Top-k Discovery under Local Differential Privacy: An Adaptive Sampling Approach

Rong Du, Qingqing Ye, Yue Fu, Haibo Hu, Kai Huang

Abstract—Local differential privacy (LDP) is a promising privacy model for data collection that protects sensitive information of individuals. However, applying LDP to top-k estimation in set-valued data (e.g., identifying most frequent k items) may yield poor results for small and sparse datasets due to high sensitivity and heavy perturbation. To address this, we propose an adaptive approach that frames the problem as a multi-armed bandit (MAB) problem, in which the decision-maker selects actions based on information collected from previous rounds to maximize the total reward over time. Inspired by this, we present two adaptive sampling schemes based on MAB: **ARBS** for identifying top-k items and **ARBSF** for both top-k item discovery and frequency estimation on these items. Furthermore, to address the potential long delay of multi-round collection, we propose an optimization technique to reduce the time complexity. Both theoretical and empirical results show that our adaptive sampling schemes significantly outperform existing alternatives.

Index Terms—Local Differential Privacy; Top-k Estimation; Multi-armed Bandit.

1 Introduction

Local differential privacy (LDP) has become popular for protecting personal data privacy, particularly in big data analysis. It has already been deployed in many real-life data collection systems, including Google Chrome [1], [2], iOS [3], and Windows 10 [4]. By perturbing data through mechanisms such as Laplace or the LDP frequency oracle protocol [1], [5], [6], [7], [8], [9], LDP enables the data collector to aggregate statistical values (e.g., mean or frequency) using methods such as likelihood estimation [1] and regression [10], while providing deniability for users' private data.

In this paper, we study top-k item discovery in a set-valued dataset under LDP. In this setting, each user has a set of private items, such as the music playlist, web search history, and location trajectories. To protect the privacy, each user perturbs his data using LDP before sending it to the data collector. The collector then analyzes the perturbed data to identify the most frequent k items. For instance, in iOS [11], users' emoji data is perturbed using LDP before being sent to Apple. This enables Apple to estimate the frequency of each emoji and identify the most popular ones, without compromising user privacy.

However, the main challenge in discovering top-k items from a set-valued dataset by conventional LDP mechanisms lies in the fact that users have varying numbers of items and the item domain (with size d) is usually extremely large, which leads to poor utility of the estimation. LDPMiner [12] and SVIM [13] address these challenges using a padding-and-sampling-based frequency oracle (PSFO) approach, in which a subset of users (e.g., 20%) report the perturbed size of their private sets to determine the padding length l, which is typically set to the 90^{th} percentile of the former. The remaining users augment their private set with dummy items to pad to the size of l. Then one item is randomly selected from the padded set and reported using a frequency oracle GRR or OLH [13]. Although PSFO increases the probability of sampling an existing item by each user, it still faces several challenges, including inaccuracies in determining the padding length l for small datasets, excessive padding length for most users, and biased frequency estimation due to some users with more items than the padding length l.

Uniform sampling is an unbiased scheme that treats all items equally over time and can effectively address PSFO issues. Specially, each user generates a d-bit string with existing item locations marked as 1 and all other locations marked as 0. The user then samples and reports a bit (i.e., 0 or 1) from the string uniformly at random. However, uniform sampling can lead to most users uploading non-existent items, which contributes little information to the results. To overcome this limitation, the data collector can adjust the sampling scheme based on the information collected over time. This process can be regarded as a Multi-armed bandit (MAB) problem. In the MAB problem, the decision-maker encounters a d-armed bandit, where each arm corresponds to a unique probability distribution. The decision-maker adaptively samples n times from d arms to obtain its corresponding reward and achieve their intended targets, such as returning the k arms with the highest reward. In our LDP setting, the data collector can be seen as a decsion-maker and Each set values represent arms with the unique probability distribution. The discovery of the top-k items of set value data from n sampling times is equivalent to selecting k arms with the highest reward with n times in

- *Rong Du, Qingqing Ye, Yue Fu and Haibo Hu are with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong, China. E-mail: roong.du@connect.polyu.hk, qqing.ye@polyu.edu.hk, yuesandy.fu@connect.polyu.hk, haibo.hu@polyu.edu.hk*
- *Kai Huang is with the School of Computer Science and Engineering, Macau University of Science and Technology, Macau. Email: kylehuangk@gmail.com.*
- *Corresponding author: Qingqing Ye.*

a MAB problem.

In this paper, we explore the problem of identifying top-k set-valued data under LDP settings, with the goal of returning the sets of top-k items along with their frequencies, given a fixed number of samples. However, traditional MAB solutions need to be revised due to limited sample sizes, privacy budgets, and time constraints in LDP systems. To overcome these challenges, we propose an adaptive sampling algorithm, namely Adaptive-RR Bandit Sampling (ARBS), which divides users into multiple partitions and sequentially lets users in each partition report their local data in light of the previous estimation results. Consequently, it adaptively increases the sampling probabilities for those frequent items. On the other hand, we further introduce ARBS with frequency (ARBSF) for estimating frequencies of the identified items and a Delay-contrained Batch Sampling algorithm (DBS) for optimizing the user allocation when given fixed rounds of interactions between the users and the collector, ensuring both accuracy and timeliness. To summarize, our contributions are three folded.

- To the best of our knowledge, this is the first work to formulae set-valued data collection under LDP as an MAB problem, which inspires us to enhance the estimation results significantly.
- We propose ARBS for identifying top-k frequent items and ARBSF for estimating the frequencies of them, in which we utilize data characteristics and adaptively adjust the sampling scheme to obtain better utility.
- We propose a minimal error scheme to ensure the constraint on limited rounds is fully satisfied, and we demonstrate its effectiveness through extensive empirical analysis.

The remainder of this paper is organized as follows. In Section 2, we introduce some preliminaries of LDP. In Section 3, we formally give the problem definition. In Section 4, we present the details of the proposed adaptive sampling scheme , followed by a delay-constrained solution in Section 5. In Section 6, we provide extensive experimental evaluations. Then we review the related studies in Section 7, and conclude the paper in Section 8.

2 PRELIMINARIES

2.1 Local Differential Privacy

LDP provides a strict privacy guarantee for individual data in untrusted environments where (centralized) differential privacy (DP) [14], [15], [16] cannot work without a trusted data collector. A randomized algorithm $\mathcal{A}$ satisfies ϵ-local differential privacy (a.k.a. ϵ-LDP, ϵ>0), if and only if for any two values x and x', and any possible output $y \in$ Range($\mathcal{A}$), the following condition holds:

$$e^{-\epsilon} \leq \frac{P[\mathcal{A}(x) = y]}{P[\mathcal{A}(x') = y]} \leq e^{\epsilon}.$$

The intuition of LDP is the probabilistic nature that $\mathcal{A}$ maps any particular input to some output according to an underlying distribution, controlled by the privacy budget ϵ. Therefore, anyone is unable to tell any individual's true answer from observing an output y.

2.2 Multi-armed Bandit vs. Randomized Response

The MAB (or bandits) is a powerful hypothesis generation and exploration technique. It can be considered as a collection of arms with real Bernoulli distributions $D = \{D_1, \ldots, D_n\}$. Let $\{p_1, \ldots, p_d\}$ be the mean values associated with these reward distributions. Each arm i has the value 1 with probability p_i and has the value 0 with probability $q_i = 1 - p_i$. The probability mass function f of this distribution, over a possible output z, is

$$f(z; p_i) = \begin{cases} p, & \text{if } z = 1, \\ q & \text{if } z = 0. \end{cases}$$

The gambler iteratively plays one arm per round and observes the associated reward. This associated reward can be viewed as a form of the Randomized Response technique [17].

The Randomized Response (RR) is the most well-known perturbation algorithm to achieve ϵ-LDP. This algorithm is frequently used in sensitive "yes or no" problems where a user makes a decision based on a biased coin flip to provide either a true or a false answer to the collector with a certain probability. The algorithm returns the output b' based on the given input b as follows.

$$b' = \begin{cases} b, & w.p. \quad \frac{e^{\epsilon}}{e^{\epsilon}+1}, \\ 1-b & w.p. \quad \frac{1}{e^{\epsilon}+1}. \end{cases}$$

In the MAB problem, a decision-maker aims to maximize expected gain by adaptively selecting arms to sample, seeking to minimize samples while identifying optimal arms. This approach is useful in sampling under LDP constraints, where data collectors adaptively choose set-valued responses to enhance utility and performance within ϵ-LDP limits. By treating LDP sampling as a MAB problem, adaptive strategies can improve estimation accuracy and reduce costs while maintaining privacy.

2.3 Hoeffding Bounds vs. Empirical Bernstein Bounds

Hoeffding bounds. Hoeffding's inequality, as proven by Wassily Hoeffding in 1963 [21], gives an upper bound for the likelihood that the sum of constrained independent random variables deviates from its expected value. Let $\{X_1, \ldots, X_t\}$ be real-valued i.i.d. random variables with the range R ($R = 1$ for binary data) and the mean μ, and let $\overline{X_t} = \frac{1}{t}\sum_{i=1}^{i=t} X_i$. With a probability of at least $1 - \delta$, Hoeffding's inequality states that:

$$|\overline{X_t} - \mu| <= R\sqrt{\frac{log(2/\delta)}{2t}}. \quad (1)$$

Hoeffding's inequality has been extensively applied to online learning scenarios due to its generality. One limitation of the bound is that it scales linearly with the range R, but not with the variance of X_i. In cases where the variance bound is known and relatively small compared to the range, Bernstein's inequality is a more appropriate choice due to its better handling of small variance bounds. However, when we rarely have prior knowledge of variance bounds, Hoeffding's inequality is more practical.

Empirical Bernstein bounds. The empirical Bernstein bound [18] provides a tighter bound than Hoeffding's inequality, depending on the empirical standard deviation

$\overline{\sigma}_t^2 = \frac{1}{t}\sum_{i=1}^{t}(X_i - \hat{x}_t)^2$. The empirical Bernstein bound states that with a probability of at least $1-\delta$,

$$|Xt - \mu| \leq \overline{\sigma_t}\sqrt{\frac{2log(3/\delta)}{t}} + \frac{3Rlog(3/\delta)}{t}. \quad (2)$$

The rate of decline for the term associated with range R is t^{-1}, and it becomes insignificant as the variance increases. In contrast, the square root term is dependent on $\overline{\sigma_t}$ rather than R. Therefore, when $\overline{\sigma_t}$ is much smaller than R, the empirical Bernstein bound is much tighter than the Hoeffding bound.

3 Problem Definition and Naive Solutions

3.1 Problem Definition

There is a data collector and n users in the system. Each user maintains a private set of items, which of each is from the domain $V = \{v_1, v_2, ..., v_d\}$ with d distinct items. Let p_i denote the frequency of item v_i, and without loss of generality, we assume that

$$p_1 \geq p_2 \geq \cdots \geq p_d. \quad (3)$$

For the sake of privacy, each user perturbs and reports her set of items to the data collector, by following LDP protocols. Upon receiving the perturbed data from all users, the goals of the data collector are to identify the most frequent k items and to estimate their frequencies.

The first goal can be formulated as a top-k discovery problem, which involves finding a set of most frequent items while preserving a ranking that is close to the real one. Formally, it can be defined as follows.

Definition 3.1. *(Top-k discovery) Given the items set $V = \{v_1, \ldots, v_d\}$ with the true frequency in decreasing order, the top-k problem finds the item set T, where $T \subseteq V$, $|T| = k$ and for any item $v_i \in T$ with frequency p_i and for any item $v_j \in T^{\complement} = V - T$ with frequency p_j, we have:*

$$p_i \geq p_j. \quad (4)$$

In particular, given the k-th most frequent item v_k's frequency f_k, we know that $\forall v_i \in T$, $\exists \theta \geq 0$

$$p_i \geq p_k + \theta. \quad (5)$$

The second goal is to ensure the frequency estimation accuracy of items in T. Specifically, given the top-k frequent item set T, we estimate each p_i for the item $v_i \in T$. This can be defined as a top-k item frequency estimation problem as follows.

Definition 3.2. *(Top-k frequency estimation) The top-k frequency estimation problem involves estimating the frequency p_i of each item $v_i \in T$, where T is a set of top-k frequent items. The frequency p_i is defined as the proportion of v_i's occurrences among all users.*

3.2 Uniform Sampling

Intuitively, each user can encode her set of items using a d bit string $\{0,1\}^d$, where 1 (resp. 0) indicates the existence (resp. non-existence) of an item in her set. Then a bit (0 or 1) can be sampled uniformly at random, perturbed by RR mechanism (as described in Section 2.2), and sent to the data collector. Then the collector estimates the frequency of each item based on the perturbed data from all user. In particular, the expected frequency of item v_i observed by the data collector follows a Bernoulli distribution

$$f_i = p_i\frac{e^{\epsilon}}{e^{\epsilon}+1} + (1-p_i)\frac{1}{e^{\epsilon}+1}. \quad (6)$$

where p_i is the real frequency of item v_i, and ϵ is the privacy budget. The above procedure adopts a **uniform sampling** strategy, where each item is sampled from the domain with equal probability. However, there are several issues that need to be further considered. Firstly, the approach neglects the differences between items. To be more specific, items with smaller frequencies can be sampled less frequently, which would allow for more users to be allocated towards the frequent items. To take this strategy further, in order to gather information on different items, the sampling process should be carried out in multiple rounds and the sampling strategy should be adjusted in a timely manner, thereby achieving optimal results.

4 Adaptive Sampling

In this section, we propose an adaptive sampling algorithm for identifying the top-k items and estimating their frequencies. The inspiration for our adaptive sampling approach comes from the Adaptive Bounds in Practice method presented in the paper [19]. This prior work not only developed a dynamic arm selection strategy, but also derived a bound to effectively allocate the sampling budget. Our approach builds on this by additionally incorporating LDP noise and deriving a better bound. Furthermore, in Section 4.5, we address the challenge of dealing with a large number of arms. Of course, our method is concerned not only with identifying the top-k items, but also with accurately estimating the frequencies of those top-k items. We first present a system overview of the proposed scheme and then describe the implementation details.

