

From Randomized Response to Randomized Index: Answering Subset Counting Queries with Local Differential Privacy

Qingqing Ye*, Liantong Yu*, Kai Huang[†], Xiaokui Xiao[‡], Weiran Liu[§], Haibo Hu*
 *Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University
 Email: qqing.ye@polyu.edu.hk, liantong2001.yu@connect.polyu.hk, haibo.hu@polyu.edu.hk

[†]Faculty of Information Technology, Macau University of Science and Technology
 Email: kylehuangk@gmail.com

[‡]School of Computing, National University of Singapore
 Email: xkxiao@nus.edu.sg

[§]Alibaba Group
 Email: weiran.lwr@alibaba-inc.com

Abstract—Local Differential Privacy (LDP) is the predominant privacy model for safeguarding individual data privacy. Existing perturbation mechanisms typically require perturbing the original values to ensure acceptable privacy, which inevitably results in value distortion and utility deterioration. In this work, we propose an alternative approach — instead of perturbing values, we apply randomization to indexes of values while ensuring rigorous LDP guarantees.

Inspired by the deniability of randomized indexes, we present CRIAD for answering subset counting queries on set-value data. By integrating a multi-dummy, multi-sample, and multi-group strategy, CRIAD serves as a fully scalable solution that offers flexibility across various privacy requirements and domain sizes, and achieves more accurate query results than any existing methods. Through comprehensive theoretical analysis and extensive experimental evaluations, we validate the effectiveness of CRIAD and demonstrate its superiority over traditional value-perturbation mechanisms.

1. Introduction

As big data analytics become more prevalent, service providers are increasingly eager to collect extensive usage data to improve their services. However, much of this data, particularly when gathered from individuals, includes sensitive information such as biometrics, personal identification, health data, financial transactions, and location trajectories. Directly collecting such data for model training or statistical analysis raises significant privacy concerns.

To address these privacy challenges, Local Differential Privacy (LDP) [9], [20] has emerged as the predominant privacy model in many large-scale distributed systems, used by tech giants such as Apple [26], Google [15] and Microsoft [6], to protect end users’ data. Despite its benefits, LDP has faced criticism for its low utility, as the values collected must undergo significant or even unbounded perturbation by noise injection [13], [14], [27] or value randomization [19], [29], [33] to ensure acceptable privacy.

In the literature, there are many existing works on LDP for set-value data [7], [25], [28], [31], [32], [36]. Specifically, each user possesses a set of private items, and the data collector aims to estimate the frequency of a specific item or identify the most frequent items among users. However, in real-world applications, items are often organized into categories and there is a need to estimate statistics for the set of items belonging to a specific category. The following are two examples.

- Amazon sells millions of books, each belonging to a specific category, e.g., fiction, poetry. To predict sales and manage inventory effectively, Amazon needs to obtain the sales volume for a specific category among thousands of titles.
- To optimize advertising strategies, IMDb needs to determine which movie genre attracts the largest audience. This involves analyzing the number of users interested in a specific genre (e.g., thriller) that encompasses a set of movies.

In this study, we formulate such problem as a **subset counting query**, which returns the total count of items within a subset of the item domain. To answer this query in the context of LDP, there are two solutions adapted from existing works. First, each user counts her items belonging to that subset and then employs a numerical-value perturbation mechanism (e.g., Laplace Mechanism [13] or Piecewise Mechanism [27]) to inject random noise to the count and then reports the sanitized count. Alternatively, each user employs a categorical-value perturbation mechanism (e.g., Randomized Response [19], [33]) to perturb her item set and then reports the perturbed items.

However, both two solutions introduce perturbation errors into the original value. In mission-critical applications, particularly in medical and financial applications, value perturbation does not apply as distorted values become useless or even detrimental. In fact, significant distortion of values can overshadow the original value and potentially make it unbounded. In this work, we propose an alternative

approach: instead of perturbing values, we apply randomization to indexes of values, while ensuring a rigorous LDP guarantee. The following example illustrates how randomized index ensures plausible deniability, while the original values remain intact.

Example. *In a ballot with 10 candidates, each voter independently votes yes/no to each candidate. An interviewer wants to estimate the average number of “yes” votes of all voters. To ensure privacy of these votes, each voter randomly samples a candidate index from $\{1, 2, \dots, 10\}$, and then faithfully reports her vote (i.e., yes/no) for the selected candidate, but without indicating the index she has sampled.*

Inspired by the deniability of the above randomized index, we propose CRI (short for Counting via Randomized Index) protocol for answering subset counting queries, while satisfying ϵ -LDP. Although perturbation noise is not necessary in most cases, to ensure an unbiased estimation, CRI still suffers from utility loss by re-introducing certain perturbation noise in few extreme cases. To address this issue, we develop a dummy-assisted solution CRIAD (short for CRI with Augmented Dummies) to eliminate perturbation noise, which thus achieves significantly higher accuracy. On the other hand, we further enhance the scalability of CRIAD by developing a multi-dummy, multi-sample and multi-group strategy that can support a wide range of privacy requirements (specified by ϵ) and domain sizes. Through theoretical analysis and empirical studies, we show the effectiveness of CRIAD. To summarize, our contributions are three-folded.

- We formulate the problem of answering subset counting queries under LDP, and design randomized-index-based solutions that can ensure rigorous LDP guarantees. To the best of our knowledge, this is the first LDP mechanism based on the deniability of randomized indexes.
- We develop a scalable solution, CRIAD, which accommodates flexible privacy requirements and domain sizes. By leveraging augmented dummy items, CRIAD eliminates perturbation noise injected into the original value while satisfying ϵ -LDP.
- Through comprehensive theoretical and experimental analysis, we demonstrate that CRIAD achieves significantly higher accuracy than existing value-perturbation LDP mechanisms.

The rest of the paper is organized as follows. Section 2 introduces preliminaries, problem definition, and existing solutions. Section 3 presents our baseline solution CRI via randomized index deniability. Section 4 proposes the optimized solution CRIAD. Section 5 presents experimental evaluations. Finally, we review existing work in Section 6 and conclude this paper in Section 7.

2. Preliminaries and Problem Definition

In this section, we introduce preliminaries on LDP, formulate the problem of answering a subset counting query under LDP, and then present two naive solutions that are directly adapted from existing works.

2.1. Local Differential Privacy

Differential privacy (DP) [12], [13] works in both centralized and local settings. Centralized DP [21] requires the data curator to be fully trusted to collect all data, while local DP does not rely on this assumption. In the local setting [9], [20], each user locally perturbs her data before reporting them to an untrusted data collector, which makes it more secure and practical in real-world applications. The formal definition is as follows.

Definition 2.1. (Local Differential Privacy, LDP) A randomized algorithm \mathcal{A} satisfies ϵ -LDP if for any two input records w and w' , and any set W of possible outputs of \mathcal{A} , the following inequality holds.

$$\frac{\Pr[\mathcal{A}(w) \in W]}{\Pr[\mathcal{A}(w') \in W]} \leq e^\epsilon \quad (1)$$

In the above definition, ϵ is called the privacy budget, which controls the deniability of a randomized algorithm taking w or w' as its input. As with centralized DP, LDP also has the property of sequential composition [23] as below, which guarantees the overall privacy for a sequence of randomized algorithms.

Theorem 2.2. (Sequential Composition) Given t randomized algorithms $\mathcal{A}_i (1 \leq i \leq t)$, each providing ϵ_i -local differential privacy, then the sequence of algorithms $\mathcal{A}_i (1 \leq i \leq t)$ collectively provides $(\sum \epsilon_i)$ -local differential privacy.

2.2. Problem Definition

We assume there are n users $\mathcal{U} = \{u_1, u_2, \dots, u_n\}$, and each user u_i possesses a set of private items $R_i \subseteq D$, where $D = \{r_1, r_2, \dots, r_{|D|}\}$ is the item domain. The domain has some categories $\{c_1, c_2, \dots\}$, and each item belongs to one or more categories. In other words, items belonging to a category is a subset of the domain. For ease of presentation, we use $d = |c_i|$ to denote the sub-domain size of category c_i , i.e., the number of items in the domain D belonging to category c_i . Now we formally define the subset counting query as follows.

Definition 2.3. (Subset Counting Query) Denoted by $Q(c)$, a subset counting query takes as input a category c , and returns the count of items belonging to c among the user population. Formally,

$$Q(c) = \sum_{i=1}^n \sum_{r \in R_i} \mathbb{1}_c(r), \quad (2)$$

where $\mathbb{1}_c(r)$ is an indicator function, which returns 1 if an item r belongs to the category c , and returns 0 otherwise.

TABLE 1. NOTATIONS

Symbol	Description
ϵ	the privacy budget
n	the number of users
u_i	the i -th user in user population, $i \in \{1, \dots, n\}$
D	item domain
R_i	a set of items possessed by user u_i , $R_i \subseteq D$
c_i	the i -th category of items in the domain, $i \in \{1, 2, \dots\}$
d	the size of category, $d = c_i $
m	the number of dummy items in CRIAD
g	the number of groups in CRIAD
s	the number of samples in CRIAD

2.3. Solutions from Existing Work

To answer a subset counting query $Q(c)$ of a category c , there are two naive solutions adapted from existing works.

Solution 1: Numerical Value Perturbation (NVP). Each user u_i locally counts her items that belong to category c . Then she perturbs it and reports a sanitized count t_i^* by a numerical-value perturbation mechanism $\mathcal{A}(\cdot)$, e.g., Laplace Mechanism (LM) [13], Piecewise Mechanism (PM) [27] or Square Wave mechanism (SW) [22]. Formally,

$$t_i^* = \mathcal{A}\left(\sum_{r \in R_i} \mathbb{1}_c(r)\right),$$

Then the subset counting query result is the summation of noisy reports from all users, i.e., $\sum_{i=1}^n t_i^*$.

Solution 2: Padding-and-Sampling Perturbation (PSP). Each user first pads (with dummy items) or truncates the item set in her records that belongs to category c into a fixed padding length $\eta \ll |c|$. Then she samples one item from η and perturbs it by a categorical-value perturbation mechanism, e.g., k -ary Randomized Response (kRR) [19], Optimized Unary Encoding [29] or Optimal Local Hashing (OLH) [29]. To compensate the effect of sampling, upon receiving the sanitized reports from all users, the data collector aggregates the count and scales it up by a factor of η . Then the subset counting query $Q(c)$ can be estimated by summing up the counts of all items. This method is adapted from the padding-and-sampling protocol [31], which has been proved to achieve good performance when the domain size is large.

