Locally Private Set-Valued Data Analyses: Distribution and Heavy Hitters Estimation

Shaowei Wang, *Member, IEEE*, Yuntong Li, *Member, IEEE*, Yusen Zhong, Kongyang Chen, Xianmin Wang, Zhili Zhou, *Senior Member, IEEE*, Fei Peng, *Member, IEEE*, Yuqiu Qian, Jiachun Du, and Wei Yang, *Member, IEEE*

***Abstract*—In many mobile applications, user-generated data are presented as set-valued data. To tackle potential privacy threats in analyzing these valuable data, local differential privacy has been attracting substantial attention. However, existing approaches only provide sub-optimal utility and are expensive in computation and communication for set-valued data distribution estimation and heavy-hitter identification. In this paper, we propose a utility-optimal and efficient set-valued data publication method (i.e., *Wheel mechanism*). On the user side, the computational complexity is only $O(\min\{m\log m, me^{\epsilon}\})$ and communication costs are $O(\epsilon + \log m)$ bits, where m is the number of items, d is the domain size and ϵ is the privacy budget, while existing approaches usually depend on $O(d)$ or $O(\log d)$ ($d \gg m$). Our theoretical analyses reveal the estimation errors have been reduced from the previously known $O(\frac{m^2 d}{n\epsilon^2})$ to the optimal rate $O(\frac{md}{n\epsilon^2})$. Additionally, for heavy-hitter identification, we present a variant of the Wheel mechanism as an efficient frequency oracle, entailing only $O(\sqrt{n})$ computational complexity. This heavy-hitter protocol achieves an identification bar of $\tilde{O}(\frac{1}{\epsilon}\sqrt{\frac{m}{n}\log d})$, reducing by a factor of $\sqrt{m}$ relative to existing protocols. Extensive experiments demonstrate our methods are 3-100x faster than existing approaches and have optimized statistical efficiency.**

***Index Terms*—Distributed data aggregation, frequency estimation, heavy-hitter identification, local differential privacy.**

Manuscript received 18 December 2022; revised 17 November 2023; accepted 8 December 2023. Date of publication 12 December 2023; date of current version 2 July 2024. This work was supported in part by the National Natural Science Foundation of China under Grants 62372120, 62102108, 62072055, 62372128, 62072132, 62002074, 62072127, and 62002076, in part by the Natural Science Foundation of Guangdong Province of China under Grants 2022A1515010061, 2023A1515011774, 2023A1515011575, and 2019A1515110213, in part by CNKLSTISS under Project no. 6142111180404, and in part by Guangzhou Basic and Applied Basic Research Foundation under Grants 202201010194, 202002030131, and 202201020139. Recommended for acceptance by J. Ren. *(Corresponding authors: Kongyang Chen; Xianmin Wang.)*

Shaowei Wang, Yuntong Li, Yusen Zhong, Kongyang Chen, Xianmin Wang, Zhili Zhou, and Fei Peng are with the Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, Guangdong 511370, China (e-mail: wangsw@gzhu.edu.cn; 2112006152@e.gzhu.edu.cn; 2112106277@e.gzhu.edu.cn; kychen@gzhu.edu.cn; xianmin@gzhu.edu.cn; zhou_zhili@163.com; eepengf@gmail.com).

Yuqiu Qian and Jiachun Du are with the Interactive Entertainment Group, Tencent Games, Shenzhen, Guangdong 518054, China (e-mail: yuqiuqian@tencent.com; kevinjcdu@tencent.com).

Wei Yang is with the Institute of Advanced Research, University of Science and Technology of China, Hefei, Anhui 230052, China (e-mail: qubit@ustc.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TMC.2023.3342056, provided by the authors.

Digital Object Identifier 10.1109/TMC.2023.3342056

I. Introduction

USER data, such as service usage data, are pivotal to the success of numerous online services and mobile applications. When cast in bit-vector formats, these data can be characterized as high-dimensional set-valued data (that is, set-valued data with an expansive data domain). Examples include visited website URLs, shopping cart or recently used Apps on mobile phones [1], [2], [3], as well as activity data on intelligent wearable devices or other loT devices [4], [5].

Analyzing user data from mobile applications is beneficial for improving the quality of services from the perspective of service providers. However, it raises privacy concerns for users since service providers can trace user interests or mine social statuses [6] by harvesting sensitive data from user devices in mobile environment. Consequently, numerous regulations and laws, such as the GDPR regulation [7] in the Europe Union, and the CCPA act [8] in the state of California, have been enacted recently. To further ensure data privacy and adhere to these privacy regulations, delving into privacy-preserving data analysis techniques is essential for the welfare of both users and service providers.

Local differential privacy (LDP) [9] has became as the *de facto* standard for data privacy protection in computer networking systems. It sanitizes user data on their local devices, without trusting any third parties (e.g., service providers or edge servers). This concept stems from centralized differential privacy (DP) [10] for databases, which needs a trusted data curator to collect and manage raw data from users (e.g., in [11], [12], [13], [14]). In contrast to classic generalization-based or perturbation-based methods for privacy preservation (e.g., k-anonymity [15] and ℓ-diversity [16] in [17], [18], [19]), LDP approaches are more robust to adversaries with background knowledge. Furthermore, compared to privacy-preserving methods founded on secure multi-party computation (e.g., in [20], [21], [22]), the computation and communication costs of LDP on the user side are often lower [23]. Hence, LDP-based data analysis has been extensively explored in mobile or distributed environments, spanning topics like mean estimation for numerical data [9], distribution estimation on categorical data [9], [24], location data [25] and set-valued data [26]. It is also been integrated into mainstream platforms, including Google Chrome browser [27], [28] and Apple iOS/Mac OS systems [29], [30]).

TABLE I
COMPARISON OF ϵ-LDP APPROACHES FOR SET-VALUED DATA DISTRIBUTION ESTIMATION WHEN $\epsilon = O(1)$

approaches	user	user-server	server	MSE
d-RAPPOR [27]	$O(d)$	$O(d)$	$O(nd)$	$O(\frac{m^2 d}{n\epsilon^2})$
sampling RR [26]	$O(m)$	$O(\log d)$	$O(n+d)$	$O(\frac{m^2 d}{n\epsilon^2})$
sampling 1-bit [26]	$O(m)$	$O(d)+O(1)$	$O(d)$	$O(\frac{m^2 d}{n\epsilon^2})$
PA-GRR [31]	$O(m)$	$O(\log(d))$	$O(n+d)$	$O(\frac{m^2 d}{n\epsilon^2})$
PrivSet [32]	$O(d)$	$O(d)$	$O(nd)$	-
this work	$O(m)$	$O(\log m)$	$O(nd)$	$O(\frac{md}{n\epsilon^2})$

A. Item Distribution Estimation

Regarding high dimensional set-valued data, existing ϵ-LDP approaches suffer from high computational or communication overheads, which is intolerable for mobile devices. We summarize the complexities of representative ϵ-LDP distribution estimation approaches for set-valued data in Table I. The parameter ϵ is termed the privacy budget or privacy level, typically set within the range $(0.01, 3.0]$. The *user/server* column shows the computational complexity on the user/server side correspondingly, the *user-server* column shows the communication complexity between a user and the server, and the *MSE* column shows the mean squared error bounds for distribution estimation (i.e. $\sup_\theta \mathbb{E}[\|\hat{\theta} - \theta\|_2^2]$, where θ and $\hat{\theta}$ represent the true and estimated item distribution respectively).

For distribution estimation of *set-valued data*, the RAPPOR mechanism [27] exhibit computational and communication costs of $O(d)$ on the user side are both, with an error bound of $O(\frac{m^2 \cdot d}{n\epsilon^2})$. Following this, [26] decomposes set-valued data into categorical data by random item sampling, then employing standard categorical ϵ-LDP methods (e.g., the randomized response [34] or the 1-bit mechanism [24]). Though the sampling-based approaches are almost efficient as ones for categorical data, their estimation error bounds depend on a factor of m^2 due to the sampling. The PrivSet mechanism [32] randomly responses with a subset for set-valued data in an end-to-end manner, which however needs $O(d)$ computation and $O(d)$ communication costs on the user side. It shows remarkable empirical results but lacks theoretical error guarantees. Indeed, a pivotal question in set-valued data distribution estimation under ϵ-LDP – *what is the minimax lower error bound?* – remains unaddressed in existing literature.

B. Heavy-Hitter Identification

When dealing with set-valued data encompassing a vast domain (e.g., website URLs, words or phrases, and online commercial products), the server is often more interested in items with high occurrences (i.e., heavy hitters). One naive approach to heavy-hitter identification is first calculating an item distribution estimator and then enumerating the data domain to find heavy items, as seen in [26]. However, it incurs unacceptable $O(d)$ computational costs, where d can range in the millions or even billions. Inspired by non-private heavy-hitter identification protocols that have sub-linear complexities, researchers have proposed several locally private approaches to categorical heavy-hitter identification (e.g., in [33], [35], [36], [37], [38]). They only incur $O(n) \ll d$ computational costs, and some of them reach error bounds close to frequency estimation (e.g., the $O(\frac{1}{\epsilon}\sqrt{n \log \frac{d}{\beta} \log \frac{1}{\beta}})$ in [35] and the $O(\frac{1}{\epsilon}\sqrt{n \log \frac{d}{\beta}})$ in [36]).

In real-world applications, users usually possess a set of items (e.g., a list of URLs, a basket of products). As for set-valued heavy-hitter identification with local differential privacy (see Table II), the current literatures either naively enumerate over the item domain (e.g., in [26]), or sample one item then utilize categorical protocols (e.g., in [33]). Specifically, though the approach in [26] is efficient on the user side, it suffers from high computational costs of $O(d)$ on the server side for enumerating over the item domain and filtering out heavy hitters. The approach in [33] proposes to sample only one sequence from each user and then iteratively construct heavy alphabets in a prefix tree. It incurs error rate of $\tilde{O}(\frac{1}{\epsilon}\sqrt{nm^2})$ due to sequence sampling. Heavy-hitter identification protocols have pervasive applications in decentralized network systems, such as collecting browsing histories (e.g., in Google Chrome browser [27]) and novel words discovery (e.g., in Apple iOS system [39]). Efficient and accurate heavy-hitter identification protocols for locally private set-valued data collection are still lacking in the community.

C. Our Contributions

In summary, existing ϵ-LDP approaches for set-valued data distribution estimation and heavy-hitter identification are suffering from the following two drawbacks:

I. *Domain dependence:* the computational/communication overheads on the user side and the storage costs on the server side depend on the item domain size d, which is inefficient for high-dimensional set-valued data;
II. *Statistical inefficiency:* the distribution estimator and heavy-hitter identification bar suffer from sub-optimal rates, and thus more samples from the user population (i.e. a larger n) are needed to achieve desirable estimation accuracy.

In this paper, we propose an efficient and optimal ϵ-LDP mechanism (i.e., *Wheel mechanism*) for set-valued distribution estimation and heavy-hitter identification. It is domain-agnostic (i.e., independent of dimension size d) and utility-optimal, and is thus more adaptive to resource constrained features in mobile environment. Specifically, the Wheel mechanism maps the set-valued data to multiple points in a circled wheel, and then designs a calibrated probability distribution that satisfies ϵ-LDP based on these points. After that, it samples one numerical value as the output from the wheel according to the designed probability distribution. The mechanism needs only $O(\epsilon + \log m)$-bits communication bits between a user and the server. Since the mapping procedure is done by a user-specific hash function, its computation costs are $O(m)$. By carefully designing the randomization distribution, the server could derive an unbiased item distribution estimator over the data domain with the optimal error bound. The Wheel mechanism can also be adapted as an efficient frequency oracle, which is a core building block for heavy-hitter identification.

TABLE II
COMPARISON OF ϵ-LDP APPROACHES FOR SET-VALUED DATA HEAVY-HITTER IDENTIFICATION WHEN $\epsilon = O(1)$

approaches	user	user-server	server	MAE
sampling RR [26]	$O(m)$	$O(\log d)$	$O(n+d)$	$O(\frac{1}{\epsilon}\sqrt{\frac{m^2}{n}\log\frac{d}{\beta}})$
sampling 1-bit [26]	$O(m)$	$O(d)+O(1)$	$O(d)$	$O(\frac{1}{\epsilon}\sqrt{\frac{m^2}{n}\log\frac{d}{\beta}})$
TriHH [33]	$O(m)$	$O(1)$	$O(n)+\tilde{O}(\sqrt{n})$	$O(\frac{1}{\epsilon}\sqrt{\frac{m^2}{n}\log\frac{d}{\beta}\log\frac{1}{\beta}})$
this work	$O(m)$	$O(\log m)$	$O(nm)+\tilde{O}(\sqrt{n})$	$O(\frac{1}{\epsilon}\sqrt{\frac{m}{n}\log\frac{d}{\beta}\log\frac{1}{\beta}})$

The MAE column presents the identification bar of retrieving all heavy-hitters with probability 1–β.