4.1 System Overview

Our proposed framework employs an adaptive by-round strategy to overcome the limitations of naive solutions, as shown in Fig. 1. The data collection process begins with a uniform sampling of all items from n_0 users, establishing an initial knowledge base called Initialization. The remaining users are subsequently divided into R partitions, with $n_1, n_2, \ldots, n_R$ users respectively, for R rounds of data uploading. In each round, users in corresponding partition report their perturbed values only once to maintain ϵ-LDP privacy guarantee as defined in [20]. In particular, the sampling strategy in the upcoming round is in light of the information collected in the previous rounds. After completing all rounds, the data collector aggregates the top-k items and their frequencies. In this section, we first consider a simplified case where there involves only one user in each of R round, i.e., $n_1 = n_2 = \cdots = n_R = 1$. This enables us to adjust our sampling strategy in real time.

4.2 Knowledge Initialization

To estimate the frequency of each item, let t_i denote the number of times item v_i is sampled and let $\{x_1^i, \ldots, x_{t_i}^i\}$ denote the reported values. The observed frequency is given by $\hat{f}_i = 1/t_i \sum_{j=1}^{t_i} x_j^i$, which is an unbiased estimation of

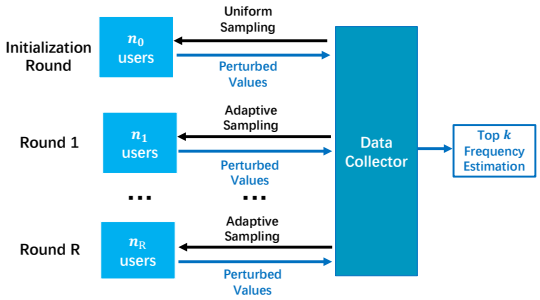


Fig. 1. Illustration of the framework

f_i in Equ. 6. It is unjustifiable to set different t_i without any knowledge on v_i. Therefore, to implement the adaptive sampling scheme, it is crucial to establish some prior knowledge about the items (e.g., item frequency distribution), which can be accomplished in a similar manner as uniform sampling.

Algorithm 1 shows our idea. Note that in uniform sampling, each user randomly selects an item to report, which generally results in each item being chosen approximately the same number of times. However, there still exists some approximation error due to randomness. Algorithm 1 corrects this by letting the data collector directs each user to sample a specific item (and then perturb it), which effectively controls the number of each item being sampled. Consequently, the data collector collects each item a fixed number of times $t = \frac{n_0}{d}$ (line 3), estimates the frequency for all items (line 4) and returns $\hat{f}_i$ as prior knowledge for further use (line 5).

Algorithm 1 Initialization(n_0)

Input: Privacy budget ϵ and initialization parameter n_0
Output: The aggregated frequency $\{\hat{f}_1, \ldots, \hat{f}_d\}$

1: $t = \frac{n_0}{d}$
2: **for** $i = 1$ to d **do**
3: Let t users report v_i by RR, denoted by $\{x_1^i, ..., x_t^i\}$
4: Calculate v_i's frequency as $\hat{f}_i = \sum_{j=1}^{t} x_j^i / (\frac{n_0}{d})$
endfor
5: **return** $\{\hat{f}_1, \ldots, \hat{f}_d\}$

It is critical to determine an appropriate value for n_0 during initialization to ensure accurate results. A small n_0 may result in inaccurate information collected in the initialization round, leading to error accumulation in subsequent rounds. On the contrary, with a limited number of users, an excessively large n_0 may cause too many users to be concentrated in the initialization round, resulting in fewer users learning and utilizing prior knowledge in subsequent rounds. Previous studies have proposed various scales for n_0 [19], [21]; however, these values cannot be directly applied to our scheme. For example, Chen et al. [21] suggest a minimum n_0 of 5 to guarantee proper prior knowledge estimation. In the LDP setting, we cannot establish such a minimum n_0 because when ϵ is closer to 0, the prior knowledge becomes less accurate, and a larger n_0 is required. Therefore, under the constraints of LDP noise and a limited number of users, selecting a suitable n_0 is essential for performance enhancement.

To determine the optimal value of n_0, we first need to quantify the prior knowledge derived from the Algorithm 1. This is done by defining the **initialization error and confidence**.

Definition 4.1. *(Initialization error and confidence) Let e_0 denote the initialization error, and $1 - e_0$ denote the initialization confidence in a top-k discovery problem. For an estimated top-k item set $\hat{T}$, let $\hat{T}^{\complement} = V - \hat{T}$. $\exists e_0 \in (0, 1)$, $\forall \theta > 0$, $v_i \in \hat{T}$, and $v_j \in \hat{T}^{\complement}$, we have $\hat{f}_i \geq f_j - \theta$ with the probability of $1 - e_0$.*

This definition does not take into account the order of the returned items and a larger initialization confidence (i.e., a smaller initialization error) indicates better performance in identifying the top-k items.Inspired by the multi-armed bandit (MAB) literature [19], we obtain the initialization error by applying the union bound and Hoeffding's inequality after sampling all items $\frac{n_0}{d}$ times, as demonstrated in the following theorem:

Theorem 4.1. *$\forall \theta > 0$, the probability of error e_0 of Initialization(n_0) in the top-k discovery satisfies*

$$e_0 \leq d e^{-\frac{\theta^2 n_0}{2d}}. \tag{7}$$

Proof. Recall that T is the top-k item set and items in $T^{\complement}$ are ranked the top-k. $\forall \theta$, according to Equ. 3 and Equ. 5 we can connect T with $T^{\complement}$ as follows:

$$\forall i \in T, \ \forall j \in T^{\complement} : (f_i - f_j > \theta). \tag{8}$$

After initialization, according to Definition 4.1, there exists a $\hat{T} \subset V$ such that $|\hat{T}| = k$, and $\forall i \in \hat{T}$, $\forall j \in (V - \hat{T})$: $(\hat{f}_i \geq \hat{f}_j)$. An item v_j in $T^{\complement}$ can occur in $\hat{T}$, only if there is some items in T such that $\hat{f}_i < \hat{f}_j$. In turn, Equ. 8 implies that the latter event only occurs if $\hat{f}_i \leq f_i - \frac{\theta}{2}$ or $\hat{f}_j \geq f_j + \frac{\theta}{2}$. Using $\frac{\theta}{2}$ instead of θ in multi-armed bandit problems helps control errors and improve estimation accuracy. The goal is to avoid incorrectly identifying non-optimal items as part of the optimal set $\hat{T}$. Splitting θ into $\frac{\theta}{2}$ creates a symmetrical "safety buffer", which helps detect significant deviations and balances false positives and false negatives. This method [22] stems from statistical analysis to ensure theoretical guarantees and algorithm performance

In terms of probabilities, by applying the union bound and Hoeffding's inequality, we obtain:

$$\begin{aligned} &P(\exists v_j \in T^{\complement} : (v_j \in \hat{T})) \\ &\leq \textstyle\sum_{i \in T} P\left(\hat{f}_i \leq f_i - \frac{\theta}{2}\right) + \sum_{j \in T^{\complement}} P\left(\hat{f}_j \geq f_j + \frac{\theta}{2}\right) \\ &\leq |T| \, e^{-\frac{\theta^2 n_0}{2d}} + |T^{\complement}| e^{-\frac{\theta^2 n_0}{2d}} \\ &\leq \left(|T| + |T^{\complement}|\right) e^{-\frac{\theta^2 n_0}{2d}} \\ &\leq d e^{-\frac{\theta^2 n_0}{2d}}. \end{aligned} \tag{9}$$

□

We reduce the selection of n_0 as an optimization problem, with the goal of maximizing the total initialization confidence held by all users. We simplify this problem into two stages: the Initialization round and the subsequent rounds. During the Initialization round, the confidence is zero, and there are n_0 users are collected, and their confidence is $n_0 \dot{0}$. After initialization, the remaining users have an initialized confidence of $1 - de^{-\frac{\theta^2 n_0}{2d}}$, then the total confidence of the remaining users is $(n - n_0)(1 - de^{-\frac{\theta^2 n_0}{2d}})$. Thus, the selection of n_0 aims to maximize the overall initialized confidence I

while considering the value of ϵ. The expression for I is given as follows:

$$\arg\max_{n_0} I = n_0 \cdot 0 + (n - n_0)(1 - de^{-\frac{\theta^2 n_0}{2d}}), \quad (10)$$

where θ is a parameter defined as

$$\theta = \frac{e^\epsilon}{e^\epsilon + 1} - \frac{1}{e^\epsilon + 1} = \frac{e^\epsilon - 1}{e^\epsilon + 1}.$$

Intuitively, as ϵ increases and θ decreases, a larger n_0 is necessary to achieve a satisfactory level of initialization confidence. While obtaining an analytical solution for the transcendental equation presented in Equation 10 is challenging, numerical methods (e.g., Newton's method [23]) can be used to approximate n_0. If $n_0 > n$, it indicates that the initial information is insufficient.

Remark. While an unbiased frequency $\hat{p}_i$ can be further derived from $\hat{f}_i$ (see Equ. 15), we choose $\hat{f}_i$ instead of $\hat{p}_i$ to determine n_0. Intuitively, the larger the item frequency distances are, the less accurate top-k discovery becomes. Therefore, an adaptive sampling scheme should be designed based on the observed item frequency distances. Applying $\hat{p}_i$ on an adaptive sampling scheme is good if data are collected without perturbation. However, when perturbation exists, the data collector obtains $\hat{f}_i$ rather than $\hat{p}_i$. If we still apply $\hat{p}_i$ under LDP, items near the k-th item will be undersampled, which degrades the utility of top-k discovery. Therefore, designing adaptive sampling strategy based on $\hat{f}_i$ is able to achieve better performance.

4.3 Top-k Items Discovery

The uniform sampling scheme samples each item equally and ignores the distribution of items. However, when the items in T and $T^\complement$ are clearly separated, fewer users are needed to identify the top-k items. In such cases, it is more effective to assign more users to report items whose frequencies are near p_k, i.e., the frequency of the k-th most frequent item. In this subsection, we propose an adaptive sampling scheme, called Adaptive-RR Bandit Sampling (**ARBS**), which formalizes the sampling process as an MAB problem.

Design of ARBS. Before starting a new round, we have estimated the frequency of all items, which should ideally be $\hat{f}_1 \geq \dots \hat{f}_d$. However, due to limited statistical counts, we may not obtain an accurate ordering. Thus, we set the order we obtain as $\hat{f_{\xi_1}} \geq \cdots \geq \hat{f_{\xi_i}} \geq \cdots \geq \hat{f_{\xi_d}}$, where $\hat{f_{\xi_i}}$ denotes the i-th most frequent item we estimated. We need to adjust the sampling probability for each item based on this information. To accomplish this, we draw inspiration from literature [19] and invert the concentration inequalities. Specifically, we calculate the error probability δ_i for each item, which reflects the distance between $\hat{f}_i$ and a boundary point μ of $\hat{T}$ and $\hat{T^\complement}$. Intuitively, μ should be drawn from the interval $[\hat{f_{\xi_k}}, \hat{f}_{\xi_{k+1}}]$, where $\hat{f}_{\xi_k}$ can be regarded as the lower bound of item frequency in the set $\hat{T}$, while $\hat{f}_{\xi_{k+1}}$ is the upper bound of the set $\hat{T^\complement}$. The closer $\hat{f}_i$ is to μ, the more users are required to determine whether its frequency is greater than μ.

Choosing an appropriate μ is critical, and a straightforward approach is to take the mean value of $\hat{f_{\xi_k}}$ and $\hat{f}_{\xi_{k+1}}$. However, as an item's frequency approaches μ, more users are required to estimate its frequency accurately. Therefore, we aim to choose μ in $[\hat{f_{\xi_k}}, \hat{f}_{\xi_{k+1}}]$ that maximizes the distance between it and all $\hat{f}_i$'s. To optimize the sampling scheme, we set two different values of μ to maximize the distance between them. The choice of μ is described as follows:

$$\mu = \begin{cases} \hat{f}_{\xi_{k+1}}, & \text{if} \quad \hat{f}_i \leq \hat{f}_{\xi_k}, \\ \hat{f}_{\xi_k} & \text{otherwise.} \end{cases}$$

The choice of μ in such two cases can maximize the distance between μ and $\hat{f}_i$'s and thus optimize the sampling while retaining the relative order among items.

After the initialization round, each item v_i has t_i reports from users, i.e., $\{x_1^i, \cdots, x_{t_i}^i\}$. Thus we can derive an estimated frequency $\hat{f}_i = \frac{1}{t_i}\sum_{j=1}^{t} x_j^i$, and an empirical variance $\overline{\sigma}_i = \sqrt{\frac{1}{t_i}\sum_{j=1}^{t_i}(x_j^i - \hat{f}_i)^2}$ of all reports. We then use Bernstein's inequality, as presented in Equ. 2, to derive a tighter bound than that of Hoeffding's inequality. The resulting formula for δ_i is as follows.

Definition 4.2. *Given a frequency μ, for item v_i that is observed t_i times with f_i, the inverted error δ_i can be obtained using the inverse of Bernstein's inequality (in Equ. 2).*

$$\delta_i = 3e^{(-\frac{\overline{\sigma_i^2}t_i + \overline{\sigma_i}t_i\sqrt{\overline{\sigma_i^2} + 6|\hat{f}_i - \mu|} + 3t_i|\hat{f}_i - \mu|}{9})^2 t_i}, \quad (11)$$

where $\overline{\sigma}_i^2 = \frac{1}{t_i}\sum_{j=1}^{t_i}(x_j^i - \hat{f}_i)^2$ and $\hat{f}_i = \frac{1}{t_i}\sum_{j=1}^{t} x_j^i$.