2.4. Pitfalls of Existing Solutions

NVP intuitively treats the local count as a numerical value and employs a numerical-value perturbation mechanism. However, such perturbation may incur large noise, as the local count can vary a lot among users, which could be as low as 0 (i.e., a user has no items belonging to a specific category c) or as high as $|c|$ (i.e., a user has all items belonging to the category c). This inherently results in a large sensitivity for the perturbation mechanism and thus a low utility of subset counting query result.

PSP applies perturbation to a single item instead of the local count on item set, which enables users' reports more informative. However, the effectiveness depends on an

appropriate padding length η . Generally, a large η reduces the number of valid items and increases the estimation variance, whereas a small η underestimates the subset counting query result [31]. In practice, setting an appropriate value for η a priori can be challenging, which hinders its practical application. In addition, for each user it only samples one item and scales it up by η , which will incur large estimation variance.

To summarize, both solutions suffer from large utility loss due to the noise introduced by heavy perturbation. Additionally, an inappropriate parameter setting further diminishes the utility of PSP.

3. Randomized Index for Subset Counting

In this section, we present a novel design for answering subset counting query under LDP, namely Counting via Randomized Index (CRI). We first elaborate on its design rationale, and present a new solution for count aggregation. Then we show the implementation details, followed by comprehensive privacy and utility analysis.

3.1. Randomized Response vs. Randomized Index

In general, given a set of items from each user, a subset counting query returns the total counts of items that belong to a given category. In the literature, all existing works study either frequency estimation of a specific item or top-frequent ones (i.e., heavy hitter identification) [7], [25], [31]. There is no work in counting a set of items that belong to a category.

The existing methods randomize a user's real data to ensure "response-level" deniability, which inevitably incurs utility loss to the query result. Our key idea is to randomize the indexes of the items being counted to ensure the "index-level" deniability, so that the item itself does not need perturbation. The following two examples illustrate the difference between the traditional response-level deniability (Example I) and index-level deniability (Example II).

Example I. *In a survey there are 10 sensitive yes/no questions. Each user adopts PSP in Section 2.3. A user first samples one question from them, randomize her true response, and then reports the sanitized response and the question index to which she answers. So the deniability is guaranteed by the randomized response.*

Example II. *In the same survey, each user randomly samples a question, and then sends her true answer to the data collector, without indicating which question she answers to. So the deniability is guaranteed by the randomized index.*

We observe that Example I exhibits a larger variance and consequently lower accuracy compared to Example II. This is because the former involves both sampling and perturbation error, whereas the latter has the same amount of sampling error but zero perturbation error. This observation motivates us to develop a Counting via Randomized Index (CRI) protocol for answering subset counting queries under the ϵ -LDP guarantee.

3.2. Overview of CRI Protocol

As shown in Figure 1, the CRI protocol for subset counting queries consists of three steps. Step ① pre-processes each user's item set by filtering items unrelated to category c and grouping the filtered items. Step ② produces a *bit vector* of values in category c , each bit corresponding to the existence of one item. Then a bit is sampled from the vector, and sent to the data collector. Upon receiving the sampled bits from all users, the collector estimates the subset counting query result $Q(c)$ in Step ③.

There remains a privacy issue in the above procedure. Recall that in Figure 1, the subset size of the specified category c is 4. The user u_1 has only one item (i.e., R_{11}) belonging to c , resulting in the encoded bit vector '1000'. Thus the sampled bit can be either '1' or '0' with some probability. However, for user u_2 , who has all the items in c and an encoded bit vector of '1111', and user u_3 , who has none of the items in c and an encoded bit vector of '0000', their sampled bits must be '1' and '0' respectively, i.e., without any deniability. This absolutely violates ϵ -LDP. To address this issue, for those all '1' and all '0' cases, we can randomly flip a bit to ensure there are both bits '1' and '0' in each vector. In Figure 1, the flipped bits are shown in red.

In what follows, we will elaborate on the CRI protocol, with a focus on its correctness and privacy analysis.

3.3. CRI: Counting via Randomized Index

Recall that given a subset counting query $Q(c)$ on category c , each user u_i first filters those unrelated items of c and then encodes her filtered item set into a bit vector $V_i = \{V_i^1, V_i^2, \dots, V_i^d\}$ of length $d = |c|$, where each bit V_i^l ($l \in \{1, 2, \dots, d\}$) is defined as

$$V_i^l = \begin{cases} 1, & \text{if } \exists r \in R_i, \mathbb{1}_c(r) = 1, \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

Table 2 shows different cases of bit vectors according to the number of bit '1', where π_t denotes the proportion of those cases whose number of bit '1' is t . Note that each case π_t involves up to $\binom{d}{t}$ different bit vectors. For example, π_1 is the proportion of vectors '100...0', '010...0', '001...0', ..., '000...1' among all n users. Thus, we have $\sum_{t=0}^d \pi_t = 1$.

TABLE 2. CASES OF USERS' ENCODED BIT VECTORS

Proportion	# of 1	# of 0	Pr[1]	Pr[0]
π_0	0	d	0	1
π_1	1	$d-1$	$1/d$	$(d-1)/d$
π_2	2	$d-2$	$2/d$	$(d-2)/d$
π_3	3	$d-3$	$3/d$	$(d-3)/d$
...
π_{d-1}	$d-1$	1	$(d-1)/d$	$1/d$
π_d	d	0	1	0

Based on the above $d+1$ cases, according to Eq. 2, the counting query result on category c is

$$Q(c) = \sum_{t=0}^d n\pi_t \cdot t = n \sum_{t=1}^d t\pi_t. \quad (4)$$

In Table 2, we also show $\Pr[1]$ (resp. $\Pr[0]$), the probability when a user randomly samples and reports bit '1' (resp. '0') from the encoded bit vector. Except for the cases of π_0 and π_d , all cases have non-zero probabilities to report either '0' or '1'. To satisfy ϵ -LDP, a random '0' (resp. '1') should be flipped to '1' (resp. '0') in the case of π_0 (resp. π_d) before sampling. Let z_i denote the sampled index, then the data collector can derive a noisy count from these sampled bits as

$$\bar{\theta} = d \sum_{i=1}^n V_i^{z_i}. \quad (5)$$

And its expectation is

$$\begin{aligned} \mathbb{E}[\bar{\theta}] &= n\pi_0 \cdot 1 + \sum_{t=1}^{d-1} n\pi_t \cdot t + n\pi_d \cdot (d-1) \quad (6) \\ &= n \left(\sum_{t=1}^{d-1} t\pi_t + \pi_0 + (d-1)\pi_d \right). \quad (7) \end{aligned}$$

The term $n\pi_0 \cdot 1$ in Eq. 6 means there is a bit '1' in the case of π_0 after flipping, while the term $n\pi_d \cdot (d-1)$ means there are $d-1$ bit '1' in the case of π_d after flipping. The gap between Eqs. 4 and 7 must be calibrated from $\bar{\theta}$ in Eq. 5 to ensure an unbiased estimation $\tilde{\theta}$:

$$\tilde{\theta} = \bar{\theta} + n(\pi_d - \pi_0). \quad (8)$$

We are yet to derive $\pi_d - \pi_0 = \Delta\pi$ in Eq. 8, which is estimated by a privacy budget ϵ' allocated from the overall budget ϵ . Each user reports a local status flag y_i that indicates whether her case is π_d , π_0 , or neither of them:

$$y_i = \begin{cases} 1, & \text{if } V_i = \{1\}^d, \\ -1, & \text{if } V_i = \{0\}^d, \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

The flag is sanitized into y'_i by kRR [19] with the privacy budget ϵ' :

$$\Pr[y'_i = x] = \begin{cases} \frac{e^{\epsilon'}}{2+e^{\epsilon'}}, & \text{if } x = y_i, \\ \frac{1}{2+e^{\epsilon'}}, & \text{if } x \in \{1, -1, 0\} \setminus y_i. \end{cases} \quad (10)$$

Upon receiving the sanitized status flags of all users, we can estimate $\Delta\pi$ as

$$\Delta\pi = \frac{(2 + e^{\epsilon'}) \sum_{i=1}^n y'_i}{n(e^{\epsilon'} - 1)}. \quad (11)$$

Therefore, the subset counting query result on the category c can be estimated as

$$\tilde{Q}(c) = \tilde{\theta} = \bar{\theta} + n\Delta\pi. \quad (12)$$

In Section 3.4, we will provide the proof of Eq. 11 together with the proof of unbiasedness of Eq. 12.

Algorithm 1 summarizes the workflow of CRI protocol for answering a subset counting query. Given a query on category c , each user u_i first extracts a filtered record set R_i^* from her original R_i (Line 2) and then encodes the filtered item set R_i^* into a bit vector V_i with length $d = |c|$ (Line 3). Subsequently, the user randomly flips a bit if all bits are 1 or 0 (Lines 6 and 9). Meanwhile, the user also sets her

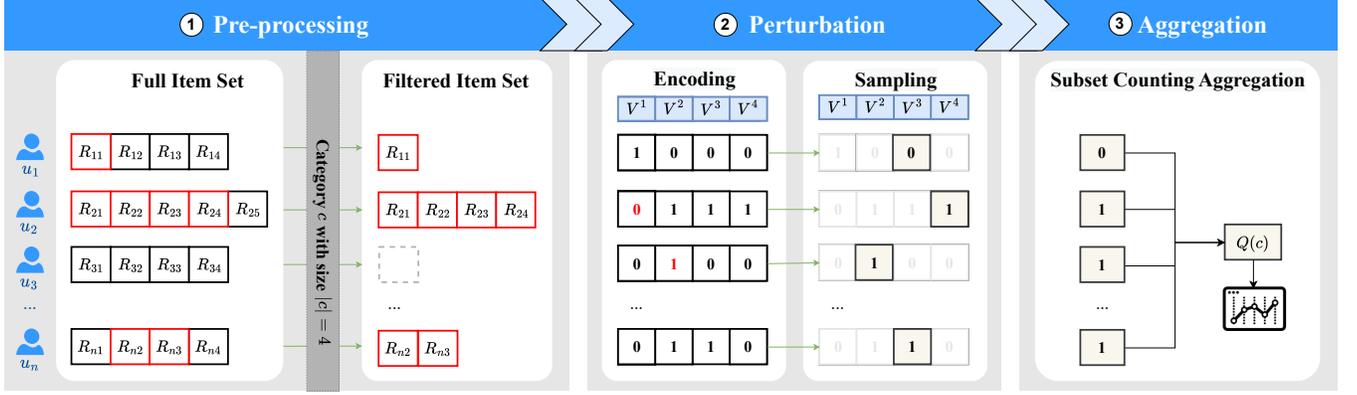


Figure 1. Workflow of CRI for Answering Subset Counting Queries.

