Differentially Private Binary- and Matrix-Valued Data Query: An XOR Mechanism

Tianxi Ji
Case Western Reserve University
txj116@case.edu

Pan Li
Case Western Reserve University
pxl288@case.edu

Emre Yilmaz
University of Houston-Downtown
yilmaze@uhd.edu

Erman Ayday
Case Western Reserve University
Bilkent University
exa208@case.edu

Yanfang (Fanny) Ye
Case Western Reserve University
yanfang.ye@case.edu

Jinyuan Sun
The University of Tennessee,
Knoxville
jysun@utk.edu

ABSTRACT

Differential privacy has been widely adopted to release continuous- and scalar-valued information on a database without compromising the privacy of individual data records in it. The problem of querying binary- and matrix-valued information on a database in a differentially private manner has rarely been studied. However, binary- and matrix-valued data are ubiquitous in real-world applications, whose privacy concerns may arise under a variety of circumstances. In this paper, we devise an exclusive or (XOR) mechanism that perturbs binary- and matrix-valued query result by conducting an XOR operation on the query result with calibrated noises attributed to a matrix-valued Bernoulli distribution. We first rigorously analyze the privacy and utility guarantee of the proposed XOR mechanism. Then, to generate the parameters in the matrix-valued Bernoulli distribution, we develop a heuristic approach to minimize the expected square query error rate under ϵ-differential privacy constraint. Additionally, to address the intractability of calculating the probability density function (PDF) of this distribution and efficiently generate samples from it, we adapt an Exact Hamiltonian Monte Carlo based sampling scheme. Finally, we experimentally demonstrate the efficacy of the XOR mechanism by considering binary data classification and social network analysis, all in a differentially private manner. Experiment results show that the XOR mechanism notably outperforms other state-of-the-art differentially private methods in terms of utility (such as classification accuracy and F_1 score), and even achieves comparable utility to the non-private mechanisms.

PVLDB Reference Format:
Tianxi Ji, Pan Li, Emre Yilmaz, Erman Ayday, Yanfang (Fanny) Ye, and Jinyuan Sun. Differentially Private Binary- and Matrix-Valued Data Query: An XOR Mechanism. PVLDB, 14(5): 849-862, 2021.
doi:10.14778/3446095.3446106

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/nestgroup/The-XOR-mechanism.

1 INTRODUCTION

Data sharing and releasing are undoubtedly critical for building a data-driven future. However, data sovereignty, regulations, and privacy concerns may prevent data holders from sharing their data, and hence hinder the development of data-driven applications. To handle this problem, differential privacy [13, 15] has been employed as a *de facto* standard for releasing privacy-preserving statistical information queries of databases. By applying differentially private mechanisms, a trusted database curator can answer queries requested by data consumers, like companies and research organizations, and guarantee that the released results are independent of the presence or absence of an individual data record.

In the literature, quite a few mechanisms have been proposed to conduct continuous- and scalar-valued queries in a differentially private manner [13–15, 40]. By treating vectors or matrices as collections of scalar values, these mechanisms can also be extended to perturb continuous- and vector-/matrix-valued queries by adding independent and identically distributed (i.i.d.) noises to each element of the query results [13, 15]. However, this approach usually results in suboptimal solutions or utility degradation, because it overlooks the underlying structural information like data correlation and dependency in the query results. Some mechanisms exploiting the data correlations are proposed in [8, 18–21, 23, 31, 37–39, 49, 57, 62, 66, 68], such as Bayesian differential privacy, pufferfish privacy, and the MVG mechanism, yet the structural information of the query results is still largely underinvestigated.

Moreover, all of the aforementioned mechanisms can hardly be applied when the query results are in the form of a binary matrix. In fact, binary- and matrix-valued query results are ubiquitous in real-world applications, whose privacy concerns may arise under a variety of circumstances. One example is social network analysis, which usually requires access to the networks' topologies characterized by binary adjacency matrices containing sensitive social relationship information [28, 55]. The leakage of social relationship information can lead to severe problems. For instance, it has been shown that social network users' relationships can be used to discover their identities and geo-locations and even track them [39, 43]. Some other application examples involving binary- and matrix-valued data include the coarse quantization for data compression and storage [6, 17, 65], and the recently emerged XNOR-Nets [54] (where the queried data is the binarized training images). As a result, designing a differentially private mechanism for binary- and matrix-valued queries is in dire need.

Proceedings of the VLDB Endowment, Vol. 14, No. 5 ISSN 2150-8097.
doi:10.14778/3446095.3446106

The fundamental challenges of this problem are twofold. First, the designed differentially private mechanism must maintain the binary property of the data. Second, it needs to take into account the structural information of the original query results so as to avoid potential utility degradation. To address these challenges, we develop a novel output perturbation mechanism, which conducts an exclusive or (XOR) operation on the query result with carefully designed binary- and matrix-valued noises. Specifically, our proposed XOR mechanism takes the structural information of the binary- and matrix-valued query results into account when perturbing them under differential privacy guarantee. We consider that all pairs of elements in the query result could be correlated (modeled by log-linear association parameters). Therefore, the perturbation noise in our mechanism is attributed to a matrix-valued Bernoulli distribution with a quadratic exponential dependence structure [41]. We *first* present a sufficient condition for the XOR mechanism to achieve ϵ-differential privacy and rigorously analyze its utility guarantee by investigating the expected *square query error rate*. *Second*, since designing the perturbation noises is in fact very challenging, we develop a heuristic approach to minimize the expected square query error rate by optimizing the eigenvalues of the parametric matrices under ϵ-differential privacy constraint, and then synthesize those parametric matrices based on the eigenvalues. *Third*, we adapt an Exact Hamiltonian Monte Carlo based sampling scheme to address the intractability of calculating the normalizing constant of the matrix-valued Bernoulli distribution and efficiently generate samples (i.e., noises) from it.

We summarize the main contributions of this paper as follows:

- We propose the XOR mechanism to protect the ϵ-differential privacy of binary- and matrix-valued queries.
- We derive a sufficient condition for the proposed XOR mechanism to achieve ϵ-differential privacy.
- We analyze the utility of the XOR mechanism through the lens of expected square query error rate of a given query.
- We devise a heuristic approach to generate the parameters in the matrix-valued Bernoulli distribution so as to minimize the expected square query error rate while satisfying ϵ-differential privacy constraint. We also adapt an Exact Hamiltonian Monte Carlo based scheme to generate samples from the desired distribution.
- We discuss the application of the XOR mechanism on special binary matrices, i.e., adjacency matrices of undirected and unweighted graphs, and investigate the privacy leakage of an arbitrary edge in graphs under edge-differential privacy.
- We evaluate the XOR mechanism on real-world applications. Experiment results show that the XOR mechanism outperforms state-of-the-art methods and achieves utility (e.g., classification accuracy and F_1 score) that is close to the case when the non-private query result is used.

The rest of the paper is organized as follows. In Section 2, we review the related works, which is followed by some preliminaries for this study in Section 3. After that, we introduce the proposed XOR mechanism, and theoretically analyze its privacy and utility guarantees in Section 4. We devise a heuristic approach to generate the parametric matrices in the matrix-valued Bernoulli distribution, provide the Exact Hamiltonian Monte Carlo sampling procedure to generate noise samples from the distribution, and present a toy example in Section 5. In Section 6, we discuss how to use the XOR mechanism to query the adjacency matrix of a graph. In Section 7 and 8, we investigate case studies on binary data classification and social network analysis. Finally, Section 9 concludes the paper.

2 RELATED WORK

We first review related works on differential privacy, and then discuss its application on graph analysis.

Mechanisms with Data Correlation. We discuss some representative differentially private mechanisms that handle data correlation, and elaborate their main differences with ours.

Bayesian Differential Privacy (BDP) [62] extends the original definition of differential privacy in a Bayesian way; it proposes to upper bound the ratio between two posterior probabilities of a randomized algorithm returning identical outcomes. Our mechanism differs with BDP in that BDP models correlated data with Gaussian Markov random field, whereas we consider a log-linear model that can better capture the correlation among binary entries [10].

Dependent Perturbation Mechanism (DPM) [39] accounts for the probabilistic dependence between tuples in a database. It achieves dependent differential privacy guarantee by augmenting the Laplace mechanism with a dependence coefficient, which computes the query sensitivity of dependent data. The main difference between our mechanism and DPM is that DPM considers a fixed dependence size L, i.e., any entry in the dataset is dependent on $L-1$ other entries. We relax this assumption and allow more flexibility in dependency size, i.e., all pairwise binary data records can potentially be correlated, and the correlations are modeled using the log-linear association parameters in the matrix-valued Bernoulli distribution.

Wasserstein Mechanism (WM) [57] calibrates Laplace noise by exploiting the Wasserstein distance between distributions of randomized output given neighboring input datasets. A special case of WM is Markov Quilt Mechanism (MQM), which models correlation between data entries by a Bayesian network. Although WM is general, it is impractical to apply it to real-world problems due to high computational complexity. MQM, while computationally light, may be also limited, as the Bayesian network describes correlated data as a directed acyclic graph, where parent and child nodes are correlated, and sibling nodes are not. In contrast, the XOR mechanism considers all pairwise correlations among the data entries.

Matrix-Variate Gaussian Mechanism (MVGM)[8] conducts differentially private matrix-valued queries by perturbing the output with noises drawn from a matrix-variate Gaussian distribution. MVGM only considers row-wise and column-wise data correlation in the matrix query result. In contrast, by exploiting the matrix-valued Bernoulli distribution with a quadratic exponential dependence structure, our proposed mechanism considers data correlation with finer granularity, i.e., the log-linear association at the element level.

Other differentially private mechanisms considering data correlation include zero-knowledge privacy [18, 19], Pufferfish privacy [31], Blowfish privacy [23], PrivBayes [66], membership privacy [37], correlated iteration mechanism [68]. However, none of them can directly protect data in a binary- and matrix-format. The proposed XOR mechanism is developed to bridge the gap between differential privacy and dichotomous correlated data.

Mechanisms without Data Correlation. To protect the privacy of matrix-valued data, one can also add i.i.d. noises to each element of the matrix using the traditional Laplace, Gaussian, or Exponential mechanism [15]. Binomial Mechanisms [1, 13] perturb discrete values in a differentially private manner using i.i.d. noises drawn from the binomial distribution. All these mechanisms usually result in utility degradation in real-world applications, because the overlook of data correlation makes the additive noises have large magnitudes and compromise their utility.

Differentially Private Graph Analysis. Quite a few works have investigated the problem of querying specific graph statistics under differential privacy [4, 5, 12, 26, 27, 30, 53, 59, 60]. In particular, Blocki et al. [5] propose restricted sensitivity as an alternative to global and smooth sensitivity to improve subgraph counting accuracy. Day et al. [12] develop (θ, ω)-Histogram and θ-Cumulative-Histogram using graph projection to publish graph degree distribution with node differential privacy. Karwa et al. [30] study the problem of releasing the number of isomorphic copies of subgraphs, e.g., a triangle, k-star or k-triangle, under edge-differential privacy. Iftikhar et al. [26] develop a dK-Microaggregation framework which anonymizes graphs by perturbing the dK-distributions (probability distribution defined on connected subgraphs of size d [42]) of the original graphs. However, all of the aforementioned works focus on specific queries, such as degree distribution, and subgraph number. Thus, they are not directly applicable to other graph based tasks, such as community detection and spectral analysis.

Releasing the entire graph under differential privacy also draws attention. Pygmalion [55] releases graph topology under ϵ-differential privacy guarantee by first extracting a given graph's detailed structure into degree correlation statistics, then injecting Laplace noises into the resulting statistics, and finally generating a synthetic graph by using the dK-graph model [42]. Most recently, Wang et al. [61] propose a probabilisitic generative model (called privateSBM) to synthesize and release weighted social network. However, privateSBM is based on Variational Bayesian Expectation and Maximization (EM), and the differential privacy guarantee can only be achieved when the EM algorithm arrives at the global optimal, which makes their models impractical in real world applications. In [25], Huang et al. develop an approach called PBCN (which is a combination of nodes clustering via K-means, graph reconstruction after degree sequence perturbation, and noise nodes generation) to release a noisy graph. Whereas, their approach is biased by the predetermined number of clusters, and may have unstable utilities. A generative adversarial network (GAN) based differentially private graph synthesizing framework is proposed in [16], yet the sensitivity of their framework scales with the maximum node degree, which may lead to unbounded sensitivity and degrade the utility of the synthesized graph. Ahmed et al. [2] propose to release a differentially private low dimensional approximation of the original adjacency matrix by first projecting it to a low dimension via the multiplication with a random matrix and then perturb the projected matrix using Gaussian noise. However, it is not clear how to obtain a valid adjacency matrix (graph structure) from the low dimensional noisy matrix. Qin et al. [52] propose the LDPGen framework, which incrementally clusters nodes based on their connections to different partitions of the whole population in a differentially private manner. Then, LDPGen synthesizes the sanitized graph by calculating the probability of two nodes being connected given the clusters.

3 PRELIMINARIES

Throughout this paper, $\mathcal{B}$ and $\mathbf{B}$ are used to denote matrix-valued Bernoulli random variable and its corresponding instance, respectively. We also denote $\mathbf{B}_k$ as a binary matrix of $\{0,1\}^{N\times P}$. Similarly, we let $\mathbf{b}_k = \text{vec}(\mathbf{B}_k^T) \in \mathcal{T} = \{0,1\}^{NP}$, $|\mathcal{T}| = 2^{NP}$, and $\mathbf{b} = \text{vec}(\mathbf{B}^T)$ for notation simplicity, where $\text{vec}(\cdot)$ is the vectorization operator that stacks columns of a matrix into a vector. We denote by D a database. Table 1 lists the frequently used notations in the paper.

Table 1: Frequently used notations in the paper.

Notions	Descriptions
$\mathcal{B}$	matrix-valued Bernoulli random variable
$\mathbf{B}, \boldsymbol{b}$	an instance of $\mathcal{B}$, vectorization of $\text{vec}(\mathbf{B})$
$\mathbf{B}_k, \boldsymbol{b}_k$	a matrix in $\{0,1\}^{N\times P}$, vectorization of $\text{vec}(\mathbf{B}_k)$
$\mathbf{\Theta}$	feature association parametric matrix
$\mathbf{\Lambda}_{i,j}$	association parametric matrix of object i and j
$\mathbf{\Pi}$	parameter in the multivariate Bernoulli distribution
$\mathbf{J}_{ij}$	single-entry matrix with 1 at the (i,j)-th position
D and D'	a pair of neighboring datasets
s_f	sensitivity of binary- and matrix-valued query
$\mathbf{A}$	binary adjacency matrix

3.1 Differential Privacy

Definition 1. *(ϵ-differential privacy [13, 15]). A randomized algorithm $\mathbb{M}$ with domain $\mathcal{D}$ satisfies ϵ-differential privacy if for any two neighboring datasets $D, D' \in \mathcal{D}$ (differ in only one data record), and for any $\Omega \subseteq \text{Range}(\mathbb{M})$, it holds that* $\Pr[\mathbb{M}(\mathrm{D}) \in \Omega] \leq \mathrm{e}^{\epsilon}\Pr[\mathbb{M}(\mathrm{D}') \in \Omega]$.