Proof. In our paper, $R = 1$ and let $H = \sqrt{\frac{\log(3/\delta_i)}{t_i}}$. According to Equation 2, we have $3H^2 + \sqrt{2}\overline{\sigma}_i H + |\hat{f}_i - \mu| = 0$. The solution for H^2 is

$$H^2 = \frac{|\hat{f}_i - \mu|}{3} + \frac{\overline{\sigma}_i\sqrt{\overline{\sigma}_i^2 + 6|\hat{f}_i - \mu|}}{9} + \frac{\overline{\sigma}_i^2}{9}.$$

Then, by eliminating H, δ can be expressed as:

$$\delta = 3e^{-H^2 t_i} = 3e^{(-\frac{|\hat{f}t - \mu|}{3} + \frac{\overline{\sigma}_i\sqrt{\overline{\sigma}_i^2 + 6|\hat{f}t - \mu|}}{9} + \frac{\overline{\sigma}_i^2}{9})t_i}.$$

□

The parameter δ_i reflects the error in determining whether f_i is larger or smaller than μ, and items with larger δ are more likely to require additional users. To incorporate δ_i into the sampling scheme, users sample an item v_i by following the probability distribution

$$P(v_i) = \frac{\delta_i}{\sum_{j=1}^{d}\delta_j}. \quad (12)$$

where $i \in \{1, 2, \cdots, d\}$. Equ. 12 normalizes each inverted error δ_i values across all items, ensuring that the sampling probabilities of all item sum up to 1. In each round of sampling, each user samples an item v_i with probability $P(v_i)$, and then update the sampling probability distribution by Equ. 11 and Equ. 12 after the current-round collection.

Algorithm of ARBS. The proposed ARBS scheme is given by Algorithm 2. At the beginning of the algorithm, each item is sampled $\frac{n_0}{d}$ times during Initialization (line 1). Then, the probabilities are updated based on the current knowledge using Equ. 12 (line 2). Next, the algorithm enters a by-round sampling phase, where only one item is sampled in each round (line 3). Specifically, an item v_s is sampled from the item set V with probability $P(v_s)$ (line 4). The estimates are updated using Equ. 11 and the sampling

probabilities are updated using Equ. 12 (lines 5-6). After all the users upload the perturbed data, the estimated top-k items are determined by selecting the items with the highest k frequencies (line 7).

Algorithm 2 Adaptive-RR Bandit Sampling (ARBS)

Input: Privacy budget ϵ and initialization parameter n_0
Output: Top-k frequent item set $\hat{T}$

1: Initialization(n_0);
2: Update $\{P(v_1),,...,P(v_d)\}$ according to Equ. 12;
3: **for** $k = n_0 + 1$ to n **do**
4: Sample an item v_s from $V = \{v_1, ..., v_d\}$ following probability distribution $\{P(v_1), ...P(v_d)\}$;
5: Update δ_s according to Equ. 11;
6: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 12;
7: Find $\hat{T} \subset V$ such that $|\hat{T}| = m$, and $\forall v_i \in \hat{T}, \forall v_j \in (V - \hat{T})$: $(\hat{f}_i \geq \hat{f}_j)$.
endfor
8: **return** $\hat{T}$

Error analysis. To address the limitation of the error bounds estimated by Equ. 1 and Equ. 2, which tend to be impractically large due to the limited number of users available for each item, we utilize the p-value of a one-sided two-sample T-test [24] to obtain more accurate error estimates for ARBS. The p-value is the probability of obtaining test results that are at least as extreme as a result observed, assuming the null hypothesis is true. For example, when testing the null hypothesis $f_i \leq f_j$, the p-value obtained from observing the samples v_i and v_j is 0.05, indicating a 0.05 probability of rejecting the null hypothesis. We use $pv_{i,j}$ to denote the p-value of v_i and v_j and present the following theorem based on these p-values.

Theorem 4.2. *Given the null hypothesis, i.e., $\forall v_i$ in $\hat{T}$ and $\forall v_j$ in $\hat{T}^{\complement}$ satisfy*

$$f_i > f_j,$$

and for any pair of items $\{v_i, v_j\}$ with p-value $pv_{i,j}$, the error on the validity of this observation can be written as:

$$\Delta = 1 - \prod_{i=1}^{i=k}(1 - pv_{i,k+1}) \prod_{i=k+1}^{i=d}(1 - pv_{i,k}).$$

Given the perturbed frequency of v_k is f_k, and for any positive value γ, we have $f_i - f_k > \gamma$ for all v_i in $\hat{T}$, and $f_k - f_j > \gamma$ for all v_j not in $\hat{T}$, the upper bound of the error is:

$$\Delta_{worst} = 1 -$$
$$\prod_{i=1}^{i=k}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} \overline{F}^j (1 - (\overline{F}))^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1 - \underline{F})^{j-g})$$
$$\prod_{i=k+1}^{i=d}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1 - f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1 - (\underline{F}))^{j-g}),$$

where $\overline{F} = f_k + \gamma$ and $\underline{F} = f_k - \gamma$.

Proof. As we do not require the k values to be returned in any particular order, we only need to compare each item in $\hat{T}$ with $\hat{f}_{k+1}$ and compare those in $\hat{T}^{\complement}$ with $\hat{f}_k$. Let δ_{pi} denote the doubt on v_i. We have:

$$\delta_{pi} = \begin{cases} pv_{i,k+1}, & \text{if } \hat{p}_i \leq \hat{p}_k, \\ pv_{i,k} & \text{if } \hat{p}_i \geq \hat{p}_{k+1}. \end{cases}$$

And thus, we have the top-k error:

$$\begin{aligned} \Delta &= 1 - \prod(1 - \delta_{pi}) \\ &= 1 - \prod_{i=1}^{i=k}(1 - pv_{i,k+1}) \prod_{i=k+1}^{i=d}(1 - pv_{i,k}). \end{aligned} \tag{13}$$

For the theoretical top-k error bound, we consider the worst-case scenario where all user frequencies are close to f_k. In other words, for any $\gamma > 0$, we have $f_i - f_{k+1} > \gamma$ (for all v_i in $\hat{T}$), and $f_k - f_j > \gamma$ (for all v_j not in $\hat{T}$). Here, we analyze the p-value for values v_k and v_j.

Since γ is very small, the sampling count allocated to each user can be approximated by n/d. To test the null hypothesis $H_0 : f_k > f_j$, we have two independent binomial experiments on these two adjacent values, with: $\hat{f}_k \sim \mathrm{B}(\frac{n}{d}, f_k)$ and $\hat{f}_j \sim \mathrm{B}(\frac{n}{d}, f_j)$.

For each possible value of g, we need to calculate the probability mass function $P(\frac{n}{d}\hat{f}_k - \frac{n}{d}\hat{f}_j = g)$, and sum these to obtain the p-value:

$$\begin{aligned} pv_{k,j} &= P(\frac{n}{d}\hat{f}_k - \frac{n}{d}\hat{f}_j \geq 0 \mid H_0) \\ &= \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1 - f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_j^g (1 - f_j)^{j-g}. \end{aligned}$$

Similarly, we obtain the p-value for $pv_{i,k+1}$:

$$\begin{aligned} pv_{i,k+1} &= P(\frac{n}{d}\hat{f}_i - \frac{n}{d}\hat{f}_{k+1} \geq 0 \mid H_0) \\ &= \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_i^j (1 - f_i)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_{k+1}^g (1 - f_{k+1})^{j-g}. \end{aligned}$$

Substituting $pv_{k,j}$ and $pv_{i,k+1}$ into Equ. 13, we obtain

$$\Delta_{worst} = 1 -$$
$$\prod_{i=1}^{i=k}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_i^j (1 - f_i)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_{k+1}^g (1 - f_{k+1})^{j-g})$$
$$\prod_{i=k+1}^{i=d}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1 - f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_j^g (1 - f_j)^{j-g}).$$

Let $\overline{F}$ denote $f_k + \gamma$ and $\underline{F}$ denote $f_k - \gamma$. We use f_k solely to represent the error, which can be rewritten as:

$$\Delta_{worst} = 1 -$$
$$\prod_{i=1}^{i=k}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} \overline{F}^j (1 - (\overline{F}))^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1 - \underline{F})^{j-g})$$
$$\prod_{i=k+1}^{i=d}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1 - f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1 - (\underline{F}))^{j-g}).$$

□

4.4 Frequency Estimation of Top-k Item Set

Besides the top-k discovery, we further present how to estimate their frequencies. While ARBS performs well in top-k discovery, few users are assigned to items with high frequencies, which affects the accuracy of frequency estimation. To address this issue, we present Adaptive-RR Bandit Sampling with frequency (**ARBSF**) in this subsection that

can achieve accurate frequency estimation of $\hat{T}$ while also maintaining effective top-k discovery performance.

Designing of ARBSF. We continue to use the ARBS algorithm to identify the top-k items, but with a modification that involves allocating some users for estimating the frequencies of items. We treat the items in sets $\hat{T}$ and $\hat{T^{\complement}}$ separately. For an item $v_j \in \hat{T^{\complement}}$, we compute its δ_j as in the ARBS algorithm. For items $v_i \in \hat{T}$, we compute δ_i to achieve the same variance $\overline{\sigma_0}$. Specifically, we firstly calculate the variance based on the distribution information for items in $\hat{T}$, and select the minimum variance as $\overline{\sigma_0}$. We set the δ_i values for all items in $\hat{T}$ to achieve $\overline{\sigma_0}$ based on Equ. 2 as

$$\begin{aligned}\overline{\sigma_0} &= \frac{1}{t_i}|f_i - \hat{f}_i|^2 \\ &\leq \frac{1}{t_i}(\overline{\sigma_i}\sqrt{\frac{2log(3/\delta_i)}{t_i}} + \frac{3log(3/\delta_i)}{t_i})^2.\end{aligned} \tag{14}$$

Then based on all $\delta_i (1 \leq i \leq d)$, we can obtain the sampling probability for v_i according to Equ. 12. This modification improves the accuracy of ARBS in estimating the frequency of top-k items by allocating more users for the estimation.

Calibration and privacy analysis. The data collector receives perturbed data from all users, and counts bit "1"s among the t_i reports for item v_i. Then the frequencies of items in $\hat{T}$, denoted by $\{\hat{p}_1, \ldots, \hat{p}_k\}$ can be derived by Equ. 15 for noise calibration. Theorem 4.3 proves that $\hat{p_i}$ is an unbiased frequency estimation for these items.

$$\hat{p}_i = \frac{\hat{f}_i - (1 - \frac{e^\epsilon}{e^\epsilon+1})}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)}. \tag{15}$$

Theorem 4.3. *$\hat{p_i}$ in Equ. 15 is an unbiased estimator of the true frequency of the v_i.*

Proof.

$$\begin{aligned}E[\hat{p_i}] &= E[\frac{\hat{f}_i - (1 - \frac{e^\epsilon}{e^\epsilon+1})}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)}] \\ &= \frac{(p_i\frac{e^\epsilon}{e^\epsilon+1} + (1-p_i)(1 - \frac{e^\epsilon}{e^\epsilon+1})) - (1 - \frac{e^\epsilon}{e^\epsilon+1})}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)} \\ &= \frac{p_i\frac{e^\epsilon}{e^\epsilon+1} + 1 - p_i + p_i\frac{e^\epsilon}{e^\epsilon+1} - \frac{e^\epsilon}{e^\epsilon+1} - 1 + \frac{e^\epsilon}{e^\epsilon+1}}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)} = p_i.\end{aligned}$$

□

All sampling methods satisfy the ϵ-LDP, if each user samples only one value with a privacy budget of ϵ.

Theorem 4.4. *For sampling mechanism $\mathcal{M}$, if for each user is sampled if and only one value with single value with privacy budget ϵ, the sampling algorithm $\mathcal{M}$ satisfy ϵ-LDP.*

Proof. For user i, the j-th set value is collected with privacy budget ϵ, and thus the data collector cannot obtain any valid information from the uncollected values, which is equivalent to having a privacy budget of 0. We have:

$$\prod_{i=1}^{d-1} e^0 e^{-\epsilon} \leq \frac{P[\mathcal{M}(x_1, \ldots, x_d) = Y]}{P[\mathcal{M}(x'_1, \ldots, x'd) = Y]} \leq e^\epsilon \prod_{i=1}^{d-1} e^0.$$

Therefore, we can conclude:

$$e^{-\epsilon} \leq \frac{P[\mathcal{M}(x_1, \ldots, x_d) = Y]}{P[\mathcal{M}(x'_1, \ldots, x'_d) = Y]} \leq e^\epsilon.$$

Thus, $\mathcal{M}$ satisfies ϵ-local differential privacy. □

Algorithm of ARBSF. Algorithm 3 presents the ARBSF algorithm, which shares similarities with ARBS, but with three notable distinctions. Initially, in lines 5-6, we determine $\overline{\sigma_t}$ and subsequently adjust δ_i to allocate more users to the top-k items. Secondly, in line 9, it is necessary to estimate $\hat{p_i}$ for items in $\hat{T}$. In the end, the algorithm at line 10 returns the items in $\hat{T}$, and their corresponding frequencies $\hat{p_i}$.

