Mathematical Biosciences and Engineering

MBE, 18(3): 2733–2763.
DOI: 10.3934/mbe.2021139
Received: 01 February 2021
Accepted: 09 March 2021
Published: 22 March 2021

http://www.aimspress.com/journal/MBE

Research article

Set-valued data collection with local differential privacy based on category hierarchy

Jia Ouyang[1], Yinyin Xiao[1,*], Shaopeng Liu[2], Zhenghong Xiao[2] and Xiuxiu Liao[2]

[1] School of Cyber Security, Guangdong Polytechnic Normal University, Guangzhou 510665, China
[2] College of Computer Science, Guangdong Polytechnic Normal University, Guangzhou 510665, China

* **Correspondence:** Email: gsxyy@gpnu.edu.cn; Tel: +8613430337590.

Abstract: Set-valued data is extremely important and widely used in sensor technology and application. Recently, privacy protection for set-valued data under differential privacy (DP) has become a research hotspot. However, the DP model assumes that the data center is trustworthy, consequently, increasingly attention has been paid to the application of the local differential privacy model (LDP) for set-valued data. Constrained by the local differential privacy model, most methods randomly respond to the subset of set-valued data, and the data collector conducts statistics on the received data. There are two main problems with this kind of method: one is that the utility function used in the random response loses too much information; the other is that the privacy protection of the set-valued data category is usually ignored. To solve these problems, this paper proposes a set-valued data collection method (SetLDP) based on the category hierarchy under the local differential privacy model. The core idea is to first make a random response to the existence of the category, continue to disturb the item count if the category exists, and finally randomly respond to a candidate itemset based on the new utility function. Theory analysis and experimental results show that the SetLDP can not only preserve more information, but also protect the category private information in set-valued data.

Keywords: privacy preservation; set-valued data; local differential privacy; utility function; data collection

1. Introduction

Set-valued data is widely used in online retailers, sensor technology and application, such as: shopping basket analysis, user behavior analysis, trajectory analysis etc. With the rapid development of big data applications, massive set-valued data containing rich knowledge has been collected. However, the collection and analysis of set-valued data will lead to the disclosure of private information, which will have serious consequences on individuals [1,2]. Set-valued data collection has been a research hotspot recently [3–7]. Most works assume that the data collector is trustworthy, and the user sends real data to the data center. The collector publishes the data under DP for data analysis or query tasks. Although the data collectors claim that the sensitive information of users will not to be compromised, users' privacy cannot be guaranteed because the data can be used for profit. Unlike the centralized differential privacy model, LDP is aimed at the untrusted third-party data collectors. With LDP, the user independently perturbs the data with random response technology at the user side before the data is collected.

There are many works based on LDP for set-valued data, and they aim to effectively protect the content and length of the set-valued data. These works generally deal with the set-valued data with equal length first, and then randomly respond to a subset of the set-valued data to send to the data center. However, there are two main problems in these works.

Firstly, the utility function adopted in the random response method focuses on the information of a single item and ignores the relationship between items, resulting in too much information loss. Secondly, category hierarchy privacy is frequently ignored. As shown in Figure 1, user u has set-valued data of {grape, durian, pregnancy test stick}, each of these items belongs to a certain category. For example, the category of grape is fruit, while the pregnancy test stick is family planning. For users, the family planning information belongs to personal private information, which needs to be disturbed for privacy protection. To the best of our knowledge, there are relatively few works about privacy protection for category hierarchy, which is of great theoretical and practical significance.

