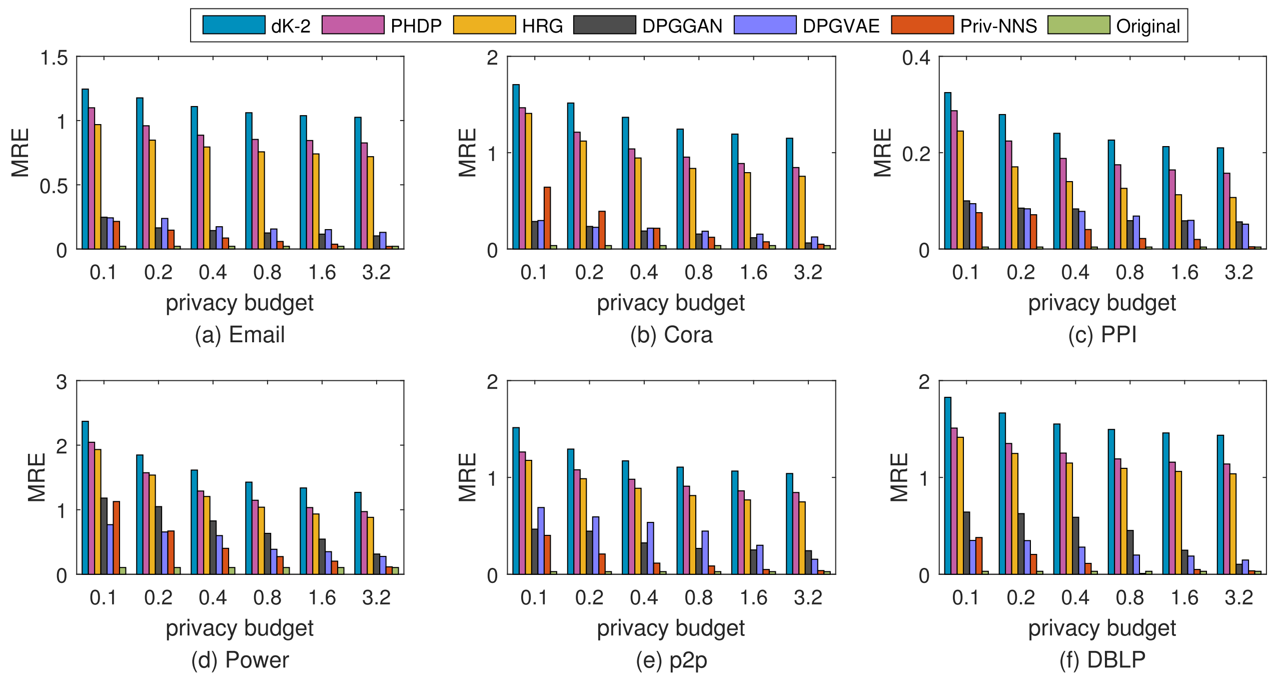


FIGURE 6. **Comparison of differential privacy-based methods in terms of MRE of average degree.**