Algorithm 3 Adaptive-RR Bandit Sampling with frequency(ARBSF)

Input: Privacy budget ϵ and initialization parameter n_0
Output: Top-k frequent item set $\hat{T}$, frequency estimation $\{\hat{p_1}, \ldots, \hat{p_k}\}$

1: Initialization(n_0);
2: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 12;
3: **for** $k = n_0 + 1$ to n **do**
4: Sample an item v_s from $V = \{v_1, ..., v_d\}$ following probability distribution $\{P(v_1), ...P(v_d)\}$;
5: Determine the minimum variance $\overline{\sigma_0}$ of items in $\hat{T}$;
6: **for** $k = n_0 + 1$ to n **do**
7: **if** $\hat{f}_i < \hat{f}_j$ **then**
8: Update δ_s according to Equ. 11;
9: **else**
10: Update δ_s according to Equ. 14;
endfor
11: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 12;
12: Find $\hat{T} \subset V$ such that $|\hat{T}| = m$, and $\forall v_i \in \hat{T}, \forall v_j \in (V - \hat{T}) : (\hat{f}_i \geq \hat{f}_j)$.
endfor
13: Calculate $\hat{p_i}$ for $i \in \{1, \ldots, k\}$ according to Equ. 15;
14: **return** $\hat{T}, \{\hat{p_1}, \ldots, \hat{p_k}\}$

4.5 Large-Scale Solution

When dealing with a large domain d (e.g., tens of thousands of items), the data collector first needs to focus on a narrower candidate set value range. This is essential as sampling methods cannot accurately capture valid information from the entire set value range. To achieve this, users are evenly divided into two groups for efficient processing. The first group is tasked with pruning the large domain into a smaller range, while the second group performs adaptive sampling methods within this reduced range. Specifically, each user in the first group selects a value randomly from their private value set and perturbs it using a perturbation mechanism such as OUE, OLH, etc. This process identifies the 10k most frequent items from d.

While this process introduces a bias in frequency estimation, at this initial phase, our primary goal is only to roughly identify the top $10k$ items without focusing too much on precise frequency estimations. Additionally, this procedure satisfies ϵ-LDP, which is proven in Theorem 4.4. It is important to note that this phase is unnecessary if the original domain size is close to or fewer than $10k$ items. In our experiments, we utilized OUE, and then reported the results.

4.6 Which one to use

Both ARBS and ARBSF are capable of finding the top-k items, but the question is which one performs better. This is greatly influenced by the relationship between the value domain d and the target k. ARBS has the highest allocated users around the k-th item, with gradual declines on both sides (for items with higher or lower frequencies), while ARBSF allocates more users in the collection of top-k items, and then decreases the number of allocated users as the item frequency drops. If the domain size d is much larger than k, ARBS has to allocate more users to the large number of non-top-k items. This reduces the number of users ARBS can allocate around the top-k items, compared to ARBSF. As a result, ARBS's performance on top-k retrieval may not be as good as ARBSF.

When the difference between the domain size d and the target k is relatively small, we recommend using ARBS to find the top-k items. When the difference between d and k is large, we recommend using ARBSF to find the top-k items. As for estimating the frequency of the top-k items, we recommend using ARBSF.

5 Delay-Constrained Solution

In traditional MAB problems, the interaction between the gambler and the arms is relatively straightforward and short-lived. However, in the LDP setting, the interaction between users and the data collector can be much more prolonged. This means that, unlike traditional MAB problems, the multiple rounds of communication between users and the data collector can be time-consuming, which becomes a new challenge for both Adaptive Recommender Bandit Selection (ARBS) and Adaptive Recommender Bandit Selection with Feedback (ARBSF). On the other hand, in each round, a single user may only contribute limited information. To address these challenges, in this section, we propose a batch processing method, which can help reduce the system delay. By processing user contributions in batches, we can mitigate the issues caused by the prolonged interaction between users and the data collector in the LDP setting.

5.1 Delay-Constrained Solution

To reduce communication overhead between users and the data collector, an intuitive solution is to let a batch of users send reports in a round, thus reducing the total rounds needed. However, simply allocating the same number of users in each round is not an optimal solution, as the impact of the data collected in earlier rounds is greater than that of the later rounds. A more effective sampling scheme is to allocate fewer users in earlier rounds to allow for timely adjustments, while ensuring sufficiently accurate information collected from these rounds. Striking a balance between these trade-offs is critical in developing an optimal sampling scheme. In this section, we propose the Delay-contrained Batch Sampling algorithm (**DBS**), which models and derives an optimal user allocation scheme when the number of rounds is fixed to R. RBS guarantees both high accuracy of top-k discovery and low communication overhead by dynamically allocating users based on each user's potential to contribute to the discovery of top-k items.

User allocation in each round. Let n_i denote the number of users in round i and N_r denote the number of remaining users after $r-1$ rounds, which can be written as:

$$N_r = n - n_0 - \sum\nolimits_{i=1}^{r-1} n_i.$$

As shown in Equ. 11, δ_i represents the inverted error for v_i. Let e_{r-1}^I denote the cumulative error obtained from δ_i after the $r-1$ rounds:

$$e_{r-1}^I = \sum\nolimits_{i=1}^{d} \delta_i. \tag{16}$$

An ideal user allocation scheme is to minimize the overall error over all rounds, by dynamically assigning a batch of users to each round. However, this is not feasible because this information can only be obtained after each round of data collection. Therefore, we apply a greedy search to determine the optimal number of next-round users based on the current-round information. Specifically, we combine the r-th round and the $(r+1)$-th round and split them into two virtual phases to simulate the upcoming sampling process. We collect n_r^I users in phase I to simulate the occurrence of round r and n_r^{II} users in phase II to simulate the occurrence of round $r+1$. The total number of users for these two virtual phases is denoted by $n_{(r,r+1)}$, and we have:

$$n_{(r,r+1)} = n_r^I + n_r^{II}. \tag{17}$$

Let t_i represent the number of reports from users for v_i before the virtual phase I, and it changes to t_i' after the virtual phase I and can be calculated as

$$t_i' = t_i + n_r^I P(v_i),$$

where $P(v_i)$ can be calculated by Equ. 12.

Since the virtual process is not actually implemented, we are unable to obtain the variance of each item in virtual phase I. Therefore, we process the error after virtual phase I by using the inverse of Hoeffding's inequality instead of Bernstein's inequality, as:

$$\delta_i' = \frac{2}{exp(2t_i'(\hat{f}_i - \mu)^2)}, \tag{18}$$

where $\hat{f}_i$ and μ are set based on the latest real knowledge obtained in round $r-1$. Similar to Equ. 16, we can simulate the cumulative error e_{r-1}^{II} after virtual phase I based on δ_i'. Specifically,

$$e_{r-1}^{II} = \sum\nolimits_{i=1}^{d} \delta_i'.$$

Theorem 5.1. *e_{r-1}^{II} is a theoretical lower bound, such that all items simultaneously satisfy $|\hat{f}_i - \mu| \leq B_i$, where $B_i = \sqrt{\frac{2\log(2/\delta_i')}{t_i'}}$.*

Proof. The error bound reflects our overall requirement for all items to simultaneously satisfy the condition $|\hat{f}_i - \mu| \leq B_i$, so we have:

$P(|\hat{f}_1 - \mu| \leq B_1, |\hat{f}_2 - \mu| \leq B_2, ..., |\hat{f}_d - \mu| \leq B_d) \geq 1 - \delta$,

where δ is the overall error tolerance, which represents our required confidence level for the joint probability. And $B_i = \sqrt{\frac{2\log(2/\delta_i')}{t_i'}}$.

According to the Boole inequality, the joint probability is bounded from below by one minus the sum of the individual probabilities. Specifically, we have:

$$P(|\hat{f}_1 - \mu| \leq B_1, |\hat{f}_2 - \mu| \leq B_2, ..., |\hat{f}d - \mu| \leq B_d) \geq 1 - \sum_{i=1}^{d} \delta_i'.$$

□

To find the optimal number of users allocated to the current round, we formalize the problem as minimizing the cumulative user error, i.e., the sum of cumulative errors for certain users, in two virtual phases. Specifically, in phase I, n_r^I users' data are collected with cumulative error e_{r-1}^I, while n_r^{II} users' data are collected with cumulative error e_{r-1}^{II} in phase II. We can express the cumulative user error for n_r^I and n_r^{II} users as:

$$\begin{aligned} &\arg\min_{n_r^I} n_r^I e_1^{r-1} + n_r^{II} e_2^{r-1}, \\ \text{s.t.} \quad &sup(n_{(r,r+1)}) = \frac{2N_r}{R-(r-1)}. \end{aligned} \tag{19}$$

Here, $sup(n_{(r,r+1)})$ denotes the upper bound on the combined number of users across the two virtual phases. This condition can be inferred from the following equation according to $n_r < n_{r+1}, r \in \{1, \ldots, R-1\}$ established in Theorem 5.2:

$$\begin{aligned} n_{(r,r+1)} &\leq n_r + n_{r+1} \\ &\leq \frac{N_r}{R-(r-1)} + \frac{N_r - \frac{N_r}{R-(r-1)}}{R-(r-1)} = \frac{2N_r}{R-(r-1)}. \end{aligned}$$

According to Equ. 19, we obtain a numerical solution for n_r^1, which represents the number of users allocated to round r. The number of users allocated to round $r+1$ can be estimated using either n_{r+1}^1 from phase I results or n_r^2 from phase II results in round r. We select n_{r+1}^1 as it has aggregated more information. Formally, we have:

$$n_r = n_r^1. \tag{20}$$

Algorithm of DBS. The batchwise procedure, as presented in Algorithm 4, can be summarized as follows. Specifically, when a new round r commences, the number of users n_r allocated to the current round is determined (line 4). Subsequently, n_r users are collected based on the current $P(v_i)$ (lines 5-6). The frequency estimation $\hat{f}_i$, the inverted error, and sampling probability $P(v_i)$ are independently updated at the end of each round in both ARBS and ARBSF (lines 7-9).

Algorithm 4 Delay-contrained Batch Sampling (DBS)

Input: Privacy budget ϵ, initialization parameter n_0 and number of round R
Output: Top-k frequent item set $\hat{T}$, frequency estimation $\{\hat{p_1}, \ldots, \hat{p_k}\}$

1: Initialization(n_0);
2: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 12;
3: **for** $r = 1$ to R **do**
4: Calculate n_r according to Equ. 19 and 20;
5: **for** $i = 1$ to n_r **do**
6: Sample an item v_s from $V = \{v_1, ..., v_d\}$ following probability distribution $\{P(v_1), ...P(v_d)\}$;
endfor
7: **ARBS**: Update δ_s according to Equ. 11;
8: **ARBSF**: Determine the minimum variance $\overline{\sigma_0}$ of the items in $\hat{T}$; Update δ_s according to Equ. 11 or Equ. 14;
9: Update $\{P(v_1), ...P(v_d)\}$ according to Equ. 12;
10: Find $\hat{T} \subset V$ such that $|\hat{T}| = m$, and $\forall v_i \in \hat{T}\ \forall v_j \in (V - \hat{T}) : (\hat{f}_i \geq \hat{f}_j)$.
endfor
11: Calculate $\hat{p_i}$ for $i \in \{1, \ldots, d\}$ according to Equ. 15.
12: **return** **ARBS**: $\hat{T}$;
13: **ARBSF**: $\hat{T}, \{\hat{p_1}, \ldots, \hat{p_k}\}$

Monotonicity analysis. In DBS, n_r progressively increases with each round. This growth pattern results in a smaller cumulative user error for all users compared to other sampling orders, as demonstrated in the following theorem:

Theorem 5.2. *Given a set of numbers of sampled users in each round $n_1, ..., n_R$, the ordering that satisfies the inequality*

$$n_1 < \cdots < n_r < \cdots < n_R.$$

guarantees an optimal cumulative user error for all users.

Proof. With each round, the number of collected users increases, which leads to a continuous decrease in the error. Recall e_1^{r-1} denotes the cumulative error for all items after $r-1$ rounds (same as Equ. 16). This establishes the following inequality:

$$e_1^{r-1} > e_1^r, r \in \{1, \ldots, R\}.$$

In a similar manner to Equ. 19, which defines the cumulative user error for all users, we can propose the following equation:

$$M = n_1 e_1^0 + n_2 e_1^1 + ... + n_{R-1} e_1^{R-1}. \tag{21}$$

For every choice of real numbers $n_1 < n_2 < \cdots < n_R$ and $n_{\xi_1}, n_{\xi_2}, \ldots, n_{\xi_R}$ is a permutation of $n_1, ..., n_R$, the rearrangement inequality [25] leads us to the following inequality:

$$\begin{aligned} n_R e_1^0 + n_{R-1} e_1^1 + ... + n_1 e_1^{R-1} > \\ n_{\xi_1} e_1^0 + n_{\xi_2} e_1^1 + \cdots + n_{\xi_h} e_1^R > \\ n_1 e_1^0 + n_2 e_1^1 + \cdots + n_R e_1^{R-1}. \end{aligned}$$

Therefore, we deduce that in order to achieve the minimal error for Equ. 21, the sampling sequence should satisfy

$$n_1 < \cdots < n_r < \cdots < n_R.$$

□

6 Experimental Results

6.1 Experimental Setup

We conduct evaluations of the proposed schemes, ARBS and ARBSF, on both synthetic and real-world datasets to validate their accuracy. For comparison purpose, we also show the results of uniform sampling (i.e., each user randomly selects and perturbs one item), SVIM (based on either GRR or OLH) [13] and Mwheel (designed for multiset frequency estimation) [26]. The experiments are conducted on a PC equipped with an AMD Ryzen 7 2700X eight-core processor, 64GB RAM, and Windows 10, using MATLAB R2019b and Python 3.10. All datasets and code are available online[1].

6.1.1 Utility Metrics

We apply normalized cumulative rank (NCR) and mean square error (MSE) to measure the accuracy of top-k item discovery and frequency estimation, respectively.