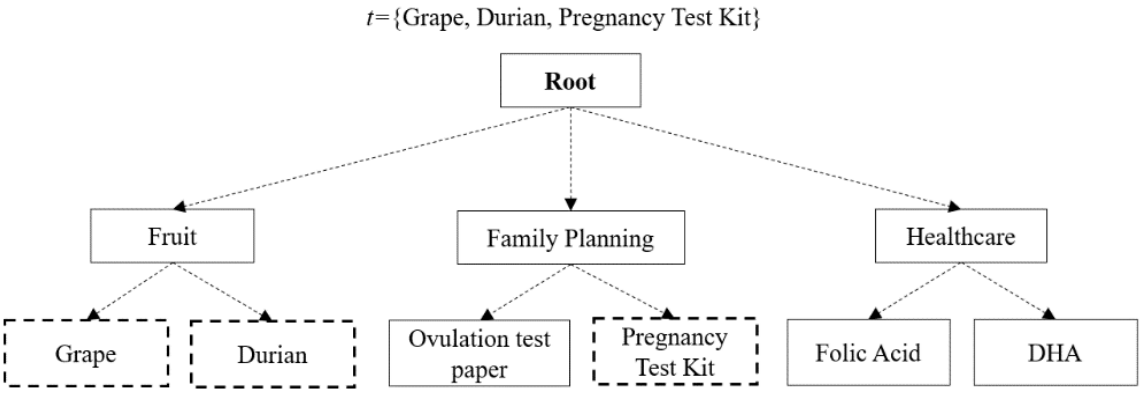

Figure 1. Category hierarchy.

Based on LDP, we propose a novel algorithm, called the category hierarchy-oriented method (SetLDP), for collecting set-valued data from the user. The user counts the number of items in each category of set-valued data owned locally; this is called the value count. There are four kinds of random responses about the existence of a category: exist -> exist, exist ->not exist, not exist -> exist, and not exist -> not exist. If the category exists after perturbation, then the value count is disturbed

via the Laplace or exponential mechanism; otherwise, let the value count be 0. Finally, the exponential mechanism is used to randomly sample an itemset from the candidate set under each category, and then the selected itemset is spliced into complete set-valued data for all category to send to the data collector. The overall process of the SetLDP is described in Session 3. Our contributions to the literature are as follows:

1) We randomly respond to the existence of an item category so that the user can protect the private information of category hierarchy;

2) We design a new utility function to disturb (based on the Laplace mechanism) or randomly respond to (based on the exponential mechanism) the value count in each category to protect the length information of data;

3) We randomly sample an itemset based on the exponential mechanism to effectively guarantee the utility of set-valued data and protect the content information of data.

The remainder of the paper is organized as follows. In Section 2, we review other related works. In Section 3, we introduce some basic knowledge and assumptions, and formally define the problem. Our SetLDP method is described in Section 4, and then we formally analyze the error bound and the privacy guarantee in Section 5. Section 6 presents the experimental results with the generated data to demonstrate the performance of SetLDP. Lastly, we provide some conclusions in Section 7.

2. Related Work

This section briefly introduces the existing privacy protection methods of set-valued data that have been widely studied. These methods can be classified into two types according to the credibility of the collector: centralized methods and localized methods. Centralized methods can be divided into traditional *k*-anonymous privacy models and differential privacy models, while localization methods can be divided into local differential privacy models and compressed local differential privacy models.

2.1. Centralization

Since the data center is trustworthy in the centralized environment, the user can directly send the real data to the server. Before publishing statistics and big data analysis, the data will be randomized first based on the privacy model, where *k*-anonymity [8] and *l*-diversity [9] are typical representatives. Previous studies have provided give a series of set-valued data privacy protection methods [10–12] based on the *k*-anonymity privacy model, which mainly assumes that the attacker's background knowledge is very rich, and artificially divides items into sensitive and non-sensitive items.

Because the *k*-anonymity privacy model makes too many assumptions about the attacker's background knowledge, which eliminates and generalizes a lot of information, there is a lack of privacy and utility of high-dimensional set-valued data for privacy protection. Differential privacy has strong privacy and high efficiency, and Chen et al. proposed a differential privacy publishing method for trajectory data based on real trajectory data in Montreal, Canada [10]. OuYangjia et al. constructed a complete trie tree with an item from the set-valued database, and based on the compressed sensing technology, the noise satisfying the differential privacy constraint was added to the tree. The frequent itemset mining task was carried out on the noisy tree, thereby guaranteeing the utility of the data [11]. For privacy protection in a set-valued data-publishing scenario under a distributed structure, the optimization objective of utility has been combined with the differential privacy constraint [12]; based on global and local data, two solutions were designed to solve the

distributed nonlinear programming model safely.

2.2. Localization

The centralized privacy protection method assumes that the data center is trustworthy, but in reality the data in the server will inevitably leak. In contrast, the localization method based on local differential privacy [13,14] assumes that the data center is untrustworthy, and the user perturbs the data randomly and locally [15]. Therefore, the data collected by the collector is not real, but lots of statistical information is retained. Random response technology is commonly used in the collection of set-valued data. For personalized privacy requirements, random response technology has been applied to frequent itemset mining [16]. Moreover, an enhancement mechanism has been proposed to ensure that the ratio of the priori probability to the posteriori probability of an attacker's knowledge about the private information is within a specified range [17].