Algorithm 1 Workflow of CRI Protocol

Input: A category c
All users' item sets $\{R_i, R_i, \dots, R_n\}$
Privacy budget ϵ

Output: Estimated subset counting query result $\tilde{Q}(c)$

Procedure:

//User side

- 1: **for** each user $u_i \in \mathcal{U}$ **do**
- 2: Extract the item set R_i^* from R_i with items belonging to c
- 3: Encode R_i^* into a bit vector $V_i = \{V_i^1, V_i^2, \dots, V_i^d\}$ by Eq. 3, where $d = |c|$
- 4: **if** $V_i = \{1\}^d$ **then**
- 5: Set flag $y_i = 1$
- 6: Randomly flip a bit to 0
- 7: **else if** $V_i = \{0\}^d$ **then**
- 8: Set flag $y_i = -1$
- 9: Randomly flips a bit to 1
- 10: **else**
- 11: Set flag $y_i = 0$
- 12: Randomly sample an index $z_i \in \{1, 2, \dots, d\}$
- 13: Perturb y_i to y'_i by Eq. 10 with budget $\epsilon' = \epsilon - \log(d-1)$
- 14: Send $V_i^{z_i}$ and y'_i to the data collector

//Collector side

- 15: Calculate the noisy count $\bar{\theta}$ by Eq. 5
- 16: Calculate $\Delta\pi$ by Eq. 11
- 17: Estimate the counting query result $\tilde{Q}(c)$ by Eq. 12
- 18: **return** Query result $\tilde{Q}(c)$

local status flag y_i according to Eq. 9 (Lines 5, 8 and 11). Then the user randomly samples an index $z_i \in \{1, 2, \dots, d\}$ (Line 12) and perturbs her status flag y_i to y'_i by Eq. 10 with privacy budget ϵ' (Line 13). The computation of ϵ' will be elaborated in Theorem 3.1. Finally, the sampled bit and the sanitized status are sent to the data collector (Line 14). Upon receiving all reports from users, the collector calculates a noisy count $\bar{\theta}$ of bit '1' from all sampled bits by Eq. 5, and calculates $\Delta\pi$ from all status flags by Eq. 11, and then estimates the counting query result $\tilde{Q}(c)$ (Lines 15-17).

3.4. Privacy and Utility Analysis

In this subsection, we establish the privacy and utility guarantee of our CRI protocol for subset counting query. In particular, Theorem 3.1 proves that Algorithm 1 satisfies ϵ -LDP. Theorem 3.2 ensures the estimated result is unbiased,

and Theorems 3.3 and 3.3 provide the error bound of the estimation variance.

Theorem 3.1. Algorithm 1 satisfies ϵ -LDP, where $\epsilon = \ln(d-1) + \epsilon'$, d is the size of query category, and ϵ' is the privacy budget for perturbing the status flag by Eq. 10.

PROOF. For a specific category c , each user sends a sampled bit and her sanitized status flag to the data collector. For the bit sampling, we know that each user may sample a bit '1' or '0'. From Table 2, the highest probability to sample a bit 1 (or 0) is $\frac{d-1}{d}$, while the lowest probability is $\frac{1}{d}$. Therefore, for any two users with filtered item sets R_i^* and R_j^* regarding to category c , and for any sampled bit $b \in \{0, 1\}$, we have

$$\frac{\Pr[\text{CRI}(R_i^*) = b]}{\Pr[\text{CRI}(R_j^*) = b]} \leq \frac{(d-1)/d}{1/d} = e^{\ln(d-1)}.$$

Therefore, sampling a bit by CRI satisfies $\ln(d-1)$ -LDP. On the other hand, for any status y' reported by CRI, we know from Eq. 10 that

$$\frac{\Pr[\text{CRI}(R_i^*) = y']}{\Pr[\text{CRI}(R_j^*) = y']} \leq \frac{e^{\epsilon'}/(2+e^{\epsilon'})}{1/(2+e^{\epsilon'})} = e^{\epsilon'}.$$

It means reporting the status flag by CRI satisfies ϵ' -LDP. Then according to the sequential composition in Theorem 2.2, Algorithm 1 satisfies ϵ -LDP, where $\epsilon = \ln(d-1) + \epsilon'$. \square

Theorem 3.2. The estimated counting query result on any category c by Eq. 12 is unbiased, i.e., $\mathbb{E}[\tilde{Q}(c)] = Q(c)$.

PROOF. As for the count estimation, according to Eq. 10, we know when $y_i = 1$, $\mathbb{E}[y'_i] = \frac{e^{\epsilon'}}{2+e^{\epsilon'}} \cdot 1 + \frac{1}{2+e^{\epsilon'}} \cdot (-1) + \frac{1}{2+e^{\epsilon'}} \cdot 0 = \frac{e^{\epsilon'}-1}{2+e^{\epsilon'}}$. Similarly, when $y_i = -1$, $\mathbb{E}[y'_i] = \frac{1-e^{\epsilon'}}{2+e^{\epsilon'}}$, and when $y_i = 0$, $\mathbb{E}[y'_i] = 0$. Therefore,

$$\begin{aligned} \sum_{i=1}^n \mathbb{E}[y'_i] &= c_1 \cdot \frac{e^{\epsilon'}-1}{2+e^{\epsilon'}} + c_{-1} \cdot \frac{1-e^{\epsilon'}}{2+e^{\epsilon'}} \\ &= \frac{(c_1 - c_{-1})(e^{\epsilon'} - 1)}{2 + e^{\epsilon'}}, \end{aligned}$$

where c_1 and c_{-1} are the real counts of status flags 1 and -1 respectively among all users. Then by Eq. 11, we have

$$\begin{aligned}\mathbb{E}[\Delta\pi] &= \frac{(2 + e^{\epsilon'}) \sum_{i=1}^n \mathbb{E}[y'_i]}{n(e^{\epsilon'} - 1)} = \frac{c_1 - c_{-1}}{n} \\ &= \frac{\sum_{i=1}^n \mathbb{1}(V_i = \{1\}^d)}{n} - \frac{\sum_{i=1}^n \mathbb{1}(V_i = \{0\}^d)}{n} \\ &= \pi_d - \pi_0,\end{aligned}$$

which mean $\Delta\pi$ by Eq. 11 is an unbiased estimation of $\pi_d - \pi_0$. By substituting the above $\mathbb{E}[\Delta\pi]$ and $\mathbb{E}[\tilde{\theta}]$ in Eq. 7 to Eq. 12, we have

$$\begin{aligned}\mathbb{E}[\tilde{Q}(c)] &= \mathbb{E}[\tilde{\theta}] + n(\mathbb{E}[\Delta\pi]) \\ &= n \left(\sum_{t=1}^{d-1} t\pi_t + \pi_0 + (d-1)\pi_d \right) + n(\pi_d - \pi_0) \\ &= n \sum_{t=1}^d t\pi_t = Q(c).\end{aligned}$$

As such, the estimation of the subset counting query by Eq. 12 is unbiased. \square

Theorem 3.3. Given a category c , the number of users n , and privacy budget ϵ' for status flag perturbation, the estimation variance of the counting query result by CRI in Algorithm 1 is bounded by $\frac{1}{4}nd^2 + \frac{2n(e^{\epsilon'}+2)}{(e^{\epsilon'}-1)^2}$.

PROOF. According to Eq. 5,

$$\begin{aligned}\text{Var}[\tilde{\theta}] &= d^2 \cdot \text{Var}[\sum_{i=1}^n V_i^{z_i}] = d^2 \cdot \sum_{i=1}^n \text{Var}[V_i^{z_i}] \\ &= nd^2 \left((\pi_0 + \pi_d) \frac{d-1}{d^2} + \sum_{t=1}^{d-1} \pi_t \frac{t(d-t)}{d^2} \right) \\ &\leq nd^2 \left(\frac{\pi_0 + \pi_d}{4} + \sum_{t=1}^{d-1} \frac{\pi_t}{4} \right) \\ &= \frac{1}{4}nd^2.\end{aligned}$$

Let c_1 and c_{-1} denote the real counts of status flags 1 and -1 respectively, and c'_1 and c'_{-1} denote observed counts based on users' reports. Then we know

$$\begin{aligned}\text{Var}[c'_1] &= \frac{2e^{\epsilon'} \cdot c_1}{(2 + e^{\epsilon'})^2} + \frac{(1 + e^{\epsilon'})(n - c_1)}{(2 + e^{\epsilon'})^2}, \\ \text{Var}[c'_{-1}] &= \frac{2e^{\epsilon'} \cdot c_{-1}}{(2 + e^{\epsilon'})^2} + \frac{(1 + e^{\epsilon'})(n - c_{-1})}{(2 + e^{\epsilon'})^2}, \\ \text{Cov}[c'_1, c'_{-1}] &= -\frac{e^{\epsilon'}(c_1 + c_{-1})}{(2 + e^{\epsilon'})^2} - \frac{n - c_1 - c_{-1}}{(2 + e^{\epsilon'})^2}.\end{aligned}$$

Therefore,

$$\begin{aligned}\text{Var}[\sum_{i=1}^n y'_i] &= \text{Var}[c'_1 - c'_{-1}] \\ &= \text{Var}[c'_1] + \text{Var}[c'_{-1}] - 2\text{Cov}[c'_1, c'_{-1}] \\ &= \frac{n(1 + 5e^{\epsilon'}) + 3(n - c_1 - c_{-1})(1 - e^{\epsilon'})}{(2 + e^{\epsilon'})^2}.\end{aligned}$$