Our contributions can be summarized as follows:

- We give tight minimax lower bounds for the ϵ-LDP set-valued distribution estimation problem, and show that the mean squared error $O(\frac{md}{n\epsilon^2})$ is optimal.
- For set-valued data distribution estimation under ϵ-LDP in mobile network environment, the proposed Wheel mechanism costs $\min\{O(me^{\epsilon}), O(m\log m)\}$ computation and $O(\epsilon+\log m)$ communication resources. The theoretical error bound for item distribution estimation achieves the optimal rate $O(\frac{md}{n\epsilon^2})$.
- We develop a discrete version of the Wheel mechanism for efficient oracle access (e.g., with large domain), which has only $O(\sqrt{n})$ computation overhead for each item frequency query.
- Building upon the Wheel frequency oracle, we design an optimized protocol for set-valued heavy-hitter identification and improve upon existing approaches on identification bar by a factor of $\sqrt{m}$.
- Through extensive experiments on distribution estimation and heavy-hitter identification, we validate the design/analyses of the Wheel mechanism, demonstrate its efficiency and effectiveness, and show about 3-100x running speed boost on the user side.

The remainder of the paper is organized as follows. Section II formally introduces set-valued data, local differential privacy and the distribution estimation framework. Section III gives tight lower bounds for set-valued data under the minimax risk framework. The Wheel mechanism for set-valued data along with its theoretical analysis shown in Section IV. Section V presents the efficient frequency oracle and the heavy-hitter identification protocol. Section VI gives utility guarantees of the proposed identification protocol. Section VII evaluates the Wheel mechanism for item distribution estimation. Section VIII tests the proposed protocol for heavy-hitter identification. Finally, Section IX concludes the paper.

II. PRELIMINARIES

We here briefly introduce set-valued data, local differential privacy (ϵ-LDP) and the framework of ϵ-LDP distribution estimation. Notations across the paper are summarized in Table III.

A. Set-Valued Data

Let $\mathcal{X} = \{X_1, X_2, \ldots, X_d\}$ denote the item domain, the set-valued data $\mathbf{x} \subseteq \mathcal{X}$ is a subset of the domain. In many practical applications, the size of the item domain $d = |\mathcal{X}|$ is relatively large (e.g., $d > 100$), such as website URLs, English words, and all possible Apps.

TABLE III
LIST OF NOTATIONS

Notation	Description
$\mathcal{X}$	The item domain of user data
d	The size of the item domain
i	The index of items
m	The cardinality of set-valued data
$\mathcal{X}^m$	The set of subsets that $\mathbf{x} \subseteq \mathcal{X}$ with size m
n	The number of users or participants
j	The index of users
$\mathbf{x}^j$	The set-valued data of user j
$\mathbf{z}^j$	The private view from user j
θ	User data's distribution over item domain
$\hat{\theta}$	The estimation of θ
ϵ	The privacy budget (privacy level)
v_i	The hashed value of item X_i in $[0.0, 1.0)$
C_{v_i}	The coverage area of item X_i
p	The length parameter of a coverage area
P_t	True coverage probability of each item
P_f	False coverage probability of each item

Let m represent the number of items present in the set-valued data $\mathbf{x}$. For ease of analysis, we operate under the assumption that the size of each set-valued data (specifically, m) remains consistent across all users. It is worth noting that categorical data serves as a distinct subset of set-valued data, where $m = 1$. For users possessing fewer or greater than m items, adjustments can be made by either introducing stub items or selecting a sample of m items, as seen in [26], [32]. The specific value of m can be determined based on pre-existing knowledge of the cardinalities of user data or through a noisy cardinality histogram estimation procured from locally private cardinality aggregation. We note that adding stub items is optional. Our protocols and their theoretical guarantees are applicable even when the input set-valued data contains fewer than m items; we fix the cardinality primarily for simplicity in notation and analyses.

B. Local Differential Privacy (ϵ-LDP)

The ϵ-LDP guarantees that privacy attackers can only gain a limited multiplicative factor over their prior knowledge after observing a privatized output z. Formally, let K denote a randomization mechanism that intends to provide ϵ-LDP. Assuming that each user's true value $\mathbf{x}$ belongs to the input

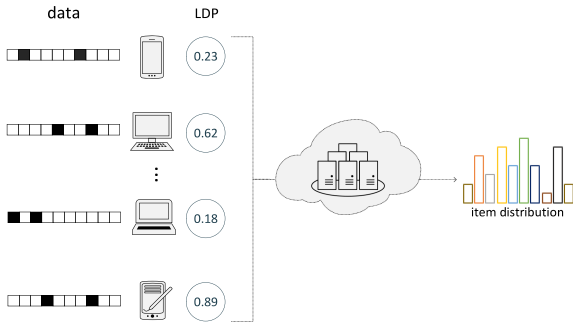


Fig. 1. Model of ϵ-LDP set-valued data publication and distribution estimation with the wheel mechanism.

data domain $\mathcal{X}^m = \{\mathbf{x} \mid \mathbf{x} \subseteq \mathcal{X}\ and\ |\mathbf{x}| = m\}$, the definition of (non-interactive) ϵ-LDP is given in the Definition 1.

Definition 1 ((non-interactive) ϵ-LDP [9]): Let $\mathcal{D}_K$ denote the output domain, a randomized mechanism K satisfies local ϵ-differential privacy iff for any data pair $s, s' \in \mathcal{X}^m$, and any output $t \in \mathcal{D}_K$, the following inequality

$$\mathbb{P}[K(s) = t] \leq \exp(\epsilon) \cdot \mathbb{P}[K(s') = t]$$

holds, where ϵ is called the privacy level or privacy budget.

Practical values for ϵ fall in (0.01,3.0] so that the privacy attacker's knowledge cannot grow too much.

If user j adopts a randomization mechanism K^j that depends on former outputs $\{\mathbf{z}^1, \ldots, \mathbf{z}^{j-1}\}$, we call such randomization mechanism provides interactive ϵ-LDP (in Definition 2).

Definition 2 (Interactive ϵ-LDP [9]): Let $\mathcal{D}_{K^j}$ denote the output domain of randomized mechanism K^j ($j \in [1, n]$). The random variable $\mathbf{z}^j$ is an ϵ-LDP view of $\mathbf{x}^j$, if

$$\sup_{t \in \mathcal{D}_{K^j}} \frac{\mathbb{P}[K^j(s) = t \mid \mathbf{x}^j = s, \mathbf{z}^1 = z^1, \ldots, \mathbf{z}^{j-1} = z^{j-1}]}{\mathbb{P}[K^j(s') = t \mid \mathbf{x}^j = s', \mathbf{z}^1 = z^1, \ldots, \mathbf{z}^{j-1} = z^{j-1}]} \leq e^\epsilon$$

holds for all $z^1 \in \mathcal{D}_{K^1}, \ldots, z^{j-1} \in \mathcal{D}_{K^{j-1}}$, any data pair $s, s' \in \mathcal{X}^m$, and any output $t \in \mathcal{D}_{K^j}$. We call such mechanism $\mathbf{K} = \{K^1, \ldots, K^n\}$ satisfies ϵ-LDP in an interactive setting.

Note that the non-interactive version of ϵ-LDP (in Definition 1) is a special case when $K^j \equiv K$.

C. Distribution Estimation

Distribution estimation is a fundamental building block for many statistical analyses (e.g., histogram estimation, hypothesis testing, causal inference, and machine learning). In the data distribution estimation framework (shown in Fig. 1), every user u^j independently randomizes their true value $\mathbf{x}^j$ to a publishable view $\mathbf{z}^j$ through a ϵ-LDP mechanism and sends it to the server. The server side then estimates the item distribution over $\mathcal{X}$ of all users based on all collected $\mathbf{z}^j$. Specifically, we denote the true item distribution as $\theta = \{\theta_1, \theta_2, \ldots, \theta_d\}$, where

$$\theta_i = \frac{1}{n} \cdot \#\{\mathbf{x}^j \mid j \in [1, n] \text{ and } X_i \in \mathbf{x}^j\}, \tag{1}$$

and $\hat{\theta}$ as the estimated item distribution.

A good ϵ-LDP mechanism should have low computation and communication burden on the user side, and minimum estimation error that matches minimax lower bounds for the service provider.

III. Tight Minimax Lower Bounds

As an extension to lower bounds on ϵ-LDP distribution estimation problem on categorical data [9], [40], we here present lower bounds for ϵ-LDP set-valued data distribution estimation. By decomposing the set-valued data into multiple categorical data, we can follow a similar procedure for categorical data in [9], [40].

A. Classic Minimax Risks

Let $\mathcal{P}$ denote all possible probability distributions over the data universe $\mathcal{X}^m$. For each distribution $P \in \mathcal{P}$, denote $\theta(P)$ as the true parameter of interests. Suppose $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n\}$ are n i.i.d. observations that are drawn according to some distribution $P \in \mathcal{P}$, and $\hat{\theta} : \mathcal{X}^m \mapsto \Psi$ is the estimation of $\theta(P)$. The minimax risk $\mathcal{M}_n$ under metric $\Phi \circ \rho$ can be defined as the following saddle point problem:

$$\mathcal{M}_n(\theta(\mathcal{P}), \Phi \circ \rho) := \inf_{\hat{\theta}} \sup_{P \in \mathcal{P}} \mathbb{E}_P[\Phi(\rho(\hat{\theta}(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n), \theta(P)))]$$

where $\rho : \Psi \times \Psi \mapsto \mathbb{R}^+$ is a semi-metric function, and $\Phi : \mathbb{R}_+ \mapsto \mathbb{R}_+$ is a non-decreasing function with $\Phi(0) = 0$.

B. Local Private Minimax Risks

We now proceed to the definition of local private minimax risk, which measures the fundamental difficulty of an estimation problem under ϵ-LDP constraints.

Given the privacy budget ϵ, we denote $\mathcal{K}_\epsilon$ as the set of all possible mechanisms $\{K^1, \ldots, K^n\}$ that satisfy interactive ϵ-LDP. Taking as input samples $\{\mathbf{x}^j\}_{j=1}^n$, some mechanism $\mathbf{K} \in \mathcal{K}_\epsilon$ can produce a sequence of private observations $\{\mathbf{z}^j\}_{j=1}^n$. If the estimator $\hat{\theta} = \hat{\theta}(\mathbf{z}^1, \ldots, \mathbf{z}^n)$ only depends on these private views $\{\mathbf{z}^j\}_{j=1}^n$ while cannot access to input samples $\{\mathbf{x}^j\}_{j=1}^n$, the minimax risk as a function of privacy budget ϵ can be defined as:

$$\mathcal{M}_n(\theta(\mathcal{P}), \Phi \circ \rho, \epsilon) := \inf_{\mathbf{K} \in \mathcal{K}_\epsilon} \inf_{\hat{\theta}} \sup_{P \in \mathcal{P}} \mathbb{E}_{p,\mathbf{K}}[\Phi(\rho(\hat{\theta}(\mathbf{z}^1, \ldots, \mathbf{z}^n), \theta(P)))].$$

In the following sections, we first theoretically give minimax lower bounds for ϵ-LDP distribution estimation on set-valued data (in Section III-C), and then design attainable mechanisms taking into consideration efficiency (i.e., the Wheel mechanism, shown in Section IV).

C. Lower Bounds for ϵ-LDP Set-Valued Data

Motivated by the lower bounding procedure for ϵ-LDP categorical distribution estimation in [9], we utilize Assouad's method [41] and its local private form (Lemma 1, [42]) to

derive sharp lower bound of minimax risks for set-valued data distribution estimation under interactive ϵ-LDP.