The local differential privacy model has been successfully used in many industrial applications [18], such as Google's Chrome browser [13,14], Apple's iOS, and Microsoft's Windows 10 OS [19], which are all based on LDP to collect the user's private data and conduct statistics and analysis (mean calculations, histogram statistics, etc.)

In academia, LDP is widely used in the collection of different types of data, such as category data [20,21], location data [22,23], and set-valued data [24–27]. The idea from [24] considers that the method from [26] has a privacy budget for each item and adds too much noise to the data, and [24] proposed an efficient candidate set sampling algorithm based on the exponential mechanism, in which the sampling probability of each candidate set was related to the intersection of the original data. Nevertheless, we find that the utility proposed in [24] only considers whether there exists an intersection between the candidate set and the original data, if we only consider an intersection, we become faced with too much information lost and less utility. We consider that the number of intersections should also be taken into account.

The concept of geo indistinguishability was proposed in [22] to provide deidentification of locations. Similarly, the condensed local differential privacy (CLDP) model [28,29] introduces the distance measurement into differential privacy, and randomly responds to a real value based on the distance matrix. Compared with LDP, CLDP's utility is higher. Lin [30] propose a novel hiding-missing-artificial utility (HMAU) algorithm which hide sensitive itemsets through transaction deletion. A novel Privacy-Preserving and Security Mining Framework (PPSF) is present in [31], which focuses on privacy-preserving data mining and data security. Based on chaotic mapping, a wearables environment authentication protocol is proposed in [32].

3. Preliminaries and problem definition

3.1. Set-valued data

Set-valued data is a kind of unstructured data, as shown in Table 1. Unlike with relational data, each record t_i is a collection of arbitrary items, $t_i \subseteq I$, and I is the domain field. Such as: web query records containing multiple search keywords; shopping records containing purchased items; click streams containing URLs, etc. Various data mining tasks can be executed for set-valued data, such as association rule mining, user behavior prediction, making recommendations, and retrieving

information retrieval.

Table 1. Example of set-valued data.

ID	Items
1	a, b, c
2	b, d
3	b, e
4	a, b, d
5	a, e

3.2. Local differential privacy

Definition 1. (ε-local differential privacy [14]). A random algorithm $\Re$ satisfies ε-local differential privacy, where $\varepsilon > 0$, if and only if for any input t, t' and for any output t^*, we have:

$$\Pr\left[\Re(t) = t^*\right] \le e^{\varepsilon} \times \Pr\left[\Re(t') = t^*\right] \tag{3-1}$$

Intuitively, when the output is t^*, the data collector cannot infer whether the input data is t or t' with a high degree of confidence through ε control (ε is the privacy parameter controlling the level of indistinguishability, and a lower ε yields higher privacy). This is fundamentally different from centralized differential privacy. The input of central differential privacy is a neighboring dataset that differs in only one row.

Differential privacy has sequence combination and parallel combination properties. Sequence combination emphasizes that privacy budget ε can be allocated in different steps of the method, while parallel combination ensures that the privacy of the algorithm satisfies differential privacy on the disjointed subset of its dataset. Although DP focuses on the nearest neighbor dataset and LDP focuses on the two input values, they still have the same combination property because of their identical form of privacy guarantee.

3.3. Random response

Random response (RR) is a technique developed for the answer "yes or no" in interviewees to give random answer to a sensitive Boolean question so that they can achieve the purpose of privacy protection. For example, for a specific sensitive question such as "are you an HIV patient? ", the interviewee answers the true value with a probability of p and gives the opposite value with a probability of 1 – p. RR has been the predominant perturbation mechanism for LDP. To adapt RR to satisfy ε-LDP, we set p as follows:

$$p = \frac{e^{\varepsilon}}{e^{\varepsilon} + 1} \tag{3-2}$$

Note that the percentage of "true" (denoted as f) directly obtained from all perturbed answers is biased. To correct this, the data collector needs to calibrate it and reports f':

$$f' = \frac{p - 1 + f}{2 \cdot p - 1} \tag{3-3}$$

Therefore, with RR, the data collector can obtain the corresponding statistical information accurately without getting the real value.