According to Eqs. 10 and 11,

$$\begin{aligned}\text{Var}[\Delta\pi] &= \frac{(2 + e^{\epsilon'})^2 \cdot \text{Var}[\sum_{i=1}^n y'_i]}{n^2(e^{\epsilon'} - 1)^2} \\ &= \frac{n(1 + 5e^{\epsilon'}) + 3(n - c_1 - c_{-1})(1 - e^{\epsilon'})}{n^2(e^{\epsilon'} - 1)^2} \\ &\leq \frac{n(1 + 5e^{\epsilon'}) + 3n(1 - e^{\epsilon'})}{n^2(e^{\epsilon'} - 1)^2} \\ &= \frac{2(e^{\epsilon'} + 2)}{n(e^{\epsilon'} - 1)^2}.\end{aligned}$$

According to Eq. 12, we have

$$\begin{aligned}\text{Var}[\tilde{Q}(c)] &= \text{Var}[\tilde{\theta}] + n^2 \cdot \text{Var}[\Delta\pi] \\ &\leq \frac{1}{4}nd^2 + \frac{2n(e^{\epsilon'} + 2)}{(e^{\epsilon'} - 1)^2},\end{aligned}$$

which proves that the variance of frequency estimation by CRI is bounded by $\frac{1}{4}nd^2 + \frac{2n(e^{\epsilon'}+2)}{(e^{\epsilon'}-1)^2}$. \square

By Theorem 3.3, we observe that the estimation error of subset counting query by CRI comes from two sources, namely the sampling and perturbation process. While ensuring deniability for extreme cases where bits are all '1's or '0's, the perturbation comes at a price of an estimation variance of $\frac{2n(e^{\epsilon'}+2)}{(e^{\epsilon'}-1)^2}$. In other words, to derive an estimation of $\Delta\pi$ in Eq. 11 and thus make the counting query result unbiased, **CRI sacrifices its utility by re-introducing the perturbation error**. In the next section, we find an alternative way to ensure deniability for extreme cases, and propose CRIAD which eliminates the perturbation error and thus enhances the utility.

4. CRIAD: Counting via Randomized Index with Augmented Dummies

In this section, we present a utility-enhanced CRI solution to counting queries. The main idea is to augment a user's encoded bit vector with dummy bits, so the protocol is called Counting via Randomized Index with Augmented Dummies (CRIAD). In this section, we first present the skeleton of CRIAD, followed by its customization to cope with various privacy requirements and category sizes. Finally, we summarize its overall procedure, together with the privacy and utility analysis.

4.1. Randomized Index with Augmented Dummies

To ensure the deniability of two extreme cases (all '0's and '1's), we augment a user's bit vector with dummy 0/1 bits, so that either bit can be sampled in both extreme cases and therefore perturbation is no longer needed. Table 3 illustrates this effect when a bit '1' is added to each bit vector, so in the case of π_0 , both $\Pr[1]$ and $\Pr[0]$ are non-zero.

Apparently we can add another dummy bit '0' to each bit vector to fix the case of π_d as well. Nonetheless, these

TABLE 3. CASES OF BIT VECTORS WITH AN AUGMENTED BIT ‘1’

Proportion	No. of 1	No. of 0	Pr[1]	Pr[0]
π_0	1	d	$1/(d+1)$	$d/(d+1)$
π_1	2	$d-1$	$2/(d+1)$	$(d-1)/(d+1)$
π_2	3	$d-2$	$3/(d+1)$	$(d-2)/(d+1)$
π_3	4	$d-3$	$4/(d+1)$	$(d-3)/(d+1)$
...
π_{d-1}	d	1	$d/(d+1)$	$1/(d+1)$
π_d	$d+1$	0	1	0

dummies come at a price of sampling error, because they dilute the original bit distribution in these vectors. For example, in the π_1 case, the sampling probability of bit ‘1’ changes from $\frac{1}{d}$ (Table 2) to $\frac{2}{d+1}$ (Table 3), and will further change to $\frac{3}{d+2}$ if 2 dummies are added. This obviously increases the sampling variance. Our key idea is that in real-world applications, the π_d case (i.e., all bits are ‘1’s) is too rare to contribute to the overall count. This is especially true when the category size is large. Therefore, we can just suppress a π_d cases to a π_{d-1} case by randomly flipping one bit 1 to 0 to refrain from adding a dummy bit ‘0’.

Algorithm 2 Procedure of CRIAD with One Dummy Bit

Input: A category c
All users’ item sets $\{R_i, R_i, \dots, R_n\}$

Output: A sampled bit $V_i^{z_i}$

Procedure:

//User side

- 1: **for** each user $u_i \in \mathcal{U}$ **do**
- 2: Extract the item set R_i^* from R_i with items belonging to c
- 3: Encode R_i^* into a binary vector $V_i = \{V_i^1, V_i^2, \dots, V_i^d, 1\}$ by Eq. 3, where $d = |c|$
- 4: **if** $V_i = \{1\}^{d+1}$ **then**
- 5: Randomly flip a bit to 0
- 6: Randomly sample an index $z_i \in \{1, 2, \dots, d+1\}$
- 7: Report $V_i^{z_i}$ to the data collector

//Collector side

- 8: Estimate the subset counting query result $\tilde{Q}(c)$ by Eq. 13
- 9: **return** Query result $\tilde{Q}(c)$

Algorithm 2 shows the pseudo-code of the above procedure, where one dummy bit ‘1’ is appended as the $(d+1)$ -th bit in each user’s bit vector (Line 3). If all bits in V_i are ‘1’s, the user randomly flips one of them to 0 (Lines 4-5). Then each user randomly samples an index $z_i \in \{1, 2, \dots, d+1\}$ and reports the sampled bit $V_i^{z_i}$ to the data collector (Lines 6-7). At the collector side, the impact of added dummy bits ‘1’ on the counting query can be eliminated by subtracting n from the aggregated bits, as each of n user contributes a dummy bit ‘1’,

$$\tilde{Q}(c) = (d+1) \sum_{i=1}^n V_i^{z_i} - n, \quad (13)$$

where $V_i^{z_i}$ is the reported bit from user u_i . The following two theorems establish the privacy and correctness guarantee of Algorithm 2, respectively.

Theorem 4.1. Algorithm 2 satisfies $\ln d$ -LDP.

PROOF. By Algorithm 2, the case of π_d is reduced to π_{d-1} by randomly flipping a bit ‘1’ to ‘0’. So for any two

filtered item sets R_i^* and R_j^* regarding category c from users, and any bit $b \in \{0, 1\}$ reported, we know

$$\begin{aligned} \frac{\Pr[\text{CRI}(R_i^*) = b]}{\Pr[\text{CRI}(R_j^*) = b]} &\leq \frac{\Pr[b = 1 | V_i \in \pi_{d-1}]}{\Pr[b = 1 | V_j \in \pi_0]} \\ &= \frac{d/(d+1)}{1/(d+1)} = e^{\ln d}. \end{aligned}$$

Therefore, Algorithm 2 satisfies $\ln d$ -LDP. \square

Theorem 4.2. The estimated counting query result $\tilde{Q}(c)$ by Eq. 13 is unbiased if each user’s number of bit ‘1’ in the bit vector does not exceed $d-1$, and the estimation variance is bounded by $\frac{1}{4}n(d+1)^2$.

PROOF. According to Eq. 13, the expectation of the subset counting query result is

$$\begin{aligned} \mathbb{E}[\tilde{Q}(c)] &= (d+1) \cdot \mathbb{E}[\sum_{i=1}^n V_i^{z_i}] - n \\ &= (d+1) \left(\sum_{t=0}^{d-1} \frac{t+1}{d+1} \cdot n\pi_t + \frac{d}{d+1} n\pi_d \right) - n \\ &= n \left(\sum_{t=1}^d t\pi_t + \sum_{t=0}^d \pi_t - \pi_d \right) - n \\ &= Q(c) - n\pi_d. \end{aligned}$$

Since the number of bit ‘1’ does not exceed $d-1$, $\pi_d = 0$. Therefore, $\mathbb{E}[\tilde{Q}(c)] = Q(c)$, i.e., $\tilde{Q}(c)$ is unbiased.

As for the estimation variance, we have

$$\begin{aligned} \text{Var}[\tilde{Q}(c)] &= (d+1)^2 \text{Var}[\sum_{i=1}^n V_i^{z_i}] \\ &= (d+1)^2 \sum_{t=0}^d n\pi_t \frac{(t+1)(d-t)}{(d+1)^2} \\ &\leq (d+1)^2 \sum_{t=0}^d \frac{n\pi_t}{4} \\ &= \frac{1}{4}n(d+1)^2. \end{aligned}$$

Therefore, the estimation variance of the subset counting query is bounded by $\frac{1}{4}n(d+1)^2$. \square

4.2. Customizing CRIAD

CRIAD is generic in terms of the number of dummies and samples in each user. In this subsection, we show the customization of CRIAD to support a wide range of privacy requirements and category sizes.

4.2.1. Multiple Dummies. In Algorithm 2, only one dummy bit ‘1’ is added to each user’s bit vector. Table 4 shows the impact of m dummies on the sampling probabilities of bits ‘1’s and ‘0’s respectively. We observe that for the first $d+1-m$ cases (i.e., from π_0 to π_{d-m}), the sampling probability of bit ‘1’ (resp. 0) ranges from $\frac{m}{d+m}$ to $\frac{d}{d+m}$ (resp. from $\frac{d}{d+m}$ to $\frac{m}{d+m}$). As more dummy ‘1’s are added, the sampling probability gradually approaches 1 (resp. 0). This motivates us to confine the number of bit ‘1’s to $d-m$. As such, the probability ratio of any two sampled bits can be bounded by d/m and thus m dummies can satisfy $(\ln \frac{d}{m})$ -LDP (see Theorem 4.3 for complete proof).