Assouad's method [41] transforms an estimation problem into multiple binary hypothesis testing problems. For some $d \in \mathbb{N}$, it defines a hypercube $\mathcal{V} = \{-1, 1\}^d$ and a family of distributions $\{P_\nu\}_{\nu \in \mathcal{V}}$ indexed by the hypercube. Assouard's method says that the distribution family induces a 2δ-Hamming separation for the loss $\Phi \circ \rho$ if there exists a vertex mapping (a function $\kappa : \theta(\mathcal{P}) \mapsto \{-1, 1\}^d$) satisfying:

$$\Phi(\rho(\theta, \theta(P_\nu))) \geq 2\delta \sum_{j=1}^{d} \mathbf{1}\{[\kappa(\theta)]_j \neq \nu_j\}.$$

The work [42] gives the following private version of Assouad's method for lower bounding ϵ-LDP estimation problems.

Lemma 1 (Private Assouad's bound [42]): Let P_{+j} denote $\frac{1}{2^{d-1}} \sum_{\nu:\nu_j=1} P_\nu$ and P_{-j} denote $\frac{1}{2^{d-1}} \sum_{\nu:\nu_j=-1} P_\nu$, we have

$$\mathcal{M}_n(\theta(\mathcal{P}), \Phi \circ \rho, \epsilon) \geq d \cdot \delta \cdot \left(1 - \left(\frac{n(e^\epsilon - 1)^2}{2d} F_{\mathbb{B}_\infty(\mathcal{X}^m)}\right)^{\frac{1}{2}}\right),$$

where $\mathbb{B}_\infty(\mathcal{X}^m)$ denote the collection of function γ with supremum norm bounded by 1 as:

$$\mathbb{B}_\infty(\mathcal{X}^m) := \{\gamma : \mathcal{X}^m \mapsto \mathbb{R} \mid \|\gamma\|_\infty \leq 1\},$$

and maximum possible discrepancy $F_{\mathbb{B}_\infty(\mathcal{X}^m),\mathcal{P}}$ is defined as:

$$\sup_{\gamma \in \mathbb{B}_\infty(\mathcal{X}^m)} \sum_{i=1}^{d} \left(\int_{\mathcal{X}^m} \gamma(x)(dP_{+j}(x) - dP_{-j}(x))\right)^2.$$

Relying on Lemma 1, we can construct a class of $\frac{2\delta^2 m^2}{d^2}$-hamming separated distributions and bound the maximum possible marginal discrepancy $F_{\mathbb{B}_\infty(\mathcal{X}^m)}$ under $\frac{8\delta^2 m}{d}$. Hence we get lower bounds (in Theorem 1, proved in Appendix A, available online) for the problem of locally private set-valued data distribution estimation.

Theorem 1: For the set-valued data distribution estimation problem where $m \leq \frac{d}{2}$, and for any non-interactive or interactive ϵ-LDP mechanism, there exists a universal constant $c > 0$ such that for all $\epsilon \in (0, 1]$,

$$\mathcal{M}_n(\theta(\mathcal{P}), \|\cdot\|_2^2, \epsilon) \geq c \cdot \min\left\{\frac{m^2}{d}, \frac{dm}{n\epsilon^2}\right\}.$$

For better understanding the minimax rate, let us consider the non-private error rate of set-valued data distribution estimation, where we have:

$$\mathbb{E}[\|\hat{\theta} - \theta\|_2^2] \leq \sum_{i=1}^{d} \mathbb{E}[\|\hat{\theta}_i - \theta_i\|_2^2] \leq \frac{m}{n}. \tag{2}$$

Hence for estimation with ℓ_2-norm error, the enforcement of ϵ-LDP causes that the effective sample size decreases from n to $n\epsilon^2/d$. Comparing with the ϵ-LDP lower bound $O(\frac{d}{n\epsilon^2})$ for categorical data with domain size d, the mean squared error lower bound on set-valued data is scaled by a factor of m. This is hinted by the fact that under the same domain size the squared ℓ_2-norm of m-sized set-valued data is m times of categorical data's.

Theorem 2: For the set-valued data distribution estimation problem where $m \leq \frac{d}{2}$, for any non-interactive or interactive ϵ-LDP mechanism, there exists a universal constant $c > 0$ such that for all $\epsilon \in (0, 1]$,

$$\mathcal{M}_n(\theta(\mathcal{P}), \|\cdot\|_1, \epsilon) \geq c \cdot \min\left\{\frac{m}{2}, \frac{d\sqrt{m}}{\sqrt{n\epsilon^2}}\right\}.$$

Similarly, under the ℓ_1-norm metric, the estimation error lower bounds of set-valued data can be derived as $O(\frac{d\sqrt{m}}{\sqrt{n\epsilon^2}})$ (see Theorem 2, proved in in Appendix A, available online). Compared to the non-private error rate on (decomposable) set-valued data:

$$\mathbb{E}[\|\hat{\theta} - \theta\|_1] \leq \sum_{a=1}^{m} \sum_{j=1}^{d/m} \mathbb{E}\left[\|\hat{\theta}_{a,j} - \theta_{a,j}\|_2\right] < m\sqrt{\frac{d/m}{n}},$$

it also indicates that the constraint of ϵ-LDP causes that the effective sample size decreases from n to $n\epsilon^2/d$.

IV. Set-Valued Data Distribution Estimation

We present an attainable mechanism (i.e., the Wheel mechanism) with optimal error bounds. It is highly efficient in terms of computation/communication costs on the user side.

A. User-Side Randomization

From a Shannon entropy perspective, existing ϵ-LDP mechanisms, which either output a single item [34] or a set of items [32], demand at least $O(\log(Poly(d)))$ bits, particularly when the privacy budget is minimal (meaning when ϵ nears 0). This is because their outputs closely mirror a uniformly random generation. Intuitively, as the privacy budget shrinks, the output is permitted to retain less information about the input. Consequently, it should be feasible to represent the input's information with fewer bits. With this understanding, we introduce the Wheel mechanism that curtails the inherent randomness in the output and substitutes it with pseudo-randomness generated by a user-specific hash function, effectively reducing communication overheads.

Supposing that the j-th user u^j holds a set-valued data $\mathbf{x}^j = \{X_1, \ldots, X_m\}$, the wheel mechanism (in Algorithm 1 and Algorithm 2) follows three steps, where the coverage parameter p is the length of coverage area to control true/false coverage probabilities.

1) Using the user id or a random-generated number as the seed, the user-specific hash function $h : \mathcal{X} \to [0.0, 1.0)$ maps every item $X_i \in \mathbf{x}^j$ to v_i.
2) To satisfy ϵ-LDP constraint, the numerical values $\mathbf{v} = \{v_1, .., v_m\}$ are then randomized with a calibrated probability distribution $Q[y \mid \mathbf{v}]$ over $[0.0, 1.0)$. The definition of Q with coverage parameter p is given as follows:

$$Q[y \mid \mathbf{v}] = \begin{cases} \frac{e^\epsilon}{\Omega}, & \text{if } y \in [v_i, v_i + p) \text{ or } [0, v_i + p - 1) \\ & \text{for any } i \in [1, m]; \\ \frac{\Omega - l \cdot e^\epsilon}{(1-l)\Omega}, & \text{otherwise.} \end{cases} \tag{3}$$

Algorithm 1: Set-Valued Data Randomization.

Input: Set-valued data $\mathbf{x} \in \mathcal{X}^m = \{c \mid c \subseteq \mathcal{X} \text{ and } |c| = m\}$, hash function $h : \mathcal{X} \to [0.0, 1.0)$ with random seed s, privacy level ϵ, the coverage parameter $p = \frac{1}{2m-1+me^{\epsilon}}$.

Output: A private view $z \in [0.0, 1.0)$ that satisfies ϵ-LDP.

// User-specific hash mapping

$\mathbf{v} = \{v_1, .., v_m\} = h_s(\mathbf{x})$

// Initialize ranges

$\text{left} = \{\min\{b \cdot p, 1.0\} \mid b \in [1, \lceil \frac{1}{p} \rceil]\}$

$\text{right} = \{(b-1) \cdot p \mid b \in [1, \lceil \frac{1}{p} \rceil]\}$

// Merge coverage ranges

for $v_i \in \mathbf{v}$ **do**

  $b = \lceil \frac{v_i}{p} \rceil$

  $\text{left}_b = \min\{v_i, \text{left}_b\}$

  if $b < \lceil \frac{1}{p} \rceil$ **then**

    $\text{right}_{b+1} = \max\{v_i + p, \text{right}_{b+1}\}$

  else

    $\text{right}_1 = \max\{v_i + p - 1, \text{right}_1\}$

  end

end

for $b \in [1, \lceil \frac{1}{p} \rceil - 1]$ **do**

  $\text{left}_b = \max\{\text{left}_b, \text{right}_b\}$

  $\text{right}_b = \text{right}_{b+1}$

end

$b = \lceil \frac{1}{p} \rceil$

$\text{left}_b = \max\{\text{left}_b, \text{right}_b\}$

$\text{right}_b = \text{right}_1 + 1$

// Randomization based to coverage ranges

$l = \sum_{b \in [1, \lceil \frac{1}{p} \rceil]} (\text{right}_b - \text{left}_b)$

r =UniformRandom(0.0, 1.0)

$a = 0.0$

for $b \in [1, \lceil \frac{1}{p} \rceil]$ **do**

  $a = a + \frac{e^{\epsilon}(\text{right}_b - \text{left}_b)}{\Omega}$

  if $a > r$ **then**

    $z = \text{right}_b - \frac{(a-r)\Omega}{e^{\epsilon}}$

    break

  end

  $a = a + \frac{(\Omega - l \cdot e^{\epsilon})(\text{left}_{b\%\lceil \frac{1}{p} \rceil + 1} + \lfloor b \cdot p \rfloor - \text{right}_b)}{(1-l)\Omega}$

  if $a > r$ **then**

    $z = \text{left}_{b\%\lceil \frac{1}{p} \rceil + 1} - \frac{(a-r)(1-l)\Omega}{\Omega - l \cdot e^{\epsilon}}$

    break

  end

end

$z = z \bmod 1.0$

return (s, z)

where $\Omega := m \cdot p \cdot e^{\epsilon} + (1 - m \cdot p)$ is normalization factor, and l is the length of the union coverage area $C_{\mathbf{v}}$:

$$C_{\mathbf{v}} = \{y \mid \text{if } y \in [v_i, v_i + p) \text{ or } [0, v_i + p - 1) \text{ for any } i \in [1, m]\}.$$

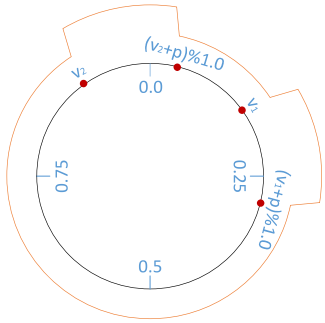


Fig. 2. Probability distribution of the private view y demonstrated over a circled wheel when coverage areas are disjoint. The probability density in the coverage area is $\frac{e^{\epsilon}}{m \cdot p \cdot e^{\epsilon} + (1 - m \cdot p)}$, while the probability density of the rest area is $\frac{1}{m \cdot p \cdot e^{\epsilon} + (1 - m \cdot p)}$.

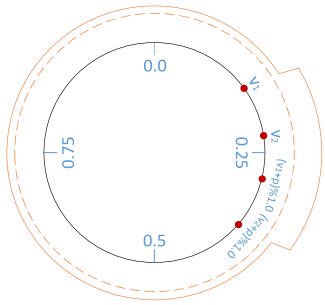


Fig. 3. Probability distribution of the private view y demonstrated over a circled wheel when coverage areas are overlapping. The probability density in the coverage area is $\frac{e^{\epsilon}}{m \cdot p \cdot e^{\epsilon} + (1 - m \cdot p)}$, while the density of the rest area is $\frac{\Omega - l \cdot e^{\epsilon}}{(1-l)\Omega}$. The dashed circle represents the density of $\frac{1}{m \cdot p \cdot e^{\epsilon} + (1 - m \cdot p)}$, which is lower than $\frac{\Omega - l \cdot e^{\epsilon}}{(1-l)\Omega}$.

Figs. 2 and 3 show examples that the probability distribution on the wheel with disjoint/overlapping coverage areas respectively.

3) Sample a value $z \in [0.0, 1.0)$ from distribution $Q[y \mid \mathbf{v}]$, then send it to the server (along with the seed if necessary).