Definition 2. *(Binary- and matrix-valued query). Given a dataset $D \in \mathcal{D}$, a binary- and matrix-valued query is to query the result of $f(D) \in \{0,1\}^{N\times P}$, which contains P binary features of N objects, where $f(\cdot) : D \longrightarrow \{0,1\}^{N\times P}$ is a query function.*

Definition 3. *(Sensitivity of binary- and matrix-valued query). Given a pair of neighboring datasets D and D', and a binary- and matrix-valued query function $f(\cdot)$, the sensitivity (s_f) of the query function is defined as $s_f = \sup_{f(D), f(D')} \|f(D) \oplus f(D')\|_F^2$.*

3.2 Background on Statistics

Definition 4. *[41] A random matrix $\mathcal{B} \in \{0,1\}^{N\times P}$ is said to have a matrix-valued Bernoulli distribution with quadratic exponential dependence structure, i.e., $\mathcal{B} \sim \text{Ber}_{N,P}(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$,[1] if its probability density function (PDF) can be written as*

$$f_{\mathcal{B}}(\mathbf{B}) = C(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N}) \exp\left\{ \text{Tr}[\mathbf{B}\,\mathbf{\Theta}\,\mathbf{B}^T] + \textstyle\sum_{i=1}^{N} \sum_{j\neq i}^{N} \text{Tr}[\mathbf{J}_{ij}\,\mathbf{B}\,\mathbf{\Lambda}_{i,j}\,\mathbf{B}^T] \right\}.$$

[1] $\mathbf{\Theta} = \begin{bmatrix} \theta^1 & \theta^{1,2} & \cdots & \theta^{1,P} \\ \theta^{2,1} & \theta^2 & \cdots & \theta^{2,P} \\ \vdots & \vdots & \ddots & \vdots \\ \theta^{P,1} & \theta^{P,2} & \cdots & \theta^P \end{bmatrix}$, $\mathbf{\Lambda}_{i,j} = \mathbf{\Lambda}_{j,i}^T = \begin{bmatrix} \lambda_{i,j}^1 & \phi_{i,j}^{1,2} & \cdots & \phi_{i,j}^{1,P} \\ \phi_{i,j}^{2,1} & \lambda_{i,j}^2 & \cdots & \phi_{i,j}^{2,P} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{i,j}^{P,1} & \phi_{i,j}^{P,2} & \cdots & \lambda_{i,j}^P \end{bmatrix}$.
$\mathbf{\Theta} \in \mathcal{R}^{P\times P}$, and $\mathbf{\Lambda}_{i,j} \in \mathcal{R}^{P\times P}$.

The normalization constant is

$$C(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$$
$$= \Big[\sum_{\mathbf{B}_k} \exp\Big\{ \mathrm{Tr}[\mathbf{B}_k \, \mathbf{\Theta} \, \mathbf{B}_k^T] + \textstyle\sum_{i=1}^{N} \sum_{j \neq i}^{N} \mathrm{Tr}[\mathbf{J}_{ij} \, \mathbf{B}_k \, \mathbf{\Lambda}_{i,j} \, \mathbf{B}_k^T] \Big\} \Big]^{-1}, \tag{1}$$

where $\mathbf{B}_k \in \{0,1\}^{N \times P}$*, and* $\mathbf{J}_{ij}$ *is the single-entry matrix of order* $N \times N$ *with* 1 *at the* (i, j)*-th position and 0 elsewhere.*

In the PDF of the matrix-valued Bernoulli distribution, there are three types of parameters, i.e., θ, λ, and ϕ, that are interpreted as:[2]

- *"pure feature-association"* parameters, i.e., θ^p and $\theta^{p,q}$ ($p, q \in [1, P]$). They are log-linear parameters describing the association structure of the features, and they are symmetric and usually assumed to be common for all objects, i.e., $\theta_i^p = \theta^p$, $\theta_i^{p,q} = \theta_i^{q,p} = \theta^{p,q} = \theta^{q,p}, \forall i \in [1, N]$.
- *"pure object-association"* parameters, i.e., $\lambda_{i,j}^p$ ($i, j \in [1, N]$, $p \in [1, P]$). They describe the intra-objects dependence with respect to (w.r.t.) each feature. They also satisfy the symmetry constraints, i.e., $\lambda_{i,j}^p = \lambda_{j,i}^p, \forall i, j \in [1, N], p \in [1, P]$.
- *"mixed features/objects-association"* parameters, i.e., $\phi_{i,j}^{p,q}$ ($i, j \in [1, N]$, $p, q \in [1, P]$). They describe the intra-objects dependence w.r.t. a particular combination of two features, and also satisfy the symmetry constraints, i.e., $\phi_{i,j}^{p,q} = \phi_{i,j}^{q,p}$.

The following Lemma connects the matrix-valued Bernoulli distribution with the multivariate Bernoulli distribution.

Lemma 1. *[41] If* $\mathcal{B} \sim \mathrm{Ber}_{N,P}(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$*, then* $\mathrm{vec}(\mathcal{B}^T)$ *has a multivariate Bernoulli distribution with parameter* $\mathbf{\Pi}$*, i.e.,* $\mathrm{vec}(\mathcal{B}^T) \sim \mathrm{Ber}_{NP}(\mathbf{\Pi})$*, and the PDF is*

$$f_{\mathrm{vec}(\mathcal{B}^T)}(\mathrm{vec}(\mathcal{B}^T) = \mathbf{b}) = C(\mathbf{\Pi}) \exp\{\mathbf{b}^T \, \mathbf{\Pi} \, \mathbf{b}\}, \tag{2}$$

where the parameter $\mathbf{\Pi}$ *and the normalization constant* $C(\mathbf{\Pi})$ *are*

$$\mathbf{\Pi} = \mathbf{I}_N \otimes \mathbf{\Theta} + \textstyle\sum_{i=1}^{N} \sum_{j \neq i}^{N} \mathbf{J}_{ij} \otimes \mathbf{\Lambda}_{i,j}, \; C(\mathbf{\Pi}) = \Big[\sum_{\mathbf{b}_k \in \mathcal{T}} \exp\{\mathbf{b}_k^T \, \mathbf{\Pi} \, \mathbf{b}_k\} \Big]^{-1}. \tag{3}$$

4 THE EXCLUSIVE OR MECHANISM

Now, we formally introduce the proposed XOR mechanism, prove its privacy guarantee, and investigate its utility guarantee.

Definition 5. *(XOR Mechanism). Given a binary- and matrix-valued query* $f(D) \in \{0,1\}^{N \times P}$*, the XOR mechanism is defined as*

$$\mathbb{XOR}(f(D), \mathcal{B}) = f(D) \oplus \mathcal{B},$$

where $\oplus$ *is the XOR operator,*[3] *and* $\mathcal{B} \sim \mathrm{Ber}_{N,P}(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$.

4.1 Privacy Guarantee

We give the privacy guarantee of the XOR mechanism in the following theorem, which not only provides a sufficient condition for it to achieve ϵ-differential privacy, but also presents a constraint on choosing the parameters in $\mathrm{Ber}_{N,P}(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$.

Theorem 1. *The XOR mechanism achieves* ϵ*-differential privacy of a matrix-valued binary query if* $\mathbf{\Theta}$ *and* $\mathbf{\Lambda}_{i,j}$ *satisfy*

$$s_f \Big(||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 + \textstyle\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} ||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2 \Big) \leq \epsilon, \tag{4}$$

where s_f *is the sensitivity of the binary- and matrix-valued query in Definition 3, and* $||\boldsymbol{\lambda}(\mathbf{\Theta})||_2$ *and* $||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2$ *are the* l_2 *norm of the vectors composed of eigenvalues of* $\mathbf{\Theta}$ *and* $\mathbf{\Lambda}_{i,j}$*, respectively.*

Proof. Please refer to Appendix A for the detailed proof. Here, we only provide the sketch of the proof:

1. Let D and D' be a pair neighboring datasets, and $\mathcal{B}_D$ (or $\mathcal{B}_{D'}$) be the matrix-valued Bernoulli distributed noise used to perturb $f(D)$ (or $f(D')$). To achieve ϵ-differential privacy, we need to have $\Pr(f(D) \oplus \mathcal{B}_D \in \mathcal{S}) \leq \Pr(f(D') \oplus \mathcal{B}_{D'} \in \mathcal{S})$, where $\mathcal{S}$ is the range of $\mathbb{XOR}(f(D), \mathcal{B})$. This inequality can be shown to be equivalent as $\mathrm{Tr}[\mathbf{B}_D \, \mathbf{\Theta} \, \mathbf{B}_D^T - \mathbf{B}_{D'} \, \mathbf{\Theta} \, \mathbf{B}_{D'}^T] + \sum_{i=1}^{N} \sum_{j \neq i}^{N} \mathrm{Tr}[\mathbf{J}_{ij}(\mathbf{B}_D \, \mathbf{\Lambda}_{i,j} \, \mathbf{B}_D^T - \mathbf{B}_{D'} \, \mathbf{\Lambda}_{i,j} \, \mathbf{B}_{D'}^T)] \leq \epsilon$, which is a sufficient condition for the XOR mechanism to achieve ϵ-differential privacy.

2. Bounding each trace term in the derived sufficient condition using the eigenvalues of the parametric matrices. □

Remark 1. *The condition developed in (4) can be intuitively interpreted as follows. Since* $\mathbf{\Theta}$ *and* $\mathbf{\Lambda}_{i,j}$ *are symmetric matrices, we have* $||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 = ||\mathbf{\Theta}||_F$ *and* $||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2 = ||\mathbf{\Lambda}_{i,j}||_F$*. Thus, given the sensitivity* s_f*, a small value of* ϵ *will make the log-linear parameters* θ_i^p*,* $\theta_i^{p,q}$*,* $\lambda_{i,j}^p$*, and* $\phi_{i,j}^{p,q}$ *close to 0, which means the ratio of the corresponding probabilities is close to 1. Take* θ_i^p *as an example, if* θ_i^p *is close to 0, then,* $\frac{\Pr(\mathrm{B}_{ip}=1|rest=0)}{\Pr(\mathrm{B}_{ip}=0|rest=0)}$ *will be close to 1, which suggests that the probability of a single noise element being 1 or 0 given the rest are 0's is approximately the same. Thus,* $f(D)$ *will be XORed by a nearly random binary matrix, which results in low utility but high privacy of the released query result. If* $\epsilon = 0$*, then,* $\mathbf{\Theta} = \mathbf{\Lambda}_{1,2} = \cdots = \mathbf{\Lambda}_{N-1,N} = \mathbf{0}$*, which means all elements in* $\mathcal{B}$ *are mutually independent, and an instance of* $\mathcal{B}$ *is a random binary matrix, whose elements equal to* 1 *with probability* $\frac{1}{2}$*. This leads to the highest privacy guarantee but the lowest utility, because it completely ignores all the potential structure dependency among data entries.*

Given a privacy budget $\epsilon > 0$, we need to select the matrix parameters satisfying (4) to guarantee that the perturbed $f(D)$ is ϵ-differentially private. In Section 5, we develop a heuristic approach to construct appropriate parameters, which satisfies (4) and minimizes the expected square query error rate of the XOR mechanism.

4.2 Utility Guarantee

Considering that the XOR mechanism achieves ϵ-differential privacy by perturbing the query with noises that affect the accuracy of the results, we define the *square query error rate* as follows.

Definition 6. *The square query error rate of the XOR mechanism given a query function and a dataset, i.e.,* $f(\cdot)$ *and* D*, is* $r(f(D), \mathcal{B}) = ||f(D) \oplus \mathcal{B} - f(D)||_F^2 / ||f(D)||_F^2$.

[2] Pure feature-association parameters are defined as

$$\theta_i^p = \log\Big\{ \frac{\Pr(\mathbf{B}_{ip} = 1|rest = 0)}{\Pr(\mathbf{B}_{ip} = 0|rest = 0)} \Big\},$$
$$\theta_i^{p,q} = \log\Big\{ \frac{\Pr(\mathbf{B}_{ip} = 1, \mathbf{B}_{iq} = 1|rest = 0)}{\Pr(\mathbf{B}_{ip} = 0, \mathbf{B}_{iq} = 1|rest = 0)} \frac{\Pr(\mathbf{B}_{ip} = 0, \mathbf{B}_{iq} = 0|rest = 0)}{\Pr(\mathbf{B}_{ip} = 1, \mathbf{B}_{iq} = 0|rest = 0)} \Big\}.$$

Pure object-association parameters are defined as

$$\lambda_{i,j}^p = \log\Big\{ \frac{\Pr(\mathbf{B}_{ip} = 1, \mathbf{B}_{jp} = 1|rest = 0)}{\Pr(\mathbf{B}_{ip} = 0, \mathbf{B}_{jp} = 1|rest = 0)} \frac{\Pr(\mathbf{B}_{ip} = 0, \mathbf{B}_{jp} = 0|rest = 0)}{\Pr(\mathbf{B}_{ip} = 1, \mathbf{B}_{jp} = 0|rest = 0)} \Big\}.$$

Mixed features/objects-association parameters are defined as

$$\phi_{i,j}^{p,q} = \log\Big\{ \frac{\Pr(\mathbf{B}_{ip} = 1, \mathbf{B}_{jq} = 1|rest = 0)}{\Pr(\mathbf{B}_{ip} = 0, \mathbf{B}_{jq} = 1|rest = 0)} \frac{\Pr(\mathbf{B}_{ip} = 0, \mathbf{B}_{jq} = 0|rest = 0)}{\Pr(\mathbf{B}_{ip} = 1, \mathbf{B}_{jq} = 0|rest = 0)} \Big\}.$$

[3] For any binary numbers u, v, we have $u \oplus v = \overline{u} \times v + u \times \overline{v}$.

To analyze the utility guarantee of the XOR mechanism, we explore the expected value of the square query error rate, i.e, $\mathbb{E}[r(f(D),\mathcal{B})]$. In particular, we have the following theorem.