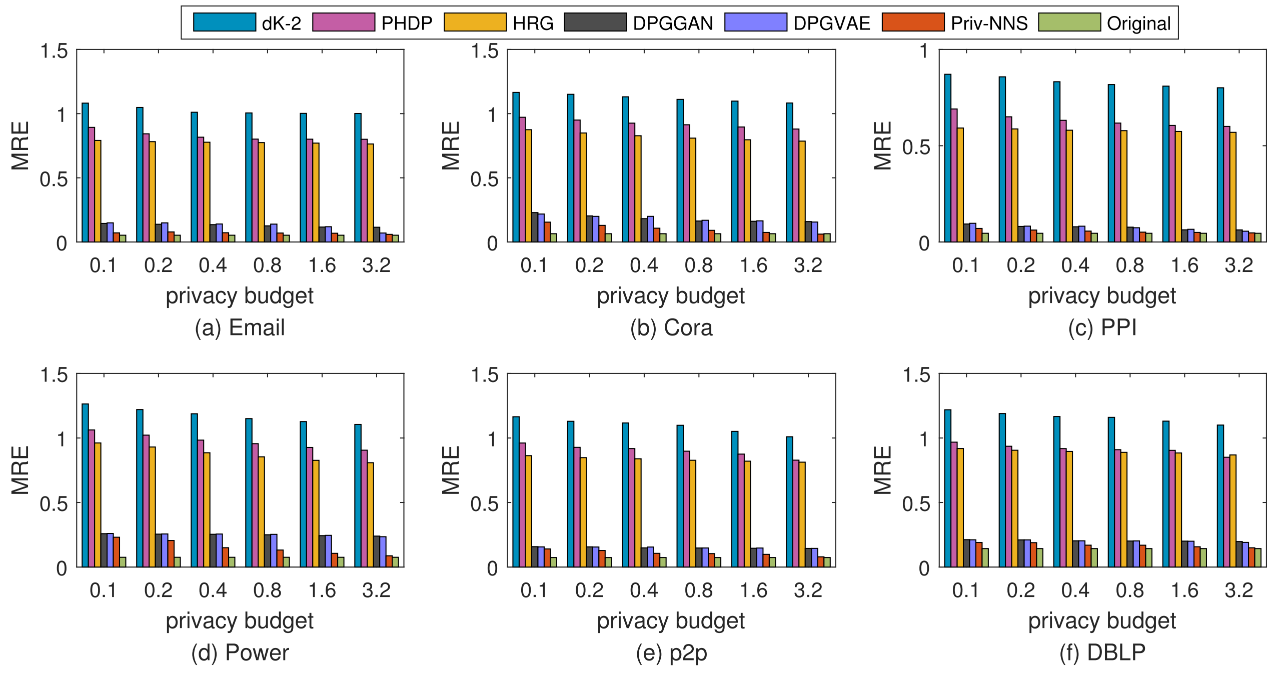


FIGURE 7. **Comparison of differential privacy-based methods in terms of MRE of power law exponent.**

of all the methods on all the datasets are depicted in Figure 6 while the privacy budget ε varies from 0.1 to 3.2. In summary, Priv-NNS achieves good accuracy on average degree over all datasets. When privacy budget is relatively large, e.g., $\varepsilon = 3.2$, its MRE always stays below 0.17. With the decrease of ε, the accuracy drops but it is still smaller than 0.8 on Arenas Email, Cora, PPI and p2p-Gnutella even when $\varepsilon = 0.1$. Compared with the other private methods, Priv-NNS clearly outperforms them in all cases, simply because it preserves the neighbor structure from an optimization perspective that provides more regulation space for the utility-privacy tradeoff. Also, the quality of results from dK-2, PHDP and HRG tends to be rather poor due to the large noise injection required to obey edge-differential privacy.