Practically, by sorting coverage areas [43], the union of coverage areas $C_{\mathbf{v}}$ can be computed in $\Theta(m \log m)$ time. However, considering that the coverage range of each C_{v_i} is of the same length p on the wheel, we can reduce the complexity to $\Theta(\lceil \frac{1}{p} \rceil) = \Theta(me^{\epsilon})$ by dividing [0.0,1.0) into $\lceil \frac{1}{p} \rceil$ buckets (as shown in Algorithm 1). For the b-th bucket, we record the current start/end point of coverage areas in the bucket as $left_b/right_b$ respectively. At first, the start point is the maximum value in the bucket and the end point is the minimum value in the bucket, which implies the bucket is empty (with no coverage area). When a new coverage area C_{v_i} comes, the start/end point of two corresponding buckets will be updated. Finally we merge continuous areas based on start/end points of neighboring buckets.

The overall algorithm follows five steps:

1) Use user-specific hash function to map set-valued data x^j to numerical values $\mathbf{v}$ and get m coverage areas C_{v_i}.
2) Divide [0.0,1.0) to $\lceil \frac{1}{p} \rceil$ buckets, and use $left_b/right_b$ to denote the start/end point of the b-th bucket respectively.

3) For each p-length coverage range C_{v_i}, record the start/end points of neighboring buckets (see lines 4 to 12).
4) For each bucket, merge ranges when the end point is lower than the start point, and then split the coverage areas into disjoint ranges (see lines 13 to 19).
5) Given disjoint coverage ranges and their relative weights against non-covered ranges (computed using the union coverage length l), draw a random sample as the output.

The ϵ-LDP guarantee of the randomization process for set-valued data is given in Theorem 3.

Theorem 3: The wheel mechanism for set-valued data in Algorithm 1 satisfies ϵ-LDP.

Proof: The output of Algorithm 1 follows the probability distribution in Equation (3). Hence to prove its ϵ-LDP guarantee, it's enough to show that $\frac{1}{\Omega} \leq \frac{\Omega - l \cdot e^{\epsilon}}{(1-l)\Omega} \leq \frac{e^{\epsilon}}{\Omega}$. For any $\epsilon \geq 0.0$, we have $\Omega \leq e^{\epsilon}$ and then $\frac{\Omega - l \cdot e^{\epsilon}}{1-l} \leq e^{\epsilon}$, which ensures $\frac{\Omega - l \cdot e^{\epsilon}}{(1-l)\Omega} \leq \frac{e^{\epsilon}}{\Omega}$. Given that $l \leq m \cdot p$ and obviously $1 - l + l \cdot e^{\epsilon} \leq 1 - m \cdot p + m \cdot p \cdot e^{\epsilon}$, it further implies $\frac{1}{\Omega} \leq \frac{\Omega - l \cdot e^{\epsilon}}{(1-l)\Omega}$.

B. Server-Side Distribution Estimation

It can be observed from Figs. 2 and 3: when a category X_i is in the set-valued data $\mathbf{x}$, z appears in the coverage area C_{v_i} with a relatively high probability; when a category X_i is not in the set-valued data $\mathbf{x}$, the probability that z appears in the coverage area C_{v_i} is relatively low. This implies the server side could estimate whether each item is included in set-valued data $\mathbf{x}$ according to the value of z.

Formally, we can compute the true/false coverage probability in Lemma 2 and 3 respectively. The true coverage probability P_t is always greater than the false coverage probability P_f, which further confirms previous observations.

Lemma 2 (True Coverage Probability): When a category X_i is in the set-valued data $\mathbf{x}$, the probability that the private view $z \in [0.0, 1.0)$ of $\mathbf{x}$ appears in the coverage area C_{v_i} is:

$$\mathrm{P}_t = \mathbb{P}[z \in C_{v_i} \mid X_i \in \mathbf{x}] = \frac{p \cdot e^{\epsilon}}{m \cdot p \cdot e^{\epsilon} + (1 - m \cdot p)}.$$

Lemma 3 (False Coverage Probability): When a category X_i is not in the set-valued data $\mathbf{x}$, assuming that $h(X_i)$ is uniformly random and independent from $h(\mathbf{x})$ (the hash function is perfect), the expected probability that the private view $z \in [0.0, 1.0)$ of $\mathbf{x}$ shows in the coverage area C_{v_i} is:

$$\mathrm{P}_f = \mathbb{E}_{h_s}[\mathbb{P}[z \in C_{v_i} \mid X_i \notin \mathbf{x}]] = p.$$

Proof: Under the assumption that hash functions are perfect, the mapped value v of X_i distributes on $[0.0, 1.0)$ uniformly at random. Considering that C_{v_i} is a p-length circular arc on wheel, when v_i rotate around the wheel, the accumulative probability density of the private view z on any point in C_{v_i} is 1.0 and the traveled length is also 1.0. Hence P_f is equal to the expected probability density 1.0 multiplied the interval length p, which equals p. □

Given the true/false coverage probabilities of items, we present the estimator of the item distribution for set-valued data in Algorithm 2. According to Lemma 4, this estimator is unbiased. Further, its computational complexity is $O(n \cdot d)$.

Algorithm 2: Set-Valued Data Distribution Estimation.

Input: Private views and random seeds $(S, Z) = [\ (s^1, z^1), (s^2, z^2), ..., (s^n, z^n)\ ]$ from n users, coverage parameter p.
Output: An unbiased estimator $\hat{\theta}$ of the true item distribution $\theta = \{\theta_1, \theta_2, ..., \theta_d\}$.

1 $F = \{0\}^d$
`// Record frequencies of items.`
2 **for** $(s^j, z^j) \in (S, Z)$ **do**
3 **for** $X_i \in \mathcal{X}$ **do**
4 $v_i^j = h_{s^j}(X_i)$
5 **if** $z^j - p < v_i^j \leq z^j$ *or* $z^j - p + 1 < v_i^j < 1$ **then**
6 $F_i = F_i + 1$
7 **end**
8 **end**
9 **end**
`// Estimate item distribution from observed frequencies.`
10 **for** $i = 1$ *to* d **do**
11 $\hat{\theta}_i = \frac{1}{n} \cdot \frac{F_i - n \cdot \mathrm{P}_f}{\mathrm{P}_t - \mathrm{P}_f}$
12 **end**
13 **return** $\hat{\theta}$

Specifically, based on the coverage probabilities, let $\mathbb{P}[z \in C_{v_i}]$ denote the probability that z shows in the coverage area C_{v_i} of item X_i, we have:

$$\mathbb{P}[z \in C_{v_i}] = \theta_i \cdot \mathrm{P}_t + (1.0 - \theta_i) \cdot \mathrm{P}_f. \tag{4}$$

Consequently, given that the frequency $\frac{F_i}{n} = \frac{\sum_{j=1}^{n} \{z^j \in C_{v_i}\}}{n}$ is an unbiased estimation of $\mathbb{P}[z \in C_{v_i}]$, a linear transformation of $\frac{F_i}{n}$ provides an estimator of item distribution (shown in Algorithm 2). In the algorithm, lines 1 to 10 record the number of z falling in the coverage area of item X_i for each item, while lines 12 to 14 establish unbiased estimation of item distribution θ_i from recorded numbers according to the formula (4).

We theoretically prove that the estimator (i.e., Algorithm 2) is unbiased according to Lemma 4.

Lemma 4: The estimator $\hat{\theta}$ in Algorithm 2 is an unbiased estimation of the true item distribution θ, where the expectation is taken over the (pseudo) randomness of perfect hash functions and the randomness of sampling in Algorithm 1.

Proof: Proving $\hat{\theta}$ is an unbiased estimation of θ is equivalent to prove that $\hat{\theta}_i$ is an unbiased estimation of θ_i. Obviously, the observed frequency F_i is the summation of $n \cdot \theta_i$ Bernoulli variables with success probability P_t and $n - n \cdot \theta_i$ Bernoulli variables that their expected success probability is P_f. Consequently, we have:

$$\begin{aligned}
&\mathbb{E}\left[\frac{F_i - n \cdot \mathrm{P}_f}{\mathrm{P}_t - \mathrm{P}_f}\right] \\
&= \frac{n \cdot \theta_i \cdot \mathrm{P}_t + (n - n \cdot \theta_i) \cdot \mathrm{P}_f - n \cdot \mathrm{P}_f}{\mathrm{P}_t - \mathrm{P}_f} \\
&= n \cdot \theta_i
\end{aligned}$$

which concludes that $\frac{1}{n} \cdot \frac{F_i - n \cdot \mathrm{P}_f}{\mathrm{P}_t - \mathrm{P}_f}$ is an unbiased estimation of the θ_i in item distribution. □

It should be noted that the randomization procedure of the Wheel mechanism requires no information about the item domain $\mathcal{X}$ or the domain size d, making it domain-independent. As a result, the server can derive the distribution of target items on demand without any knowledge of the entire domain. This approach remains effective even when the domain undergoes changes over time.

C. Optimal Utility Guarantees

We here give mean squared error bounds on distribution estimation of the wheel mechanism (shown in Theorem 4, proved in in Appendix B, available online). Note that the error bound proved here is $O(\frac{md}{n\epsilon^2})$, which significantly improves state-of-art results (i.e., $O(\frac{m^2 d}{n\epsilon^2})$ in [26]) by a factor of m.

Theorem 4 (Mean Squared Error Bound): For set-valued data domain $\mathcal{X}^m$ that $|\mathcal{X}| = d$ and privacy budget $\epsilon = O(1)$, the distribution estimation error of Algorithms 1 and 2 is:

$$\sum_{i \in [1,d]} \mathbb{E}\left[|\hat{\theta}_i - \theta_i|^2\right] = O\left(\frac{dm}{n\epsilon^2}\right),$$

where the expectation is taken over the randomness of user-specific hash functions and randomization in Algorithm 1.

The error bound mentioned above is data-independent. It holds for any possible set-valued samples with any true item distribution θ. Given the claim in [9] that the population minimax rate (shown in Theorem 1) lower bounds conditional sample risk (shown in Theorem 4) up to some multiplicative constant, we can conclude that the wheel mechanism is minimax optimal (when $\epsilon = O(1)$ and $n \ll 1$). Furthermore, combining Equation (2) and above bounds, we conclude that the minimax error bounds in Theorem 1 are tight.

The Wheel mechanism can be viewed as a continuous analogy of the PrivSet mechanism. Through its continuous design, it not only achieves the optimal utility with theoretical guarantees but also substantially reduces both computation and communication costs. While the PrivSet mechanism might also attain optimal utility, proving this is particularly challenging due to the discrete nature of its parameter. Hence, such a continuous analogy of the previous mechanisms with discrete parameters (e.g., in [27], [32], [34]) not only reduces computation/communication costs, but also facilitates theoretical analyses on error bounds.

V. Set-Valued Data Heavy-Hitter Identification

Instead of analyzing the entire item domain, which may have a vast or even infinite size (e.g., millions of online products, web urls), service providers are typically more interested in items with the relatively high occurrences, a.k.a. heavy-hitters. Given a set-valued frequency oracle (e.g., the distribution estimation mechanism in the previous section), one naïve approach to identifying heavy-hitters is to enumerate over the candidate domain $\mathcal{X}$ and find items with frequencies that exceed a threshold Δ (e.g., in [26], [31]). However, this approach suffers a computational complexity of $\Theta(|\mathcal{X}|)$, rendering it impractical.

Algorithm 3: An Efficient Frequency Oracle.

Input: Set-valued data $\mathbf{x}^j$ belonging to $\mathcal{X}^m = \{c \mid c \subseteq \mathcal{X} \text{ and } |c| = m\}$ from n users. Hash functions $\{h_s\}_{s \in S}$ that map $\mathcal{X}$ to $[0.0, 1.0]$, where S is the pool of seeds. Discretization parameter w.

Output: Item Frequency oracle access to locally private dataset $\{\mathbf{x}^j\}_{j \in [1:n]}$.

```
// Initialize bucketed counts for every seed
1  for s ∈ S do
2      for i ∈ [1 : w] do
3          F_i^s = 0
4      end
5  end
6  for j ∈ [1 : n] do
       // Sanitize set-valued data
7      (s, z) is the private view of x^j outputted by Algorithm 1
8      i = ⌈z · w⌉
9      F_i^s = F_i^s + 1
10 end
11 for an incoming query v ∈ X do
12     a(v) = (Σ_{s∈S} Σ_{h_s(v)·w ≤ i ≤ h_s(v)·w + p·w} F_i^s − n·P_f) / (P_t − P_f)
13     return (v, a(v))
14 end
```

Ideally, the goal is to identify heavy-hitters with computational costs of $\tilde{O}(\frac{n}{\Delta})$. Here the notion of $\tilde{O}$ means logarithmic factors about $|\mathcal{X}|$ are omitted.