Theorem 2. *Let $\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N}$ be the matrix parameters satisfying the sufficient condition to achieve ϵ-differentially private $f(D)$ in Theorem 1, then the expected square query error rate is*

$$\underset{\mathcal{B}\sim\mathrm{Ber}_{N,P}(\mathbf{\Theta},\mathbf{\Lambda}_{1,2},\cdots,\mathbf{\Lambda}_{N-1,N})}{\mathbb{E}}\left[r(f(D),\mathcal{B})\right] = \frac{1}{||f(D)||_F^2}\frac{\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\mathbf{\Pi}\,\mathbf{b}_k)||\,\mathbf{b}_k\,||_2^2}{\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\mathbf{\Pi}\,\mathbf{b}_k)}.$$

Proof. This theorem is proved by first showing the multivariate Bernoulli distribution is an exponential family distribution, then applying Lemma 1, and finally replacing integration by differentiation to calculate the expected value of exponential family distributed random variable. Please refer to Appendix B for details. □

Remark 2. *In Theorem 2, the parameters of the matrix-value Bernoulli distribution are embedded in $\mathbf{\Pi}$ given in (3). The result can be interpreted as a scaled weighted average of all possible $||\,\mathbf{b}_k\,||_2^2$'s ($||\,\mathbf{b}_k\,||_2^2 = k$), the weight is $\omega_k = \frac{\exp(\mathbf{b}_k^T\,\mathbf{\Pi}\,\mathbf{b}_k)}{\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\mathbf{\Pi}\,\mathbf{b}_k)}$ ($\sum_{\omega_k}\omega_k = 1$), and the scale factor is $\frac{1}{||f(D)||_F^2}$. If the privacy budget is low, then according to (4), $||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 = ||\,\mathbf{\Theta}\,||_F, ||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2 = ||\,\mathbf{\Lambda}_{i,j}\,||_F$ will be small, which make elements in $\mathbf{\Pi}$ close to 0, thus, lead to a high expected square query error rate. Particularly, when $\epsilon = 0$, we have the largest expected square query error rate, i.e., $\frac{1}{||f(D)||_F^2}\frac{1}{2^{NP}}\left(\sum_{t=0}^{NP}\binom{NP}{t}t\right) = \frac{NP2^{NP-1}}{2^{NP}||f(D)||_F^2} = \frac{NP}{2||f(D)||_F^2}$. It suggests that high privacy and high utility are conflicting objectives need to be balanced.*

5 GENERATION OF THE MATRIX-VALUED BERNOULLI DISTRIBUTED NOISE

Recall that in Theorem 1, the sufficient condition for the XOR mechanism to achieve ϵ-differential privacy only depends on the eigenvalues of the parametric matrices in the matrix-valued Bernoulli distribution. Thus, as long as (4) holds, there are infinite numbers of $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$'s in the design space of the perturbation noises. In this section, we first propose a heuristic approach to generate the parametric matrices. Then, we adapt an Exact Hamiltonian Monte Carlo based sampling scheme to generate samples (noises). Finally, we visualize the effect of the XOR mechanism using a toy example.

5.1 Generating Parameters in the Distribution: a Heuristic Approach

We generate the parametric matrices of the matrix-valued Bernoulli distribution from a utility boosting perspective. In other words, we propose to minimize the expected square error rate of the XOR mechanism by optimizing the eigenvalues of $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$ while satisfying the ϵ-differential privacy guarantee. Once having the eigenvalues, $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$ can be synthesized via $\mathbf{U}\mathrm{diag}(\boldsymbol{\lambda}(\mathbf{\Theta}))\mathbf{U}^{-1}$ and $\mathbf{U}\mathrm{diag}(\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j}))\mathbf{U}^{-1}$, where $\mathrm{diag}(\cdot)$ represents the diagonal matrix generated from a vector, and $\mathbf{U}$ is any orthonormal basis in $\mathcal{R}^P$. We consider $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$ are all positive definite (PD) matrices with positive entries, i.e., $\mathbf{\Theta}, \mathbf{\Lambda}_{i,j} \in \mathcal{PD}^{P\times P}$ and $\mathbf{\Theta} > 0, \mathbf{\Lambda}_{i,j} > 0$, and formulate the following optimization problem

$$\begin{aligned}\min_{\substack{\mathbf{\Theta}>0,\mathbf{\Lambda}_{i,j}>0,\mathbf{\Theta},\mathbf{\Lambda}_{i,j}\in\mathcal{PD}^{P\times P},\\ i,j\in[1,N],i\neq j}} \quad & \underset{\mathcal{B}\sim\mathrm{Ber}_{N,P}(\mathbf{\Theta},\mathbf{\Lambda}_{i,j},\cdots,\mathbf{\Lambda}_{N-1,N})}{\mathbb{E}}\left[r(f(D),\mathcal{B})\right] \\ \textbf{s. t.} \quad & \epsilon\text{-differential privacy condition in (4).}\end{aligned} \tag{5}$$

Since $\mathbb{E}[r(f(D),\mathcal{B})] \propto g(\Pi) = \frac{\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\mathbf{\Pi}\,\mathbf{b}_k)||\,\mathbf{b}_k\,||_2^2}{\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\mathbf{\Pi}\,\mathbf{b}_k)}$, and minimizing the ratio of two summations is generally an NP hard problem [46], we instead consider an upper bound of $g(\Pi)$, which is provided in the following proposition.

Proposition 1. *$g(\Pi)$ is upper bounded by $\frac{U(\Pi)}{L(\Pi)}$, i.e., $g(\Pi) \leq \frac{U(\Pi)}{L(\Pi)} = \frac{\sum_{\mathbf{b}_k\in\mathcal{T}}\exp\left((\lambda_{\max}(\mathbf{\Theta})+2\sum_{i=1}^{N-1}\sum_{j>i}^{N}\lambda_{\max}(\mathbf{\Lambda_{i,j}}))||\,\mathbf{b}_k\,||_2^2\right)||\,\mathbf{b}_k\,||_2^2}{2^{NP-1}\left(2+N||\boldsymbol{\lambda}(\mathbf{\Theta})||_2+\sum_{i=1}^{N-1}\sum_{j>i}^{N}||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2\right)}$, when $\mathbf{\Theta} > 0, \mathbf{\Lambda}_{i,j} > 0, \mathbf{\Theta}, \mathbf{\Lambda}_{i,j} \in \mathcal{PD}^{P\times P}, i, j \in [1,N], i \neq j$.*

Proof. This upper bound can be obtained by first upper bounding the numerator of $g(\Pi)$ using eigenvalues of the parametric matrices, and then, lower bounding the denominator of $g(\Pi)$ via Taylor approximation. Please refer to Appendix C for the proof. □

As a consequence, we can reformulate (5) as

$$\begin{aligned}\min_{\mathbf{\Theta}>0,\mathbf{\Lambda}_{i,j}>0,\mathbf{\Theta},\mathbf{\Lambda}_{i,j}\in\mathcal{PD}^{P\times P},i,j\in[1,N],i\neq j} \quad & U(\Pi)/L(\Pi) \\ \textbf{s. t.} \quad & \epsilon\text{-differential privacy condition in (4).}\end{aligned} \tag{6}$$

We develop a heuristic solution to (6) based on the following observation: to reduce $U(\Pi)/L(\Pi)$, we need to decrease $\lambda_{\max}(\mathbf{\Theta})$ and $\lambda_{\max}(\mathbf{\Lambda}_{i,j})$ and increase $||\boldsymbol{\lambda}(\mathbf{\Theta})||_2$ and $||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2$ under the ϵ-differential privacy constraint in (4). Since $\mathbf{\Theta}, \mathbf{\Lambda}_{i,j} \in \mathcal{PD}^{P\times P}$, the smallest $\lambda_{\max}(\mathbf{\Theta})$ and $\lambda_{\max}(\mathbf{\Lambda}_{i,j})$ are $\lambda_{\max}(\mathbf{\Theta}) = ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2/\sqrt{p}$ and $\lambda_{\max}(\mathbf{\Lambda}_{i,j}) = ||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2/\sqrt{p}$, respectively. As a result, the heuristic solution to (6), which is also a feasible solution to (5), is obtained by letting $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$ all have 1 eigenvalue with multiplicities P, i.e., $\lambda_1(\mathbf{\Theta}) = \cdots\lambda_P(\mathbf{\Theta}) = ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2/\sqrt{p}$ and $\lambda_1(\mathbf{\Lambda}_{i,j}) = \cdots\lambda_P(\mathbf{\Lambda}_{i,j}) = ||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2/\sqrt{p}, i, j \in [1,N], i \neq j$. The privacy budget allocated to $||\boldsymbol{\lambda}(\mathbf{\Theta})||_2$ and $||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2$'s is controlled by a parameter $\alpha \in (0,1)$. We present the procedure to generate $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$ in Algorithm 1.

Algorithm 1: Parametric matrices generation procedure

```
Input  : Privacy budget $\epsilon$ provided by the data consumer, the
         privacy budget allocation parameter $0 < \alpha < 1$.
Output : $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}, i, j \in [1,N], i \neq j$.
Allocate budget $\alpha\frac{\epsilon}{s_f}$ to $||\boldsymbol{\lambda}(\mathbf{\Theta})||_2$.
Let $\lambda_1(\mathbf{\Theta}) = \cdots = \lambda_P(\mathbf{\Theta}) = \frac{\alpha\epsilon}{s_f\sqrt{p}}$.
$\mathbf{\Theta} \leftarrow \mathbf{U}\mathrm{diag}(\boldsymbol{\lambda}(\mathbf{\Theta}))\mathbf{U}^{-1}$. // U is any orthonormal basis in $\mathcal{R}^P$.
for $i \in [1, N-1]$ do
    for $j \in [i+1, N]$ do
        Allocate budget $\frac{1-\alpha}{N(N-1)/2}\frac{\epsilon}{s_f}$ to $||\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j})||_2$.
        Let $\lambda_1(\mathbf{\Lambda}_{i,j}) = \cdots = \lambda_P(\mathbf{\Lambda}_{i,j}) = \frac{(1-\alpha)\epsilon}{s_f\sqrt{p}N(N-1)/2}$.
        $\mathbf{\Lambda}_{i,j} \leftarrow \mathbf{U}\mathrm{diag}(\boldsymbol{\lambda}(\mathbf{\Lambda}_{i,j}))\mathbf{U}^{-1}$.
        $\mathbf{\Lambda}_{j,i} \leftarrow \mathbf{\Lambda}_{i,j}^T$.
    end
end
```

5.2 Generating Samples in the Distribution: Exact Hamiltonian Monte Carlo

It is computationally prohibitive to calculate the normalization constants in both (1) and (3), as they require the summation of 2^{NP} items. Besides, there is no available toolbox to generate samples from the matrix-valued Bernoulli distribution, which hinders the success of applying our proposed mechanism. To solve this problem, we propose to generate samples of the matrix-valued Bernoulli noise using the Exact Hamiltonian Monte Carlo (EHMC) approach [50], which efficiently samples generic distributions in the support of $\{-1,+1\}^d$, where d is the dimension of interest.

To map random variable from $\{0,1\}^{NP}$ to $\{-1,+1\}^{NP}$, we perform variable transformation on $\mathbf{b}$, i.e., $\mathbf{s} = g(\mathbf{b}) = 2\,\mathbf{b}-1$. Thus, we have $\mathbf{s} \in \{-1,+1\}^{NP}$ with PDF $f_{\mathbf{S}}(\mathbf{s}) = \sum_{\mathbf{b}\in g^{-1}(\mathbf{s})} f_{\mathrm{vec}(\mathcal{B}^T)}(\mathbf{b}) = \frac{1}{Z} f_{\mathrm{vec}(\mathcal{B}^T)}(\frac{\mathbf{s}+1}{2}) \propto e^{\{\mathbf{s}^T\,\mathbf{\Pi}\,\mathbf{s}+2\,\mathbf{s}^T\,\mathbf{\Pi}\mathbf{1}\}}$, where $f_{\mathrm{vec}(\mathcal{B}^T)}(\mathbf{b})$ is defined in (2), and Z is a normalization factor, whose value is not required in practice, and $\mathbf{1}$ is an all-ones vector.

According to [50], EHMC generates samples from $f_{\mathbf{S}}(\mathbf{s})$ by first sampling $\mathbf{y}$ from $f_{\mathbf{Y}}(\mathbf{y}) = f_{\mathbf{S}}(\mathbf{s}) f_{\mathbf{Y}|\mathbf{S}}(\mathbf{y}\,|\,\mathbf{s})$ using the standard HMC procedure, where $f_{\mathbf{Y}|\mathbf{S}}(\mathbf{y}\,|\,\mathbf{s})$ is a truncated Gaussian distribution, i.e., $f_{\mathbf{Y}|\mathbf{S}}(\mathbf{y}\,|\,\mathbf{s}) = (\frac{2}{\pi})^{NP/2}\exp(-\frac{\mathbf{y}^T\mathbf{y}}{2})$, if $\mathrm{sign}(\mathbf{y}) = \mathbf{s}$, and $f_{\mathbf{Y}|\mathbf{S}}(\mathbf{y}\,|\,\mathbf{s}) = 0$, otherwise. Then, $\mathbf{s}$ is obtained by $\mathbf{s} = \mathrm{sign}(\mathbf{y})$, where $\mathrm{sign}(\cdot)$ is an element-wise sign function, and $\mathbf{y} \in \mathcal{R}^{NP}$. We summarize the steps to generate samples (noises) of the matrix-valued Bernoulli distribution in Algorithm 2. Specifically, we first generate samples of the multivariate Bernoulli distribution via EMHC (line 2-3), then resize it according to Lemma 1 to obtain noises with the desired matrix-valued Bernoulli distribution (line 4).

Algorithm 2: EHMC based noise samples generation

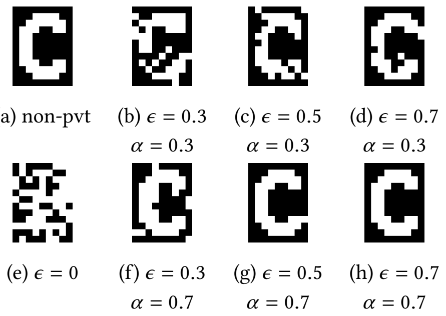
```
Input :
    • $\mathbf{\Theta}$ and $\mathbf{\Lambda}_{i,j}$, $i, j \in [1, N]$, $i \neq j$ generated by Algorithm 1.
    • $\mathbf{\Pi} \leftarrow \mathbf{I}_N \otimes \mathbf{\Theta} + \sum_{i=1}^{N}\sum_{j\neq i}^{N} \mathbf{J}_{ij} \otimes \mathbf{\Lambda}_{i,j}$.
    • The potential energy function, i.e.,
      $U(\mathbf{y}) = -\log f_{\mathbf{Y}|\mathbf{S}}(\mathbf{y}\,|\,\mathbf{s}) - \log\exp(\mathbf{s}^T\,\mathbf{\Pi}\,\mathbf{s}+2\,\mathbf{s}^T\,\mathbf{\Pi}\mathbf{1})$.
    • The momentum variable $\mathbf{q} \in \mathcal{R}^{NP}$, and the kinetic energy function
      $K(\mathbf{q}) = \frac{\mathbf{q}^T\mathbf{q}}{2}$.
    • The continuous Hamiltonian $H(\mathbf{y},\mathbf{q}) = U(\mathbf{y}) + K(\mathbf{q})$.
    • Particle moving duration $T$, and starting position $\mathbf{y}(0) \sim \mathcal{N}(\mathbf{0},\mathbf{I})$.
Output: An instance of $\mathbf{B} \sim \mathrm{Ber}_{N,P}(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$.
1 Sample the momentum variable $\mathbf{q}(0) \sim \mathcal{N}(\mathbf{0},\mathbf{I})$.
2 Let the particle move according to the equations of motion
    $\dot{\mathbf{y}(t)} = \frac{\partial H}{\partial \mathbf{q}(t)}$, $\dot{\mathbf{q}(t)} = -\frac{\partial H}{\partial \mathbf{y}(t)}$, i.e.,
    $y_i(t) = y_i(0)cos(t) + q_i(0)sin(t)$,
    $q_i(t) = -y_i(0)sin(t) + q_i(0)cos(t)$, $\forall i \in \{1, 2, \cdots, NP\}$.
3 Set $\mathbf{s} = \mathrm{sign}(\mathbf{y}(T))$, and $\mathbf{b} = \frac{\mathbf{s}+1}{2}$.
4 Set $\mathbf{B} = \big(\mathrm{Resize}(\mathbf{b}, P, N)\big)^T$, and return $\mathbf{B}$.
```

One of the most important property of EHMC based binary distribution sampling scheme is that it can draw samples from the exactly same binary distribution without approximation [50, 51]. The reason that it is tuning parameter-free and solves the two-state differential equations in the defined Hamiltonian dynamics exactly (i.e., line 2 of Algorithm 2), instead of using numerical integrators (such as the Leapfrog integrator) to approximate the Hamiltonian systems of equations. In contrast, traditional HMC sampling approaches will have numerical integrator's error on the order of τ^3 per iteration step and τ^2 globally, where τ is the step-size tuning parameter in HMC [34]. As a result, Algorithm 2 generates samples from the desired matrix-valued Bernoulli distribution and guarantees the ϵ-differential privacy of the XOR mechanism.