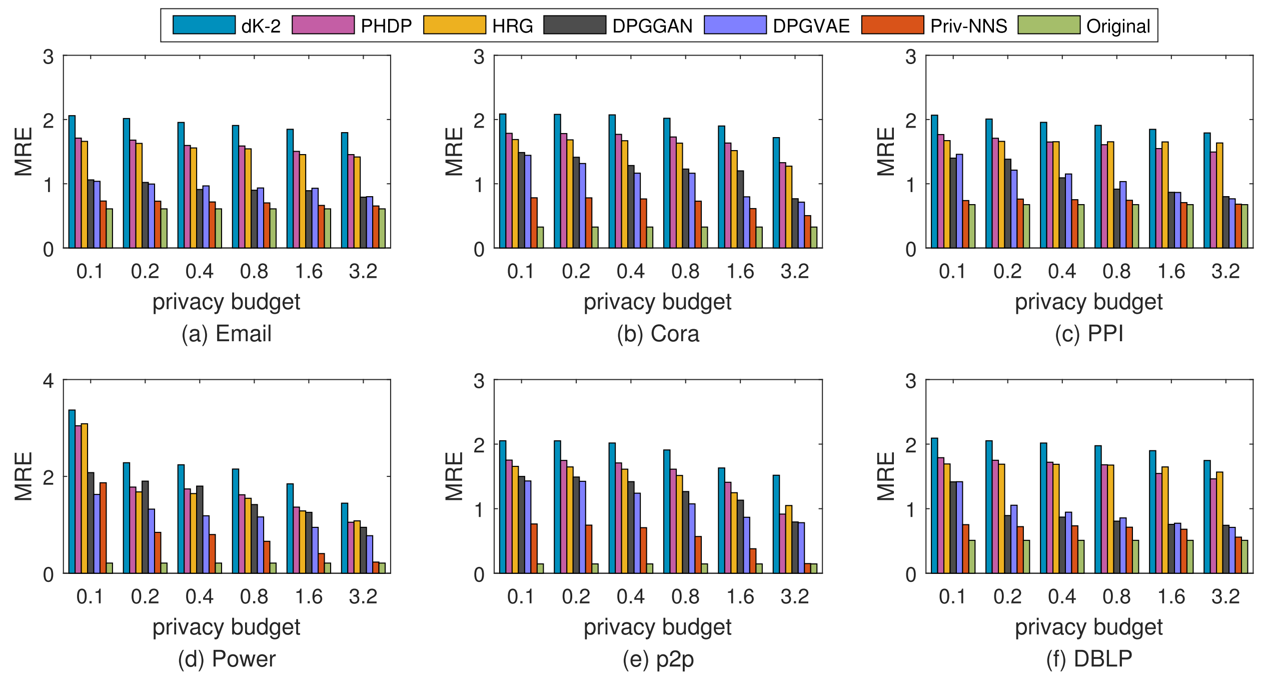


FIGURE 8. Comparison of differential privacy-based methods in terms of MRE of triangle count.