A. An Efficient Frequency Oracle

To circumvent the exhaustive search across the entire domain $\mathcal{X}$, most heavy-hitter identification solutions adopt the strategy of recovering heavy alphabets sequentially. This process requires multiple frequency estimations over some bounded domains that are much smaller than the $\mathcal{X}$. Therefore, we start with constructing an efficient and accurate frequency oracle, which is a discrete variant of the Wheel mechanism.

Denoting all set-valued data $\{\mathbf{x}^1, \ldots, \mathbf{x}^n\}$ from n users as M, the Algorithm 3 provides a frequency oracle about $v \in \mathcal{X}$. The oracle employs the ϵ-LDP Wheel mechanism in Algorithm 1 for sanitizing $\mathbf{x}^j$, and turns the private view (s, z) into w-bucketed counts for discretizing z (at lines 8-9), and finally estimates querying frequency from these counts (at lines 11-13). The bucketization parameter w controls the granularity of discretization, and is chosen as $\max\{2, 2e^{\epsilon}\sqrt{\frac{n}{3\log(2d/\beta)}}\}$ for balancing utility and efficiency (see detail in Theorem 5).

Note that the estimator in Algorithm 2 itself provides a frequency oracle, but the computational cost of processing each query is $\Theta(n)$. By comparison, the cost of Algorithm 3 is $\Theta(w)$, which scales with $O(\sqrt{n})$.

We now shift our focus to ensuring the utility of the frequency oracle. Using the (α, β)-accurate metric as defined in Definition 3, we first demonstrate that the original Wheel mechanism is $(\frac{32}{\epsilon}\sqrt{\frac{m}{n}\log\frac{d}{\beta}}, \beta)$-accurate in Lemma 5. Moreover, with

proper discretization of z, the more efficient oracle presented in Algorithm 3 also achieves $(O(\frac{1}{\epsilon}\sqrt{\frac{m}{n}\log\frac{d}{\beta}}),\beta)$-accuracy (see Theorem 5).

Definition 3 ((α,β)-accurate): A frequency oracle $\hat{\mathbf{S}}$ is (α,β)-accurate if with probability of at least $1-\beta$, it answers every frequency query $i\in[1,d]$ with $\tilde{\theta}_i$ satisfying $|\tilde{\theta}_i-\theta_i|\leq\alpha$.

Lemma 5 (Accuracy of the Wheel Mechanism): Given $\epsilon\leq 1$ and n set-valued data over domain $\mathcal{X}^m$, the $(m,\epsilon,\frac{1}{2m-1+m\cdot e^\epsilon})$-Wheel mechanism in Algorithm 1 and 2 is $(\frac{32}{\epsilon}\sqrt{\frac{m}{n}\log\frac{2d}{\beta}},\beta)$-accurate.

Proof: When both the randomness of selecting a user-specific hash function and the randomness of outputting z^j are considered, the binary value $[z^j-p<h_{s^j}(X_i)\leq z^j$ or $z^j-p+1<h_{s^j}(X_i)<1]$ is a Bernoulli variable with success rate of either P_f or P_t. Since the frequency F_i is a summation of n independent Bernoulli variables, denoting u as the estimation bias of $\tilde{\theta}_i$, according to the multiplicative Chernoff bound and the fact that $e^\epsilon P_f\geq P_t\geq P_f$, we have $\mathbb{P}[n\cdot|u|>\eta\cdot\frac{1}{P_t-P_f}]\leq 2\exp(\frac{-\eta^2}{3nP_t})$.

Therefore, when $P_f=\frac{1}{2m-1+m\cdot e^\epsilon}$, with probability of $1-\beta/d$, we have:

$$\begin{aligned}&n\cdot|\theta_i-\tilde{\theta}_i|\\ &\leq\frac{e^\epsilon(2m\cdot e^\epsilon+m-1)\sqrt{m\cdot e^\epsilon+2m-1}}{(m\cdot e^\epsilon+m-1)(e^\epsilon-1)}\sqrt{3n\log(2d/\beta)}\\ &\leq\frac{e^\epsilon\sqrt{m\cdot e^\epsilon+2m-1}}{e^\epsilon-1}\sqrt{12n\log(2d/\beta)},\end{aligned}$$

which implies $|\theta_i-\tilde{\theta}_i|\leq\frac{32}{\epsilon}\sqrt{\frac{m}{n}\log(2d/\beta)}$ when $e^\epsilon<3$. Based on the union bound of probabilities on all $i\in[1:d]$, $|\theta-\tilde{\theta}|_{+\infty}\leq\frac{32}{\epsilon}\sqrt{\frac{m}{n}\log(2d/\beta)}$ holds with probability $1-\beta$. □

In pursuit of efficiency for answering frequency queries, the Wheel oracle in Algorithm 3 records only the bucketed counts of z, introducing additional bias/variance on frequencies. However, with a carefully calibrated bucketization parameter w, Theorem 5 assures that the oracle still attains $(O(\frac{1}{\epsilon}\sqrt{\frac{m}{n}\log(\frac{d}{\beta})}),\beta)$-accuracy.

Theorem 5 (Accuracy of the Wheel Frequency Oracle): Given $\epsilon\leq 1$ and n set-valued data over domain $\mathcal{X}^m$, the Wheel oracle with parameter $w\geq\max\{2,2e^\epsilon\sqrt{\frac{n}{3\log(2d/\beta)}}\}$ in Algorithm 3 is $(\frac{128}{\epsilon}\sqrt{\frac{m}{n}\log\frac{2d}{\beta}},\beta)$-accurate.

The proof of this theorem is provided in Appendix C, available online.

B. Efficient Heavy-Hitter Identification

We proceed to describe data structures and procedures for efficient set-valued data heavy-hitter identification. Drawing from classical solutions [24], [35], [36] for retrieving heavy-hitters, we adopt *sequence representation* of each item $v\in\mathcal{X}$ for recovering every heavy hitter's bits sequentially, and then use pseudo-random function (i.e., hash) to map each v to $[1:T]$ (i.e., *sketch encoding*) for reducing the chance of interference among multiple heavy hitters. Given the common assumption that heavy hitters have at least $\Omega(\sqrt{n\cdot m})$ occurrences, we anticipate the recovery of up to $O(\sqrt{n\cdot m})$ heavy hitters. When there is only one sketch, for almost perfect avoidance of interference, the hashed domain size T should scale with $O(n^2m^2)$. To reduce the size T, we leverage R replicas of the sketch encoding to reduce the interference probability exponentially. The nuances of these components are elaborated upon in the ensuing paragraphs.

Error-correcting Sequence Representation: For quickly recovering indexes of heavy-hitters, every element $x\in\mathcal{X}$ is now represented as L bits from binary alphabet $\{0,1\}$, where $L\in\mathbb{R}_+$ and $2^L\geq|\mathcal{X}|$ holds. That is, the sequence representation $Seq(x)$ of x is a list of symbols as

$$[Seq(x)_1,Seq(x)_2,\ldots,Seq(x)_L],$$

where each symbol $Seq(x)_j\in\{0,1\}$. Since there are interference from other items and privatization noises, we can encode the sequence with error-correction code. Let $Seq'(x)$ denote the $\frac{1}{8}$-error correcting encode of $x\in\mathcal{X}$, which can resist at most $\frac{L}{8}$ bit corruptions. Let L' denote the length of $Seq'(x)$. The $L'=O(\log|\mathcal{X}|)$ is sufficient for $\frac{1}{8}$-error correcting with encode/decode complexity of $O(\log|\mathcal{X}|)$ [44].

Sketch Encoders: To identify multiple heavy hitters, it is imperative to separately record (approximate) counts of their associated bits. One efficient approach involves using a hash function $h:\mathcal{X}\mapsto[1:T]$ that maps every item to buckets ranging from 1 to T. When T is adequately large, one can expect that every heavy hitter will not interfere with the counts of others. For a given element $x\in\mathcal{X}$, at every index $j\in[1:L']$, the data structure that logs hashed values is denoted as sketch $Skt_j(x)\in\{0,1\}^{[1:T]\times[0:1]}$, wherein only the $(h(x),Seq(x)_j)$-th value of $Skt_j(x)$ is 1, while remaining values stand at 0. Now considering a set-valued data $\mathbf{x}\in\mathcal{X}^m$, every element x within $\mathbf{x}$ can be mapped to produce a sketch $Skt_j(x)\in\{0,1\}^{[1:T]\times[0,1]}$ for every index $j\in[1:L']$. The sketch for the set-valued data, denoted as $Skt_j(\mathbf{x})\in\{0,1\}^{[1:T]\times[0,1]}$, is consequently defined as $Skt_j(\mathbf{x})_{t,b}=\max\{Skt_j(x)_{t,b}\}_{x\in\mathbf{x}}$ for every $t\in[1:T],b\in[0:1]$. This implies that values in the sketch are merely overwritten when two or more elements map to the same bucket and share an identical index value, as detailed in Algorithm 4. This adaptation ensures that $Skt_j(\mathbf{x})$ can be considered a set-valued data with no more than m elements.

Sketch Randomization and Estimation: The encoded sketch $Skt_j(\mathbf{x})$ can be interpreted as set-valued data with domain size of $2T$ and a cardinality up to m. For preserving ϵ-LDP, the sketch is subsequently fed into the randomization process for set-valued data in Algorithm 1. The server collects private views, and functions as a frequency oracle for the sketch summation $\mathbf{S}_j=\sum_{i=1}^n Skt_j(\mathbf{x}^i)$ sourced from n users (refer to lines 9 to 13 in Algorithm 4).

Sketch Decoder: Given the noiseless sketch summary $\mathbf{S}_j\in\mathbb{R}^{[1:T]\times[0:1]}$, one can simply enumerate all cells in every bucket, and assemble all bits with a frequency greater than some threshold Δ. Taking into account the noises due to local privacy, if the absolute noise is at most α with a high probability, given a

Algorithm 4: Set-Valued Heavy Hitter Identification.

Input: Random partition of $[1, n]$ to $R \cdot L'$ disjoint subsets $I_{1,1}, ..., I_{R,L'}$, the set-valued data $\mathbf{x}^j \in \mathcal{X}^m = \{c \mid c \subseteq \mathcal{X} \text{ and } |c| = m\}$ from n users, hash functions $\{h_1, ..., h_R\}$ that maps $\mathcal{X}$ to $[1, T]$.

Output: An accurate list of heavy hitters.

// Construct frequency oracle for each index and cohort

for $r \in [1, R]$ **do**
  for $i \in [1, L']$ **do**
    get $Oracle_{r,i}$ by Algorithm 3 with users in $I_{r,i}$ and sketched set-valued data $\{Skt_i(\mathbf{x}^j)\}_{j \in I_{r,i}}$
  end
end

// Decode heavy indexes to ensemble candidate heavy-hitters

initialize a list $H = \Phi$

for $r \in [1, R]$ **do**
  for $t \in [1, T]$ **do**
    initialize indexes $\hat{c}_{r,t}$ as $\{0\}^{[1,L']}$
    for $i \in [1, L']$ **do**
      $\hat{c}_{r,t,i} = \arg\max_{b \in \{0,1\}} Oracle_{r,i}(t, b)$
    end
    decode $\hat{v}_{r,t} \in \mathcal{X}$ from $\hat{c}_{r,t}$
    add $\hat{v}_{r,t}$ to H
  end
end

use Wheel oracle in Algorithm 3 with privacy budget $\frac{\epsilon}{2}$ to obtain frequency $F(\hat{v}_{r,t})$ of $\hat{v}_{r,t} \in H$ for all $(r, t) \in [1, R] \times [1, T]$

return $\{(\hat{v}_{r,t}, F(\hat{v}_{r,t}))\}_{(r,t) \in [1,R] \times [1,T]}$

heavy hitter x with more than $\alpha + \Delta$ occurrences, then all corresponding bits $(h(x), Seq(x)_j)_{j \in [1:l]}$ will exceed the threshold Δ (with a high probability). Therefore, for every $t \in [1 : T]$, one can record all bits with more than Δ frequencies, and ensemble full indexes of candidate symbols.