3.4. Problem description

This paper studies the problem of set-valued data collection with LDP. Without loss of generality, we define the domain of the item as $I=\{a_1,\cdots,a_n\}$, the set of categories for the item is $C=\{c_1,\cdots,c_d\}$, there are *d* categories in total, the sub-domain under category $c_j\,(i\in[1,d])$ is IC_j, the candidate set under category c_j is $CandIC_j$, user u_i owns the data t_i, and the items in t_i under category c_j are defined as $t_{i,j}$. The main notations are listed in Table 2.

Table 2. Summary of notations.

Symbol	Description
u_i	the *i*-th user
t_i	the set-valued data of the *i*-th user
I	domain of item
C	category set
$t_{i,j}$	items of *i*-th user under category *j*
IC_j	sub-domain of category c_j
$CandIC_j$	the candidate set under category c_j

We define the private information model of set-valued data as the < *c*, *b*, *v* > triple, where *c* is a specific category, *b* indicates whether *c* exists or not (1 for exists, 0 for does not exist), and *v* is the number of items of t_i under category *c* (also called the value count). With this model, the information triples for user u_i under all category sets is $\left[\langle c_1,b_1,v_1\rangle,\langle c_2,b_2,v_2\rangle,\cdots,\langle c_d,b_d,v_d\rangle\right]$. The problem can be described as in Figure 2. The user's set-valued data is {*abdeh*}, which has four categories.

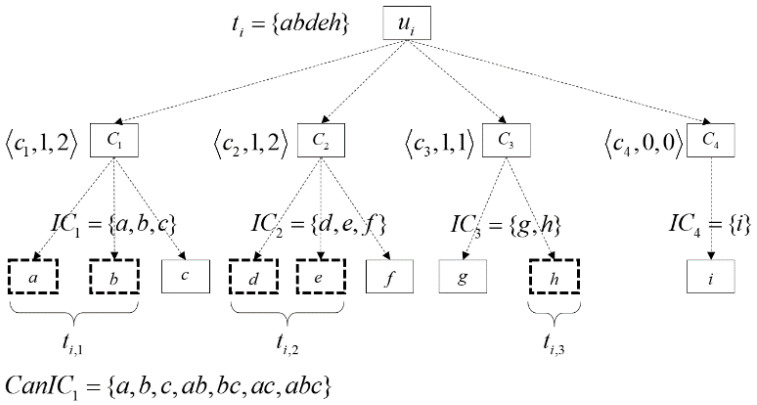

Figure 2. Problem description.

1) b' is obtained with a random response of b to protect the private information of category hierarchy. For example, the first triple may change to $\langle c_1,0,2\rangle$ (1- > 0), or it may remain unchanged;

2) For the value count *v*, we design a utility function u_v according to the length L_c of *IC*. Then, based on the exponential mechanism, we obtain v' for v to protect the length of set-valued

data. For example, if we get $\langle c_1, 1, 2\rangle$ in step 1, then the v' in step 2 may be $\langle c_1, 1, 1\rangle$. Another method for disturbing *v* is based on using the Laplace mechanism to add calibrated noise.

3) Based on the v' obtained in step 2, the utility function $u_{itemset}$ is designed for all categories of the real data. If $b' = 1$, the random response is an itemset from the candidate itemset of the sub-domain under this category. For example, in step 2, for $v' = 1$, the sub-domain in the first category is $IC_1 = \{abc\}$. Then, the itemset with length 1 in candidate set $CandIC_1$ is $\{a\}$, $\{b\}$, $\{c\}$, and the returned result itemset may be $\{a\}$. Another method is to directly respond with one of all candidate itemsets *C* and IC_1 based on the utility function but without considering the length of the itemset.

4) The candidate sets under each category are spliced together to form a new set-valued data, which is sent to the data center. For example, if the first category gets $\{a\}$, the second category gets $\{ef\}$, the third category gets an empty itemset because of $b = 0$, and the fourth category gets $\{i\}$, then the final data is $\{aefi\}$.

4. SetLPD method

In this chapter, the SetLDP method will be described in detail. Section 4.1 introduces the overall procedure of the SetLDP. Then, each sub-procedure will be introduced in turn, namely, category perturbation (CP), value perturbation (VP), and random itemset selection from candidate set (RS). At the end of this section, we conduct a theoretical analysis for SetLDP.

4.1. Overall process of the SetLDP method

The SetLDP method has 4 steps, Initialization, CP, VP, and RS, as shown in Figure 3.