TABLE 4. CASES OF BIT VECTORS WITH m DUMMY BITS ‘1’

Proportion	No. of 1	No. of 0	Pr[1]	Pr[0]
π_0	m	d	$m/(d+m)$	$d/(d+m)$
π_1	$1+m$	$d-1$	$(1+m)/(d+m)$	$(d-1)/(d+m)$
π_2	$2+m$	$d-2$	$(2+m)/(d+m)$	$(d-2)/(d+m)$
...
π_{d-m}	d	m	$d/(d+m)$	$m/(d+m)$
π_{d-m+1}	$d+1$	$m-1$	$(d+1)/(d+m)$	$(m-1)/(d+m)$
...
π_{d-1}	$d+m-1$	1	$(d+m-1)/(d+m)$	$1/(d+m)$
π_d	$d+m$	0	1	0

Then each user u_i randomly samples an index $z_i \in \{1, 2, \dots, d+m\}$ and reports the bit $V_i^{z_i}$ to the data collector. Based on the reports from all users, the estimated subset counting query result $\tilde{Q}(c)$ can be derived as

$$\tilde{Q}(c) = (d+m) \sum_{i=1}^n V_i^{z_i} - mn,$$

where the second term mn is the number of dummy ‘1’s added by all users.

4.2.2. Multiple Samples. In Algorithm 2, each user randomly samples and reports one bit to the data collector, and the overall algorithm satisfies $\ln d$ -LDP (see Theorem 4.1). However, this becomes a privacy bottleneck when the privacy budget $\epsilon > \ln d$, as the extra budget has to be wasted. CRIAD can benefit from a large privacy budget by having users report multiple samples. Note that this is different from directly applying sequential composition (i.e., Theorem 2.2) to repeatedly perform one-bit sampling multiple times, as in CRIAD, bits are sampled without replacement to cover as many data bits as possible. Table 5 shows the impact of number of samples s on the probabilities of bits ‘1’s and ‘0’s respectively, where m ($m \geq s$) dummies are added. Note that the table only shows the first $d-m+1$ cases (i.e., from π_0 to π_{d-m}), as the others are reduced to the case of π_{d-m} before sampling, in the same way as in Section 4.2.1.

TABLE 5. CASES OF BIT VECTORS WITH m DUMMY BITS ‘1’ AND s SAMPLES

Proportion	No. of 1	No. of 0	Pr[$\{1\}^s$]	Pr[$\{0\}^s$]
π_0	m	d	$\binom{m}{s}/\binom{d+m}{s}$	$\binom{d}{s}/\binom{d+m}{s}$
π_1	$1+m$	$d-1$	$\binom{m+1}{s}/\binom{d+m}{s}$	$\binom{d-1}{s}/\binom{d+m}{s}$
π_2	$2+m$	$d-2$	$\binom{m+2}{s}/\binom{d+m}{s}$	$\binom{d-2}{s}/\binom{d+m}{s}$
...
π_{d-m}	d	m	$\binom{d}{s}/\binom{d+m}{s}$	$\binom{m}{s}/\binom{d+m}{s}$

Let $z_i = \{z_i[1], z_i[2], \dots, z_i[s]\}$ denote the s indexes sampled by user u_i . Based on all users’ reports, the estimated counting query result $\tilde{Q}(c)$ can be derived as

$$\tilde{Q}(c) = \frac{d+m}{s} \sum_{i=1}^n \sum_{x=1}^s V_i^{z_i[x]} - mn.$$

4.2.3. Multiple Groups. Although m dummies can satisfy $(\ln \frac{d}{m})$ -LDP, when the category size d is large, it is difficult to satisfy a small privacy budget. To address this issue, we further propose a grouping strategy to divide a large category over $\{1, 2, \dots, d\}$ into g disjoint and equal-sized groups

$\{G_1, G_2, \dots, G_g\}$, i.e., $|G_r| = \frac{d}{g}$ and $\cup_{r=1}^g G_r = \{1, 2, \dots, d\}$. Each group becomes a new (sub)category and thus the above multi-dummy and multi-sample strategies can still work in each group. The users are also divided into g equal-sized groups $\{U_1, U_2, \dots, U_g\}$, i.e., $\frac{n}{g}$ users for each group, and each user reports s samples drawn from her bit vector with m dummy ‘1’s in her corresponding group.

Upon receiving reports from all users, the data collector first counts bit ‘1’s in each group, and then collectively derives the estimated counting query result based on the counts from g groups. Specifically, for group G_r , the count in user group U_r can be estimated as

$$\gamma_r = \frac{d+gm}{gs} \sum_{i=1}^{|U_r|} \sum_{x=1}^s V_{U_r[i]}^{z_i[x]} - m \cdot |U_r|,$$

where $U_r[i]$ denotes the i -th user in U_r , and $\sum_{i=1}^{|U_r|} \sum_{x=1}^s V_{U_r[i]}^{z_i[x]}$ is the sum of all returned samples in group U_r . Finally, the estimated counting query result becomes

$$\tilde{Q}(c) = \sum_{r=1}^g \gamma_r \cdot g = \frac{d+gm}{s} \sum_{i=1}^n \sum_{x=1}^s V_i^{z_i[x]} - nmg. \quad (14)$$

4.3. CRIAD: Putting Things Together

Algorithm 3 shows the complete CRIAD procedure with a multi-dummy, multi-sample, and multi-group strategy. As such, Algorithm 2 can be considered as a special case where $m = s = g = 1$. Given a subset counting query on the category c , the data collector first derives three parameters, namely, the number of dummies m , samples s and groups g , based on the given privacy budget ϵ and category size $d = |c|$ (Line 1), which will be elaborated by Theorem 4.7 in Section 4.4. The collector then broadcasts m , s and g to all users (Line 2). At the user side, the domain $\{1, 2, \dots, d\}$ of category c is first divided into g groups $\{G_1, G_2, \dots, G_g\}$ uniformly at random (Line 3), then each user samples a group G_r for reporting (Line 5). Each user extracts a filtered record set R_i^* from R_i with items belonging to G_r (Line 6), and then encodes it into a bit vector with length $|G_r|$ (Line 7). Then m dummy bits 1 are added to the encoded bit vector (Line 8). If the number of bit ‘0’s is fewer than m , the user needs to randomly flip some ‘1’s to ensure at least m ‘0’s (Lines 9-11). Then s indexes are randomly sampled from $\{1, 2, \dots, d+m\}$ and the user sends these sampled bits to the data collector (Lines 12-13). Finally, based on all the reports from users, the collector estimates the subset counting query result by Eq. 14.

4.4. Privacy and Utility Analysis of CRIAD

In this subsection, we will address the pending problem of choosing parameters m , s and g , given category c and privacy budget ϵ . We will first provide privacy and utility analysis in Theorems 4.3 to 4.6, based on which we derive the optimal setting for three parameters in Theorem 4.7.

Algorithm 3 Workflow of CRIAD

Input: A category c
All users' item set $\{R_i, R_i, \dots, R_n\}$
Privacy budgets ϵ_1 and ϵ_2 for count and mean estimation

Output: Estimated subset counting query result $\tilde{Q}(c)$

Procedure:
//Collector side
1: Set parameters: $m, s, g \leftarrow \text{ParaSelect}(d, \epsilon)$ by Theorem 4.7
2: Broadcast m, s and g to all users
//User side
3: Divide the full domain $\{1, 2, \dots, d\}$ of category c into g groups $\{G_1, G_2, \dots, G_g\}$ uniformly at random
4: **for** each user u_i ($1 \leq i \leq n$) **do**
5: Randomly sample a group G_r for $r \in \{1, 2, \dots, g\}$
6: Extract the item set R_i^* from R_i with items belonging to G_r
7: Encode R_i^* into a binary vector $V_i = \{0, 1\}^{|G_r|}$ by Eq. 3
8: Add m dummy bits (i.e., $\{1\}^m$) to V_i
9: Set c' as the number of bit '0's in V_i
10: **if** $c' < m$ **then**
11: Randomly flip $m - c'$ bits '1' in V_i
12: Randomly sample s indices $z_i = \{z_i[1], \dots, z_i[s]\}$ from $\{1, 2, \dots, d + m\}$
13: Send $V_i^{z_i}$ to the data collector
//Collector side
14: Estimate the counting query result $\tilde{Q}(c)$ by Eq. 14
15: **return** Query result $\tilde{Q}(c)$

Theorem 4.3. With m dummies, s samples and g groups, CRIAD satisfies $\ln \left(\binom{d/g}{s} / \binom{m}{s} \right)$ -LDP.

PROOF. Recall that in Table 4, the last m cases (i.e., $\pi_{d-m+1}, \pi_{d-m+2}, \dots, \pi_d$) are reduced to π_{d-m} by suppressing the number of bit '1's in the encoded bit vector. Therefore, for any two filtered item sets R_i^* and R_j^* regarding a subset, and any bit $b \in \{0, 1\}$ reported by CRIAD, we have

$$\begin{aligned} \frac{\Pr[\text{CRIAD}(R_i^*) = b]}{\Pr[\text{CRIAD}(R_j^*) = b]} &\leq \frac{\Pr[\text{CRIAD}(V_i \in \pi_{d-m}) = 1]}{\Pr[\text{CRIAD}(V_j \in \pi_0) = 1]} \\ &= \frac{d/(d+m)}{m/(d+m)} = \frac{d}{m}. \end{aligned}$$

Then with increasing $s > 1$, let $\mathbf{b} = \{0, 1\}^s$ denote a bit vector of length s reported by a user. Recall that in Table 5, the above ratio further becomes

$$\begin{aligned} \frac{\Pr[\text{CRIAD}(R_i^*) = \mathbf{b}]}{\Pr[\text{CRIAD}(R_j^*) = \mathbf{b}]} &\leq \frac{\Pr[\text{CRIAD}(V_i \in \pi_{d-m}) = \{1\}^s]}{\Pr[\text{CRIAD}(V_j \in \pi_0) = \{1\}^s]} \\ &= \frac{\binom{d}{s} / \binom{d+m}{s}}{\binom{m}{s} / \binom{d+m}{s}} = \frac{\binom{d}{s}}{\binom{m}{s}}. \end{aligned}$$

Then with increasing $g > 1$, the group size changes from $d + m$ to $\frac{d}{g} + m$. So the above ratio becomes $\binom{d/g}{s} / \binom{m}{s}$. Therefore, CRIAD satisfies ϵ -LDP, where $\epsilon = \ln \left(\binom{d/g}{s} / \binom{m}{s} \right)$. \square

Theorem 4.4. The estimated count $\tilde{Q}(c)$ by Algorithm 3 is unbiased if the number of bits '1's in the encoded bit vector does not exceed $d/g - m$.