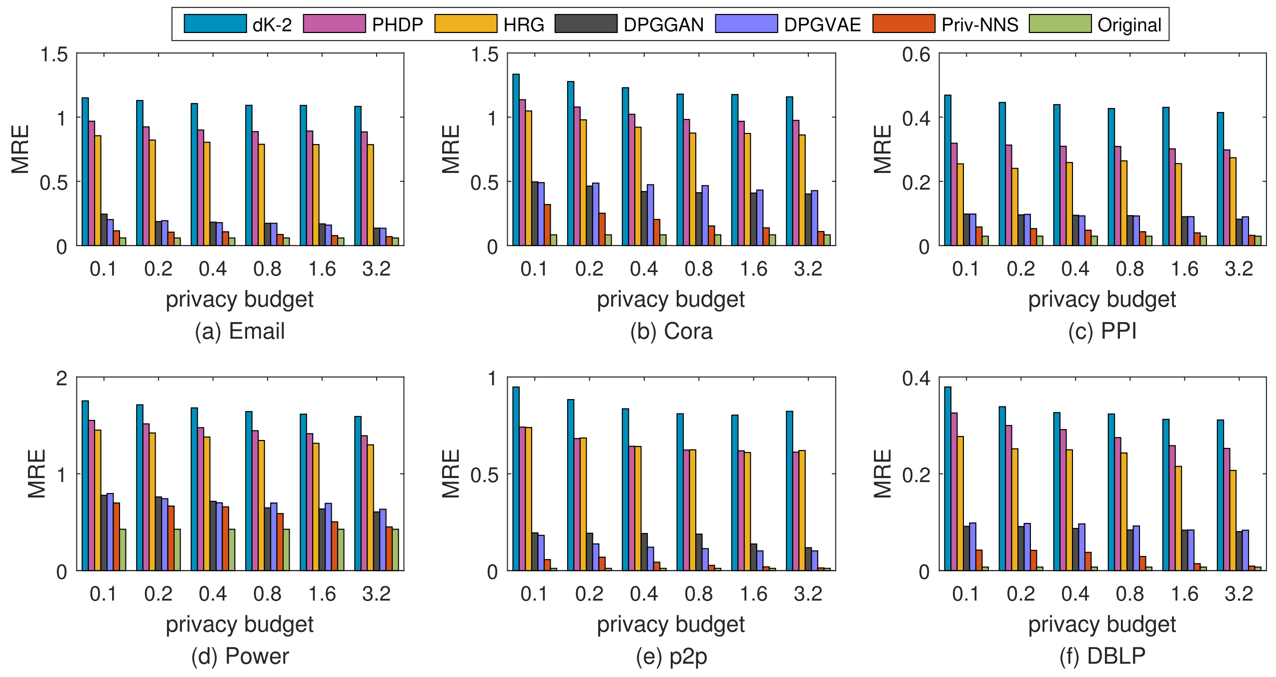


FIGURE 9. Comparison of differential privacy-based methods in terms of MRE of characteristic path length.

4) POWER LAW EXPONENT

Next we evaluate different methods for power law exponent in Figure 7. Priv-NNS keeps returning extremely accurate results for large ε, and reasonably good results for small ε. Meanwhile, it is still the best solution with an ε-differential privacy guarantee in all settings. The general trend in the results of power law exponent is the same as that of average degree across datasets and differentially private approaches. Empirically, we can conclude that preserving the neighbor structure between nodes is the basis of retaining statistical data utilities.

5) TRIANGLE COUNT

Figure 8 presents the results of triangle count. Our proposed approach Priv-NNS once again achieves good accuracy over all datasets, and consistently outperforms dK-2, PHDP and

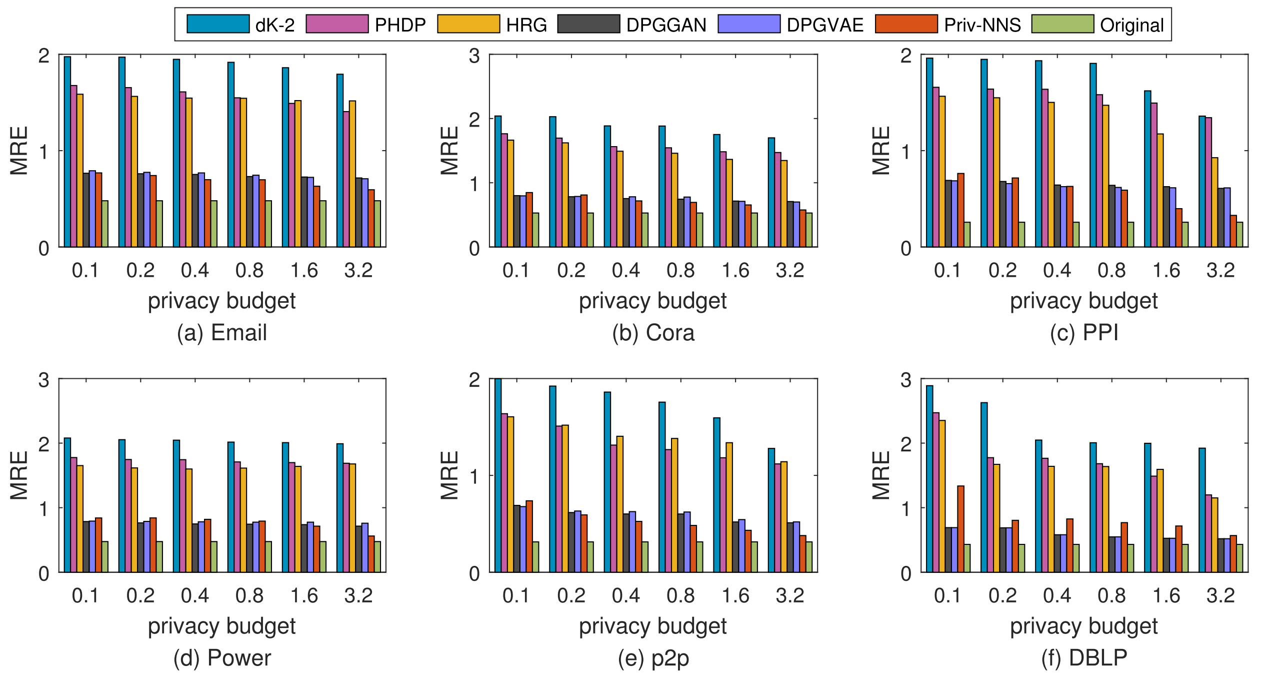


FIGURE 10. **Comparison of differential privacy-based methods in terms of MRE of clustering coefficient.**