- NCR [13] is proposed to reveal the true rank information of the top-k values. For each item v_i, we first calculate its ranked value V_i as

$$V_i = \frac{q(v_i)|v_i \in T}{q(v_i)|v_i \in \hat{T}} = \frac{q(v_i)|v_i \in T}{k(k+1)/2}.$$

1. https://github.com/RONGDUGithub/ARBSF

TABLE 1
Statistical characteristics of the dataset

Dataset	n	Domain	Max count	Min count	Average Count
Linear	10000	100	5	28	15
Beta	10000	100	2	27	12.2
Gamma	100000	10000	436	641	612.2
Laplace	100000	9883	29	92	59.9
$Word_{google}$	10000	26	1	8	5.6
$World_{mit}$	10000	26	1	7	5.6
Retail	540455	2602	1	7	4.4
Kosarak	990002	41270	1	2498	8.1

where $q(\cdot)$ is a quality function defined as

$$q(v_i) = \begin{cases} k-i+1, & \text{if} \quad v_i \in T, \\ 0, & \text{if} \quad v_i \in \hat{T}. \end{cases}$$

Then NCR is calculated based on the ranked values across all items. Formally,

$$NCR = \frac{\sum_{v_i \in \hat{T}} V_i}{\sum_{v_j \in T} V_j}.$$

- MSE [27] measures the frequency estimation accuracy in terms of squared errors as

$$MSE = \frac{1}{k} \sum\nolimits_{v_i \in T} (p_i - \hat{p_i})^2,$$

where the estimated frequencies $\hat{p_i}$ are set to 0 for items that are not successfully identified by the protocol.

6.1.2 Datasets

The experiments are conducted over four synthetic datasets and four real datasets. The dataset statistics are detailed in TABLE 1. The last three columns are the minimum, maximum and average numbers of items possessed by each user.

Synthetic Datasets. We generate two synthetic datasets with 10,000 users and 100 items. Users' item ownership follows different distributions: **Linear** and **Beta**. Additionally, we generate two large-scale datasets with 100,000 users and 10,000 items, where users' item ownership follows **Gamma** and **Laplace** distributions, respectively.

Real Datasets. We use four publicly available set-valued datasets as follows. **Word**$_{google}$[2] contains the 10,000 most frequent English words sorted by frequency, and we estimated the letter frequency among these words. **Word**$_{mit}$ is similar to **Word**$_{google}$ and contains 10,000 words from 26 letters. **Retail**[3] contains all the transactions occurring between 2010 and 2011 for a UK-based and registered non-store online retail, including merchant transactions for half a million users in 2,603 categories. **Kosarak**[4] contains click streams on a Hungarian website that contains around one million users and 42 thousand categories.

6.2 Overall Results on Synthetic Datasets

In this subsection, we study the impact of ϵ and k as well as the dataset sparsity parameter s on a synthetic dataset. To adjust the dataset's sparsity, we multiply the number of user-owned items in the original dataset by our parameter s. The smaller the value of s is, the sparser the dataset becomes. To evaluate the performance on large-scale datasets, we first pruned the domain, and the specific process can be referred to in Section 4.5. The bar charts represent the NCR results, while the line plots represent the MSE results.

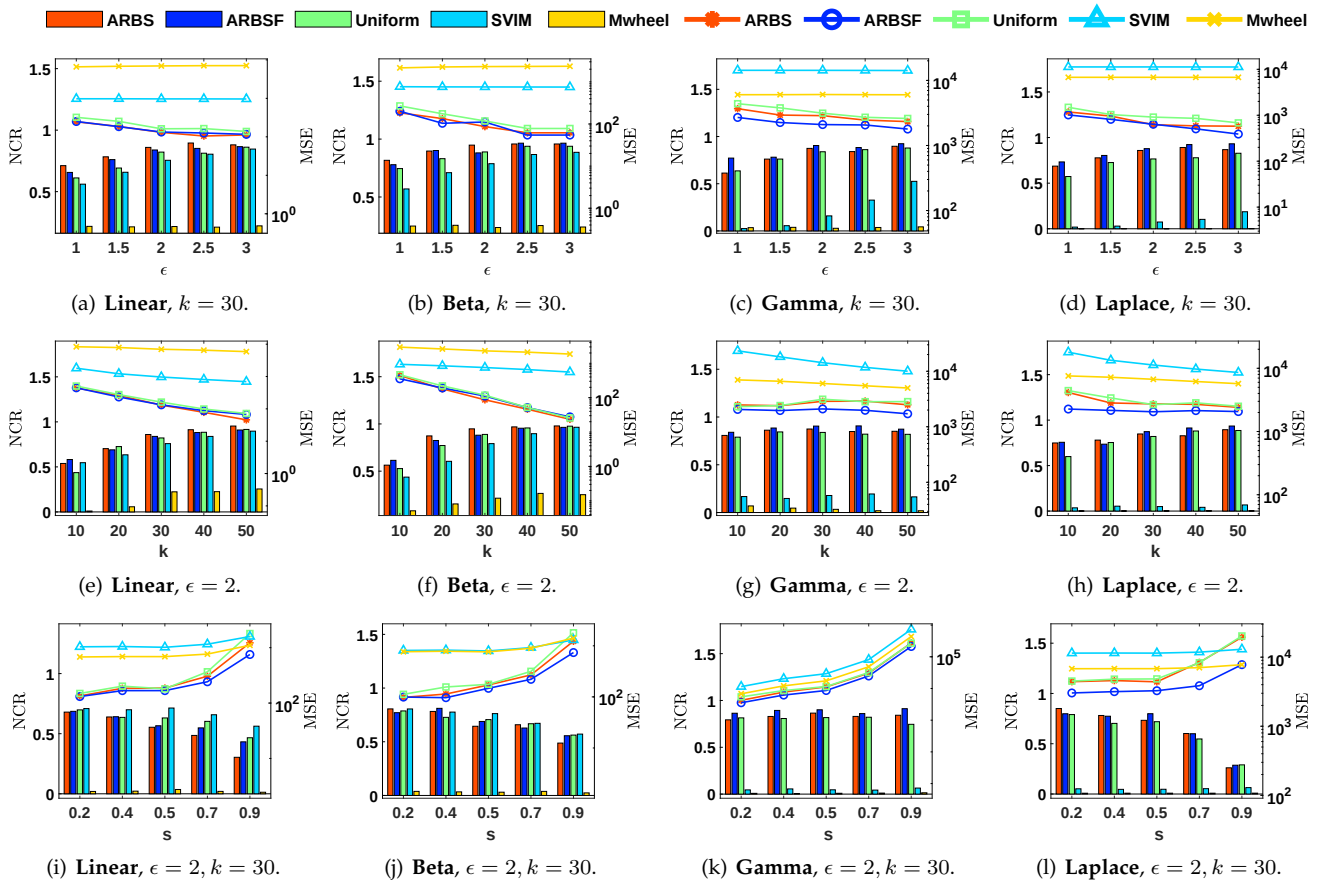
The impact of ϵ. Fig. 2 (a)-(d) illustrates the performance of various algorithms in terms of NCR and MSE by varying ϵ, with k fixed at 30 for all datasets. The histograms show the NCR for top-k discovery, while the curves indicate the MSE for frequency estimation. The results indicate that a larger ϵ leads to a higher NCR and a lower MSE across all five algorithms. Although SVIM and Mwheel do not appear to decrease with changes in ϵ, they are indeed decreasing in reality. Specifically, SVIM's mean squared error (MSE) decreases by 0.796% to 1.932% when comparing the results with $\epsilon = 1$ and $\epsilon = 3$. Similarly, Mwheel's MSE decreases by 0.056% to 0.225% over the same range of ϵ values. The amount of change is related to the parameter settings, such as k. In most cases, ARBS and ARBSF outperform SVIM, Mwheel and uniform sampling, achieving much higher NCR and lower MSE. This is because SVIM introduces a large error when the dataset is not sparse, leading to poor performance. Furthermore, the adaptive sampling schemes of ARBS and ARBSF perform better than uniform sampling by constantly learning and applying new knowledge to update their sampling schemes.We can observe that Mwheel performs poorly across all datasets. This is because Mwheel utilizes principles similar to PEM [28] to find the corresponding top-k data, and PEM tends to perform poorly for small ranges of d. Moreover, this scheme is designed to handle multi-value sets, where the same set may contain duplicate data, and it may perform better in such scenarios. Although our set-value is a special case of multi-value sets, it cannot leverage the advantages of Mwheel.

In terms of NCR, ARBS performs better than ARBSF on **Linear** and **Beta**, while ARBSF performs better on **Gamma** and **Laplace**. This is because in **Gamma** and **Laplace**, despite domain pruning, the d value remains relatively large compared to k, causing ARBS to sample a large number of users on non-top-k items, resulting in poorer performance. As for MSE, in most cases, ARBSF exhibits better performance. Overall, these results suggest that adaptive sampling algorithms such as ARBS and ARBSF can improve the utility of differentially private data analysis tasks, even for large-scale datasets.

In addition to NCR and MSE, we also employ another metric for measuring accuracy: the hit rate [29], as shown in TABLE 3. The hit rate is used to describe the proportion of correctly identified or classified instances. Unlike NCR, which focuses on the ranking of accurate values, the hit rate is concerned with the quantity of accurate values. From the data presented in the table, it is evident that our adaptive sampling methods achieve a higher hit rate. ARBSF outperforms ARBS on **Gamma** and **Laplace** with $d = 10k$, and ARBS performs better on **Linear** and **Beta**, and its performance is generally consistent with NCR. Moreover, the larger the value of ϵ, the higher the hit rate.

The impact of k. To investigate the impact of k on the performance of different algorithms, we conduct an exper-

2. https://github.com/first20hours/google-10000-english
3. https://archive.ics.uci.edu/dataset/352/online+retail
4. https://github.com/cpearce/HARM/blob/master/datasets/kosarak.dat



(a) **Linear**, $k = 30$. (b) **Beta**, $k = 30$. (c) **Gamma**, $k = 30$. (d) **Laplace**, $k = 30$.

(e) **Linear**, $\epsilon = 2$. (f) **Beta**, $\epsilon = 2$. (g) **Gamma**, $\epsilon = 2$. (h) **Laplace**, $\epsilon = 2$.

(i) **Linear**, $\epsilon = 2, k = 30$. (j) **Beta**, $\epsilon = 2, k = 30$. (k) **Gamma**, $\epsilon = 2, k = 30$. (l) **Laplace**, $\epsilon = 2, k = 30$.

Fig. 2. The results of NCR & MSE w.r.t. ϵ, k and s on synthetic datasets

TABLE 2
Hit rate of real datasets

Dataset	Scheme	$\epsilon = 1$	$\epsilon = 1.5$	$\epsilon = 2$	$\epsilon = 2.5$	$\epsilon = 3$
Linear	ARBS	0.68	0.74	0.8	0.83	0.82
	ARBSF	0.65	0.72	0.79	0.79	0.81
	Uniform	0.58	0.63	0.76	0.74	0.78
	SVIM	0.51	0.60	0.7	0.73	0.78
	Mwheel	0.23	0.23	0.22	0.21	0.22
Beta	ARBS	0.77	0.82	0.87	0.91	0.91
	ARBSF	0.73	0.85	0.84	0.91	0.91
	Uniform	0.71	0.77	0.85	0.89	0.89
	SVIM	0.55	0.67	0.76	0.83	0.85
	Mwheel	0.25	0.24	0.24	0.25	0.25
Gamma	ARBS	0.54	0.67	0.75	0.83	0.77
	ARBSF	0.66	0.80	0.83	0.88	0.81
	Uniform	0.58	0.62	0.68	0.78	0.75
	SVIM	0.02	0.05	0.12	0.25	0.41
	Mwheel	0.07	0.07	0.05	0.06	0.07
Laplace	ARBS	0.62	0.67	0.78	0.79	0.80
	ARBSF	0.67	0.75	0.77	0.82	0.86
	Uniform	0.53	0.67	0.69	0.73	0.76
	SVIM	0.02	0.03	0.05	0.08	0.14
	Mwheel	0.001	0.001	0.002	0.002	0.003

iment where we vary k while keeping ϵ fixed at 2. Fig. 2 (e)-(h) illustrates that all algorithms achieve better MSE as k increases, since more low-frequency items are estimated and the distance between the estimated frequency and the actual frequency becomes smaller. The performance of the NCR algorithm is heavily influenced by different datasets. For the **Beta** dataset, where the frequency difference between items is constant, the difficulty of distinguishing the items remains the same, and the NCR for ARBS and ARBFS does not change significantly with respect to k.

The impact of s. Fig.2 (i)-(l) demonstrates the impact of dataset sparsity s from 0.2 to 0.9. Here, as s increases, the data becomes more sparse. When $s = 0.2$, 20% of the data in the original dataset is removed, resulting in sparsity. As expected, as s increases, the utility (measured by both NCR and MSE) decreases, indicating that all algorithms perform worse when the dataset is sparse. However, since both the original data and estimated amounts decrease, leading to a smaller interpolation, the MSE might decrease. To enable a fair comparison of MSE across different values of s, we propose a normalization of MSE.

Specifically, assume that the original frequency of a dataset is f, and the estimated frequency is $\hat{f}$. Given s, the expected estimate should be $(1-s)f$, and if the actual estimate is denoted by $(1-s)\hat{f}$, the difference in the MSE expressions before and after normalization is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}((\hat{f}_i) - f_i)^2 = \frac{1}{(1-s)^2}\frac{(1-s)^2}{n}\sum_{i=1}^{n}((\hat{f}_i) - f_i)^2$$
$$= \frac{1}{(1-s)^2}\frac{1}{n}\sum_{i=1}^{n}(((1-s)\hat{f}_i) - (1-s)f_i)^2 = \frac{1}{(1-s)^2}MSE_s.$$

where MSE_s is the experimental result, and our normalization is to multiply the calculated experimental result MSE_s by $1/(1-s^2)$.

The experimental results show that as the data becomes sparser, NCR decreases, and MSE increases. This is because a sparse dataset has fewer available data points for

each user, making it more difficult to estimate the user's preference accurately. In most cases, our proposed schemes outperform the others under different s values. Although SVIM shows better NCR performance on the **linear**, its MSE is still worse than that of our methods.