PROOF. By the grouping strategy, each encoded bit vector is split into g sub-vectors. Therefore, for an encoded vector

V_i which contains t bit '1's, the length of each sub-vector is d/g and the expected count of bit 1 is t/g . In Eq. 14, $\sum_{x=1}^s V_i^{z_i[x]}$ is the count of bit '1's in s samples reported by the user u_i in a group. Similar to Theorem 4.2, if the number of bit '1's in that group does not exceed $d/g - m$, the count $\sum_{x=1}^s V_i^{z_i[x]}$ from each user is an unbiased estimation of the true count in that group. Hence, in the case of π_t , the expectation of the count $\sum_{x=1}^s V_i^{z_i[x]}$ is

$$\mathbb{E}\left[\sum_{x=1}^s V_i^{z_i[x]} \mid \pi_t\right] = \frac{s(t/g + m)}{d/g + m}.$$

Therefore,

$$\begin{aligned} \mathbb{E}\left[\sum_{x=1}^s V_i^{z_i[x]}\right] &= \sum_{t=0}^d \pi_t \cdot \mathbb{E}\left[\sum_{x=1}^s V_i^{z_i[x]} \mid \pi_t\right] \\ &= \sum_{t=0}^d \frac{s(t/g + m)}{d/g + m} \pi_t. \end{aligned}$$

By Eq. 14, the expectation of the estimated count is

$$\begin{aligned} \mathbb{E}[\tilde{Q}(c)] &= \frac{d + gm}{s} \sum_{i=1}^n \mathbb{E}\left[\sum_{x=1}^s V_i^{z_i[x]}\right] - nmg \\ &= \frac{d + gm}{s} \sum_{i=1}^n \sum_{t=0}^d \frac{s(t/g + m)}{d/g + m} \pi_t - nmg \\ &= n \sum_{t=0}^d t \pi_t = Q(c). \end{aligned}$$

Therefore, $\mathbb{E}[\tilde{Q}(c)] = Q(c)$. \square

Theorem 4.5. The expected bias error for $\tilde{Q}(c)$ by Algorithm 3 is $n \sum_{t=d-mg}^d \pi_t f(t)$, where $f(t) = t - d + mg$.

PROOF. By the grouping strategy, each encoded bit vector is split into g sub-vectors. Therefore, for an encoded vector V_i which contains t bit '1's, the length of each sub-vector is d/g and the expected count of bit 1 is t/g .

For the case where $t > d - mg$, since we suppress $\pi_{t/g}$ cases to $\pi_{d/g-m}$ case by randomly flipping one bit 1 to 0 to refrain from adding a dummy bit '0', the expected bias error after aggregation is

$$n \sum_{t=d-mg}^d \pi_t \left(\frac{t}{g} - \frac{d}{g} + m \right) g = n \sum_{t=d-mg}^d \pi_t f(t),$$

where $f(t) = t - d + mg$. Conversely, when $t \leq d - mg$, the expected bias error after aggregation is 0. In summary, the expected bias error for $\tilde{Q}(c)$ by Algorithm 3 is $n \sum_{t=d-mg}^d \pi_t f(t)$. \square

Theorem 4.6. With m dummies, s samples and g groups, the variance of the estimated count by CRIAD is bounded by $\frac{n(d+gm)^2}{4s}$.

PROOF. With g groups, m dummies and s samples, each user first selects a group uniformly at random, and then selects s sample bits in that group. From the perspective of sampling variance, this is equivalent to directly selecting s samples from the whole domain of length $d + gm$. For any encoded bit vector belonging to the case of $\pi_t \in \{\pi_0, \pi_1, \dots, \pi_{d-m}\}$, the whole domain consists of $t + gm$

bit ‘1’s and $d - t$ bit ‘0’s. As such, the variance of the estimated count $\tilde{Q}(c)$ by Eq. 14 is

$$\begin{aligned} \text{Var}[\tilde{Q}(c)] &= \frac{(d + gm)^2}{s^2} \text{Var}\left[\sum_{i=1}^n \sum_{x=1}^s V_i^{z_i[x]}\right] \\ &= \frac{n(d + gm)^2}{s^2} \sum_{t=0}^d \pi_t \text{Var}\left[\sum_{x=1}^s V_i^{z_i[x]} \mid \pi_t\right] \\ &\lesssim \frac{n(d + gm)^2}{s} \cdot \frac{(t + gm)(d - t)}{(d + gm)^2} \\ &< \frac{n(d + gm)^2}{4s}. \end{aligned}$$

Therefore, the variance of the estimated count by CRIAD is bounded by $\frac{n(d+gm)^2}{4s}$. \square

Finally, Theorem 4.7 below derives an optimal approximation of m , s and g in terms of the estimation variance.

Theorem 4.7. Given privacy budget ϵ , the optimal m , s , and $g \in \mathbb{Z}^+$ of CRIAD can be approximated by

$$\begin{aligned} m, s, g &= \arg \min_{m, s, g} \mathbb{E}[(\tilde{Q}(c) - Q(c))^2] \\ &= \arg \min_{m, s, g} \left(n \frac{(d+gm)^2}{4s} + \left(n \sum_{t=d-mg}^d \pi_t f(t) \right)^2 \right) \quad (15) \\ &\quad s.t., \quad \epsilon \geq \ln \left(\binom{d/g}{s} / \binom{m}{s} \right), \\ &\quad 1 < s \leq m \leq d/g. \end{aligned}$$

PROOF. The optimal parameter setting of m , s and g can be derived by minimizing the expected squared error, i.e., $\arg \min_{m, s, g} \mathbb{E}[(\tilde{Q}(c) - Q(c))^2]$. Note that the first term in Eq. 15 is the overall variance derived from Theorem 4.6, and the second term is the expected bias error due to suppression derived from Theorem 4.5. As for the two conditions, the first is due to Theorem 4.3, where CRIAD satisfies $\ln \left(\binom{d/g}{s} / \binom{m}{s} \right)$ -LDP. The second is because even when the true bit vector has all bit ‘0’s, the number of bit ‘1’ is still m as m dummy ‘1’s are added. And $m \leq d/g$ is because in each group we confine the number of bit ‘1’s to $d/g - m$. \square

As for π_t , we can randomly allocate a portion of users to estimate the distribution, where each user only needs to report an integer. Then we conduct a greedy search to enumerate all combinations of m , s and g , calculate their resulted variance by Eq. 15, and select the optimal one with the lowest variance. Note that $m, s, g \in \mathbb{Z}^+$, and s and g are typically small integers even for a large domain. As such, the number of combinations will not be too large, resulting in an efficient search.

5. Experimental Evaluation

In this section, we evaluate the performance of our proposed solution CRIAD to validate its effectiveness in answering subset counting queries.

5.1. Experiment Setup

Datasets. We conduct experiments on three real datasets.

- *Kosarak*¹ contains click-stream dataset from a Hungarian on-line news portal, which contains 990,002 users. The item domain size is 41,270.
- *OnlineRetail*² is transformed from the Online Retail dataset, which contains 541,909 users. The item domain is 2,603.
- *POS* [38] is a dataset on merchant transactions, which contains 515,595 users. The item domain is 1,657.

Competitors. We compare our proposed solution CRIAD with existing value-perturbation LDP solutions, namely Numerical Value Perturbation (NVP) and Padding-and-Sampling Perturbation (PSP) introduced in Section 2.3. Additionally, to demonstrate the superiority of randomized index over randomized response (RR), we also implement RR, which replaces Step ② of Figure 1 by sampling a bit from the encoded vector produced by Eq. 3, perturbing it by RR [33], and reporting the sanitized bit for aggregation.

Parameter Setting. For NVP, we implement it by integrating three state-of-the-art perturbation mechanisms, namely Laplace Mechanism (LM), Piecewise Mechanism (PM) and Square Wave mechanism (SW), which are denoted by NVP-LM, NVP-PM, and NVP-SW, respectively. For PSP, we follow the existing work [31] and use the 90th percentile of the users’ itemset sizes of the query category as the padding length η . To estimate this value, 10% of users report the length through Optimal Local Hashing (OLH) [29] in advance. For CRIAD, to derive the optimal parameter setting of m , s and g by Theorem 4.7, we allocate 10% of the users randomly to estimate the distribution of π_t via SW mechanism [22]. Note that this is for parameter selection only, and will not inject any noise to the original data.

Experiment Design. We design three sets of experiments to evaluate different methods. The first set compares the overall performance of CRIAD and its competitors across three datasets by varying the privacy budget. The second set studies the impact of category sizes on the performance of different methods. The third set validates the effectiveness of Theorem 4.7 for setting the optimal parameters for the numbers of dummies m , samples s and groups g .

Metrics. To evaluate the result accuracy, we employ the Mean Relative Error [24], which quantifies the average difference between the estimated result $\tilde{Q}(c)$ and the ground truth $Q(c)$. Formally,

$$MRE(c) = \frac{1}{N} \sum_N \frac{|\tilde{Q}(c) - Q(c)|}{Q(c)} \quad (16)$$

where N represents the number of trials for each experiment. In our study, N is set to 100.

We conduct experiments using Python 3.11.5 and the Numpy 1.24.3 library on a desktop equipped with an Intel Core i5-13400F 1.50 GHz CPU and 64GB of RAM, running Windows 11.