HRG. In particular, from the analysis of Figure 8(c), it can be observed that Priv-NNS is rather close to Original (non-private Priv-NNS) even when $\varepsilon \leq 0.2$ on PPI. Furthermore, Figure 8(d) shows that the gap between Priv-NNS and Original begins to reduce when ε increases. It is worth mentioning here that when $\varepsilon = 3.2$, our proposed Priv-NNS outperforms Original and the reason is because of the use of private Skip-gram.

6) CHARACTERISTIC PATH LENGTH

The characteristic path length is defined as the average number of edges in the shortest paths between all node pairs. Figure 9 illustrates the characteristic path length in all the datasets. The superiority of Priv-NNS is two-fold. First, it is more accurate than dK-2, PHDP, HRG, DPGGAN and DPGVAE when ε varies from 0.1 to 3.2 in most cases. Second, it maintains the neighbor structure from an optimization perspective that provides more regulation space for the utility-privacy tradeoff. In summary, Priv-NNS is a more preferable solution for private characteristic path length queries.

7) CLUSTERING COEFFICIENT

The global clustering coefficient of a graph measures the proportion of wedges, that is, paths of length 2, that are embedded in triangles. As Figure 10 shows, differentially private synthetic graphs generated by our Priv-NNS suffer a reasonably low degradation with respect to their counterparts, in terms of their ability to preserve clustering coefficient of the original graphs. The main reason is because within Priv-NNS, the objective perturbation based graph encoding procedure can generate noisy node vectors while preserving high utility on the neighbor structure. Moreover, since the objective function in Equation (13) is proper, convex, and lower semi-continuous, it guarantees that the graph reconstructing algorithm converges to the minimum.

VII. CONCLUSION

In this paper, we have investigated how to publish graphs with edge-differential privacy while preserving node nearest neighbor structure. The underlying highlights are summarized as follows. To yield node vectors that meet differential privacy simultaneously preserving the neighbor structure, we set up a private graph encoding approach with structure-awareness, which learns topological features of the decomposed subgraph by maximizing the co-occurrence probability among nodes. During this process, a novel objective perturbation approach with a random term, which only requires a scalar noise rather than a vector noise, is devised to balance the neighbor structure retained against the noise added. Finally, a synthetic graph can be reconstructed by calculating distances between nodes in the noisy vectors. Formal privacy analysis and simulation results verify that the released graph by Priv-NNS preserves high utility on node nearest neighbor structure while satisfying ε-edge differential privacy.

Despite its success, Priv-NNS only achieves the weak edge-differential privacy. As further work, we are going to develop a strict node-differentially private graph release algorithm to match the structural properties of the original graph as closely as possible while maintaining a higher level of privacy.

REFERENCES

[1] F. Cheng, C. Wang, X. Zhang, and Y. Yang, “A local-neighborhood information based overlapping community detection algorithm for large-scale complex networks,” *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 543–556, Apr. 2021.

[2] Y. Li, J. Fan, Y. Wang, and K.-L. Tan, ''Influence maximization on social graphs: A survey,'' *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 10, pp. 1852–1872, Oct. 2018.

[3] C. Dwork, F. McSherry, K. Nissim, and A. Smith, ''Calibrating noise to sensitivity in private data analysis,'' in *Proc. Theory Cryptogr. Conf.*, 2006, pp. 265–284.

[4] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, ''Sharing graphs using differentially private graph models,'' in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf.*, Nov. 2011, pp. 81–98.

[5] Q. Xiao, R. Chen, and K.-L. Tan, ''Differentially private network data release via structural inference,'' in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 911–920.

[6] Y. Wang and X. Wu, ''Preserving differential privacy in degree-correlation based graph generation,'' *Trans. Data Privacy*, vol. 6, no. 2, p. 127, 2013.