6.3 Overall Results on Real Datasets

In this subsection, we conduct experiments on the estimation of top-k items using four real datasets with varying ϵ and k. The bar charts represent the NCR results, while the line plots represent the MSE results. To evaluate the performance on **Retail** and **Kosarak**, we first prune the domain, and the specific process can be referred to in Section 4.5.

The impact of ϵ. Fig. 3 depicts the accuracy of frequency estimation with respect to ϵ. Consistent with the synthetic data, we observe an improvement in utility as ϵ increases. We can observe that for our dataset, Mwheel performs the worst. Mwheel's set domain length is an exponential of 2, which is different from the domain length of our data. After converting the data into binary, values with the same lower bits may be recovered simultaneously, leading to inaccurate results. Furthermore, our dataset does not have repeated values within a single set, so Mwheel cannot take advantage of its strengths. Even in Figures 3 (c)-(d), its NCR values are close to 0. Additionally, we notice that for small-scale datasets, the query NCR results of other methods are similar. However, for the large-scale **Retail** and **Kosarak** dataset, our adaptive schemes generally perform better than other approaches. This proves the effectiveness of our handling of large-scale data in Section 4.5.

Regarding MSE, we can observe that ARBSF performs optimally in most cases, as this method allocates a portion of the sampling budget to the discovered top-k values, thus outperforming ARBS. Regarding NCR, ARBS performs better on the $word_{google}$ and $word_{mit}$, while ARBSF performs better on the **Retail** and **Kosarak**, which is related to the values of d and k. Furthermore, we can see that the sampling-based schemes achieve better MSE than SVIM because our methods guarantee unbiasedness, resulting in more accurate frequency estimates.

The impact of k. In Fig. 4, we present the NCR and MSE values by varying k, and our adaptive algorithms demonstrate superior performance in most cases. The MSE consistently decreases as k increases, while the NCR performance is influenced by the characteristics of the dataset. In particular, for **Word**$_{google}$ and **Word**$_{mit}$, NCR improves as k increases. However, for **Retail** and **Kosarak**, NCR worsens as k increases. This discrepancy arises because the first two datasets have relatively uniform data frequency distributions. In contrast, for the latter two datasets, as k increases, the density of data points around k also increases, making it more challenging to accurately distinguish the true top-k values, thereby leading to a decline in NCR performance.

In Fig. 4 (c), when $k = 10, 40, 50$, we observe that SVIM achieves the best NCR performance. However, as k decreases, our schemes outperform SVIM. This indicates that while SVIM is not as proficient as our proposed schemes in estimating the counts for extremely high-frequency values.

TABLE 3
Hit rate of synthetic datasets

Dataset	Scheme	k 3(10)	k 6(20)	k 9(30)	k 12(40)	k 15(50)
$Word_{google}$	ARBS	0.73	0.87	0.937	0.95	0.91
	ARBSF	0.80	0.83	0.96	0.92	0.89
	Uniform	0.73	0.83	0.96	0.92	0.91
	PFSO	0.66	0.79	0.95	0.93	0.94
	Mwheel	0.08	0.198	0.26	0.27	0.29
$Word_{mit}$	ARBS	0.67	0.87	0.98	0.90	0.92
	ARBSF	0.73	0.90	1.00	0.90	0.92
	Uniform	0.67	0.87	0.93	0.93	0.89
	PFSO	0.64	0.82	0.96	0.90	0.93
	Mwheel	0.05	0.21	0.23	0.30	0.28
Retail	ARBS	0.86	0.74	0.58	0.545	0.48
	ARBSF	0.94	0.81	0.65	0.6	0.51
	Uniform	0.86	0.75	0.58	0.47	0.47
	PFSO	0.90	0.78	0.67	0.61	0.55
	Mwheel	0.006	0.006	0.008	0.013	0.020
Kosarak	ARBS	0.96	0.84	0.71	0.60	0.52
	ARBSF	0.98	0.87	0.79	0.63	0.56
	Uniform	0.92	0.86	0.65	0.60	0.52
	PFSO	0.89	0.63	0.46	0.37	0.30
	Mwheel	0.015	0.013	0.003	0.009	0.010

Additionally, since SVIM is not an unbiased estimator, its corresponding MSE performance is also inferior to that of our proposed schemes.

We also test the hit rate on real datasets with varying k, and the results are shown in TABLE 4. Consistent with the NCR results, in most cases, our adaptive sampling method yielded superior results. However, the trend of the hit rate with varying k is related to the distribution of the dataset itself. When there are more items with frequencies similar to the k-th item, the accuracy tends to be lower, and vice versa. For the **Retail** and **Kosarak** datasets, as k increases, the frequency of items near the k-th item is lower but there are more such items, resulting in a lower hit rate accuracy.

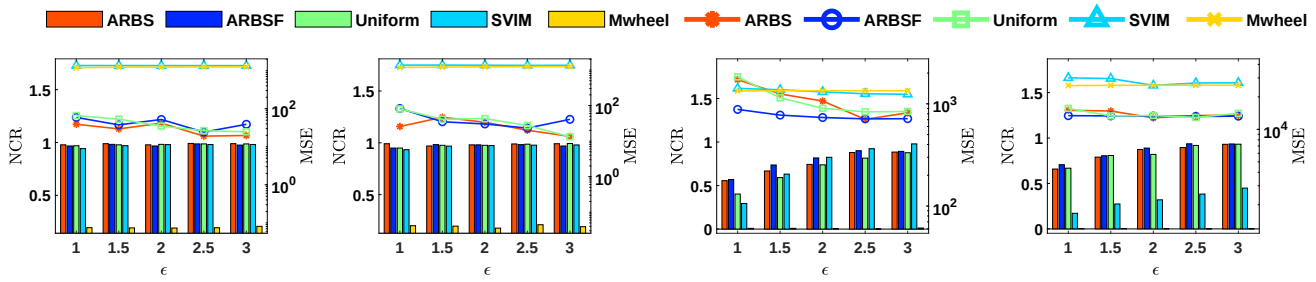
6.4 Performance of Adaptive Schemes

Finally, we study the performance of adaptive schemes in terms of sampling times allocation, error and the number of rounds. For a more intuitive understanding, we mainly execute our schemes on **Linear** $45°$, where there are 10,000 users with 100 items and the frequency monotonically increases at equal intervals from 0 to 1. Moreover, we also execute our experiments on data following the Beta(2,5) distribution.

6.4.1 Time Analysis

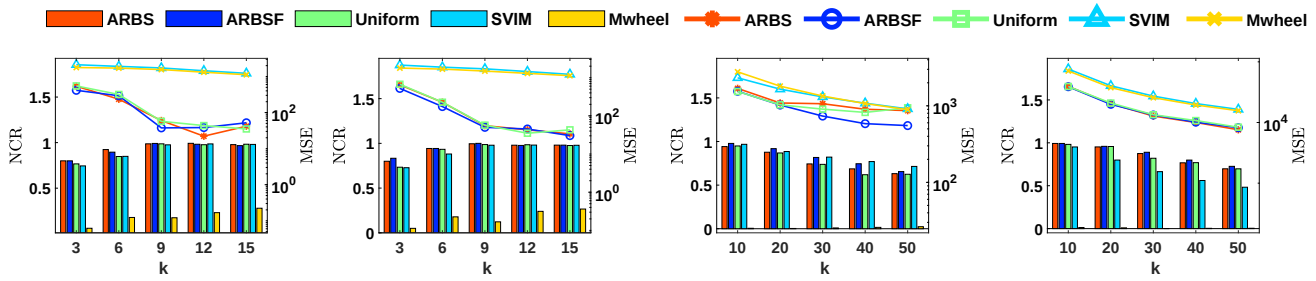
In Table 5, we analyze the time complexity across different datasets. Since we cannot accurately simulate the interaction process of the data collector, we use the numbers of interaction between a user and the data collector as a basic unit and analyze the performance for different schemes. If the sampling probability distribution is same, multiple users can interact with the data collector simultaneously, which can be considered a single interaction.

Without batch operations, the data collector needs to adjust the sampling probability after collecting data from each individual user, so the number of interactions equals the number of users. This means users cannot upload data synchronously because each user must wait for the previous user to finish uploading before the data collector updates the



(a) **Word_ google**, $k = 13$. (b) **Word _ mit**, $k = 13$. (c) **Retail**, $k = 30$. (d) **Kosarak**, $k = 30$.

Fig. 3. The results of NCR & MSE w.r.t. ϵ on real-world datasets.



(a) **Word_ google**, $\epsilon = 2$. (b) **Word_ mit**, $\epsilon = 2$. (c) **Retail**, $\epsilon = 2$. (d) **Kosarak**, $\epsilon = 2$.

Fig. 4. The results of NCR & MSE w.r.t. k on real-world datasets.

TABLE 4
Number of interactions for different datasets

Dataset	No Batch	Round 5	Round 10	Round R
Linear	10000	5	10	R
Beta	10000	5	10	R
gamma	100000	6	11	$R+1$
laplace	100000	6	11	$R+1$
$Word_{google}$	10000	5	10	R
$Word_{mit}$	10000	5	10	R
Retail	270229	6	11	$R+1$
Kosarak	990002	6	11	$R+1$

sampling probability, making the process excessively time-consuming. However, with the batch scheme, the data collector can collect data from a group of users simultaneously and then update the sampling probability before interacting with the next group of users. This approach significantly reduces the time complexity by enabling synchronous data uploads.

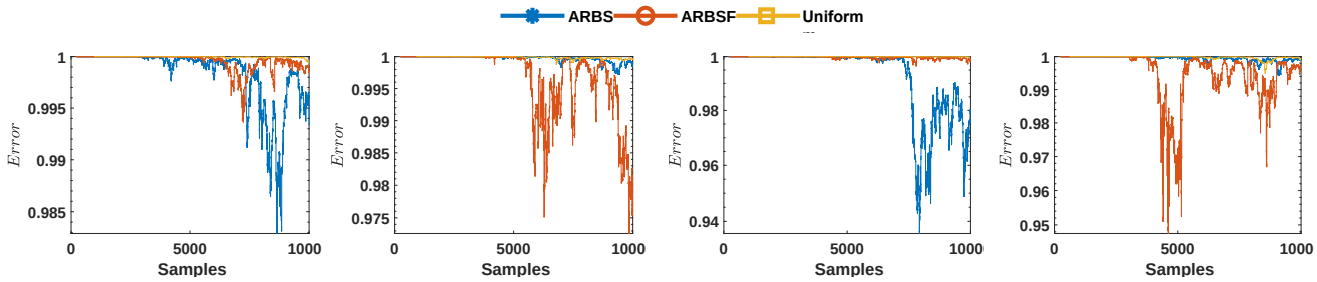
We then discuss the number of interactions for different datasets in two categories: when the dataset range is relatively small, such as **Linear** and **Beta**, the number of interactions is equal to R; when the dataset range is relatively large, such as **Retail** and **Kosarak**, the number of interactions is $R + 1$. According to Section 4.5, half of the users will first interact with the data collector to prune the domain. These users can interact with the data collector simultaneously, and then the remaining interactions are added, resulting in a total of $R + 1$ interactions.

6.4.2 *Top-k Discovery Error Δ*

In Fig. 5, we utilize a one-sided two-sample T-test to assess the error Δ in the top-k discovery, as mentioned in Theorem 4.2. Since only ARBS, ARBSF, and Uniform collect data from users one by one or across multiple rounds, we are able to dynamically capture their frequency distributions. For the other methods, data is collected in a one-time process, and aggregation can only be performed after all data has been collected. A higher Δ is indicative of a decrease in the estimation's accuracy. The yellow line represents the performance of the uniform sampling technique, which exhibits the highest error among the sampling algorithms considered. Furthermore, the error decreases slowly as the number of users increases. On the other hand, the ARBS and ARBSF schemes, shown in blue and red, respectively, exhibit lower error and decrease dramatically as the number of collected users increases. It is important to note that the error does not always decrease with an increasing number of collected users, as $\hat{p}_k$ and $\hat{p}_{k+1}$ may change during the data collection process. However, the error gradually reduces when p_k and p_{k+1} remain constant. Our scheme adapts to changing probabilities and increasing information by dynamically updating p_k and p_{k+1}, providing both flexibility and accuracy.

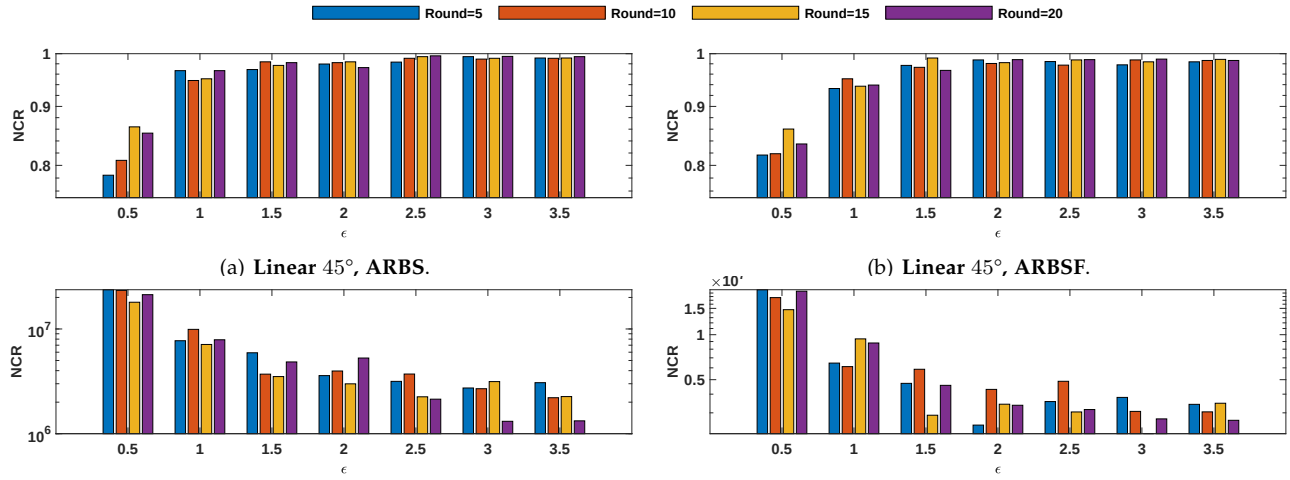
6.4.3 *Impact of Number of Rounds*

Fig. 6 shows the impact of number of rounds (i.e., 5, 10, 15, and 20) on the overall results. We observe a continuous increase in accuracy with increasing ϵ, as evidenced by the gradual increase in NCR and decrease in MSE. However, we note that the effect of increasing the number of rounds on NCR is not particularly significant, especially for larger privacy budgets (e.g., $\epsilon > 2$). In fact, as the number of rounds increases, the final result may not necessarily exhibit a monotonic increase in accuracy. This is due to the data-dependent nature of our experiments and the potential variation in the number of users assigned to each round. Such differences can impact the learning outcomes in each round and ultimately affect the final accuracy. However, we observe that MSE with more rounds generally performs better. If there are no constraints on communication overhead, increasing the number of rounds would enhance accuracy.