1. <http://fimi.uantwerpen.be/data/>
2. <https://archive.ics.uci.edu/ml/datasets/>

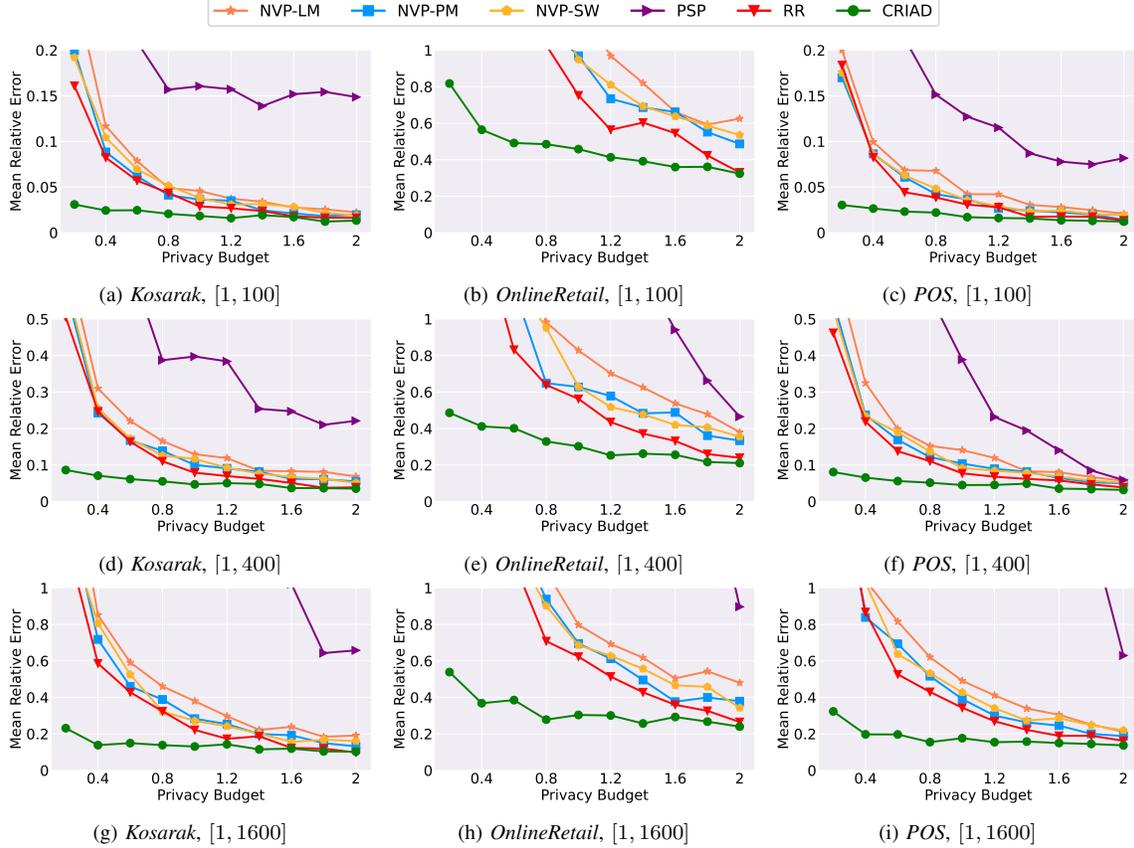


Figure 2. Overall performance on real-world datasets with varying privacy budgets.

5.2. Overall Results

In this subsection, we investigate the overall performance of different methods across three real-world datasets with varying privacy budgets. On each dataset, we evaluate three subset counting queries with categories set to $[1, 100]$, $[1, 400]$, and $[1, 1600]$, respectively. The privacy budget varies from 0.2 to 2.0, with a step size of 0.2. Figure 2 shows the results, where MRE of all methods decreases as the privacy budget increases. Overall, CRIAD performs the best, followed by RR, NVP and finally PSP. The gap between CRIAD and the competitors is particularly significant for smaller privacy budget (e.g., $\epsilon < 1.2$). This is because, a small privacy budget potentially leads to larger estimation variance, resulting in higher MRE. However, CRIAD is capable of selecting optimal parameters (e.g., by reducing the number of dummy items) to minimize the variance according to the given privacy budget, thanks to Theorem 4.7. In particular, with a small privacy budget of $\epsilon = 0.1$, CRIAD outperforms the second-best method (i.e., RR) by a factor of at least 5. On the other hand, we found PSP is much less effective compared to other methods. This is because, in each dataset, despite the large size of the category, the number of items owned by each user is small, which can lead to a large η and ultimately result in large estimation error.

5.3. Impact of Category Size

In this subsection, we study the impact of the category size on the performance of different methods, while fixing the privacy budget at $\epsilon = 1$. Specifically, the category size varies from 100 to 1600, resulting in the category domain from $[1, 100]$ to $[1, 1600]$ respectively. Figure 3 presents the results across various category sizes, where CRIAD consistently delivers the best performance in all cases, while PSP always exhibits the highest MRE, which even exceeds 1 in some cases, especially on *OnlineRetail* dataset.

On the other hand, as shown in Figures 3(a) and 3(c), we observe that the MRE of different methods increases with the category size on both *Kosarak* and *POS* datasets. This can be attributed to the distribution of items in both datasets, where the majority of items are concentrated in smaller category indexes. Consequently, as the category size increases, the number of items owned by users within the query category remains relatively stable, which ultimately leads to worse results. In *OnlineRetail* dataset, the items are more evenly distributed across users, resulting in a relatively consistent MRE across different category sizes. Overall, the performance gap between CRIAD and the other methods becomes more pronounced as the category size increases, which suggests that CRIAD scales effectively with larger category sizes.

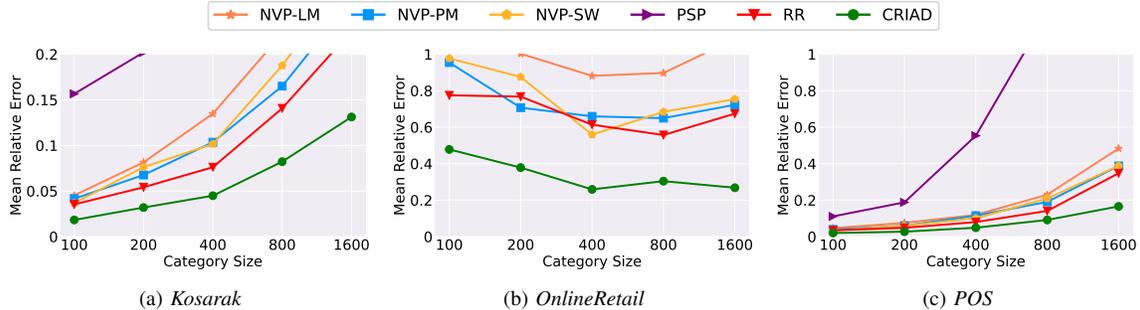


Figure 3. MRE of different methods with varying category size ($\epsilon = 1$).

5.4. Effectiveness of the Optimal Parameter Setting

In this subsection, we validate the effectiveness of Theorem 4.7 for determining the optimal parameters, in terms of the numbers of dummies m , samples s and groups g . For this study, we fix the privacy budget at $\epsilon = 1$ and set the category size $d = 400$. Note that the estimation of π_t in Theorem 4.7 introduces variability in parameter selection. Therefore, we conduct 100 independent experiments across three datasets. Table 6 presents the three most frequent (m, s, g) combinations and their occurrences out of 100. We observe that, under the condition of $\epsilon = 1$ and $d = 400$, our strategy consistently sets $(m, s, g) = (148, 1, 1)$ with the highest probability.

TABLE 6. OCCURRENCES OF THE THREE MOST FREQUENT PARAMETER COMBINATIONS OUT OF 100 EXPERIMENTS

(m, s, g)	Kosarak	OnlineRetail	POS
(148,1,1)	60	63	58
(244,2,1)	9	4	3
(288,3,1)	2	2	5

Then we fix m at 148, 244, and 288 respectively, and enumerate all feasible (m, s, g) combinations that satisfy the given privacy budget. The results of MRE of each combination over three datasets are presented in Table 7, where the results of the selected parameter combinations by Theorem 4.7 is shown as bold (i.e., parameter combinations in Table 6). Overall, we observe that those selected (m, s, g) combinations yield a relatively lower relative error than most of the cases. Although the combination (148, 3, 2) yields the smallest relative error, the difference is not substantial compared to the (148, 1, 1) case, particularly on the *Kosarak* and *OnlineRetail* datasets. It is also noteworthy that when $g = 2$, the computational cost almost doubles. Additionally, we observe that for fixed values of m and g , a larger s results in a decreasing MRE. This trend is explained by Theorem 4.7 that, while satisfying ϵ -LDP, a larger s corresponds to a smaller expected squared error.

To further investigate the effectiveness of parameter setting, we then fix $s = 1$ or $g = 1$ respectively, and randomly select 100 combinations of (m, s, g) that satisfy the given privacy budget $\epsilon = 1$. The results are presented in Figure 4, where combinations with larger MRE than the case

TABLE 7. RESULTS OF MRE OF PARAMETER COMBINATIONS OVER THREE DATASET, WITH $d = 400$, $\epsilon = 1$.

m	(s, g)	Kosarak	OnlineRetail	POS
$m = 148$	(1,1)	0.052	0.343	0.060
	(1,2)	0.078	0.593	0.074
	(2,2)	0.052	0.341	0.045
	(3,2)	0.050	0.337	0.043
$m = 244$	(1,1)	0.093	0.596	0.072
	(2,1)	0.369	0.593	0.070
$m = 288$	(1,1)	0.109	0.468	0.082
	(2,1)	0.067	0.273	0.074
	(3,1)	0.040	0.199	0.048

of (148, 1, 1) are marked in red, while those with smaller MRE are in blue.

As shown in Figures 4(a), (c), and (e), we find that when fixing the number of samples at $s = 1$, setting a smaller number of groups g tends to achieve a lower MRE. This is to ensure more valid samples for the estimation within each group. As g increases, the number of dummy items m is significantly reduced, since it is constrained by the decreasing group size. Nevertheless, the case of (148, 1, 1) selected by Theorem 4.7 still achieves lower MRE than most of parameter combinations.

Figure 4(b) shows that when $m > 300$ and $s < 16$, the corresponding parameter combinations outperform (148, 1, 1) on *Kosarak* dataset. Additionally, as evidenced by Table 6, besides the (148, 1, 1) combination, Theorem 4.7 also frequently sets parameters within this range with a high probability on *Kosarak*. Figures 4(d) and (f) demonstrate that on *OnlineRetail* and *POS* datasets, parameter combinations with $m > 350$ and $s < 40$ also achieve lower MRE than (148, 1, 1). Similarly, Table 6 indicates that, besides the combination (148, 1, 1), Theorem 4.7 frequently sets parameters within this specified range with considerable probability. Overall, although Theorem 4.7 does not always determine the optimal setting when g is fixed, it reliably suggests better-than-average parameter configurations and frequently identifies optimal combinations.