In practice, it is generally time consuming to use HMC based sampling. Algorithm 2 may cause memory overflow on ordinary devices if N and P are very large, as it requires the maintaining of $\mathbf{\Pi}$ which is of size $NP \times NP$. To solve these issues, we can partition the matrix-valued query $f(D)$ into smaller non-overlapping blocks, assign each block some fraction of the privacy budget (proportional to the size of the block), and then generate the binary noise patches for all the blocks. Thus, we can protect the differential privacy of each block of $f(D)$ separately.[4] In this case, it may lead to some utility loss since the correlation between entries in different blocks are ignored and each block only has reduced privacy budget, but as will be shown in the case studies, the proposed XOR mechanism can still achieve high utilities in real world applications.

5.3 A Toy Example on Grayscale Image

To demonstrate the effect of noises introduced by the XOR mechanism, we take a greyscale image as an example, i.e., the non-private character "C" image shown in Fig. 1 (a). We consider two choices of both privacy budget ϵ and allocation parameter α when releasing the image. Specifically, Fig. 1 (b)-(d) show the perturbed images by setting ϵ as 0.3, 0.5, and 0.7, respectively, when $\alpha = 0.3$. Fig. 1 (f)-(h) show the perturbed images by setting ϵ as 0.3, 0.5, and 0.7, respectively, when $\alpha = 0.7$. We observe that under the same ϵ, the noisy images are closer to the non-private one if α is higher (i.e., the privacy budget allocated to $\mathbf{\Theta}$ is higher). In particular, when $\epsilon = 0.7$ and $\alpha = 0.7$, the perturbed image is the same as the non-private image. Thus, in the case studies in Section 8, we will choose $0.5 < \alpha < 1$ to preserve more feature-associations while still make all $\mathbf{\Lambda}_{i,j}$ to be PD. Besides, we also show the perturbed image when $\epsilon = 0$ in Fig. 1 (e), which is a random matrix and has the lowest utility. It corroborates the analysis in Remark 2.



(a) non-pvt (b) $\epsilon = 0.3$, $\alpha = 0.3$ (c) $\epsilon = 0.5$, $\alpha = 0.3$ (d) $\epsilon = 0.7$, $\alpha = 0.3$

(e) $\epsilon = 0$ (f) $\epsilon = 0.3$, $\alpha = 0.7$ (g) $\epsilon = 0.5$, $\alpha = 0.7$ (h) $\epsilon = 0.7$, $\alpha = 0.7$

Figure 1: Visualization of a greyscale image perturbed by the proposed XOR mechansim.

[4] It is remarkable that the new arisen of quantum computing can be applied to efficiently generate desired binary noises in our study. The reason is that the calculation of normalization constant in (1) and (3) is essentially related to the computation of the permanent of matrix, which is intractable to classical computers, but has recently shown can be finished within seconds using photonic quantum computer [67].

6 APPLYING XOR MECHANISM ON GRAPHS

A special case of a binary- and matrix-valued query is the adjacency matrix which is symmetric and has zero diagonal entries. It represents the topology of an undirected graph, and its element indicates whether a particular pair of vertices are connected or not in the graph. Thus, a data consumer who is interested in the structure of a graph will conduct the query $f(\mathbf{A}) = \mathbb{XOR}(\mathbf{A}, \mathcal{B}) = \mathbf{A} \oplus \mathbf{B} \in \{0,1\}^{N \times N}$, where N is the number of nodes (here the dataset is the edge list of a graph, which is usually represented as a adjacency matrix). However, directly applying the XOR mechanism may not result in a valid adjacency matrix, as the generated matrix-valued Bernoulli distributed noise matrix may not be symmetric and its diagonal can also have nonzero entries. Thus, we need to perform post process to obtain a valid one. We summarize the procedure to query the adjacency matrix of an undirected graph with ϵ-differential privacy guarantee in Algorithm 3.

Specifically, at line 2, we use the proposed XOR mechanism to first get an intermediate result denoted as $\mathbf{A}_{temp}$, which may not be a valid adjacency matrix. Then, at line 3, we post process $\mathbf{A}_{temp}$ by first performing the element-wise logical conjunction (logical and) operation between $\mathbf{A}_{temp}$ and its transpose, and then subtracting the diagonal entries from the result of the logical conjunction. We can immediate have the following theorem to guarantee the ϵ-differential privacy of Algorithm 3.

Algorithm 3: Query adjacency matrix with ϵ-differential privacy guarantee using the proposed XOR mechanism

```
Input  : Privacy budget $\epsilon$ and allocation parameter $\alpha$ provided by
         the data consumer, the original adjacency matrix $\mathbf{A}$.
Output : $\widetilde{\mathbf{A}}$, $\epsilon$-differentially private adjacency matrix.
1 Set the parameters in the matrix-valued Bernoulli distribution and
    generate a sample, i.e., $\mathbf{B}$, from the distribution using Algorithm 1
    and 2, respectively.
2 Set $\mathbf{A}_{temp} = \mathbf{A} \oplus \mathbf{B}$.
3 Set $\widetilde{\mathbf{A}} = \mathbf{A}_{temp} \wedge \mathbf{A}_{temp}^T - \mathrm{diag}(\mathbf{A}_{temp} \wedge \mathbf{A}_{temp}^T)$.
4 Release $\widetilde{\mathbf{A}}$ to the data consumer.
```

Theorem 3. *$\widetilde{\mathbf{A}}$ obtained via Algorithm 3 is ϵ-differentially private.*

Proof. The post processing step (line 3 in Algorithm 3) is a deterministic mapping function $g : \{0,1\}^{N \times N} \to \{0,1\}^{N \times N}$. Thus, Algorithm 3 is a composition of the XOR mechanism and g, i.e., $g(\mathbb{XOR}(\mathbf{A}))$. Let $\mathbf{A}, \mathbf{A}'$ be the adjacency matrices of a pair of neighboring graphs that differ in only one edge. For any fixed event $\mathcal{H} \subset \mathcal{V}$, where $\mathcal{V}$ is the set of valid adjacency matrices of undirected graph with N nodes, define the event $\mathcal{S} = \{S \in \{0,1\}^{N \times N} : g(S) \in \mathcal{H}\}$. Then, we have $\Pr[g(\mathbb{XOR}(f(\mathbf{A}))) \in \mathcal{H}] = \Pr[\mathbb{XOR}(f(\mathbf{A})) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathbb{XOR}(f(\mathbf{A}')) \in \mathcal{S}] = e^{\epsilon} \Pr[g(\mathbb{XOR}(f(\mathbf{A}'))) \in \mathcal{H}]$, which satisfies the definition of ϵ-differential privacy. □

Adjacency matrices carry sensitive information of social actors' relationship, and Theorem 3 protects any edge in the graph from disclosure. Thus, Algorithm 3 is an ϵ-edge-differentially private mechanism [22, 55] even considering a powerful attacker [39, 62], who has access to not only the noisy query O (e.g., graph topology, community detection results, degree distribution) but also the auxiliary social connection excluding just one pair of users (i.e., $\mathbf{A}_{/\{i,j\}}$). Particularly, we can arrive at the following conclusion.

Theorem 4. *Define the privacy leakage of an arbitrary edge in a graph as* $\mathrm{PrvcLkg} = \max_{a \in \{0,1\}} \Pr(\widehat{\mathbf{A}_{i,j}} = a | O, \mathbf{A}_{/\{i,j\}})$*, which is the maximal posterior probability of the relationship* $(\widehat{\mathbf{A}_{i,j}} = a)$ *inferred by the powerful attacker. If O is the query result from an ϵ-edge-differentially private mechanism, then* $\mathrm{PrvcLkg} = \max\{\frac{1}{\zeta e^{\epsilon}+1}, \frac{\zeta e^{\epsilon}}{\zeta e^{\epsilon}+1}\}$*, where* $\zeta = \frac{\Pr(\widehat{\mathbf{A}_{i,j}}=a | \mathbf{A}_{/\{i,j\}})}{\Pr(\widehat{\mathbf{A}_{i,j}}=\overline{a} | \mathbf{A}_{/\{i,j\}})}$ *is the ratio between prior probabilities of that relationship, and is independent of the adopted mechanism.*

Proof. Please refer to Appendix D for the proof. □

7 CASE STUDY I: DIFFERENTIALLY PRIVATE BINARY DATA CLASSIFICATION

In case study I, we consider generic binary- and matrix-valued data query, and investigate the problem of binary data classification in a differentially private manner. We consider two datasets. The first is the congressional voting records dataset (Votings) [56], which contains votes on 16 issues from 435 representatives either from the democratic or the republican party. We binarize the categorical voting records to obtain binary data entries.[5] The second dataset is the Breast Cancer Wisconsin Diagnostic dataset (Cancer) [58], which contains 569 medical records that are labeled as either malignant or benign. Each record has 30 features extracted from the cell nucleus of breast cancer sample, and the feature are also binarized.[6]

We adopt logistic regression to distinguish the two classes in each of the considered dataset, and the binary- and matrix-valued queries are the training dataset. In particular, for Voting, by leaving out 45% voting records for testing, we have $f(D_{\text{Voting}}) = \{0,1\}^{240 \times 16}$. For Cancer, by leaving out 25% instances for testing, we have $f(D_{\text{Cancer}}) = \{0,1\}^{426 \times 30}$. According to Definition 3, the sensitivity s_f for $f(D_{\text{Voting}})$ and $f(D_{\text{Cancer}})$ are 16 and 30, respectively.

Note that the data objects (i.e., records) in each dataset are mutually independent (e.g., the statistics of the breast cancer cell nucleus of an individual are independent from the others), which suggests that the query result satisfy *object-independence* and leads to $\mathbf{\Lambda}_{i,j} = \mathbf{0}, \forall i, j \in [1, N], i \neq j$. This is consistent with the common assumption adopted by many learning algorithms that data samples are i.i.d. w.r.t. a certain underlying probability distribution. Hence, the sufficient condition for the XOR mechanism to be ϵ-differentially private (i.e., Theorem 1) becomes $s_f ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 \leq \epsilon$.

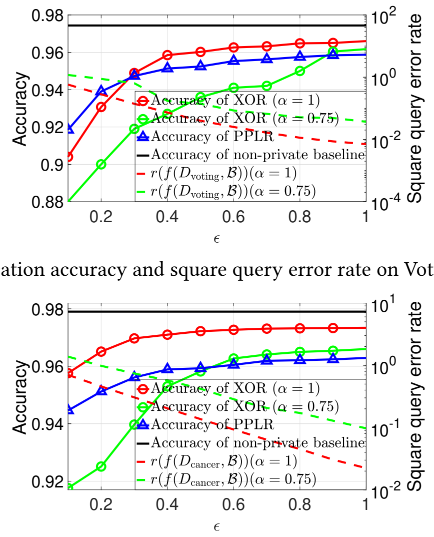
Since there is no existing work that can directly protect the differential privacy of binary- and matrix-valued query, we compare XOR mechanism with the non-private baseline, i.e., the logistic regression classifier obtained by using the original non-private training data, and the classifier obtained from privacy-preserving logistic regression (PPLR) [9], which conducts objective perturbation on the objective function of logistic regression with ϵ-differential privacy

[5] Voted for, paired for, and announced for are represented as "yes" (i.e., 1), and voted against, paired against, announced against, voted present, voted present to avoid conflict of interest, and did not vote are represented as "no" (i.e., 0).

[6] For example, if the radius of a cell nucleus (mean of distances from center to points on the perimeter) exceeds a certain threshold, then radius is set as 1, otherwise it is 0.

guarantee.[7] We use classification accuracy on the testing dataset as the evaluation metric.

We vary the privacy budget from 0.1 to 1 and show the classification accuracy on Voting and Cancer against the left y-axis in Figure 2 (a) and (b), respectively. Specifically, red and green dotted curves are the results using differentially private training dataset released by XOR mechanism, and $\alpha = 1$ corresponds to the object-independence assumption. Blue curves are the results obtained by PPLR. Black lines are the classification accuracy (97.44% for Voting and 97.9% for Cancer) obtained from the non-private baselines. Additionally, we also plot the empirical square query error rate (in red and green dashed curves) under different ϵ against the right y-axis.



(a) Classification accuracy and square query error rate on Voting Dataset

(b) Classification accuracy and square query error rate on Cancer Dataset

Figure 2: Classification accuracy and empirical square query error rate on (a) Voting and (b) Cancer.

From Figure 2 (a), we see that the XOR mechanism achieves higher accuracy than PPLR on the Voting dataset when $\epsilon \geq 0.3$ considering object-independence ($\alpha = 1$), but it can only slightly outperform PPLR when $\epsilon \geq 0.9$ without considering object-independence ($\alpha = 0.75$). This is because when data objects are independent, we do not need to preserve the object-associations in the matrix-valued query result. As a result, we can set $\alpha = 1$ in Algorithm 1 to allocate all the privacy budget to $\mathbf{\Theta}$ and preserve feature-associations only. Although we have the same privacy guarantee when $\alpha = 0.75$ and $\alpha = 1$, the prior knowledge on the structural property of the query result helps us obtain higher utility by achieving lower empirical square query error rate on the training dataset. Hence, when ϵ is high, only a small fraction of entries in $f(D_{\text{Voting}})$ will be perturbed, e.g., the empirical square query error rate is around 0.04 when $\epsilon = 0.5$. As a result, $\boldsymbol{\eta}^*_{\mathbb{XOR}}$ will be closer to $\boldsymbol{\eta}^*_{\text{Baseline}}$. In contrast, PPLR protects the privacy of training data by injecting i.i.d. noise into logistic regression, and makes $\boldsymbol{\eta}^*_{\text{PPLR}}$ deviated from $\boldsymbol{\eta}^*_{\text{Baseline}}$ even under large ϵ. This pattern is more clear in Figure 2 (b), where the proposed XOR mechanism considering object-independence outperforms PPLR given all ϵ, and is closer to the non-private baseline. For both datasets, the square query error rate decreases as ϵ increases, which validates our discussion in Remark 2.