(a) **Linear** 45°, $\epsilon = 2$, R=5. (b) **Linear** 45°, $\epsilon = 2$, R=10. (c) **Beta(2,5)**, $\epsilon = 2$, R=15. (d) **Beta(2,5)**, $\epsilon = 2$, R=20.

Fig. 5. The results of error estimated by T-test.



(a) **Linear** 45°, **ARBS**. (b) **Linear** 45°, **ARBSF**.

(c) **Beta(2,5), ARBS**. (d) **Beta(2,5), ARBSF**.

Fig. 6. The results of NCR & MSE w.r.t. ϵ with different round.

6.4.4 *Impact of Allocated User Number Per Item*

The histograms in Fig. 7 depict the number of users allocated to different items by the ARBS and the ARBSF. We sort the items from the smallest to the largest, and then set k to 30. The yellow line representing uniform sampling is flat, indicating that naive sampling treats all items equally. The blue curve represents ARBS and includes a small peak around the 70-th item, which indicates that more sampling probability is given to the items nearby the k-th item. The red curve represents the ARBSF scheme, and rises between the items of 70 and 100, illustrating how we aim to achieve higher statistical frequency accuracy while ensuring that the top-k discovery is generally correct. The experimental results are consistent with our intention, as described in Section 4.

6.4.5 *Impact of Allocated User Number per Round*

In Fig. 8, we can observe that the number of collected users increases with each round, thereby illustrating the correctness of the Theorem 5.2. It is worth noting that the last round typically has the highest number of users in each figure. This is because our scheme is locally optimal, and after the first $(r-1)$ rounds, it would be desirable to allocate all the remaining data to the final round. This ensures that we obtain the maximum amount of information from the dataset. It is noteworthy that when $R = 20$, the number of users in the final three rounds remains almost constant. This is because the decrease in error during these rounds is insignificant for **Linear**, leading to an inconsequential effect on the number of users. Nevertheless, this scenario still adheres to a progressively increasing n_r, as demonstrated in Theorem 5.2.

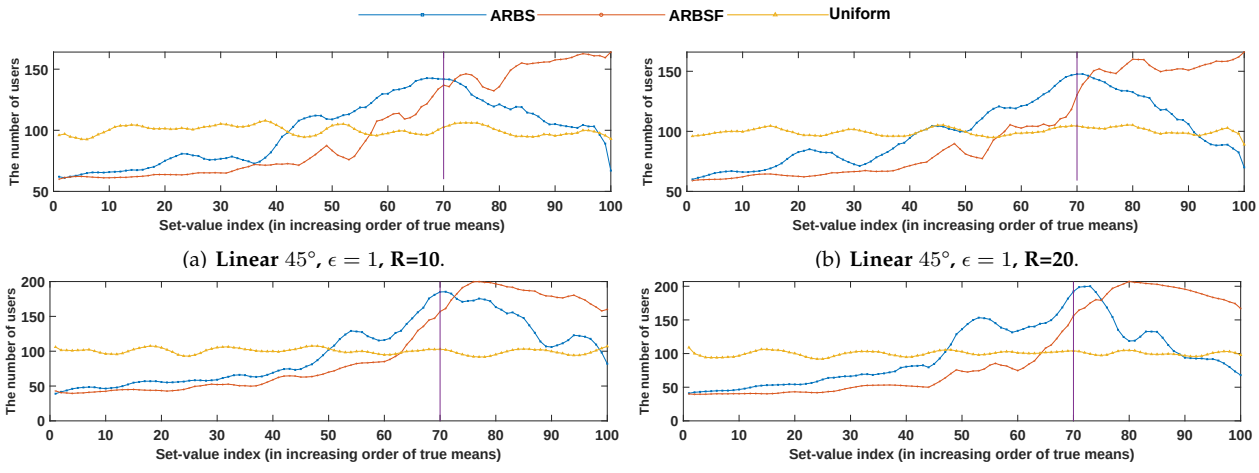
6.4.6 *Robust analysis*

Fig. 9 shows the changes in the estimated results as the number of rounds increases. We can observe that the overall trend of the frequency estimation after the first round (represented by the blue line) is already close to the ground truth (indicated by the dashed line). This is because in the initialization process, each user collected data $\frac{n_0}{d}$ times, providing an initial frequency estimation that is relatively accurate. However, we can also see many outliers in the blue line, representing cases where the frequency estimation is highly inaccurate after initialization round.

Nevertheless, as the number of rounds increases, the frequency estimation is gradually corrected and approaches the true results. This demonstrates that our mechanism can effectively rectify the frequency errors introduced during the initialization process. This is because we ensure that each sample has a probability of being sampled, and the difference in these probabilities is not too large.

6.4.7 *Ablation Experiment on Domain Pruning*

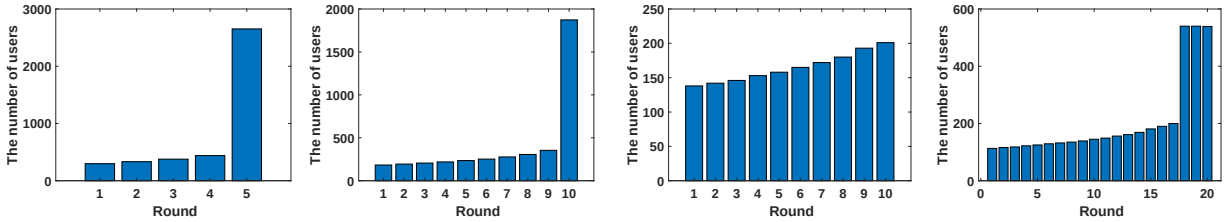
We conduct an ablation study on domain pruning in TABLE 6. Domain pruning is designed specifically for large-scale datasets, such as the extremely sparse **Retail** and **Kosarak**, where the domain of set-values is very large, but many values are only associated with a small number of users. Our



(a) **Linear** $45°$, $\epsilon = 1$, **R=10**. (b) **Linear** $45°$, $\epsilon = 1$, **R=20**.

(c) **Linear** $45°$, $\epsilon = 2$, **R=10**. (d) **Linear** $45°$, $\epsilon = 2$, **R=20**.

Fig. 7. The number of users allocated on each item.



(a) **Linear** $45°$, $\epsilon = 2$, **R=5**. (b) **Linear** $45°$, $\epsilon = 2$, **R=10**. (c) **Linear** $45°$, $\epsilon = 2$, **R=15**. (d) **Linear** $45°$, $\epsilon = 2$, **R=20**.

Fig. 8. The number of users allocated in each round.

TABLE 5
Ablation Experiment on Domain Pruning

Datasets	Scheme	NCR	Hit rate	MSE
Retail	ARBS	0.9164	0.8	1.94E+03
	ARBSF	0.9745	0.92	1.62E+03
	Ablation	0.0214	0.025	2.98E+03
Kosarak	ARBS	0.9495	0.82	1.95E+04
	ARBSF	0.9448	0.85	1.96E+04
	Ablation	0	0	3.96E+04

schemes include an initialization round, during which each value is collected $t(n_0/d)$ times to gather initial information. According to the calculation in Equation (10), the computed t exceeds n/d, where n is the number of users and d is the value domain. This indicates that n is too small relative to d to provide sufficient information from the initial round to support subsequent adaptive sampling operations. Consequently, our schemes degrade to the Randomized Response. We observe that ARBS and ARBSF with domain pruning improve the hit rate, NCR, and MSE compared to their performance without domain pruning. This improvement occurs because, without domain pruning, the value domain is larger, and each value is assigned to fewer users, leading to highly inaccurate estimates and consequently poor top-k performance.

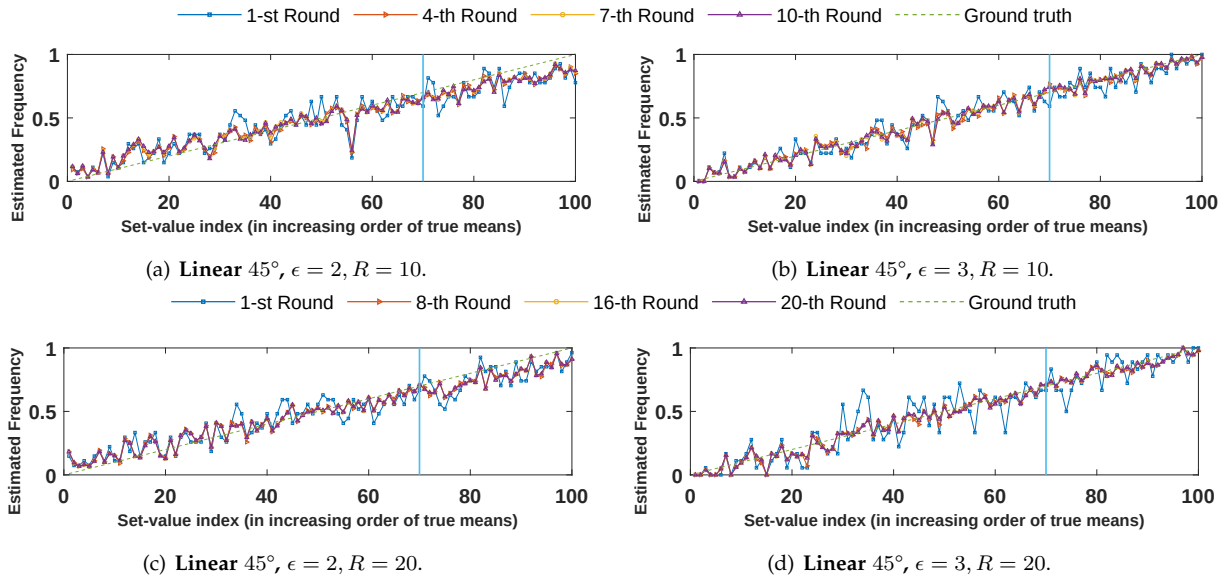
7 Related Work

7.1 Frequency Estimation with LDP

LDP [6], [7], [30], or local differential privacy, is a technique that aims to provide privacy guarantees for individual users in distributed systems. The concept of local privacy was first introduced in 1965, and the randomized response (RR) model [17], which is a simple perturbation technique proposed by Warner. Since it was proposed, LDP has been widely studied and applied. LDP has been extensively used across various fields, such as time series data release [31], [32], graph data collection [33], [34], key-value data analysis [35], [36], [37] and machine learning [38], [39]. LDP has been successfully deployed on Google's Chrome [1], Apple's iOS [3], and Windows 10 by Microsoft [4]. Samsung [40] has also conducted research on this technology.

One of the key objectives of Local Differential Privacy (LDP) is frequency estimation [1], [8]. Beyond studying single attributes, a substantial body of work has also investigated the values of multiple attributes [10], [41], [42]. The problem most closely related to collecting multi-attribute values is the collection of set-valued data. In the former case, each user has d attributes, with a single value corresponding to each attribute. In the latter case, each user has a private set containing some subset of the d attributes, where the value is the attribute itself rather than a value associated with the attribute.

Estimating frequencies in set-valued data presents significant complexities. Among the pioneering works in this domain are LDPminer [12] and SVIM [13]. The primary objective of these seminal methods is to identify the "heavy hitters" - the most frequently occurring elements - and accurately estimate the frequencies of their corresponding values within the highly sparse set-valued data setting. LDPMiner is a frequency publishing method targeted at heavy-hitter queries. First, data are collected, and the collector determines the heavy hitter set and returns it to the



(a) **Linear** $45°$, $\epsilon = 2, R = 10$.

(b) **Linear** $45°$, $\epsilon = 3, R = 10$.

(c) **Linear** $45°$, $\epsilon = 2, R = 20$.

(d) **Linear** $45°$, $\epsilon = 3, R = 20$.

Fig. 9. The results of NCR & MSE w.r.t. ϵ with different round.

user. The user then sends data corresponding to some of the items in the set to the data collector. Wang et al. [12] expanded upon LDPMiner by introducing the Set-Value Item Mining (SVIM) technique, which is adaptable to discovering frequent item sets, specifically through Set-Value ItemSet Mining (SVSM). In SVIM, the authors first focus on set-value top-k collecting problems by using PSFO mechanisms to estimate both frequent and frequent itemsets. SVIM generally employ padding and sampling techniques to mitigate the variance resulting from large domains. However, these techniques introduce bias, with a padding length smaller than the length of the user's private set. Furthermore, the frequency estimation could perform better due to the significant variance when the padding length exceeds that of most value sets. Hence, there may be better methods for accurate frequency estimation.