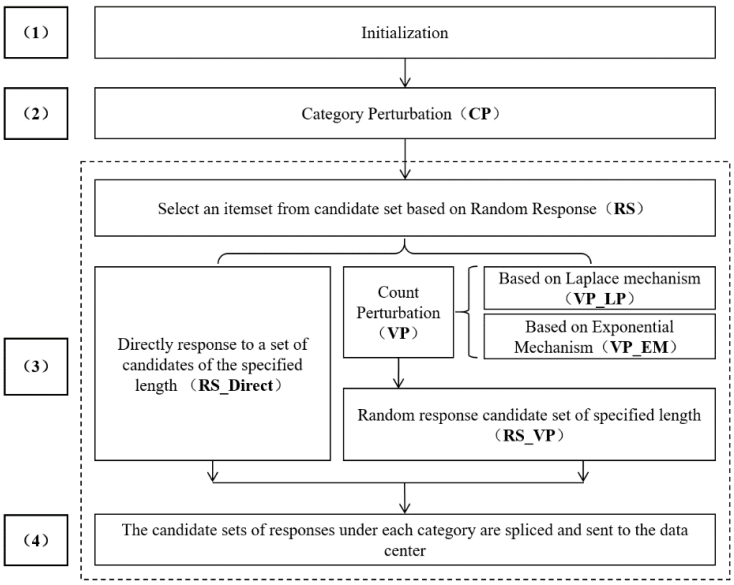

Figure 3. Overall procedure of the SetLDP method.

(1) Initialization. The user counts the number of items in each category, and identifies the category exists or not;

(2) In CP, there are four cases for perturbation on the category: 1- > 1, 1- > 0, 0- > 1, 0- > 0;

(3) In RS, there are two different methods, RS_Direct and RS_VP;

(4) Concatenate. Splice the randomly selected itemsets of all categories into complete set-valued data and send this to the data center.

The most important steps are (2), (3), and (4). Step (2) is based on the LDP model, which gets the original value with the probability of p, and inverts the value with the probability of $1 - p$. In steps (3) and (4), the sample space involved is the set of the value count and the candidate itemsets, which are finite countable discrete datasets. Therefore, the utility function can be designed based on the Exponential Mechanism (EM). We denote the utility function as $u(x,y)$ and the output as z. Then, the sensitivity of $u(x,y)$ is defined as follows:

$$\triangle u = \max_{\forall x,y} \left| u(x,z) - u(y,z) \right| \tag{4-1}$$

The Exponential Mechanism can randomly select one item from the finite countable set. If the input is t, the probability of the output being z is:

$$P\left[z \text{ is sampled}\right] \propto e^{\frac{\varepsilon \cdot u(t,z)}{2 \cdot \Delta u}} \tag{4-2}$$

Based on the Exponential Mechanism of local differential privacy, this work proposes the category perturbation algorithm (CP), value perturbation algorithm (VP), and randomly select candidate set algorithm (RS).

4.2. Category perturbation (CP)

In order to protect the private information of the category, it is necessary to make a random response to the existence of b. There are four kinds of random response regarding b: exists (1) - > exists (1), exists (1) - > does not exist (0), does not exist (0) - > exists (1), and does not exist (0) - > does not exist (0). CP is shown in Table 3.

Table 3. Category Perturbation (CP).

Input: $\langle c,b,v \rangle$, privacy budget: ε_1
Output: $\langle c,b',v \rangle$
1: if b is 1 then
2: Perturbs $\langle c,b',v \rangle$ as:
3: $\langle c,b',v \rangle = \begin{cases} \langle c,1,v \rangle & w.p. \ \frac{e^{\varepsilon_1}}{1+e^{\varepsilon_1}} \\ \langle c,0,v \rangle & w.p. \ \frac{1}{1+e^{\varepsilon_1}} \end{cases}$
4: else
5: Perturbs $\langle c,b',v \rangle$ as:
6: $\langle c,b',v \rangle = \begin{cases} \langle c,0,v \rangle & w.p. \ \frac{e^{\varepsilon_1}}{1+e^{\varepsilon_1}} \\ \langle c,1,v \rangle & w.p. \ \frac{1}{1+e^{\varepsilon_1}} \end{cases}$
7: end if
8: return $\langle c,b',v \rangle$

Theorem 4.1. Category perturbation algorithm (CP) satisfies the ε_1-local differential privacy.