8 CASE STUDY II: DIFFERENTIALLY PRIVATE SOCIAL NETWORK ANALYSIS

Now, we evaluate the performance of the proposed XOR mechanism through differentially private social network analysis.

8.1 Dataset and Tasks

We consider two social networks, i.e., (i) the Facebook friendship network [36], which has 4039 users and 10 ground-truth communities, and is constructed by merging the ego networks of 10 focal users ("ego"), and (ii) the email interaction network [35], which has 986 users and 42 ground-truth communities, and is constructed from the email interactions among institution members in 42 departments. In the experiment, we denote the pristine Facebook network (resp. email network) as G^{FB} (resp. G^{EM}), and the differential privately released Facebook network (resp. email network) using privacy budget ϵ as G^{FB}_ϵ (resp. G^{EM}_ϵ). We analyze the topology of the two social networks in a differentially private manner. In particular, we consider querying their topology (i.e., adjacency matrices) with differential privacy guarantees, based on which some network statistics are calculate, and tasks of degree distribution estimation and community detection are conducted. Note that in this case study, the dataset is considered as the edge list of a graph, which can be represented as the adjacency matrix, i.e., $D = \mathbf{A}$, thus, the query is $f(\mathbf{A})$. Since the addition or removing an edge will change two entries in $\mathbf{A}$, we have sensitivity $s_f = 2$.

8.2 The Comparing Mechanisms

We compare the XOR mechanism with ApproxDeg [28], DPCD [28], Pygmalion [55], LDPGen [52], and PBCN [25]. Specifically, ApproxDeg and DPCD are developed specially for degree distribution estimation and community detection under differential privacy guarantee.[8] Pygmalion, LDPGen, and PBCN are the state-of-the-art differentially private mechanism on graph topology releasing (discussed in Section 2). We also compare with the non-private baselines, that use the original adjacency matrices for all tasks.

8.3 Evaluation Metrics

Since all considered mechanisms in case study II can be categorized as ϵ-edge-differentially private mechanisms, according to Theorem 4, they all result in the same privacy leakage on an arbitrary edge. Thus, for a given ϵ value, it is sufficient to just compare the utilities

[7] Specifically, the classifiers from the non-private baseline, PPLR, and our proposed XOR mechanism are, respectively, obtained by solving the following optimization problems $\boldsymbol{\eta}^*_{\text{Baseline}} = \text{argmin}_{\boldsymbol{\eta}} -1/N \sum_{i=1}^N y_i \log(h(f(D)_i)) + (1 - y_i) \log(1 - h(f(D)_i))$, $\boldsymbol{\eta}^*_{\text{PPLR}} = \text{argmin}_{\boldsymbol{\eta}} -1/N \sum_{i=1}^N y_i \log(h(f(D)_i)) + (1 - y_i) \log(1 - h(f(D)_i)) + 1/N \boldsymbol{n}^T \boldsymbol{\eta}$, and $\boldsymbol{\eta}^*_{\mathbb{XOR}} = \text{argmin}_{\boldsymbol{\eta}} -1/N \sum_{i=1}^N y_i \log(h(\mathbb{XOR}(f(D))_i)) + (1 - y_i) \log(1 - h(\mathbb{XOR}(f(D))_i))$, where N is the number of samples in the training dataset, $f(D)_i$ is the ith data sample, $\mathbb{XOR}(f(D))_i$ is the ith perturbed data sample, $h(\mathbf{x}) = 1/1 + \exp(-\boldsymbol{\eta}^T \mathbf{x})$ is the sigmoid function, and $\boldsymbol{n}$ is the noise used to perturb the objective function. The norm of $\boldsymbol{n}$ is sampled from the $\Gamma(P, 2/\epsilon)$ distribution, P is the dimension of data sample, and the direction of $\boldsymbol{n}$ is selected uniformly at random.

[8] ApproxDeg is a combination of constrained global sensitivity control [14] and post-processing. DPCD performs objective perturbation on the objective function of community detection with differential privacy guarantees [9].

achieved by different mechanisms. Now, we formally define the utility metrics for the 2 social network analysis tasks.

Degree Distribution. We use the Kullback-Leibler (KL) divergence [32] to evaluate the utility of the obtained degree distributions. It is defined as $D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log(P(x)/Q(x))$, where $P(x)$ and $Q(x)$ are two probability distributions. Particularly, we let $P(x)$ be the degree distributions achieved from the differentially private mechanisms, and $Q(x)$ be that from the original social network. The smaller the KL divergence is, the higher utility the obtained degree distribution has.

Community Detection. We use average F_1 score [28, 29, 63, 64] as the utility metric for community detection. Given the detected communities, each community is matched with the most similar one of the ground-truth communities. F_1 score of two matched sets c_1 and c_2 is $F_1(c_1, c_2) = 2\frac{\text{prec}(c_1,c_2)\times\text{recall}(c_1,c_2)}{\text{prec}(c_1,c_2)+\text{recall}(c_1,c_2)}$, where $\text{prec}(c_1, c_2) = \frac{|c_1 \cap c_2|}{|c_1|}$ and $\text{recall}(c_1, c_2) = \frac{|c_1 \cap c_2|}{|c_2|}$. Average F_1 score of two sets of communities C and C^* is $\bar{F}_1(C, C^*) = \frac{1}{2|C|} \sum_{c_i \in C} F_1(c_i, C^*) + \frac{1}{2|C^*|} \sum_{c_i^* \in C^*} F_1(c_i^*, C)$, where $F_1(c_i, C^*) = \max_{c_j \in C^*} F_1(c_i, c_j)$. The higher the average F_1 score, the higher the utility.

8.4 Results

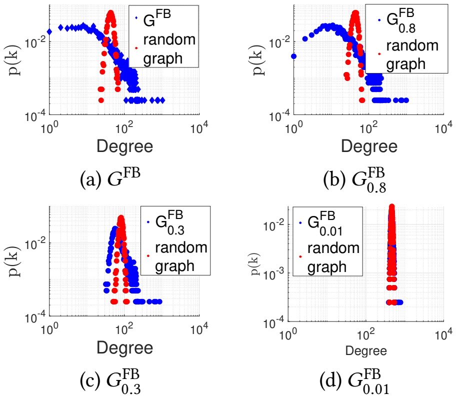
We first show the effect of privacy budget on the network statistics, and then present the experiment results on the 2 tasks.

Network Statistics. In Table 2, we show the number of edges ($|\mathcal{E}|$), network diameter, network density, and average path length (Avg. PL) of G^{FB} and G^{EM}, and those obtained by applying the XOR mechanism, Pygmalion, LDPGen, and PBCN, when the privacy budgets are 1, 0.6, and 0.2.

We observe that, as the privacy budget decreases, except for PBCN, the released social networks from the XOR mechanism, Pygmalion, and LDPGen become denser, i.e., the edge number increases, average distance shortens, and the nodes tend to cluster together. This transformation is known as graph densification [35]. From Table 2, it is clear that Pygmalion and LDPGen have higher densification speed. For example, when $\epsilon = 0.2$, $G^{\text{FB}}_{0.2}$ released by Pygmalion and LDPGen have Avg. PL. of 2.04 and 1.99, which are smaller than the theoretical mean-shortest path length of a scale-free social graph, i.e., $\ln \ln(N) \approx 2.12$ [47]. The reason is that Pygmalion extracts graph structural information into dK-2-series, which has sensitivity $4d_{\max} + 1$ (c.f. Lemma 1 in [55], $d_{\max}$ is the largest node degree). Pygmalion also assumes the edges are mutually independent and injects i.i.d. noise to perturb the dK-2-series. As a result, the introduced noises have large magnitude and compromise the utility. As for LDPGen, it heavily depends on an accurate estimation of the number of optimal groups (c.f. Section 4.2 in [52]). Specifically, LDPGen applies the Chung-Lu model [3] to calculate the probability of connecting two nodes in the synthesized graph using the group affiliation vectors. Thus, LDPGen may have good performance on preserving the clustering structure of the original graph, but its utility on other network statistics is not guaranteed. The proposed XOR mechanism outperforms Pygmalion and LDPGen, because it characterizes the dependency of edges using the log-linear association parametric matrices, which are optimized to reduce the expected square query error rate on the adjacency matrices. Although PBCN can achieve edge number and density close to the original graphs, it has no utility on distance based statistics (e.g., diameter and Avg. PL), because PBCN depends on the the Havel Theorem (c.f. Theorem 5 in [25]) to generate sanitized graphs from the perturbed degree distributions. However, not every degree sequence has a connected realization [24]. Consequently, the released social networks from PBCN may contains several disconnected components, which gives infinity path length between nodes in different components.

Degree Distribution Estimation. For the proposed XOR mechanism, the decreasing ϵ will reduce the Frobenius norms of $\mathbf{\Theta}$ and $\Lambda_{i,j}$'s, which further weakens the structural information in the released topology. According to Remark 2, when ϵ is close or equal to 0, the released the social networks from the XOR mechanism will be similar to the Erdős-Rényi graph (random graph, where nodes are connected with a given probability).

In order to corroborate the transformation to Erdős-Rényi graph as ϵ decreases, we take G^{FB} as an example and plot the degree distributions of G^{FB}, $G^{\text{FB}}_{0.8}$, $G^{\text{FB}}_{0.3}$, and $G^{\text{FB}}_{0.01}$ in Figure 3. In particular, we show in blue the degree distribution of G^{FB}, $G^{\text{FB}}_{0.8}$, $G^{\text{FB}}_{0.3}$, $G^{\text{FB}}_{0.01}$, and also in red, the degree distribution of the simulated Erdős-Rényi graphs whose mean degrees equal to that of G^{FB}, $G^{\text{FB}}_{0.8}$, $G^{\text{FB}}_{0.3}$, $G^{\text{FB}}_{0.01}$.



(a) G^{FB} (b) $G^{\text{FB}}_{0.8}$ (c) $G^{\text{FB}}_{0.3}$ (d) $G^{\text{FB}}_{0.01}$

Figure 3: Degree distribution ($p(k)$) of G^{FB}, and $G^{\text{FB}}_{0.8}$, $G^{\text{FB}}_{0.3}$, $G^{\text{FB}}_{0.01}$ released by the XOR mechanism compared with simulated Erdős-Rényi graphs. $p(k)$ is the probability of a node having degree k.

From Figure 3, we observe that as ϵ decreases, G^{FB}_{ϵ} transforms to Erdős-Rényi graphs, since the degree distributions change from monotonic to non-monotonic. The reason is that due to the preferential attachment phenomenon [44, 48], real-world social networks are scale-free graphs, whose degree distributions follow the power-law [11, 45, 47], i.e., $p(k) \propto k^{-\gamma}$, where k is a given degree, $p(\cdot)$ indicates the probability mass function, and $\gamma \geq 1$ is the parameter of the power-law distribution. Specifically, G^{FB} is a scale-free graph, whose degree distribution shows two power law regimes (separated by a critical degree $k_{crit} \approx 15$), i.e., $p(k) \propto k^{-1.0}$, if $k < k_{crit}$, and $p(k) \propto k^{-3.3}$, if $k > k_{crit}$. In contrast, the degrees of Erdős-Rényi graphs follow the Poisson distributions [47]. Besides, the degree distribution of G^{FB} deviates the most from the degree distribution of the corresponding simulated Erdős-Rényi graph. As ϵ decreases,

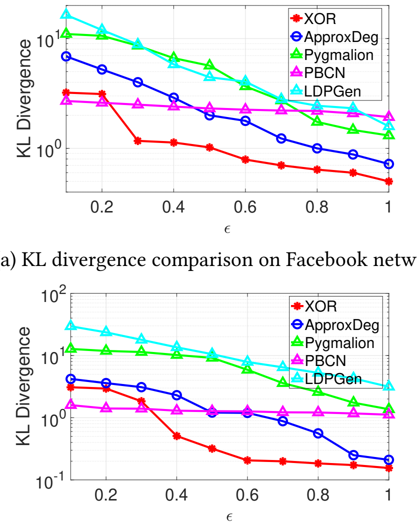
Table 2: Network statistics of the original and differentially private social networks. The statistics in bold indicate that they are the closest to the original value under a specific privacy budget.

	non-pvt	XOR			Pygmalion			PBCN			LDPGen		
statistics	G^{FB}	G_1^{FB}	$G_{0.6}^{\text{FB}}$	$G_{0.2}^{\text{FB}}$	G_1^{FB}	$G_{0.6}^{\text{FB}}$	$G_{0.2}^{\text{FB}}$	G_1^{FB}	$G_{0.6}^{\text{FB}}$	$G_{0.2}^{\text{FB}}$	G_1^{FB}	$G_{0.6}^{\text{FB}}$	$G_{0.2}^{\text{FB}}$
$\lvert\mathcal{E}\rvert(\times 10^5)$	0.88	1.13	1.48	2.73	1.64	2.18	3.91	**0.88**	**0.89**	**0.90**	1.22	1.74	2.87
Diameter	8	**5**	**4**	**4**	4	**4**	3	N/A	N/A	N/A	**5**	4	3
Density	0.011	0.014	0.018	0.034	0.020	0.027	0.048	**0.011**	**0.012**	**0.013**	0.015	0.021	0.035
Avg. PL	3.691	**2.70**	**2.44**	**2.12**	2.34	2.28	2.04	N/A	N/A	N/A	2.61	2.34	1.99

	non-pvt	XOR			Pygmalion			PBCN			LDPGen		
statistics	G^{EM}	G_1^{EM}	$G_{0.6}^{\text{EM}}$	$G_{0.2}^{\text{EM}}$	G_1^{EM}	$G_{0.6}^{\text{EM}}$	$G_{0.2}^{\text{EM}}$	G_1^{EM}	$G_{0.6}^{\text{EM}}$	$G_{0.2}^{\text{EM}}$	G_1^{EM}	$G_{0.6}^{\text{EM}}$	$G_{0.2}^{\text{EM}}$
$\lvert\mathcal{E}\rvert(\times 10^4)$	1.606	1.572	1.679	3.911	3.677	4.554	7.008	**1.633**	**1.647**	**1.693**	3.167	4.720	12.642
Diameter	7	**5**	**3**	**2**	3	**3**	**2**	N/A	N/A	N/A	3	3	2
Density	0.0331	**0.0329**	0.0348	0.0814	0.076	0.094	0.194	0.0337	**0.0334**	**0.0347**	0.065	0.097	0.261
Avg. PL	2.584	**2.565**	**2.421**	**1.922**	1.922	1.877	1.803	N/A	N/A	N/A	2.058	1.930	1.738

the divergence between the degree distributions of G_ϵ^{FB}'s and that of the corresponding simulated Erdős-Rényi graphs also decreases. We evaluate the KL divergence of the two degree distributions in Figure 3 (a), (b), (c) and (d) as 1.74, 1.44, 1.12 and 0.02, respectively.