6. Related Work

Differential privacy was first proposed in the centralized setting [13], [14], [16]. To avoid relying on a trusted data collector, local differential privacy (LDP) was proposed to let each user perturb her data locally [9]. In

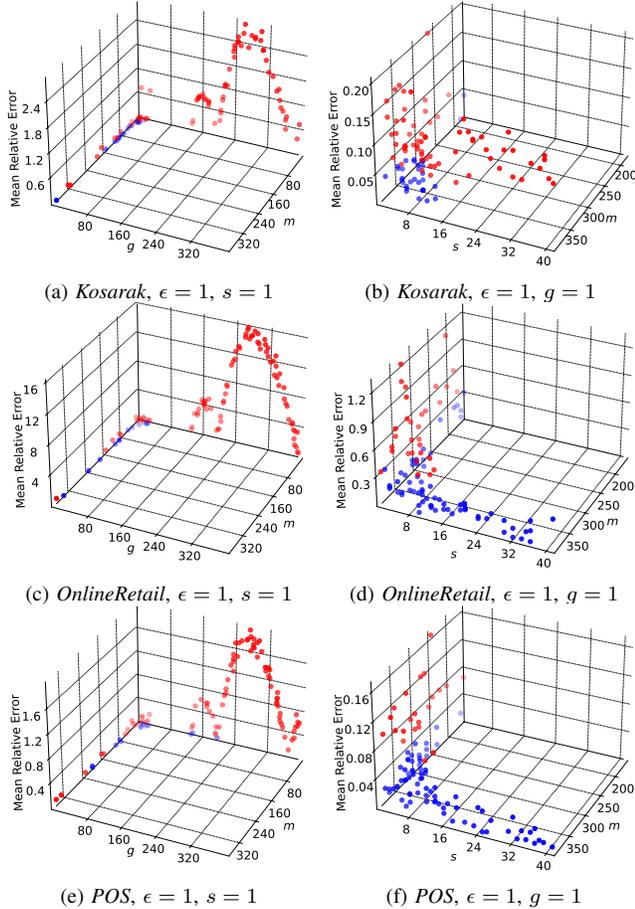


Figure 4. MRE of 100 random parameter combinations of (m, s, g) .

the literature, many LDP techniques have been proposed for various statistical collection tasks, such as frequency of categorical values [2], [15], [19], [29], and mean of numerical values [6], [27]. Recently, the research focus in LDP has been shifted to more complex tasks, such as heavy hitter identification [3], [4], itemset mining [31], marginal release [5], [37], time series data analysis [1], [30], [35], and data poisoning attacks [8], [18]. In what follows, we review existing LDP works that are relevant to ours, namely LDP mechanisms for categorical data, numerical data and set-value data, respectively.

LDP Mechanisms for Categorical Data. There are a line of LDP work developed for categorical value perturbation. Randomized Response (RR) [33] is the most straightforward mechanism for binary value, and a generalized form of RR (a.k.a., kRR) [19] is then proposed to deal with a value with domain size $d > 2$. To alleviate large perturbation noise along with the increasing domain size, Wang *et al.* propose Optimized Unary Encoding (OUE) [29] which achieves better utility. Besides, several perturbation protocols are also proposed in the literature, including RAP-POR [15], SHist [2] and subset selection [34].

LDP Mechanisms for Numerical Data. Designing LDP protocols for numerical values also attracts great attention

from the researchers. As with centralized DP, Laplace Mechanism [13] can be adopted in the local setting. On the other hand, Duchi *et al.* propose a solution for mean estimation over numerical values [10]. To address computation and space complexity of it, an improved method [11] is then proposed to perturb any numerical input into a binary output according to a certain probability. Ding *et al.* [6] also propose mechanisms to continuously collect telemetry data. More recently, Wang *et al.* [27] propose Piecewise Mechanism (PM) to improve estimation accuracy, and Li *et al.* [22] propose Square-wave (SW) mechanism to support numerical distribution estimation.

Frequency Estimation over Set-value Data. As a relevant problem to this paper, frequency estimation over set-value data has been widely studied in the context of LDP. LDPMiner [25], together with a *padding-and-sampling* protocol, is the first solution for this problem. Wang *et al.* [31] further study the privacy amplification effect of padding-and-sampling protocol with kRR, and use it for frequent itemset mining. Wang *et al.* [28] propose PrivSet to estimate both item distribution and set size distribution over set-value data. Besides, some other work also consider the setting where each user possesses a set of key-value pairs [17], [36]. Nevertheless, existing work over set-value setting all consider a full-domain item distribution or heavy hitter identification [7], [32], both of which are different from the problem studied in this work.

7. Conclusion

In this work, we study the problem of answering subset counting queries on set-value data. Unlike existing studies that perturb the original values to ensure an LDP guarantee, we propose an alternative approach that leverages the deniability of randomized indexes without perturbing the values. Our design, named CRIAD, satisfies a rigorous LDP guarantee while achieving higher accuracy than all existing methods. Furthermore, by integrating a multi-dummy, multi-sample, and multi-group strategy, CRIAD is optimized into a fully scalable solution that offers flexibility across various privacy requirements and domain sizes.

As for further work, we plan to extend CRIAD to more complex scenarios, such as the federated setting, where queries involve users across multiple parties, and item exhibit heterogeneity.

References

- [1] E. Bao, Y. Yang, X. Xiao, and B. Ding. CGM: an enhanced mechanism for streaming data collection with local differential privacy. *PVLDB*, 14(11):2258–2270, 2021.
- [2] R. Bassily and A. Smith. Local, private, efficient protocols for succinct histograms. In *STOC*, pages 127–135. ACM, 2015.
- [3] R. Bassily, U. Stemmer, A. G. Thakurta, et al. Practical locally private heavy hitters. In *NIPS*, pages 2285–2293, 2017.
- [4] M. Bun, J. Nelson, and U. Stemmer. Heavy hitters and the structure of local privacy. In *PODS*, pages 435–447. ACM, 2018.

- [5] G. Cormode, T. Kulkarni, and D. Srivastava. Marginal release under local differential privacy. In *SIGMOD*, pages 131–146. ACM, 2018.
- [6] B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. In *NIPS*, pages 3574–3583, 2017.
- [7] R. Du, Q. Ye, Y. Fu, H. Hu, and K. Huang. Top-k discovery under local differential privacy: An adaptive sampling approach. *IEEE Transactions on Dependable and Secure Computing*, 2024.
- [8] R. Du, Q. Ye, Y. Fu, H. Hu, J. Li, C. Fang, and J. Shi. Differential aggregation against general colluding attackers. In *ICDE*, pages 2180–2193. IEEE, 2023.
- [9] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *FOCS*, pages 429–438. IEEE, 2013.
- [10] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Privacy aware learning. *Journal of the ACM*, 61(6):1–57, 2014.
- [11] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521):182–201, 2018.
- [12] C. Dwork. Differential privacy. In *ICALP*, pages 1–12. Springer, 2006.
- [13] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284. Springer, 2006.
- [14] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.
- [15] Ú. Erlingsson, V. Pihur, and A. Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, pages 1054–1067. ACM, 2014.
- [16] J. Fu, Q. Ye, H. Hu, Z. Chen, L. Wang, K. Wang, and X. Ran. Dpsur: accelerating differentially private stochastic gradient descent using selective update and release. *Proceedings of the VLDB Endowment*, 17(6):1200–1213, 2024.
- [17] 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*, 2020.
- [18] K. Huang, G. Ouyang, Q. Ye, H. Hu, B. Zheng, X. Zhao, R. Zhang, and X. Zhou. LDPGuard: Defenses against data poisoning attacks to local differential privacy protocols. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3195–3209, 2024.
- [19] P. Kairouz, S. Oh, and P. Viswanath. Extremal mechanisms for local differential privacy. In *NIPS*, pages 2879–2887, 2014.
- [20] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.
- [21] N. Li, M. Lyu, D. Su, and W. Yang. Differential privacy: From theory to practice. *Synthesis Lectures on Information Security, Privacy, & Trust*, 8(4):1–138, 2016.
- [22] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škorić. Estimating numerical distributions under local differential privacy. In *SIGMOD*, pages 621–635, 2020.
- [23] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD*, pages 19–30. ACM, 2009.
- [24] A. M. Mood. Introduction to the theory of statistics. 1950.
- [25] 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 *CCS*, pages 192–203. ACM, 2016.
- [26] A. G. Thakurta, A. H. Vyrros, U. S. Vaishampayan, G. Kapoor, J. Freudiger, V. R. Sridhar, and D. Davidson. Learning new words, Mar. 14 2017. US Patent 9,594,741.
- [27] 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 *ICDE*, 2019.
- [28] S. Wang, L. Huang, Y. Nie, P. Wang, H. Xu, and W. Yang. PrivSet: Set-valued data analyses with locale differential privacy. In *INFOCOM*, pages 1088–1096. IEEE, 2018.
- [29] T. Wang, J. Blocki, N. Li, and S. Jha. Locally differentially private protocols for frequency estimation. In *USENIX Security*, pages 729–745, 2017.
- [30] T. Wang, J. Q. Chen, Z. Zhang, D. Su, Y. Cheng, Z. Li, N. Li, and S. Jha. Continuous release of data streams under both centralized and local differential privacy. In *CCS*, pages 1237–1253, 2021.
- [31] T. Wang, N. Li, and S. Jha. Locally differentially private frequent itemset mining. In *S&P*, pages 127–143. IEEE, 2018.
- [32] T. Wang, N. Li, and S. Jha. Locally differentially private heavy hitter identification. *IEEE Transactions on Dependable and Secure Computing*, 18(2):982–993, 2019.
- [33] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.
- [34] M. Ye and A. Barg. Optimal schemes for discrete distribution estimation under locally differential privacy. *IEEE Transactions on Information Theory*, 64(8):5662–5676, 2018.
- [35] Q. Ye, H. Hu, K. Huang, M. H. Au, and Q. Xue. Stateful switch: Optimized time series release with local differential privacy. In *INFOCOM*. IEEE, 2023.
- [36] Q. Ye, H. Hu, X. Meng, and H. Zheng. PrivKV: Key-value data collection with local differential privacy. In *S&P*, pages 317–331. IEEE, 2019.
- [37] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen. CALM: Consistent adaptive local marginal for marginal release under local differential privacy. In *CCS*, pages 212–229. ACM, 2018.
- [38] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In *SIGKDD*, pages 401–406, 2001.