4.3. Value perturbation algorithm I (VP_LP)

Since the value count is the numerical category, the Laplacian mechanism, which is most widely used in (local) differential privacy, is adopted first. In the Laplacian mechanism, the noise is calibrated by the privacy parameters and sensitivity. The VP_LP algorithm is showed in Table 4. The sensitivity of VP_LP is determined by the length $L_c = |IC_c|$ of the sub-domain under category *c*:

$$\Delta_{VP\_LP} = \max_{v_1, v_2 \in [0, L_c]} |v_1 - v_2|_1 = |L_c - 0| = L_c \tag{4-3}$$

Table 4. Value Perturbation algorithm based on the Laplace mechanism (VP_LP).

Input: $\langle c, \hat{b}, v \rangle$, the length of sub-domain under category c, $L_c = \|IC_c\|$, privacy budget: ε_2
Output: $\langle c, \hat{b}, v' \rangle$
1: $\Delta_{VP\_LP} = \max_{v_1, v_2 \in [0, L_c]} \|v_1 - v_2\|_1 = \|L_c - 0\| = L_c$
2: $v' = v + Lap\left(\frac{\Delta_{VP\_LP}}{\varepsilon_2}\right)$
3: return $\langle c, \hat{b}, v' \rangle$

Theorem 4.2. The Value Perturbation algorithm I (VP_LP) satisfies ε_2-LDP.

Theorem 4.3. The mean squared error (MSE) of the Value Perturbation algorithm I (VP_LP) is $2 \cdot (L/\varepsilon_2)^2$.

From Theorem 4.3, when L_c is large, the utility of the data becomes poor after adding too much noise. In the next section, based on the exponential mechanism, we will discuss the value perturbation algorithm II (VP_EM), randomly response with the value count v'.

4.4. Value perturbation algorithm II (VP_EM)

It is known that the value count of set-valued data of user *u* on category *c* is *v*, and the length of the sub-domain of category *c* is $L_c = |IC_c|$. Then, the range of v' is $[0, L_c]$, which means that the minimum value is 0, and the maximum value is the length L_c of all items under category *c*. Considering the distance between v and v', the utility function is defined as follows:

$$u_v(x, v) = \frac{1}{|x - v| + 1} \tag{4-4}$$

Intuitively, if *x* is closer to the value of *v*, then more information will be retained. The sensitivity of u_v is:

$$\triangle u_v = \max_{x, y, v \in [0, |IC|]} \left| \frac{1}{|x - v| + 1} - \frac{1}{|y - v| + 1} \right| = 1 - \frac{1}{v + 1} \leq 1 \tag{4-5}$$

Since $1/(v+1) \geq 0$, the sensitivity of u_v is 1. Based on $u_v(\cdot,\cdot)$, the probability of the randomly responded value v^* is proportional to u_v:

$$p_{v^*} \propto \exp\left(\frac{\varepsilon_2 \cdot u_v(v, v^*)}{2}\right) \tag{4-6}$$

Table 5. Value perturbation algorithm II (VP_EM).

Input: $\langle c, \hat{b}, v \rangle$, the length of the sub-domain under category c, $L_c = \lvert IC_c \rvert$, privacy budget: ε_2
Output: $\langle c, \hat{b}, v' \rangle$
1: $\Omega_v = \sum_{y \in [0, L_c]} \exp\left(\frac{\varepsilon_2 \cdot u_v(y, v)}{2 \cdot \Delta u_v}\right) = \sum_{y \in [0, L_c]} \exp\left(\frac{\varepsilon_2 \cdot u_v(y, v)}{2}\right)$
2: $u_v(v^*, v) = \frac{1}{\lvert v - v^* \rvert + 1}, v^* \in [0, L_c]$
3: v^* is sampled with probability $\Pr[y = v^*] = \exp\left(\frac{\varepsilon_2 \cdot u_v(v^*, v)}{2}\right) \Big/ \Omega_v$
4: return $\langle c, \hat{b}, v^* \rangle$

The value perturbation algorithm VP_EM is shown in Table 5, where:

$$\Omega_v = \sum_{y \in [0, L_c]} \exp\left(\frac{\varepsilon_2 \cdot u_v(y, v)}{2 \cdot \Delta u_v}\right) = \sum_{y \in [0, L_c]} \exp\left(\frac{\varepsilon_2 \cdot u_v(y, v)}{2}\right) \tag{4-7}$$

Ω_v is the normalization factor of sampling probability, and v and ε_2 are input parameters. The probability in step 3 is related to the utility function value of v^*. Based on the intuitive meaning of $u_v(\cdot,\cdot)$, we can see that the closer the selected x is to v, the higher the probability of v^* being selected, the less noise introduced, and the more the utility of data is guaranteed.