In Figure 4, we compare the degree distributions obtained using the proposed XOR mechanism with that of ApproxDeg, Pygmalion, PBCN, and LDPGen by varying ϵ from 0.1 to 1. We observe that the degree distributions obtained from the XOR mechanism have the least KL divergence with the original social networks when $\epsilon > 0.3$. Although PBCN achieves lower KL divergence when $\epsilon \leq 0.2$, its performance can hardly be improved with higher privacy budgets. It is because PBCN results in sanitized graphs with disconnected components, which have good matches with nodes with low degrees, yet bad matches with nodes with high degrees, e.g., hubs in the social networks. Pygmalion and LDPGen have the worst performance on both Facebook and Email networks. The reason is that Pygmalion generates and releases synthetic graphs using the noisy dK-2-series, which is essentially 2-node subgraphs with different combinations of node degrees, and uses injected Laplace noises proportional to the maximum node degree. In fact, for Pygmalion to achieve decent performance on degree-based metrics, it requires moderate to high privacy budget, e.g., $\epsilon \geq 5$ (c.f. page 90 [55]). Whereas, LDPGen perturbs the original graph while preserving the strong clustering structures, thus, it introduces large number of extra edges within the same cluster and destroys the power law degree distributions of the original social networks. Although ApproxDeg has similar performance with the XOR mechanism when ϵ is high, its utility degrades fast when ϵ is reduced from 0.4 to 0.1.



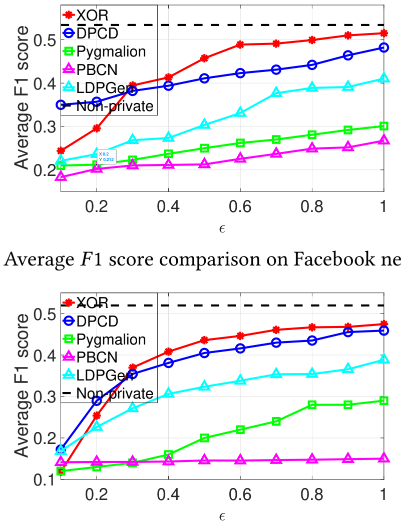
(a) KL divergence comparison on Facebook network

(b) KL divergence comparison on Email network

Figure 4: KL divergence between degree distributions of differentially private released social networks and the non-private ones.

Community Detection. We apply BIGCLAM (Cluster Affiliation Model for Big Networks) [63] to detect communities given a network topology, i.e., the released adjacency matrix. BIGCLAM can efficiently detect communities in social networks by factorizing the adjacency matrix into nonnegative affiliation matrices. The ground-truth communities are the 10 ego-nets in the Facebook network, and the 42 departments affiliation in the Email network.

We compare the community detection results obtained using the XOR mechanism with that of DPCD and other mechanisms in Figure 5 (a) and (b). The privacy budget varies from 0.1 to 1. We see that the XOR mechanism can always achieve a higher average F_1 score when $\epsilon \geq 0.3$. If $\epsilon \geq 0.7$, the XOR mechanism can have utilities that are close to the non-private baselines which use the original adjacency matrix to detect communities. The main reason is that the injected noises in the XOR mechanism retain the structural information in the social network, and the parametric matrices used to generate the noises are also optimized to achieve high query accuracy given the ϵ-differential privacy constraint. Whereas, the elements in the perturbation noises used in DPCD are i.i.d. from Laplace distributions, which ignore such information. Although LDPGen outperforms Pygmalion and PBCN, it inevitably introduces extra edges between communities in the graph synthesizing process



(a) Average $F1$ score comparison on Facebook network

(b) Average $F1$ score comparison Email network

Figure 5: Average F_1 scores on the 10 ego-nets and 42 departments. The black dash lines indicate the non-private baselines, which are 0.53 and 0.504 for G^{FB} and G^{EM}, respectively.

(c.f. Section 4.4 [52]), which generates overlapping communities that are not exist in the original graphs.

9 CONCLUSIONS

In this paper, we have investigated the problem of querying binary- and matrix-valued data from a database in a differentially private manner. To this end, we have proposed a novel output perturbation mechanism, called XOR mechanism, which perturbs the queried results using binary- and matrix-valued noise drawn from a matrix-valued Bernoulli distribution. We have theoretically proved the privacy and utility guarantee for the XOR mechanism. We have also devised a heuristic approach to generate the parameters in the desired distribution so as to minimize the expected square query error rate while satisfying ϵ-differential privacy constraint. Furthermore, we adapted an EHMC based procedure to efficiently generate the perturbation noises attributed to the matrix-valued Bernoulli distribution. Finally, we have evaluated the proposed XOR mechanism experimentally by conducting 2 study cases. Experiment results show that our mechanism provides significantly higher utility compared to the state-of-the-art mechanisms and even achieves comparable utilities to the non-private cases where the original query results are considered.

A PROOF OF THEOREM 1

Before the proof of Theorem 1, we first recall the following Lemma.

Lemma 2. *[33] For any two Hermitian matrices* $\mathbf{A}$ *and* $\mathbf{B}$ *of size* $N \times N$*, let* $\{\lambda_i(\mathbf{A})\}_1^N$ *and* $\{\lambda_i(\mathbf{B})\}_1^N$ *be the sequences of (real) eigenvalues of* $\mathbf{A}$ *and* $\mathbf{B}$ *in a non-increasing order. Then,* $\sum_i^N \lambda_i(\mathbf{A})\lambda_{N-i+1}(\mathbf{B}) \leq \text{Tr}[\mathbf{AB}] \leq \sum_i^N \lambda_i(\mathbf{A})\lambda_i(\mathbf{B})$*.*

Now, we present the proof of Theorem 1.

Proof of Theorem 1. XOR mechanism guarantees ϵ-differential privacy if $\Pr[f(D) \oplus \mathcal{B}_D \in \mathcal{S}] \leq e^{\epsilon} \Pr[f(D') \oplus \mathcal{B}_{D'} \in \mathcal{S}]$, where D and D' are an arbitrary pair of neighboring datasets, $\mathcal{B}_D$ (resp. $\mathcal{B}_{D'}$) is the noise used to perturb the query $f(D)$ (resp. $f(D')$) such that $f(D) \oplus \mathcal{B}_D = f(D') \oplus \mathcal{B}_{D'} = \mathbf{Y} \in \mathcal{S}$, and $\mathcal{S}$ stands for the range of $\mathbb{XOR}(f(D), \mathcal{B}), \forall D \in \mathcal{D}$, $\mathcal{B} \sim \text{Ber}_{N,P}(\mathbf{\Theta}, \mathbf{\Lambda}_{1,2}, \cdots, \mathbf{\Lambda}_{N-1,N})$. Due to the property of XOR operation, we have $\mathcal{B}_D = \mathbf{Y} \oplus f(D)$ and $\mathcal{B}_{D'} = \mathbf{Y} \oplus f(D')$. According to Definition 4, we obtain
$\int_{\mathcal{S}} \exp\{\text{Tr}[(\mathbf{Y} \oplus f(D))\, \mathbf{\Theta} (\mathbf{Y} \oplus f(D))^T] + \sum_{i=1}^N \sum_{i \neq j}^N \text{Tr}[\mathbf{J}_{ij} (\mathbf{Y} \oplus f(D))\, \mathbf{\Lambda}_{i,j} (\mathbf{Y} \oplus f(D))^T]\} d\,\mathbf{Y} \leq$
$e^{\epsilon} \int_{\mathcal{S}} \exp\{\text{Tr}[(\mathbf{Y} \oplus f(D'))\, \mathbf{\Theta} (\mathbf{Y} \oplus f(D'))^T] + \sum_{i=1}^N \sum_{i \neq j}^N \text{Tr}[\mathbf{J}_{ij} (\mathbf{Y} \oplus f(D'))\, \mathbf{\Lambda}_{i,j} (\mathbf{Y} \oplus f(D'))^T]\} d\,\mathbf{Y}$,
which is $\int_{\mathcal{S}} \exp\{\text{Tr}[\mathbf{B}_D\, \mathbf{\Theta}\, \mathbf{B}_D^T] + \sum_{i=1}^N \sum_{i \neq j}^N \text{Tr}[\mathbf{J}_{ij}\, \mathbf{B}_D\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_D^T]\} d\,\mathbf{Y} \leq$
$e^{\epsilon} \int_{\mathcal{S}} \exp\{\text{Tr}[\mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] + \sum_{i=1}^N \sum_{i \neq j}^N \text{Tr}[\mathbf{J}_{ij}\, \mathbf{B}_{D'}\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_{D'}^T]\} d\,\mathbf{Y}$.
Inserting $\frac{\exp\{\text{Tr}[\mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] + \sum_{i=1}^N \sum_{j \neq i}^N \text{Tr}[\mathbf{J}_{ij}\, \mathbf{B}_{D'}\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_{D'}^T]\}}{\exp\{\text{Tr}[\mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] + \sum_{i=1}^N \sum_{j \neq i}^N \text{Tr}[\mathbf{J}_{ij}\, \mathbf{B}_{D'}\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_{D'}^T]\}}$ into the integral on the left hand side of the above inequality, we can arrive at
$\frac{\exp\{\text{Tr}[\mathbf{B}_D\, \mathbf{\Theta}\, \mathbf{B}_D^T] + \sum_{i=1}^N \sum_{j \neq i}^N \text{Tr}[\mathbf{J}_{ij}\, \mathbf{B}_D\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_D^T]\}}{\exp\{\text{Tr}[\mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] + \sum_{i=1}^N \sum_{j \neq i}^N \text{Tr}[\mathbf{J}_{ij}\, \mathbf{B}_{D'}\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_{D'}^T]\}} \leq e^{\epsilon}, \forall\, \mathbf{Y} \in \mathcal{S}$. With further simplifications, we have the sufficient condition for the XOR mechanism to satisfy ϵ-differential privacy as

$$\text{Tr}[\mathbf{B}_D\, \mathbf{\Theta}\, \mathbf{B}_D^T - \mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] + \sum_{i=1}^N \sum_{j \neq i}^N \text{Tr}[\mathbf{J}_{ij} (\mathbf{B}_D\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_D^T - \mathbf{B}_{D'}\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_{D'}^T)] \leq \epsilon, \quad (7)$$

which needs to hold for all pairs of neighboring datasets D and D' and all $\mathbf{Y} \in \mathcal{S}$. Next, we analyze the two separated parts in (7).

First, we derive an upper bound of the first part of (7).

$$\text{Tr}[\mathbf{B}_D\, \mathbf{\Theta}\, \mathbf{B}_D^T - \mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] = \text{Tr}[\mathbf{\Theta}\, \mathbf{B}_D^T\, \mathbf{B}_D - \mathbf{\Theta}\, \mathbf{B}_{D'}^T\, \mathbf{B}_{D'}]$$
$$\overset{(a)}{\leq} \sum_{p=1}^P \lambda_p(\mathbf{\Theta})\lambda_p\Big(\mathbf{B}_D^T\, \mathbf{B}_D\Big) - \sum_{p=1}^P \lambda_p(\mathbf{\Theta})\lambda_{P+1-p}\Big(\mathbf{B}_{D'}^T\, \mathbf{B}_{D'}\Big)$$
$$\overset{(b)}{=} \sum_{p=1}^P \lambda_p(\mathbf{\Theta})\Big[\Big(\sigma_p(\mathbf{B}_D)\Big)^2 - \Big(\sigma_{P+1-p}(\mathbf{B}_{D'})\Big)^2\Big]$$
$$\overset{(c)}{\leq} \sqrt{\sum_{p=1}^P \Big(\lambda_p(\mathbf{\Theta})\Big)^2} \sqrt{\sum_{p=1}^P \Big[\Big(\sigma_p(\mathbf{B}_D)\Big)^2 - \Big(\sigma_{P+1-p}(\mathbf{B}_{D'})\Big)^2\Big]^2}$$
$$\leq ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 \sqrt{\Big[\sum_{p=1}^P \Big(\sigma_p(\mathbf{B}_D)\Big)^2 - \Big(\sigma_{P+1-p}(\mathbf{B}_{D'})\Big)^2\Big]^2}$$
$$= ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 \Big|\sum_{p=1}^P \Big(\sigma_p(\mathbf{B}_D)\Big)^2 - \Big(\sigma_{P+1-p}(\mathbf{B}_{D'})\Big)^2\Big|$$
$$\overset{(d)}{=} ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 \Big| ||\, \mathbf{B}_D\, ||_F^2 - ||\, \mathbf{B}'_D\, ||_F^2 \Big| \leq ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 ||\, \mathbf{B}_D - \mathbf{B}'_D\, ||_F^2 \overset{(e)}{\leq} ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2 s_f,$$

where (a) follows from Lemma 2, (b) is because $\lambda_p(\mathbf{X}^T\, \mathbf{X}) = (\sigma_p(\mathbf{X}))^2$ for any matrix $\mathbf{X}$, (c) is due to the Cauchy–Schwarz inequality, (d) follows from $||\boldsymbol{\sigma}(\mathbf{X})||_2^2 = ||\,\mathbf{X}\,||_F^2$, and (e) is because $f(D) \oplus \mathbf{B}_D = f(D') \oplus \mathbf{B}_{D'} = \mathbf{Y} \in \mathcal{S}$, which gives $\mathbf{B}_D = f(D) \oplus \mathbf{Y}$ and $\mathbf{B}_{D'} = f(D') \oplus \mathbf{Y}$ (property of the XOR operator), then, based on Definition 3, there are also at most s_f different binary entries between $\mathbf{B}_D$ and $\mathbf{B}_{D'}$, i.e., $||\, \mathbf{B}_D - \mathbf{B}'_D\, ||_F^2 \leq s_f$. Thus, we arrive at

$$\text{Tr}[\mathbf{B}_D\, \mathbf{\Theta}\, \mathbf{B}_D^T - \mathbf{B}_{D'}\, \mathbf{\Theta}\, \mathbf{B}_{D'}^T] \leq s_f ||\boldsymbol{\lambda}(\mathbf{\Theta})||_2. \quad (8)$$