Dealing with Hash Conflicts: When T is not sufficiently large, there are hash conflicts of symbols, and the above decode procedure potentially yield exponential-size false positive candidates. For n users and each holds m items, it suffices to take $T \geq \Theta(n^2 m^2)$ for avoiding hash conflicts. Nonetheless, this leads to a direct implication where the server's computational cost escalates to $\Theta(T) = \Theta(n^2 m^2 2)$.

For reducing the size of T, we run $R = O(\log(1/\beta))$ concurrent identification protocols each with $T = \tilde{\Theta}(nm)$ as in [35]. Here $T = \tilde{\Theta}(nm)$ ensures the conflicts only happens with a constant probability, and $R = O(\log(1/\beta))$ repetitions ensure the conflict/failure probability remains confined to a negligible value β.

Drawing from the Wheel frequency oracle and the components previously discussed that address the efficiency/conflict dilemma, we introduce the overall heavy-hitter identification protocol for set-valued data in Algorithm 4. This protocol only incurs $\tilde{O}(\sqrt{n})$ memory consumption and $\tilde{O}(nm)$ computational costs on the server side.

VI. Utility of Heavy-Hitter Identification

This part analyze the utility performance of the Algorithm 4. The utility of the heavy-hitter identification protocol largely hinges on the accuracy of the frequency oracle. As formally analyzed in Theorem 5, the Wheel frequency oracle for n users with domain size $2T$ is $(\frac{128}{\epsilon}\sqrt{\frac{m}{n}\log(4T/\beta)}, \beta)$-accurate. Consequently, by setting Δ at $\frac{128}{\epsilon}\sqrt{nm\log(4T/\beta)}$, we can retrieve all items with more than $\frac{256}{\epsilon}\sqrt{nm\log(4T/\beta)}$ occurrences from the noisy $\hat{\mathbf{S}}$. Choosing $R = O(\log(1/\beta))$ and $T = O(\frac{\epsilon nm}{\sqrt{R\log d}})$, we present the utility guarantee of our protocol in Theorem 6. This ensures the identification of all heavy hitters having frequency greater than $\Omega(\frac{1}{\epsilon}\sqrt{nm\log(d/\beta)\log(1/\beta)})$ with a high probability.

Theorem 6 (Effectiveness of the Heavy-hitter Identification Protocol): For $\epsilon \leq 1.0$, the set-valued heavy-hitter identification protocol with $(m \cdot l, \epsilon, \frac{1}{2m \cdot l - 1 + m \cdot l \cdot e^{\epsilon}})$-Wheel mechanism satisfies ϵ-LDP, and returns a list H of length $\tilde{O}(\sqrt{n})$ satisfying:

1) (Precision Bound) With probability $1 - \beta$, for every $(i, v) \in H$, we have $|v - n \cdot \theta_i| \leq O(\frac{1}{\epsilon}\sqrt{nm\log(n/\beta)})$.
2) (Recall Bound) With probability $1 - \beta$, for every $i \in [1, d]$ that $n \cdot \theta_i \geq O(\frac{1}{\epsilon}\sqrt{nm\log(d/\beta)\log(1/\beta)})$, we have i in the list H.

The detailed proof of Theorem 6 is provided in Appendix D, available online.

Compared to the seminal protocol in [26] that identifies heavy-hitters by iterating over raw frequency oracles, our approach reduces computational costs from $O(d)$ to $\tilde{O}(R \cdot T) = \tilde{O}(nm)$. When contrasted with the improved protocol in [33] that first samples and then utilizes categorical heavy-hitter identification, our approach reduces the identification bar from $\tilde{O}(\frac{\sqrt{nm^2}}{\epsilon})$ to $\tilde{O}(\frac{\sqrt{nm}}{\epsilon})$.

For better understanding the identification bar on set-valued data, we compare it with protocols for categorical data. Relative to the recall bound $O(\frac{1}{\epsilon}\sqrt{n\log(d)\log(1/\beta)})$ and error bound $O(\frac{1}{\epsilon}\sqrt{n\log(n/\beta)})$ in [35], our bounds for set-valued data are only amplified by a factor of $\sqrt{m}$. The optimal recall bound for categorical data is $O(\frac{1}{\epsilon}\sqrt{n\log(1/\beta)})$. The bound is achievable by employing list-recoverable code via per-index hash for further decreasing hash conflicts [36]. We conjecture that when the list-recoverable code is applied to set-valued heavy-hitter identification, and the recall lower bound in the Theorem 6 could be optimized by a factor of $\sqrt{\log(d)}$ to $O(\frac{1}{\epsilon}\sqrt{nm\log(1/\beta)})$.

When applied to categorical data with $m \equiv 1$, our protocol matches the error lower bound of $\tilde{O}(\sqrt{\frac{\log(d)}{n\epsilon^2}})$ [24], holding a similar bound to that in [35]. Our protocol differs from previous approaches [24], [35] in two aspects. First, the efficient frequency oracle in [35] is achieved by running multiple concurrent ϵ-LDP protocols on hashed items and then using the median estimator, which suffers a loose error bound of $\frac{400}{\epsilon}\sqrt{n\log(\frac{12nd}{\beta})}$. As comparison, we use the protocol in Algorithm 3 that achieves a tighter error bound of $\frac{128}{\epsilon}\sqrt{n\log(\frac{2d}{\beta})}$. Second, our protocol carefully handles set-valued data with cardinality $m > 1$. If one were to naively consider n users holding m items as $n \cdot m$ singular-item users and subsequently apply existing approaches,

TABLE IV
PARAMETERS FOR EXPERIMENTS ON DISTRIBUTION ESTIMATION

Parameter	Enumerated values
domain size d	256, 512, 1024, 2048
cardinality m	1, 2, 4, 16
number of users n	10000, 100000, 1000000
privacy budget ϵ	0.001, 0.01, 0.1, 0.2, 0.8, 1.0, 1.5, 2.0, 3.0

this would inevitably either fragment the privacy budget or induce dependencies amongst the R parallel protocols. Such an approach would fail to reach the identification threshold of $\tilde{O}(\sqrt{\frac{nm}{\epsilon^2}})$.

VII. EXPERIMENTS OF DISTRIBUTION ESTIMATION

In this section, we present an experimental evaluation focused on distribution estimation, considering both computational and statistical efficiencies. The parameters selected for this evaluation are outlined in Table IV, reflecting the configurations commonly observed in mobile network environments.

A. Experimental Settings

Datasets: Given that the performance of set-valued mechanisms is data-independent, we have created several synthetic datasets by uniformly sampling m-sized data from a d-sized domain.

Competitors: We compare the Wheel mechanism with two other prominent mechanisms: the RAPPOR mechanism [27] with a Bloom filter size of d and the PrivSet mechanism [32]. The PrivSet mechanism currently shows the best empirical utility performance for set-valued data distribution estimation.

It should be noted that the scaled distribution $\frac{\theta}{m}$ lies in the probability simplex. In contrast, the distribution estimator $\frac{\hat{\theta}}{m}$, as produced by all LDP methodologies, might not adhere to this property. With the prior knowledge that $m = \sum_{i\in[1,d]} \theta_i$, the estimated distributions during each simulation can be optimized by mapping to the probability simplex [45]. The results presented in this section are derived from these post-processed distribution estimators.

Evaluation Metrics: The *average running time* is used to measure the computational cost of the data randomization procedure on the user side. The procedures are programmed in Python3 language running on a laptop embedded with Intel i5-8250U CPU. The performance metrics regarding data utility (distribution estimation accuracy) include the *total variation error* (TVE, a.k.a. ℓ_1-norm error):

$$\|\hat{\theta} - \theta\|_1 = \sum_{i\in[1,d]} |\hat{\theta}_i - \theta_i|,$$

and the *maximum absolute error* (MAE, a.k.a. ℓ_∞-norm error):

$$\|\hat{\theta} - \theta\|_\infty = \max_{i\in[1,d]} |\hat{\theta}_i - \theta_i|.$$

The results represent the average of 200 independent trials. Data from extreme privacy conditions, such as $\epsilon = 0.0001$ or $\epsilon = 10.0$, are detailed in Table V to supplement the subsequent findings.

TABLE V
EXPERIMENTAL TVE RESULTS UNDER EXTREMELY LOW/HIGH PRIVACY BUDGETS

Mechanisms	ϵ = 0.0001	ϵ = 1.0	ϵ = 10.0
RAPPOR	7.21	4.43	0.94
PrivSet	7.20	3.42	0.18
Wheel	7.20	3.73	0.25

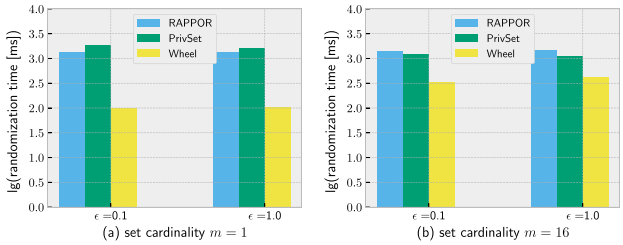


Fig. 4. Total randomization time on $n = 1\,000$ users with domain size $d = 512$ and cardinality $m = 1$ or 16.

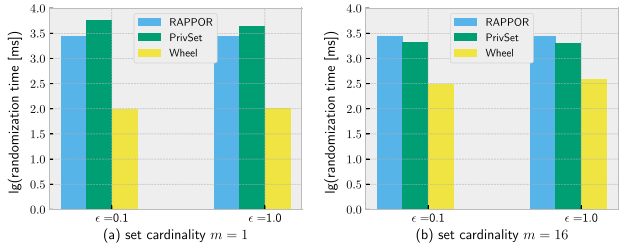


Fig. 5. Total randomization time on $n = 1\,000$ users with domain size $d = 1024$ and cardinality $m = 1$ or 16.

We have chosen not to include experimental results regarding the communication costs of the different methods. Their communication complexities are already theoretically discussed in Table I, and the required bits for transmission can be theoretically determined.

B. Computational Efficiency Evaluation

In this part, we evaluate both the client-side and server-side computational costs of competitive mechanisms.

1) Vary Item Cardinality m: Fig. 4 presents the decimal logarithm of user-side running time of three mechanisms, with varying item cardinality m and fixed domain size $d = 512$. We can see that the Wheel mechanism is extremely fast. Every user finishes data randomization procedure in a few milliseconds, while other mechanisms need tens of milliseconds. The wheel mechanism is 3-10x faster than other mechanisms.

2) Vary Domain Size d: We here study how the computational costs vary with the domain size d. Results of the user-side running time are shown in Figs. 5 and 6, when the domain size d is 1024 and 2048 respectively. Compared with results in Fig. 4, it can be observed that the costs of Wheel mechanism are domain-size independent, while the costs of RAPPOR and

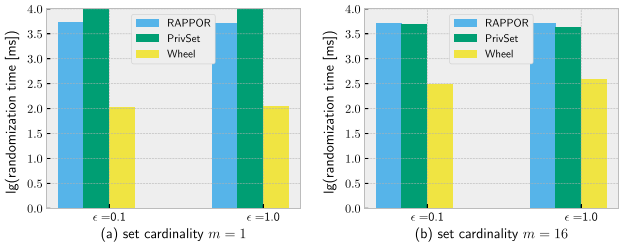


Fig. 6. Total randomization time on $n = 1\,000$ users with domain size $d = 2048$ and cardinality $m = 1$ or 16.

TABLE VI
SERVER-SIDE RUNNING TIME RESULTS IN SECONDS OF WHEEL MECHANISM

Settings	$\epsilon = 0.1$	$\epsilon = 1.0$	$\epsilon = 3.0$
$n = 10^3, d = 128, m = 2$	0.42	0.50	0.45
$n = 10^3, d = 128, m = 8$	0.47	0.53	0.41
$n = 10^5, d = 128, m = 8$	49.98	56.15	54.31
$n = 10^3, d = 4096, m = 8$	14.95	15.01	15.09

PrivSet mechanisms grow at least linearly with the domain size. The Wheel mechanism is $5 - 100$x faster than other mechanisms. When the domain size is large in real-world applications (e.g., tens of thousands of web URLs, products or Apps), the performance gap can be even larger.

3) Sever-Side Running Time: To assess the efficiency of the Wheel mechanism on the server side, we present the running time results in Table VI. The results illustrate that the server-side running time increases linearly with both the domain size d and the number of users n.

C. Statistical Efficiency Evaluation

In this part, we evaluate the server-side estimation accuracy of competitive mechanisms.