[7] A. Clauset, C. Moore, and M. E. J. Newman, ''Hierarchical structure and the prediction of missing links in networks,'' *Nature*, vol. 453, no. 7191, pp. 98–101, May 2008.

[8] J. Pei, K. Zhong, Z. Yu, L. Wang, and K. Lakshmanna, ''Scene graph semantic inference for image and text matching,'' *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 22, no. 5, pp. 1–23, May 2023.

[9] A. Kavitha, V. B. Reddy, N. Singh, V. K. Gunjan, K. Lakshmanna, A. A. Khan, and C. Wechtaisong, ''Security in IoT mesh networks based on trust similarity,'' *IEEE Access*, vol. 10, pp. 121712–121724, 2022.

[10] M. Hay, C. Li, G. Miklau, and D. Jensen, ''Accurate estimation of the degree distribution of private networks,'' in *Proc. 9th IEEE Int. Conf. Data Mining*, Dec. 2009, pp. 169–178.

[11] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, ''Private release of graph statistics using ladder functions,'' in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, May 2015, pp. 731–745.

[12] K. Nissim, S. Raskhodnikova, and A. Smith, ''Smooth sensitivity and sampling in private data analysis,'' in *Proc. 39th Annu. ACM Symp. Theory Comput.*, Jun. 2007, pp. 75–84.

[13] F. McSherry and K. Talwar, ''Mechanism design via differential privacy,'' in *Proc. 48th Annu. IEEE Symp. Found. Comput. Sci.*, Oct. 2007, pp. 94–103.

[14] E. Shen and T. Yu, ''Mining frequent graph patterns with differential privacy,'' in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2013, pp. 545–553.

[15] F. Ahmed, A. X. Liu, and R. Jin, ''Publishing social network graph eigenspectrum with privacy guarantees,'' *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 2, pp. 892–906, Apr. 2020.

[16] D. J. Mir and R. N. Wright, ''A differentially private graph estimator,'' in *Proc. IEEE Int. Conf. Data Mining Workshops*, Dec. 2009, pp. 122–129.

[17] R. Chen, B. C. M. Fung, P. S. Yu, and B. C. Desai, ''Correlated network data publication via differential privacy,'' *VLDB J.*, vol. 23, no. 4, pp. 653–676, Aug. 2014.

[18] D. Proserpio, S. Goldberg, and F. McSherry, ''Calibrating data to sensitivity in private data analysis,'' *Proc. VLDB Endowment*, vol. 7, no. 8, pp. 637–648, Apr. 2014.

[19] T. Gao and F. Li, ''PHDP: Preserving persistent homology in differentially private graph publications,'' in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2019, pp. 2242–2250.

[20] R. Arora and J. Upadhyay, ''On differentially private graph sparsification and applications,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 13399–13410.

[21] W. Q. Su, X. Guo, and H. Zhang, ''Differentially private precision matrix estimation,'' *Acta Math. Sinica, English Ser.*, vol. 36, no. 10, pp. 1107–1124, Oct. 2020.

[22] M. Eliáš, M. Kapralov, J. Kulkarni, and Y. T. Lee, ''Differentially private release of synthetic graphs,'' in *Proc. 14th Annual ACM-SIAM Symp. Discrete Algorithms*, 2020, pp. 560–578.

[23] C. Dwork and A. Roth, ''The algorithmic foundations of differential privacy,'' *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, ''Distributed representations of words and phrases and their compositionality,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[25] M. Zhang and Y. Chen, ''Weisfeiler–Lehman neural machine for link prediction,'' in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 575–583.

[26] H. Wang, Q. Meng, J. Fan, Y. Li, L. Cui, X. Zhao, C. Peng, G. Chen, and X. Du, ''Social influence does matter: User action prediction for in-feed advertising,'' in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 246–253.

[27] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, ''Differentially private empirical risk minimization,'' *J. Mach. Learn. Res.*, vol. 12, no. 3, pp. 1069–1109, 2011.

[28] J. Hua, C. Xia, and S. Zhong, ''Differentially private matrix factorization,'' in *Proc. 24th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2015, pp. 1763–1770.