Traditional methods [43], [44] involve estimating frequencies across all values and deriving heavy hitters through frequency-based ranking, yet these approaches falter in efficiency with large data domains. Addressing this, pruning-based strategies like PEM [28] and SPM [2] have been developed to segment the domain and collect sub-values across distinct user groups, enhancing efficiency. Despite these advancements in identifying heavy hitters within set-valued data, computational burdens become substantial in extensive domains, necessitating frequency estimations for all potential values. In contrast, the proposed PemSet approach targets heavy hitters by exploring prefix spaces, markedly boosting efficiency. Utilizing the exponential mechanism [45], the PrivSet framework [16] was proposed, offering enhanced accuracy but at the cost of increased communication overhead due to the necessity of submitting item subsets. The Wheel mechanism [46] theoretically reduces estimation errors while keeping both computational and communication costs low.

The most recent research work is paper [26], which introduces the PemSet framework for identifying the top-k most frequently occurring items in set-valued data. It also proposes mechanisms like MOLH, MOLH-S, MPCKV, and MWheel tailored for estimating frequencies in multisets. Among these, the MWheel mechanism is reported to perform the best. However, PemSet primarily focuses on finding the top-k items without providing their actual frequency estimates. Moreover, while the MWheel mechanism works well for multisets of equal length, it does not perform optimally with set-valued data of varying lengths. Furthermore, this approach also exhibits relatively average performance when the data domain is relatively small in scale.

7.2 Multi-armed bandits

The classical MAB problem is a formulation of the exploration and exploitation dilemma inherent in reinforcement learning [47]. The MAB problem [48] , which involves decision-making under uncertainty, has been extensively studied for decades. MAB problem has found applications in numerous fields. such as online advertising [49], clinical trials [50], networking [47], [51], and pairwise ranking [52].

Most MAB studies focused on either (i) minimizing the regret (e.g., [53], [54], [55], [47], [56], [57]) through a tradeoff between the exploration and exploitation of arms or researching pure exploration problems, which are aimed at identifying one, or (ii) pure exploration problems (e.g., [47], [53], [54], [54], [55], [56], [57]) where minimizing the number of samples taken or identifying one or multiple best possible arms while satisfying specific conditions (e.g., within a fixed number of samples or with the least cost). In this study, we focus on exploring the best reward with the given cost, which indicates that the problem is of the pure exploration type. There are already quite a few papers [19], [41], [58], [59], [60] that use MAB methods under the LDP system to achieve higher privacy protection. This paper presents how Multi-Armed Bandit (MAB) methods can be employed to enhance the performance of the existing LDP perturbation protocol, RR, in achieving better utility for top-k estimation on set-value data.

8 Conclusions

In this work, we focus on top-k estimation for set-valued data under LDP. First, we comprehensively overview ex-

isting LDP techniques and evaluate their suitability for set-valued data. Then, we offer a new perspective and present an unbiased and adaptive study of top-k estimation under LDP. Specifically, two adaptive sampling methods are deeply investigated, namely ARBS for identifying top-k items and ARBSF for both top-k item discovery and frequency estimation on these items. Additionally, we propose an optimization to reduce computational complexity while maintaining low communication overhead. The theoretical and experimental results demonstrate the effectiveness of our proposed method.

In future work, we will explore multi-dimensional data for identifying and estimating top-k items and will design suitable sampling strategies. On the other hand, we also plan to extend our work to other domains [61] involving more complicated data types, such as streaming user data.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No: 62372122, 92270123 and 62072390), the Research Grants Council, Hong Kong SAR (Grant No: 15203120, 15208923 and 15210023), and the MUST Faculty Research Grants (Grant No: FRG-24-027-FIE).

REFERENCES

[1] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014, pp. 1054–1067.

[2] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries," *arXiv preprint arXiv:1503.01214*, 2015.

[3] S. Vadhan, "Learning with privacy at scale," in *Apple Machine Learning,Journal,vol. 1, no. 8*, 2017, pp. 1–25.

[4] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[5] R. Chen, B. C. Fung, S. Y. Philip, and B. C. Desai, "Correlated network data publication via differential privacy," *The VLDB Journal*, vol. 23, no. 4, pp. 653–676, 2014.

[6] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE, 2013, pp. 429–438.

[7] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?" *SIAM Journal on Computing*, vol. 40, no. 3, pp. 793–826, 2011.

[8] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, 2015, pp. 127–135.

[9] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 729–745.

[10] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and S. Y. Philip, "Lopub: High-dimensional crowdsourced data publication with local differential privacy," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2151–2166, 2018.

[11] U. S. V. G. K. J. F. V. V. P. A. L. A. G. Thakurta, A. H. Vyrros and S. Duplinsky, "Emoji frequency detection and deep link frequency, us patent," Patent.

[12] 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 *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 192–203.

[13] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 127–143.

[14] C. Dwork, "Differential privacy: A survey of results," in *International conference on theory and applications of models of computation*. Springer, 2008, pp. 1–19.

[15] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[16] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. IEEE, 2007, pp. 94–103.

[17] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.

[18] V. Mnih, C. Szepesvári, and J.-Y. Audibert, "Empirical bernstein stopping," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 672–679.

[19] S. Kalyanakrishnan and P. Stone, "Efficient selection of multiple bandit arms: Theory and practice," in *ICML*, 2010.

[20] N. Li, M. Lyu, D. Su, and W. Yang, "Differential privacy: From theory to practice," *Synthesis Lectures on Information Security, Privacy, & Trust*, vol. 8, no. 4, pp. 1–138, 2016.

[21] C.-H. Chen, D. He, M. Fu, and L. H. Lee, "Efficient simulation budget allocation for selecting an optimal subset," *INFORMS Journal on Computing*, vol. 20, no. 4, pp. 579–595, 2008.

[22] A. Papoulis, *Probability and statistics*. Prentice-Hall, Inc., 1990.

[23] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[24] D. Semenick, "Tests and measurements: The t-test," *Strength & Conditioning Journal*, vol. 12, no. 1, pp. 36–37, 1990.

[25] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities (Cambridge mathematical library)*. cambridge university press, 1934.

[26] Y. Zhu, Y. Cao, Q. Xue, Q. Wu, and Y. Zhang, "Heavy hitter identification over large-domain set-valued data with local differential privacy," *IEEE Transactions on Information Forensics and Security*, 2023.

[27] A. M. Mood, "Introduction to the theory of statistics." 1950.

[28] T. Wang, N. Li, and S. Jha, "Locally differentially private heavy hitter identification," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 2, pp. 982–993, 2019.

[29] H. Steck, "Training and testing of recommender systems on data missing not at random," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 713–722.

[30] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin, "Private spatial data aggregation in the local setting," in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 2016, pp. 289–300.

[31] Q. Ye, H. Hu, N. Li, X. Meng, H. Zheng, and H. Yan, "Beyond value perturbation: Local differential privacy in the temporal setting," in *INFOCOM*. IEEE, 2021, pp. 1–10.

[32] Q. Ye, H. Hu, K. Huang, M. H. Au, and Q. Xue, "Stateful switch: Optimized time series release with local differential privacy," *arXiv preprint arXiv:2212.08792*, 2022.

[33] H. Sun, X. Xiao, I. Khalil, Y. Yang, Z. Qin, H. W. Wang, and T. Yu, "Analyzing subgraph statistics from extended local views with decentralized differential privacy," in *CCS*. ACM, 2019, pp. 703–717.

[34] Q. Ye, H. Hu, M. H. Au, X. Meng, and X. Xiao, "Towards locally differentially private generic graph metric estimation," in *ICDE*. IEEE, 2020, pp. 1922–1925.

[35] Q. Ye, H. Hu, X. Meng, and H. Zheng, "PrivKV: Key-value data collection with local differential privacy," in *S&P*. IEEE, 2019, pp. 317–331.

[36] X. Gu, M. Li, Y. Cheng, L. Xiong, and Y. Cao, "PCKV: locally differentially private correlated key-value data collection with optimized utility," in *USENIX Security Symposium*, 2020.

[37] Q. Ye, H. Hu, X. Meng, H. Zheng, K. Huang, C. Fang, and J. Shi, "PrivKVM*: Revisiting key-value statistics estimation with local differential privacy," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[38] H. Zheng, Q. Ye, H. Hu, C. Fang, and J. Shi, "BDPL: A boundary differentially private layer against machine learning model extraction attacks," in *ESORICS*. Springer, 2019, pp. 66–83.

[39] ——, "Protecting decision boundary of machine learning model with differentially private perturbation," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[40] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," *arXiv preprint arXiv:1606.05053*, 2016.

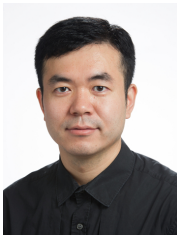
[41] R. Du, Q. Ye, Y. Fu, and H. Hu, "Collecting high-dimensional and correlation-constrained data with local differential privacy,"

in *2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2021, pp. 1–9.
[42] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu, "Collecting and analyzing multidimensional data with local differential privacy," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 638–649.
[43] R. Bassily, K. Nissim, U. Stemmer, and A. G. Thakurta, "Practical locally private heavy hitters," in *Advances in Neural Information Processing Systems*, 2017, pp. 2288–2296.
[44] J. Jia and N. Z. Gong, "Calibrate: Frequency estimation and heavy hitter identification with local differential privacy via incorporating prior knowledge," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2008–2016.
[45] S. Wang, L. Huang, Y. Nie, P. Wang, H. Xu, and W. Yang, "Privset: Set-valued data analyses with locale differential privacy," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 1088–1096.
[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," *Proceedings of the VLDB Endowment*, vol. 13, no. 8, pp. 1234–1247, 2020.
[47] S. Bubeck, N. Cesa-Bianchi *et al.*, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Foundations and Trends® in Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012.
[48] D. A. Berry and B. Fristedt, "Bandit problems: sequential allocation of experiments (monographs on statistics and applied probability)," *London: Chapman and Hall*, vol. 5, no. 71-87, pp. 7–7, 1985.
[49] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.
[50] D. A. Berry and S. G. Eick, "Adaptive assignment versus balanced randomization in clinical trials: a decision analysis," *Statistics in medicine*, vol. 14, no. 3, pp. 231–246, 1995.
[51] S. Buccapatnam, F. Liu, A. Eryilmaz, and N. B. Shroff, "Reward maximization under uncertainty: Leveraging side-observations on networks," *arXiv preprint arXiv:1704.07943*, 2017.
[52] A. Agarwal, S. Agarwal, S. Assadi, and S. Khanna, "Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons," in *Conference on Learning Theory*. PMLR, 2017, pp. 39–75.
[53] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, pp. 235–256, 2002.
[54] P. Auer and R. Ortner, "Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem," *Periodica Mathematica Hungarica*, vol. 61, no. 1-2, pp. 55–65, 2010.
[55] A. Garivier and O. Cappé, "The kl-ucb algorithm for bounded stochastic bandits and beyond," in *Proceedings of the 24th annual conference on learning theory*. JMLR Workshop and Conference Proceedings, 2011, pp. 359–376.
[56] S. Agrawal and N. Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," in *Conference on learning theory*. JMLR Workshop and Conference Proceedings, 2012, pp. 39–1.
[57] F. Liu, S. Wang, S. Buccapatnam, and N. Shroff, "Ucboost: a boosting approach to tame complexity and optimality for stochastic bandits," *arXiv preprint arXiv:1804.05929*, 2018.
[58] P. Gajane, T. Urvoy, and E. Kaufmann, "Corrupt bandits for preserving local privacy," in *Algorithmic Learning Theory*. PMLR, 2018, pp. 387–412.
[59] W. Ren, X. Zhou, J. Liu, and N. B. Shroff, "Multi-armed bandits with local differential privacy," *arXiv preprint arXiv:2007.03121*, 2020.
[60] Y. Tao, Y. Wu, P. Zhao, and D. Wang, "Optimal rates of (locally) differentially private heavy-tailed multi-armed bandits," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 1546–1574.
[61] Z. Wei, L. Lin, Y. Chen, Y. Lin, and Z. Chen, "Partial homogeneity based high-resolution nuclear magnetic resonance spectra under inhomogeneous magnetic fields," *Applied Physics Letters*, vol. 105, no. 13, 2014.

Rong Du is currently a PhD student in the Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University. She received her BEng. degree in School of Electrical Engineering from Beijing Normal University in 2017 and her Master degree in the Department of Electronic and Information Engineering from Peking University in 2020. Her research field is data privacy and security.

Qingqing Ye is an assistant professor in the Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University. She received her PhD degree from Renmin University of China in 2020. She has received several prestigious awards, including Hong Kong RGC Early Career Award, IEEE S&P Travel Award, and National Scholarship. Her research interests include data privacy and security, and adversarial machine learning.

Yue Fu is currently a PhD student in the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University. He received his BEng. degree in the Department of Mathematics from Capital Normal University in 2015 and his Master degree in the School of Electronic and Computer Engineering from Peking University in 2019. His research field is data privacy and security.



Haibo Hu is a professor in the Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University. His research interests include cybersecurity, data privacy, internet of things, and adversarial machine learning. He has published over 110 research papers in refereed journals, international conferences, and book chapters. As principal investigator, he has received over 20 million HK dollars of external research grants from Hong Kong and mainland China. He is an associate editor of ACM Transactions on Privacy and Security (TOPS). He is the recipient of a number of titles and awards, including IEEE MDM 2019 Best Paper Award, WAIM Distinguished Young Lecturer, ICDE 2020 Outstanding Reviewer, VLDB 2018 Distinguished Reviewer, ACM-HK Best PhD Paper, Microsoft Imagine Cup, and GS1 Internet of Things Award. He is a senior member of ACM, IEEE and CCF, and a certified Cisco CCNA Security Trainer.

Kai Huang is an assistant professor of School of Computer Science and Engineering, Faculty of Innovation Engineering, Macau University of Science and Technology. He was a Postdoc at the Department of Computer Science and Engineering, HKUST, under the supervision of Prof. Xiaofang Zhou. He received his PhD degree in School of Computer Science from Fudan University in 2020, and BEng degree in Software Engineering from East China Normal University in 2014. His research interests include graph database and privacy-aware data management.