Now, we continue to bound the second part of (7). First, since $\mathbf{J}_{ij}$ has only one non-zero entry at position (i, j), we have

$$\text{Tr}[\mathbf{J}_{ij} (\mathbf{B}_D\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_D^T)] = \mathbf{B}_D[j, :]\, \mathbf{\Lambda}_{i,j} \Big(\mathbf{B}_D[i, :]\Big)^T, \forall i \neq j, \quad (9)$$

where $\mathbf{B}_D[i, :]$ ($\mathbf{B}_D[j, :]$) is a row vector representing the ith (jth) row of $\mathbf{B}_D$. Since $\mathbf{\Lambda}_{i,j}$ is symmetric and $\mathbf{\Lambda}_{i,j} = \mathbf{\Lambda}_{j,i}^T$, which gives $\mathbf{\Lambda}_{i,j} = \mathbf{\Lambda}_{j,i}$ (cf. Definition 4). To invoke Lemma 2 (that requires Hermitian matrices), we consider $\mathbf{\Lambda}_{i,j}$ and $\mathbf{\Lambda}_{j,i}$ jointly and have
$\sum_{i=1}^N \sum_{j \neq i}^N \text{Tr}[\mathbf{J}_{ij} (\mathbf{B}_D\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_D^T - \mathbf{B}_{D'}\, \mathbf{\Lambda}_{i,j}\, \mathbf{B}_{D'}^T)]$
$= \sum_{i=1}^{N-1} \sum_{j=i+1}^N \text{Tr}\Big[\, \mathbf{\Lambda}_{i,j} \big(\mathbf{B}_D[i, :]^T\, \mathbf{B}_D[j, :] + \mathbf{B}_D[j, :]^T\, \mathbf{B}_D[i, :]\big) -$

$\boldsymbol{\Lambda}_{i,j}\left(\mathbf{B}_{D'}[i,:]^T\,\mathbf{B}_{D'}[j,:] + \mathbf{B}_{D'}[j,:]^T\,\mathbf{B}_{D'}[i,:]\right)\Big]$
$\overset{(a)}{=} \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \mathrm{Tr}\Big[\boldsymbol{\Lambda}_{i,j}\left((\mathbf{B}_D[i,:] + \mathbf{B}_D[j,:])^T(\mathbf{B}_D[i,:] +\right.$
$\left.\mathbf{B}_D[j,:])\right) - \boldsymbol{\Lambda}_{i,j}\left((\mathbf{B}_{D'}[i,:] + \mathbf{B}_{D'}[j,:])^T(\mathbf{B}_{D'}[i,:] + \mathbf{B}_{D'}[j,:])\right)\Big]$
$\overset{(b)}{=} \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \mathrm{Tr}\Big[\boldsymbol{\Lambda}_{i,j}\Big((\mathbf{B}_D[i,:] + \mathbf{B}_D[j,:])^T(\mathbf{B}_D[i,:] +$
$\mathbf{B}_D[j,:]) - (\mathbf{B}_{D'}[i,:] + \mathbf{B}_{D'}[j,:])^T(\mathbf{B}_{D'}[i,:] + \mathbf{B}_{D'}[j,:])\Big)\Big]$
$\overset{(c)}{\leq} \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left(\sum_{p=1}^{P}\lambda_p(\boldsymbol{\Lambda}_{i,j})\Big[\Big(\lambda_p\big(\mathbf{B}_D[i,:] + \mathbf{B}_D[j,:]\big)\Big)^2 - \Big(\lambda_{P+1-p}\big(\mathbf{B}_{D'}[i,:] + \mathbf{B}_{D'}[j,:]\big)\Big)^2\Big]\right)$
$\overset{(d)}{\leq} \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left(||\boldsymbol{\lambda}(\boldsymbol{\Lambda}_{i,j})||_2\Big\|\mathbf{B}_D[i,:] + \mathbf{B}_D[j,:] - \mathbf{B}_{D'}[i,:] - \mathbf{B}_{D'}[j,:]\Big\|_2^2\right)$
$\overset{(e)}{\leq} s_f\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}||\boldsymbol{\lambda}(\boldsymbol{\Lambda}_{i,j})||_2,$
where (a) is because $\mathrm{Tr}\left[\boldsymbol{\Lambda}_{i,j}\,\mathbf{B}_D[i,:]^T\,\mathbf{B}_D[i,:]\right] = \mathrm{Tr}\left[\boldsymbol{\Lambda}_{i,j}\,\mathbf{B}_D[j,:]^T\,\mathbf{B}_D[j,:]\right]$ $= \mathrm{Tr}\left[\boldsymbol{\Lambda}_{i,j}\,\mathbf{B}_{D'}[i,:]^T\,\mathbf{B}_{D'}[i,:]\right] = \mathrm{Tr}\left[\boldsymbol{\Lambda}_{i,j}\,\mathbf{B}_{D'}[j,:]^T\,\mathbf{B}_{D'}[j,:]\right] = 0, \forall i \neq j$, and the operands of $\mathrm{Tr}(\cdot)$ in (b) is the multiplication of two Hermitian matrices, (c) is due to Lemma 2, and (d) and (e) can be obtained by following the last 6 steps when bounding the first part of (7) and we omit them due to space limit. Combing the upper bounds of the two separate terms in (7), we complete the proof. □

B PROOF OF THEOREM 2

Before the proof of Theorem 2, we first recall the definition of Exponential family distribution and its property.

Definition 7. *(Exponential family [7]). An exponential family is a set of probability distributions whose PDFs can be expressed as* $f(\mathbf{x}|\boldsymbol{\eta}) = h(\mathbf{x})c(\boldsymbol{\eta})\exp\left(\sum_{i=1}^{k}\omega_i(\boldsymbol{\eta})t_i(\mathbf{x})\right)$*, where* $h(\mathbf{x}) \geq 0$*,* $t_i(\mathbf{x})$ *and* $\omega_i(\boldsymbol{\eta})$ $(i \in \{1, 2, \cdots, k\})$ *are real-valued functions only on* $\mathbf{x}$ *and parameter* $\boldsymbol{\eta}$*, respectively.*

Theorem 5. *[7] If* $\mathbf{X}$ *is a random variable with PDF of the form given in Definition 7, then* $\mathbb{E}[\sum_i^k \frac{\partial\omega_i(\boldsymbol{\eta})}{\partial\eta_j}t_i(\mathbf{X})] = -\frac{\partial}{\partial\eta_j}\log c(\boldsymbol{\eta})$*.*

Now, we provide the proof of Theorem 2.

Proof of Theorem 2. First, we show that the multivariate Bernoulli distribution is an exponential family distribution. This can be done by rewriting its PDF in (2) as
$\Pr[\mathrm{vec}(\mathcal{B}^T) = \mathbf{b}] = C(\boldsymbol{\Pi})\exp\{\mathbf{b}^T\,\boldsymbol{\Pi}\,\mathbf{b}\} = C(\boldsymbol{\Pi})\exp\{\mathrm{Tr}[\mathbf{b}^T\,\boldsymbol{\Pi}\,\mathbf{b}]\}$
$= C(\boldsymbol{\Pi})\exp\{\mathrm{Tr}[\boldsymbol{\Pi}\,\mathbf{b}\,\mathbf{b}^T]\} = C(\boldsymbol{\Pi})\exp\left\{\sum_{u=1}^{(NP)^2}\{\boldsymbol{\Pi}^T\}_u\{\mathbf{b}\,\mathbf{b}^T\}_u\right\}$,
where $\{\boldsymbol{\Pi}^T\}_u$ and $\{\mathbf{b}\,\mathbf{b}^T\}_u$ are the u-th element of $\boldsymbol{\Pi}^T$ and $\mathbf{b}\,\mathbf{b}^T$, respectively. By setting $h(\mathbf{b}) = 1$, $c(\boldsymbol{\eta}) = C(\boldsymbol{\Pi})$, $\omega_u(\boldsymbol{\eta}) = \{\boldsymbol{\Pi}^T\}_u$ and $t_u(\mathbf{b}) = \{\mathbf{b}\,\mathbf{b}^T\}_u$, we can represent $\Pr[\mathrm{vec}(\mathcal{B}^T) = \mathbf{b}]$ as the form of the PDF in Definition 7 with $k = (NP)^2$, which suggests that the multivariate Bernoulli distribution is in exponential family.

Next, we can have $r(f(D), \mathcal{B}) = \frac{||\overline{f(D)}\circ\mathcal{B}+f(D)\circ\overline{\mathcal{B}}-f(D)||_F^2}{||f(D)||_F^2} = \frac{||(\mathbf{1}_{N\times P}-2f(D))\circ\mathcal{B}||_F^2}{||f(D)||_F^2} \overset{*}{=} \frac{||\mathcal{B}||_F^2}{||f(D)||_F^2}$, where $*$ is because $(\mathbf{1}_{N\times P} - 2f(D)) \in \{+1, -1\}^{N\times P}$ and then the Hadamard product only changes signs of elements of $\mathcal{B}$. As a result, we only need to calculate the expectation of $||\mathcal{B}||_F^2$, i.e., $\mathbb{E}[||\mathcal{B}||_F^2]$.

Since the multivariate Bernoulli distribution is an exponential family distribution, then, by applying Theorem 5, we have

$$\mathbb{E}\left[\sum_{u=1}^{(NP)^2}\frac{\partial\{\boldsymbol{\Pi}^T\}_u}{\partial\boldsymbol{\Pi}}\{\mathrm{vec}(\mathcal{B}^T)\mathrm{vec}(\mathcal{B}^T)^T\}_u\right] = -\frac{\partial}{\partial\boldsymbol{\Pi}}\log(C(\boldsymbol{\Pi})). \quad (10)$$

For the left-hand-side (LHS) of (10), we have LHS = $\mathbb{E}\big[\sum_{u=1}^{(NP)^2}\{\mathbf{1}^T\}_u\{\mathrm{vec}(\mathcal{B}^T)\mathrm{vec}(\mathcal{B}^T)^T\}_u\big] = \mathbb{E}\big[\mathrm{vec}(\mathcal{B}^T)\mathrm{vec}(\mathcal{B}^T)^T\big]$, where $\mathbf{1}$ is an all-ones matrix of size $NP\times NP$, and it is easy to check that $\mathbb{E}[||\mathcal{B}||_F^2] = \mathrm{Tr}[\mathrm{LHS}]$. For the right-hand-side (RHS) of (10), we have RHS $= C(\boldsymbol{\Pi})\big(\sum_{\mathbf{b}_k\in\mathcal{T}}e^{\{\mathbf{b}_k^T\,\boldsymbol{\Pi}\,\mathbf{b}_k\}}\,\mathbf{b}_k\,\mathbf{b}_k^T\big)$. Hence, we have $\mathbb{E}[||\mathcal{B}||_F^2] = \mathrm{Tr}[\mathrm{LHS}] = \mathrm{Tr}[\mathrm{RHS}] = C(\boldsymbol{\Pi})\big(\sum_{\mathbf{b}_k\in\mathcal{T}}e^{\{\mathbf{b}_k^T\,\boldsymbol{\Pi}\,\mathbf{b}_k\}}||\mathbf{b}_k||_2^2\big)$. According to (3), $C(\boldsymbol{\Pi}) = [\sum_{\mathbf{b}_k\in\mathcal{T}}\exp\{\mathbf{b}_k^T\,\boldsymbol{\Pi}\,\mathbf{b}_k\}]^{-1}$. □

C PROOF OF PROPOSITION 1

Proof of Proposition 1. For the numerator of $g(\Pi)$, we have
$\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\boldsymbol{\Pi}\,\mathbf{b}_k)||\mathbf{b}_k||_2^2 = \sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathrm{Tr}[\boldsymbol{\Pi}\,\mathbf{b}_k\,\mathbf{b}_k^T])||\mathbf{b}_k||_2^2$
$\overset{(a)}{\leq} \sum_{\mathbf{b}_k\in\mathcal{T}}\exp\left(\lambda_{\max}(\boldsymbol{\Pi})\lambda_{\max}(\mathbf{b}_k\,\mathbf{b}_k^T)\right)||\mathbf{b}_k||_2^2$
$= \sum_{\mathbf{b}_k\in\mathcal{T}}\exp\left(\lambda_{\max}(\boldsymbol{\Pi})||\mathbf{b}_k||_2^2\right)||\mathbf{b}_k||_2^2$
$\overset{(b)}{\leq} \sum_{\mathbf{b}_k\in\mathcal{T}}\exp\left((\lambda_{\max}(\boldsymbol{\Theta}) + 2\sum_{i=1}^{N-1}\sum_{j>i}^{N}\lambda_{\max}(\boldsymbol{\Lambda}_{\mathbf{i},\mathbf{j}}))||\mathbf{b}_k||_2^2\right)||\mathbf{b}_k||_2^2 = U(\Pi)$,
where (a) can be obtained by applying Lemma 2 in Appendix A and using the fact that $\mathbf{b}_k\,\mathbf{b}_k^T$ is rank 1, and (b) is because $\lambda_i(\mathbf{X}+\mathbf{Y}) \leq \lambda_i(\mathbf{X}) + \lambda_i(\mathbf{Y}), \forall\,\mathbf{X}, \mathbf{Y} \in \mathcal{PD}$ and $\lambda_{\max}(\mathbf{X}\otimes\mathbf{Y}) = \lambda_{\max}(\mathbf{X})\lambda_{\max}(\mathbf{Y})$.

For the denominator of $g(\Pi)$, we have
$\sum_{\mathbf{b}_k\in\mathcal{T}}\exp(\mathbf{b}_k^T\,\boldsymbol{\Pi}\,\mathbf{b}_k) \overset{(a)}{\geq} \sum_{\mathbf{b}_k\in\mathcal{T}}\left(1 + \mathbf{b}_k^T\,\boldsymbol{\Pi}\,\mathbf{b}_k\right)$
$= 2^{NP} + \sum_{\mathbf{b}_k\in\mathcal{T}}\mathrm{Tr}[\boldsymbol{\Pi}\,\mathbf{b}_k\,\mathbf{b}_k^T] = 2^{NP} + \mathrm{Tr}\left[\boldsymbol{\Pi}\sum_{\mathbf{b}_k\in\mathcal{T}}(\mathbf{b}_k\,\mathbf{b}_k^T)\right]$
$= 2^{NP} + 2^{NP-2}\,\mathrm{Tr}[\Pi] + 2^{NP-2}\mathrm{sum}(\Pi)$
$= 2^{NP-2}\left(4 + N\,\mathrm{Tr}[\boldsymbol{\Theta}] + N\mathrm{sum}(\boldsymbol{\Theta}) + \sum_{i=1}^{N}\sum_{j\neq i}^{N}\mathrm{sum}(\boldsymbol{\Lambda}_{i,j})\right)$
$\overset{(b)}{\geq} 2^{NP-2}\left(4 + N\,\mathrm{Tr}[\boldsymbol{\Theta}] + N\,\mathrm{Tr}[\boldsymbol{\Theta}] + \sum_{i=1}^{N}\sum_{j\neq i}^{N}\mathrm{Tr}[\boldsymbol{\Lambda}_{i,j}]\right)$
$\overset{(c)}{\geq} 2^{NP-1}\left(2 + N||\boldsymbol{\lambda}(\boldsymbol{\Theta})||_2 + \sum_{i=1}^{N-1}\sum_{j>i}^{N}||\boldsymbol{\lambda}(\boldsymbol{\Lambda}_{i,j})||_2\right) = L(\Pi)$,
where (a) follows from the Taylor expansion, $\mathrm{sum}(\cdot)$ is the operator that sums all elements in a matrix, (b) is because $\boldsymbol{\Theta} > 0, \boldsymbol{\Lambda}_{i,j} > 0$, and (c) is because $\boldsymbol{\Theta}, \boldsymbol{\Lambda}_{i,j} \in \mathcal{PD}^{P\times P}$. As a result, we have $g(\Pi) \leq U(\Pi)/L(\Pi)$, which completes the proof. □