1) Vary Item Cardinality m: We simulate with 100000 users and the item domain size at 512. Fig. 7(a) and (b) show the natural logarithm of TVE errors and MAE errors for distribution estimation respectively. The wheel mechanism achieves close estimation accuracy to the cost-intensive PrivSet mechanism. Except for cases when the privacy budget is extremely low (and the estimated distribution is almost non-informative), the errors of the Wheel mechanism grow with $\sqrt{m}$, while errors of the RAPPOR mechanism grow with m.

2) Vary Domain Size d: Simulated with 100000 users, Fig. 8(a) and (b) show natural logarithm of TVE errors and MAE errors for distribution estimation respectively when the item domain size is 256, Fig. 9(a) and (b) show results when the item domain size is 1024. The TVE and MAE errors of the wheel mechanism grow roughly with $\sqrt{d}$.

3) Vary Number of Users n: Simulated with 10000 users and the item domain size at 512, Fig. 10(a) and (b) show logarithm of TVE errors and MAE errors for distribution estimation respectively. Compared to the results in Fig. 7(a) and (b), the estimation errors of all mechanisms increase roughly by $\sqrt{\frac{100000}{10000}}$.

TABLE VII
EXPERIMENTAL PARAMETERS FOR HEAVY-HITTER IDENTIFICATION

Parameter	Enumerated values
domain size d	$2^{20}, 2^{24}$
cardinality m	8, 16
number of users n	5000000
privacy budget ϵ	1.0

D. Summary

Through above experiments, we can conclude that the Wheel mechanism not only puts domain-independent and minimum computational overheads on the user side, but also achieves optimal utility for item distribution estimation on the server side. These experimental results confirm our theoretical error bounds $O(\frac{dm}{n\epsilon^2})$ (the TVE/MAE is usually proportional to the root of mean squared error). It seems that the PrivSet mechanism also has excellent practical statistical efficiency, but its computation/communication overheads on the user side are linear to the domain size d, and are much higher than the Wheel mechanism. In conclusion, for set-valued data distribution estimation under local differential privacy, the Wheel mechanism has great theoretical and practical advantages over existing approaches.

VIII. EXPERIMENTS FOR HEAVY-HITTER IDENTIFICATION

This section experimentally evaluates the effectiveness of heavy-hitter identification on set-valued data. The parameter configurations are detailed in Table VII.

A. Experimental Settings

Datasets: To comprehensively evaluate the performance of heavy-hitter identification across various settings, we create synthetic datasets with a range of parameters. Each set-valued data consists of m distinct samples from a categorical distribution of size d. The dataset includes $\min\{5m, \sqrt{n}\}$ true heavy hitters and $40m$ non-heavy hitters.

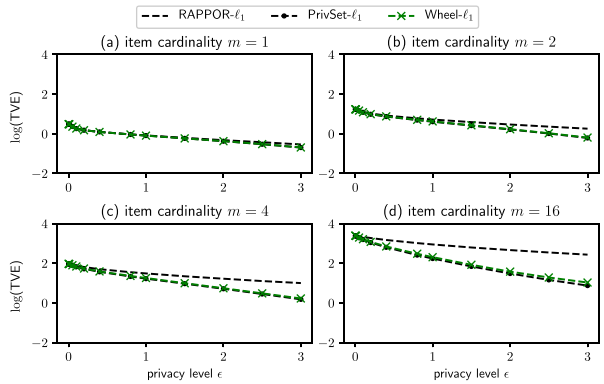
Competitors: Within the framework for set-valued heavy-hitter identification described in Algorithm 4, we contrast the Wheel-based oracle in Algorithm 3 with two other efficient set-valued frequency oracles: the RAPPOR mechanism [27] with a Bloom filter size of d and the privacy-amplified Generalized Randomized Response (PA-GRR) mechanism [31], which is a privacy-amplified version of the sampling RR [26].

Evaluation Metrics: Let $\tilde{H}$ denote true heavy hitters $\{i | i \in [1, d]\ and\ \theta_i \geq \Delta\}$, and H denote the estimated list of heavy-hitters. The performance metrics regarding the heavy-hitter identification include the *Precision* that measures the proportion of true heavy-hitters in estimation:

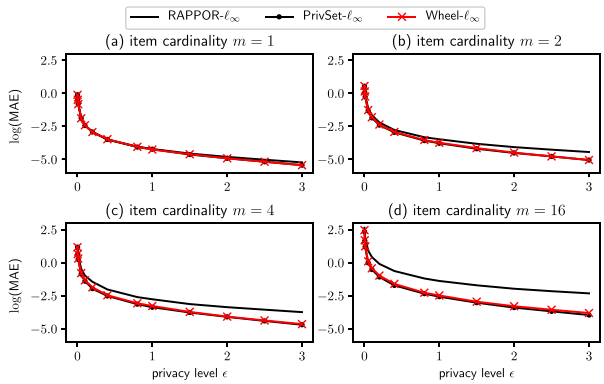
$$Precision = \frac{|H \bigcap \tilde{H}|}{|H|},$$

the *Recall* that measures the rate of successfully retrieved heavy hitters:

$$Recall = \frac{|H \bigcap \tilde{H}|}{|\tilde{H}|},$$

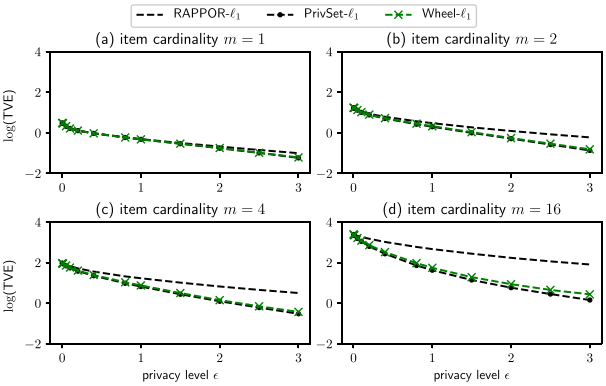


(a) Mean TVE error.

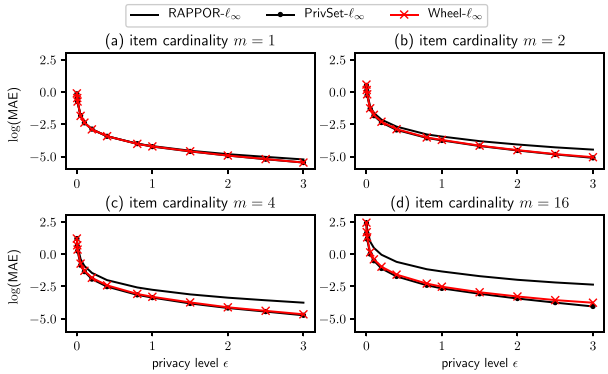


(b) Mean MAE error.

Fig. 7. Experimental results on $n = 100\,000$ users with domain size $d = 512$ and the set cardinality m ranges from 1 to 16.

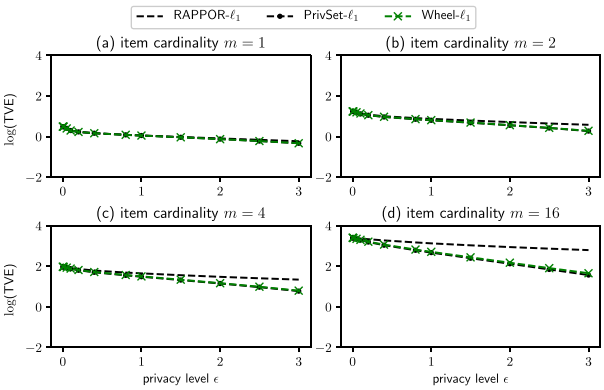


(a) Mean TVE errors.

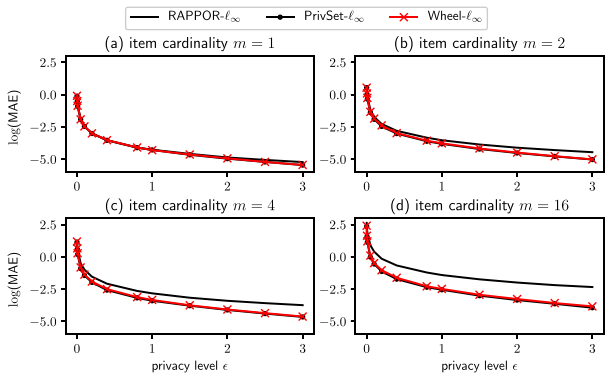


(b) Mean MAE errors.

Fig. 8. Experimental results on $n = 100\,000$ users with domain size $d = 256$ and the set cardinality m ranges from 1 to 16.



(a) Mean TVE errors.



(b) Mean MAE errors.

Fig. 9. Experimental results on $n = 100\,000$ users with domain size $d = 1024$ and the set cardinality m ranges from 1 to 16.

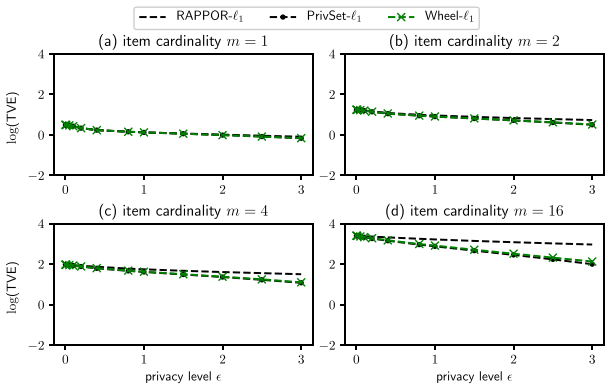
and the more balanced F_1 score:

$$F_1\text{-}Score = 2\frac{Precision \cdot Recall}{Precision + Recall}.$$

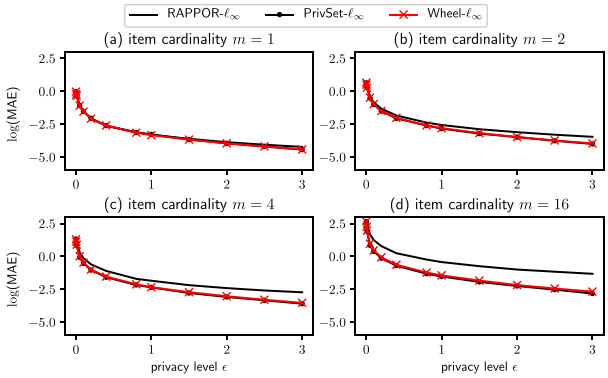
All results are the average values of 200 independent trials.

B. Results

When simulated with 5,000,000 users, the results for domain size $d = 2^{20}$ and $d = 2^{24}$ are presented in Tables VIII and IX, respectively. Compared to existing methods, the heavy-hitter identification protocol with the Wheel oracle has significant



(a) Mean TVE errors.



(b) Mean MAE errors.

Fig. 10. Experimental results on $n = 10\,000$ users with domain size $d = 1024$ and the set cardinality m ranges from 1 to 16.

TABLE VIII
HEAVY-HITTER IDENTIFICATION RESULTS WITH DOMAIN SIZE $d = 2^{20}$ AND PRIVACY BUDGET $\epsilon = 1$

	Mechanisms	Precision	Recall	F_1-Score
$m = 8$	d-RAPOOR	0.837	0.364	0.503
	PA-GRR	0.896	0.658	0.755
	Wheel	0.877	0.741	0.802
$m = 16$	d-RAPOOR	0.047	0.002	0.004
	PA-GRR	0.605	0.077	0.137
	Wheel	0.861	0.351	0.493

TABLE IX
HEAVY-HITTER IDENTIFICATION RESULTS WITH DOMAIN SIZE $d = 2^{24}$ AND PRIVACY BUDGET $\epsilon = 1$

	Mechanisms	Precision	Recall	F_1-Score
$m = 8$	d-RAPOOR	0.803	0.344	0.478
	PA-GRR	0.882	0.651	0.740
	Wheel	0.865	0.728	0.789
$m = 16$	d-RAPOOR	0.022	0.001	0.002
	PA-GRR	0.538	0.068	0.121
	Wheel	0.854	0.342	0.486

better performances (i.e., achieve over 0.8 F_1-Score when $d = 2^{20}$ and $m = 8$). When the domain size $d = 2^{24}$ that is hardly to enumerate, the Wheel oracle can also achieve about 0.79 F_1-Score with $m = 8$. Notably, as the set cardinality m increases, the performance difference compared to other methods becomes more pronounced. These outcomes underscore the efficacy of our approach, which processes the set-valued data in its entirety, without resorting to item sampling or budget division.