[29] A. Friedman, S. Berkovsky, and M. A. Kaafar, ''A differential privacy framework for matrix factorization recommender systems,'' *User Model. User-Adapted Interact.*, vol. 26, no. 5, pp. 425–458, Dec. 2016.

[30] N. S. Dash, ''Context and contextual word meaning,'' *SKASE J. Theor. Linguistics*, vol. 5, no. 2, pp. 21–31, 2008.

[31] E. M. Jin, M. Girvan, and M. E. J. Newman, ''Structure of growing social networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 64, no. 4, Sep. 2001, Art. no. 046132.

[32] V. Kalofolias, ''How to learn a graph from smooth signals,'' in *Proc. 19th Int. Conf. Artif. Intell. Statist.*, 2016, pp. 920–929.

[33] N. Komodakis and J.-C. Pesquet, ''Playing with duality: An overview of recent primal? Dual approaches for solving large-scale optimization problems,'' *IEEE Signal Process. Mag.*, vol. 32, no. 6, pp. 31–54, Nov. 2015.

[34] J. Kunegis, ''KONECT: The Koblenz network collection,'' in *Proc. 22nd Int. Conf. World Wide Web*, May 2013, pp. 1343–1350.

[35] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore, ''Automating the construction of internet portals with machine learning,'' *Inf. Retr.*, vol. 3, no. 2, pp. 127–163, 2000.

[36] C. Stark, ''BioGRID: A general repository for interaction datasets,'' *Nucleic Acids Res.*, vol. 34, no. 90001, pp. D535–D539, Jan. 2006.

[37] R. Rossi and N. Ahmed, ''The network data repository with interactive graph analytics and visualization,'' in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1–2.

[38] C. Yang, H. Wang, K. Zhang, L. Chen, and L. Sun, ''Secure deep graph generation with link differential privacy,'' in *Proc. 30th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 3271–3278.

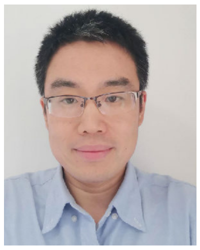
[39] H. Sun, X. Xiao, I. Khalil, Y. Yang, Z. Qin, H. Wang, and T. Yu, ''Analyzing subgraph statistics from extended local views with decentralized differential privacy,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 703–717.

[40] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, ''Fast unfolding of communities in large networks,'' *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008.

[41] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, ''Cost-effective outbreak detection in networks,'' in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2007, pp. 420–429.

[42] K. Saito, R. Nakano, and M. Kimura, ''Prediction of information diffusion probabilities for independent cascade model,'' in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst.*, 2008, pp. 67–75.

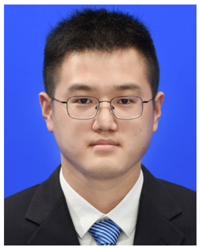
XINJIAN ZHAO received the M.S. degree in computer science and technology from Peking University, in 2010. He is currently a Senior Engineer with the Information and Telecommunication Branch, State Grid Jiangsu Electric Power Company Ltd. His current research interests include big data and network security research.



FEI XIA received the M.S. degree in communication and information systems from the Nanjing University of Posts and Telecommunications, in 2007. He is currently a Senior Engineer with the Information and Telecommunication Branch, State Grid Jiangsu Electric Power Company Ltd. His current research interests include big data and network security research.

GUOQUAN YUAN received the M.S. degree in computer science and technology from Nanjing University, in 2012. He is currently a Senior Engineer with the Information and Telecommunication Branch, State Grid Jiangsu Electric Power Company Ltd. His current research interests include big data and network security research.

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 current research interests include data mining, data privacy protection, big data technology, and complex data management.



SHI CHEN received the M.S. degree in electronic and communication engineering from Nanjing University, in 2020. He is currently an Assistant Engineer with the Information and Telecommunication Branch, State Grid Jiangsu Electric Power Company Ltd. His current research interests include electric power informatization and network security.

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 current research interests include data mining, data provenance, data privacy protection, big data technology, and complex data management.