D PROOF OF THEOREM 4

Proof of Theorem 4. Mathematically, we have
$\mathrm{PrvcLkg} = \max_{a\in\{0,1\}}\Pr(\widehat{\mathbf{A}_{i,j}} = a|\mathcal{O}, \mathbf{A}_{/\{i,j\}})$
$= \max_{a\in\{0,1\}}\frac{\Pr(\mathcal{O}|\widehat{\mathbf{A}_{i,j}}=a,\mathbf{A}_{/\{i,j\}})\Pr(\widehat{\mathbf{A}_{i,j}}=a,\mathbf{A}_{/\{i,j\}})}{\Pr(\mathcal{O},\mathbf{A}_{/\{i,j\}})}$
$= \max_{a\in\{0,1\}}\frac{\Pr(\mathcal{O}|\widehat{\mathbf{A}_{i,j}}=a,\mathbf{A}_{/\{i,j\}})}{\Pr(\mathcal{O}|\mathbf{A}_{/\{i,j\}})}\Pr(\widehat{\mathbf{A}_{i,j}} = a|\mathbf{A}_{/\{i,j\}})$
$= \max_{a\in\{0,1\}}\boxed{\frac{\Pr(\mathcal{O}|\widehat{\mathbf{A}_{i,j}} = a, \mathbf{A}_{/\{i,j\}})}{\Pr(\mathcal{O}|\widehat{\mathbf{A}_{i,j}} = \overline{a}, \mathbf{A}_{/\{i,j\}})}}\frac{\Pr(\mathcal{O}|\widehat{\mathbf{A}_{i,j}}=\overline{a},\mathbf{A}_{/\{i,j\}})}{\Pr(\mathcal{O}|\mathbf{A}_{/\{i,j\}})}\Pr(\widehat{\mathbf{A}_{i,j}} = a|\mathbf{A}_{/\{i,j\}})$
$\overset{(*)}{=} \max_{a\in\{0,1\}}\boxed{e^{\epsilon}}\frac{\Pr(\mathcal{O},\widehat{\mathbf{A}_{i,j}}=\overline{a},\mathbf{A}_{/\{i,j\}})\Pr(\mathbf{A}_{/\{i,j\}}))}{\Pr(\widehat{\mathbf{A}_{i,j}}=\overline{a},\mathbf{A}_{/\{i,j\}})\Pr(\mathcal{O},\mathbf{A}_{/\{i,j\}})}\Pr(\widehat{\mathbf{A}_{i,j}} = a|\mathbf{A}_{/\{i,j\}})$
$= \max_{a\in\{0,1\}}e^{\epsilon}\boxed{\Pr(\widehat{\mathbf{A}_{i,j}} = \overline{a}|\mathcal{O}, \mathbf{A}_{/\{i,j\}})}\frac{\Pr(\widehat{\mathbf{A}_{i,j}}=a|\mathbf{A}_{/\{i,j\}})}{\Pr(\widehat{\mathbf{A}_{i,j}}=\overline{a}|\mathbf{A}_{/\{i,j\}})}$
$\overset{(\#)}{=} \max_{a\in\{0,1\}}e^{\epsilon}\boxed{(1 - \mathrm{PrvcLkg})}\zeta$, which can be further simplified as $\mathrm{PrvcLkg} = \max\{\frac{1}{\zeta e^{\epsilon}+1}, \frac{\zeta e^{\epsilon}}{\zeta e^{\epsilon}+1}\}$. Note that at line $*$, we apply the definition of ϵ-differential privacy. At line #, $\zeta = \frac{\Pr(\widehat{\mathbf{A}_{i,j}}=a|\mathbf{A}_{/\{i,j\}})}{\Pr(\widehat{\mathbf{A}_{i,j}}=\overline{a}|\mathbf{A}_{/\{i,j\}})}$ is the ratio between prior probabilities, which is independent of the adopted ϵ-edge-differentially private mechanism. □

REFERENCES

[1] Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. 2018. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems*. 7564–7575.

[2] Faraz Ahmed, Alex X Liu, and Rong Jin. 2016. Social graph publishing with privacy guarantees. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 447–456.

[3] William Aiello, Fan Chung, and Linyuan Lu. 2000. A random graph model for massive graphs. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. 171–180.

[4] Raman Arora and Jalaj Upadhyay. 2019. On differentially private graph sparsification and applications. In *Advances in Neural Information Processing Systems*. 13399–13410.

[5] Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. 2013. Differentially private data analysis of social networks via restricted sensitivity. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*. 87–96.

[6] Petros T Boufounos and Richard G Baraniuk. 2008. 1-bit compressive sensing. In *2008 42nd Annual Conference on Information Sciences and Systems*. IEEE, 16–21.

[7] George Casella and Roger L Berger. 2002. *Statistical inference*. Vol. 2. Duxbury Pacific Grove, CA.

[8] Thee Chanyaswad, Alex Dytso, H. Vincent Poor, and Prateek Mittal. 2018. MVG Mechanism: Differential Privacy Under Matrix-Valued Query. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (Toronto, Canada) *(CCS '18)*. ACM, New York, NY, USA, 230–246. https://doi.org/10.1145/3243734.3243750

[9] Kamalika Chaudhuri and Claire Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in neural information processing systems*. 289–296.

[10] David R Cox. 1972. The analysis of multivariate binary data. *Applied statistics* 21, 2 (1972), 113–120.

[11] Gábor Csányi and Balázs Szendrői. 2004. Structure of a large social network. *Physical Review E* 69, 3 (2004), 036131.

[12] Wei-Yen Day, Ninghui Li, and Min Lyu. 2016. Publishing graph degree distribution with node differential privacy. In *Proceedings of the 2016 International Conference on Management of Data*. 123–138.

[13] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 486–503.

[14] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[15] Cynthia Dwork and Aaron Roth. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

[16] Junbin Fang, Aiping Li, and Qianyue Jiang. 2019. GDAGAN: An Anonymization Method for Graph Data Publishing Using Generative Adversarial Network. In *2019 6th International Conference on Information Science and Control Engineering (ICISCE)*. IEEE, 309–313.

[17] Jun Fang, Yanning Shen, Hongbin Li, and Zhi Ren. 2014. Sparse signal recovery from one-bit quantized data: An iterative reweighted algorithm. *Signal Processing* 102 (2014), 201–206.

[18] Johannes Gehrke, Michael Hay, Edward Lui, and Rafael Pass. 2012. Crowd-blending privacy. In *Annual Cryptology Conference*. Springer, 479–496.

[19] Johannes Gehrke, Edward Lui, and Rafael Pass. 2011. Towards privacy for social networks: A zero-knowledge based definition of privacy. In *Theory of Cryptography Conference*. Springer, 432–449.

[20] Moritz Hardt, Katrina Ligett, and Frank McSherry. 2012. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems*. 2339–2347.

[21] Moritz Hardt and Guy N Rothblum. 2010. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 61–70.

[22] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. 2009. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining*. IEEE, 169–178.

[23] Xi He, Ashwin Machanavajjhala, and Bolin Ding. 2014. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 1447–1458.

[24] Szabolcs Horvát and Carl D Modes. 2020. Connectivity matters: Construction and exact random sampling of connected graphs. *arXiv preprint arXiv:2009.03747* 1, 1 (2020), 26.

[25] Haiping Huang, Dong Jun Zhang, Fu Xiao, Kai Wang, Jiateng Gu, and Ruchuan Wang. 2020. Privacy-preserving Approach PBCN in Social Network with Differential Privacy. *IEEE Transactions on Network and Service Management* 17, 2 (2020), 15.

[26] Masooma Iftikhar, Qing Wang, and Yu Lin. 2020. dK-Microaggregation: Anonymizing Graphs with Differential Privacy Guarantees. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 191–203.

[27] Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. 2020. Locally Differentially Private Analysis of Graph Statistics. *arXiv preprint arXiv:2010.08688* (2020), 26.

[28] Tianxi Ji, Changqing Luo, Yifan Guo, Jinlong Ji, Weixian Liao, and Pan Li. 2019. Differentially Private Community Detection in Attributed Social Networks. In *Asian Conference on Machine Learning*. 16–31.

[29] Tianxi Ji, Changqing Luo, Yifan Guo, Qianlong Wang, Lixing Yu, and Pan Li. 2020. Community detection in online social networks: a differentially private and parsimonious approach. *IEEE transactions on computational social systems* 7, 1 (2020), 151–163.

[30] Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. 2011. Private analysis of graph structure. *Proceedings of the VLDB Endowment* 4, 11 (2011), 1146–1157.

[31] Daniel Kifer and Ashwin Machanavajjhala. 2014. Pufferfish: A framework for mathematical privacy definitions. *ACM Transactions on Database Systems (TODS)* 39, 1 (2014), 3.

[32] Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.

[33] Jean B Lasserre. 1995. A trace inequality for matrix product. *IEEE Trans. Automat. Control* 40, 8 (1995), 1500–1501.

[34] Benedict Leimkuhler and Sebastian Reich. 2004. *Simulating hamiltonian dynamics*. Vol. 14. Cambridge university press.

[35] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 2–es.

[36] Jure Leskovec and Julian J Mcauley. 2012. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*. 539–547.

[37] Ninghui Li, Wahbeh Qardaji, Dong Su, Yi Wu, and Weining Yang. 2013. Membership privacy: a unifying framework for privacy definitions. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 889–900.

[38] Yanan Li, Xuebin Ren, Shusen Yang, and Xinyu Yang. 2019. Impact of Prior Knowledge and Data Correlation on Privacy Leakage: A Unified Analysis. *IEEE Transactions on Information Forensics and Security* 14, 9 (2019), 2342–2357.

[39] Changchang Liu, Supriyo Chakraborty, and Prateek Mittal. 2016. Dependence Makes You Vulnberable: Differential Privacy Under Dependent Tuples.. In *NDSS*, Vol. 16. 21–24.

[40] Fang Liu. 2018. Generalized gaussian mechanism for differential privacy. *IEEE Transactions on Knowledge and Data Engineering* 31, 4 (2018), 747–756.

[41] Gianfranco Lovison. 2006. A matrix-valued Bernoulli distribution. *Journal of Multivariate Analysis* 97, 7 (2006), 1573–1585.

[42] Priya Mahadevan, Dmitri Krioukov, Kevin Fall, and Amin Vahdat. 2006. Systematic topology analysis and generation using degree correlations. *ACM SIGCOMM Computer Communication Review* 36, 4 (2006), 135–146.

[43] Kimberly McCullough. 2016. Why Government Use of Social Media Monitoring Software Is a Direct Threat to Our Liberty and Privacy. https://www.aclu.org/blog/privacy-technology/surveillance-technologies/why-government-use-social-media-monitoring

[44] Alan Mislove, Hema Swetha Koppula, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2008. Growth of the flickr social network. In *Proceedings of the first workshop on Online social networks*. ACM, 25–30.

[45] Lev Muchnik, Sen Pei, Lucas C Parra, Saulo DS Reis, José S Andrade Jr, Shlomo Havlin, and Hernán A Makse. 2013. Origins of power-law degree distribution in the heterogeneity of human activity in social networks. *Scientific reports* 3 (2013), 1783.

[46] Michael J Neely. 2012. Asynchronous control for coupled Markov decision systems. In *2012 IEEE Information Theory Workshop*. IEEE, 287–291.

[47] Mark Newman. 2018. *Networks*. Oxford university press.

[48] Mark EJ Newman. 2001. Clustering and preferential attachment in growing networks. *Physical review E* 64, 2 (2001), 025102.

[49] Aleksandar Nikolov, Kunal Talwar, and Li Zhang. 2013. The geometry of differential privacy: the sparse and approximate cases. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM, 351–360.

[50] Ari Pakman and Liam Paninski. 2013. Auxiliary-variable exact Hamiltonian Monte Carlo samplers for binary distributions. In *Advances in neural information processing systems*. 2490–2498.

[51] Ari Pakman and Liam Paninski. 2014. Exact hamiltonian monte carlo for truncated multivariate gaussians. *Journal of Computational and Graphical Statistics* 23, 2 (2014), 518–542.

[52] Zhan Qin, Ting Yu, Yin Yang, Issa Khalil, Xiaokui Xiao, and Kui Ren. 2017. Generating synthetic decentralized social graphs with local differential privacy. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 425–438.

[53] Sofya Raskhodnikova and Adam Smith. 2016. Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE,

495–504.

[54] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*. Springer, 525–542.

[55] Alessandra Sala, Xiaohan Zhao, Christo Wilson, Haitao Zheng, and Ben Y Zhao. 2011. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. 81–98.

[56] Jeffrey Curtis Schlimmer. 1987. *Concept Acquisition through Representational Adjustment*. Ph.D. Dissertation. AAI8724747.

[57] Shuang Song, Yizhen Wang, and Kamalika Chaudhuri. 2017. Pufferfish privacy mechanisms for correlated data. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 1291–1306.

[58] W. Nick Street, W. H. Wolberg, and O. L. Mangasarian. 1993. Nuclear feature extraction for breast tumor diagnosis. In *Proc.SPIE* (1993/7/29), Vol. 1905. SPIE, Bellingham, Washington USA, 10. https://doi.org/10.1117/12.148698

[59] Yingcheng Sun and Kenneth Loparo. 2019. Opinion spam detection based on heterogeneous information network. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 1156–1163.

[60] Ananda Theertha Suresh. 2019. Differentially Private Anonymized Histograms. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc., 7971–7981.

[61] Yuye Wang, Jing Yang, and Jianpei Zhang. 2020. Differential Privacy for Weighted Network Based on Probability Model. *IEEE Access* 8 (2020), 80792–80800.

[62] Bin Yang, Issei Sato, and Hiroshi Nakagawa. 2015. Bayesian Differential Privacy on Correlated Data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (Melbourne, Victoria, Australia) *(SIGMOD '15)*. Association for Computing Machinery, New York, NY, USA, 747–762. https://doi.org/10.1145/2723372.2747643

[63] Jaewon Yang and Jure Leskovec. 2013. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 587–596.

[64] Jaewon Yang, Julian McAuley, and Jure Leskovec. 2013. Community detection in networks with node attributes. In *2013 IEEE 13th International Conference on Data Mining*. IEEE, 1151–1156.

[65] Zhong-Xuan Yuan, Bo-Ling Xu, and Chong-Zhi Yu. 1999. Binary quantization of feature vectors for robust text-independent speaker identification. *IEEE Transactions on Speech and Audio Processing* 7, 1 (1999), 70–78.

[66] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)* 42, 4 (2017), 25.

[67] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, et al. 2020. Quantum computational advantage using photons. *Science* 370, 6523 (2020), 1460–1463.

[68] Tianqing Zhu, Ping Xiong, Gang Li, and Wanlei Zhou. 2014. Correlated differential privacy: Hiding information in non-IID data set. *IEEE Transactions on Information Forensics and Security* 10, 2 (2014), 229–242.