IX. CONCLUSION

For set-valued data distribution estimation with ϵ-LDP, this study provides tight minimax error bounds and proposes an efficient and optimal mechanism: the Wheel Mechanism. The mechanism has the advantage of being domain-independent and requires only $O(\min m \log m, me^\epsilon)$ computational costs on the user's side and $O(\log(me^\epsilon))$ communication costs between the user and the server. Compared with existing approaches that depend on $O(d)$ or $O(\log d)$, the proposed Wheel Mechanism is practical for large-scale set-valued data aggregation in an online service or mobile environment. For the service provider, the mechanism offers an unbiased distribution estimator with significantly improved error bounds from the previous $O(\frac{m^2 d}{n\epsilon^2})$ to the optimal rate of $O(\frac{md}{n\epsilon^2})$. For set-valued heavy-hitter identification with ϵ-LDP, we introduce a protocol that achieves $\tilde{O}(\frac{1}{\epsilon}\sqrt{\frac{m}{n}\log d})$ identification bar and yields a factor of $\sqrt{m}$ reduction when compared to existing approaches. The protocol builds upon an efficient oracle variant of the original Wheel Mechanism. Experimental results demonstrate the computational/statistical efficiency of our proposals. Specifically, it achieves a 3-100x speedup on user-side running time.

ACKNOWLEDGMENTS

This work was extended from [46] in the 46th International Conference on Very Large Data Bases (VLDB 2020).

REFERENCES

[1] Ş. Gündüz and M. T. Özsu, "A web page prediction model based on click-stream tree representation of user behavior," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2003, pp. 535–540.
[2] J. Lin, J. Niu, X. Liu, and M. Guizani, "Protecting your shopping preference with differential privacy," *IEEE Trans. Mobile Comput.*, vol. 20, no. 5, pp. 1965–1978, May 2021.
[3] T. M. T. Do, J. Blom, and D. Gatica-Perez, "Smartphone usage in the wild: A large-scale analysis of applications and context," in *Proc. ACM 13th Int. Conf. Multimodal Interfaces*, 2011, pp. 353–360.
[4] O. Arias, J. Wurm, K. Hoang, and Y. Jin, "Privacy and security in Internet of Things and wearable devices," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 1, no. 2, pp. 99–109, Second Quarter 2015.
[5] J. Li, Z. Li, G. Tyson, and G. Xie, "Characterising usage patterns and privacy risks of a home security camera service," *IEEE Trans. Mobile Comput.*, vol. 21, no. 7, pp. 2344–2357, Jul. 2022.
[6] O. Tene and J. Polonetsky, "Privacy in the age of Big Data: A time for big decisions," *Stan. L. Rev. Online*, vol. 64, 2011, Art. no. 63.
[7] P. Voigt and A. Von dem Bussche, *The EU General Data Protection Regulation (GDPR)*. Berlin, Germany: Springer, 2017, vol. 18.
[8] E. Goldman, "An introduction to the California consumer privacy act (CCPA)," Santa Clara Univ. Legal Stud. Res. Paper, 2019. [Online]. Available: https://ssrn.com/abstract=3211013
[9] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proc. IEEE 54th Annu. Symp. Found. Comput. Sci.*, 2013, pp. 429–438.

[10] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.*, Springer, 2008, pp. 1–19.
[11] R. Chen, N. Mohammed, B. C. Fung, B. C. Desai, and L. Xiong, "Publishing set-valued data via differential privacy," in *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 1087–1098, 2011.
[12] N. Li, W. Qardaji, D. Su, and J. Cao, "Privbasis: Frequent itemset mining with differential privacy," *Proc. VLDB Endowment*, vol. 5, no. 11, pp. 1340–1351, 2012.
[13] C. Zeng, J. F. Naughton, and J.-Y. Cai, "On differentially private frequent itemset mining," *Proc. VLDB Endowment*, vol. 6, no. 1, pp. 25–36, 2012.
[14] G. Cormode, C. Procopiuc, D. Srivastava, and T. T. Tran, "Differentially private summaries for sparse data," in *Proc. ACM 15th Int. Conf. Database Theory*, 2012, pp. 299–311.
[15] L. Sweeney, "k-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 05, pp. 557–570, 2002.
[16] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," in *Proc. 22nd Int. Conf. Data Eng.*, 2006, pp. 24–24.
[17] G. Ghinita, Y. Tao, and P. Kalnis, "On the anonymization of sparse high-dimensional data," in *Proc. IEEE 24th Int. Conf. Data Eng.*, 2008, pp. 715–724.
[18] M. Terrovitis, N. Mamoulis, and P. Kalnis, "Privacy-preserving anonymization of set-valued data," *Proc. VLDB Endowment*, vol. 1, pp. 115–125, 2008.
[19] Y. He and J. F. Naughton, "Anonymization of set-valued data via top-down, local generalization," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 934–945, 2009.
[20] E. De Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," in *Proc. Int. Conf. Financial Cryptography Data Secur.*, Springer, 2010, pp. 143–159.
[21] D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung, "Efficient robust private set intersection," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur.*, Springer, 2009, pp. 125–142.
[22] Y. Huang, D. Evans, and J. Katz, "Private set intersection: Are garbled circuits better than custom protocols?," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2012, pp. 272–283.
[23] Y. Lindell, "Secure multiparty computation for privacy preserving data mining," in *Encyclopedia of Data Warehousing and Mining*. Hershey, PA, USA: IGI Global, 2005, pp. 1005–1009.
[24] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Proc. 47th Annu. ACM Symp. Theory Comput.*, 2015, pp. 127–135.
[25] S. Wang, Y. Nie, P. Wang, H. Xu, W. Yang, and L. Huang, "Local private ordinal data distribution estimation," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.
[26] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 192–203.
[27] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 1054–1067.
[28] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries," *Proc. Privacy Enhancing Technol.*, no. 3, pp. 41–61, 2016.
[29] A. Greenberg, "Apple's 'differential privacy'is about collecting your data–but not your data," *Wired*, Jun. 2016. [Online]. Available: https://www.wired.com/2016/06/apples-differential-privacy-collecting-data/
[30] J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang, "Privacy loss in Apple's implementation of differential privacy on MacOS 10.12," 2017, *arXiv: 1709.02753*.
[31] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *Proc. IEEE Symp. Secur. Privacy*, 2018, pp. 127–143.
[32] S. Wang, L. Huang, Y. Nie, P. Wang, H. Xu, and W. Yang, "Privset: Set-valued data analyses with locale differential privacy," in *Proc. IEEE Conf. Comput. Commun.*, 2018, pp. 1088–1096.
[33] W. Zhu, P. Kairouz, B. McMahan, H. Sun, and W. Li, "Federated heavy hitters discovery with differential privacy," in *Proc. Int. Conf. Artif. Intell. Statist.*, PMLR, 2020, pp. 3837–3847.
[34] P. Kairouz, K. Bonawitz, and D. Ramage, "Discrete distribution estimation under local privacy," in *PRoc. Int. Conf. Mach. Learn.*, 2016, pp. 2436–2444.
[35] R. Bassily, K. Nissim, U. Stemmer, and A. G. Thakurta, "Practical locally private heavy hitters," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2288–2296.
[36] M. Bun, J. Nelson, and U. Stemmer, "Heavy hitters and the structure of local privacy," *ACM Trans. Algorithms*, vol. 15, no. 4, pp. 1–40, 2019.
[37] D. Boneh, E. Boyle, H. Corrigan-Gibbs, N. Gilboa, and Y. Ishai, "Lightweight techniques for private heavy hitters," in *Proc. IEEE Symp. Secur. Privacy*, 2021, pp. 762–776.
[38] H. Wu and A. Wirth, "Asymptotically optimal locally private heavy hitters via parameterized sketches," in *Proc. Int. Conf. Artif. Intell. Statist.*, PMLR, 2022, pp. 7766–7798.
[39] A. G. Thakurta et al., "Learning new words," U.S Patent 9,594,741, Mar. 14 2017.
[40] M. Ye and A. Barg, "Optimal schemes for discrete distribution estimation under locally differential privacy," *IEEE Trans. Inf. Theory*, vol. 64, no. 8, pp. 5662–5676, Aug. 2018.
[41] B. Yu, "Assouad, fano, and Le cam," in *Festschrift for Lucien Le Cam*. Berlin, Germany: Springer, 1997, pp. 423–435.
[42] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Minimax optimal procedures for locally private estimation," *J. Amer. Stat. Assoc.*, vol. 113, no. 521, pp. 182–201, 2018.
[43] M. L. Fredman and B. Weide, "On the complexity of computing the measure of [ai, bi]," *Commun. ACM*, vol. 21, no. 7, pp. 540–544, 1978.
[44] P. Elias, "Error-correcting codes for list decoding," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 5–12, Jan. 1991.
[45] W. Wang and M. A. Carreira-Perpin án, "Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application," 2013, *arXiv:1309.1541*.
[46] S. Wang, Y. Qian, J. Du, W. Yang, L. Huang, and H. Xu, "Set-valued data publication with local privacy: Tight error bounds and efficient mechanisms," *Proc. VLDB Endowment*, vol. 13, no. 8, pp. 1234–1247, 2020.
[47] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
[48] N. Alon and J. H. Spencer, *The Probabilistic Method*. Hoboken, NJ, USA: John Wiley & Sons, 2016.

Shaowei Wang (Member, IEEE) received the PhD degree from the School of Computer Science and Technology, University of Science and Technology of China (USTC). He is an associate professor with the Institute of Artificial Intelligence and Blockchain, Guangzhou University. His research interests are data privacy, federated learning and recommendation systems. He has published more than 20 papers on top-tier conferences and journals, such as INFOCOM, VLDB, IJCAI, *IEEE Transactions on Parallel and Distributed Systems*, and *IEEE Transactions on Knowledge and Data Engineering*.

Yuntong Li (Member, IEEE) is currently working toward the master's degree with Guangzhou University. His research interests include differential privacy and federated learning.

Yusen Zhong is currently working toward the master's degree with Guangzhou University. His research interests are differential privacy and federated learning.

Kongyang Chen received the PhD degree in computer science from the University of Chinese Academy of Sciences, China. He is currently an associate professor with the Institutes of Artificial Intelligence and Blockchain, Guangzhou University, China. His main research interests are artificial intelligence, privacy computing, edge computing, and distributed systems.

Xianmin Wang received the BS degree from Suzhou University, Jiangsu, China, in 2006, and the MS degree in computer science from the Jiangxi University of Science and Technology, Jiang Xi, China, in 2013, the PhD degree in computer science from Beihang University, in 2017. Currently, he is working with Institutes of Artificial Intelligence and Blockchain, Guangzhou University, China. His research interests include deep learning, image processing, and understanding.

Zhili Zhou (Senior Member, IEEE) received the MS and PhD degrees in computer application from the School of Information Science and Engineering from Hunan University, in 2010 and 2014, respectively. He is currently a professor with the Institute of Artificial Intelligence, Guangzhou University. His current research interests include multimedia security, artificial intelligence security and information hiding. He has been selected as "World's Top 2% Scientists" from 2019 to 2022 by Stanford University and Elsevier. He received ACM Rising Star Award and got Guangdong Natural Science Funds for Distinguished Young Scholar.

Fei Peng (Member, IEEE) received the PhD degree in circuits and systems from the South China University of Science and Technology, Guangzhou, China, in 2006. He was a visiting fellow with the Department of Computer Science, The University of Warwick, U.K., from 2009 to 2010. He was a visiting professor with the SeSaMe Centre, School of Computing, National University of Singapore, in 2016. He is currently a professor with the Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou. His areas of interest include digital watermarking and digital forensics.

Yuqiu Qian received the PhD degree in computer science from the University of Hong Kong, in 2019. She is currently a senior researcher with Tencent in Shenzhen, China. Her research interests include distributed computing, recommendation systems, and machine learning.

Jiachun Du received the PhD degree in mathematic from the University of Science and Technology of China (USTC), in 2007. He is currently a senior researcher with Tencent in Shenzhen, China. His research interests include recommendation systems and game data mining.

Wei Yang (Member, IEEE) received the PhD degree in computer science from the University of Science and Technology of China (USTC), in 2007. He is an associate professor with the School of Computer Science and Technology, USTC. His research interests include information security, quantum information, and human-computer interaction.