Theorem 4.4. VP_EM satisfies ε_2 - local differential privacy.

Theorem 4.5. The MSE of VP_EM reaches the maximum upper bound:

$$ErrorMSE_{VP\_EM} \leq \frac{v \cdot (2 \cdot v + 1)}{6}$$

when $\forall y_1, y_2 \in [0, v], p_{y_1} = p_{y_2}$, i.e., all sampling probabilities are the same.

4.5. Basic idea behind the randomly select candidate set (RS)

The basic idea of the RS is to randomly respond with an itemset from the candidate set C and IC_c under category c based on the exponential mechanism. The intuitive meaning of the utility function $u_{itemset}$ defined in RS is the intersection length of the candidate set with the real data under category c:

$$u_{itemset}(s, t_c) = \frac{\lvert s \cap t_c \rvert}{\lvert t_c \rvert} \tag{4-8}$$

$s \in |CandIC_c|, v' = |s|$, and the sensitivity of utility function $u_{itemset}$ is:

$$\Delta u_{itemset} = \max_{x,y \in CandIC, t} \left\| \frac{|x \cap t|}{|t|} - \frac{|y \cap t|}{|t|} \right\| \leq 1 \tag{4-9}$$

Similar to VP_ EM, based on the utility function $u_{itemset}$, the probability of the selected itemset s^* is proportional to $u_{itemset}$:

$$\Pr\left[s = s^* \right] \propto \exp\left(\frac{\varepsilon_3 \cdot u_{itemset}\left(s_y, t_c\right)}{2} \right) \tag{4-10}$$

We use the exponential mechanism to implement the algorithm (RS). Based on the intuitive meaning of $u_{itemset}$, the larger the intersection with t_c, the more likely the itemset is to be selected, and the more utility retained. RS satisfies ε_3-local differential privacy.

Theorem 4.6. RS satisfies ε_3- local differential privacy.

The basic idea of the RS is to randomly select an itemset from the candidate set based on the utility function. The main problem is that the sample space of the candidate set is exponentially related to the size of the itemset domain. We do not directly select the itemset from the sample space of the candidate set. Based on the utility function, the intersection size *inter* of t_c' with t_c is randomly selected first to determine the subspace of the sample space. The intersection size of all the candidate sets in the subspace with t_c are *inter*, and then the *inter* items are randomly selected from t_c. Based on this idea, we propose two methods: RS_Direct and RS_VP. Both methods need to preprocess the set-valued data with equal length first, and the length of set-valued data under category c of all users is m_c. There are differences between RS_Direct and RS_VP. RS_Direct specifies the uniform length m_c for each category c, and all users will be grouped under category c. Meanwhile, RS_VP randomly responds with a length to get $m_c = v'$ for all category c items, $v' \in [0, L_c]$. Then, users with the same length v' are divided into the same group, thereby generating user group L_c. The intersection size of the returned candidate set t_c' and the original data t_c is an important index to measure the effectiveness of the RS algorithm. Theorem 4.7 gives the upper bound of the MSE of the intersection size and the condition to reach the upper bound.

Theorem 4.7. Set $y = |t_c \cap t_c'|$ is the intersection size of t_c and t_c', the probability is p_y, and the MSE of y is:

$$ErrorMSE_{RS} = E\left(|v - y|^2\right) = \sum_{y=0}^{v} p_y \cdot \left(y^2 - 2 \cdot v \cdot y\right) + v^2 \tag{4-11}$$

where $v' = |t_c'|$, $v = |t_c|$, and $r = \min\{v, v'\}$. When $p_y = \frac{1}{1+r}$, $ErrorMSE_{RS}$ reaches the maximum upper bound:

$$ErrorMSE_{RS} \leq \frac{v \cdot (v+1)}{1+r} \cdot \left(\frac{1 - 4 \cdot v}{6} \right) + v^2 \tag{4-12}$$

In particular, where r=0, $ErrorMSE_{RS}$ reaches the maximum upper bound: $v \cdot (v+1) \cdot \left(\frac{1-4 \cdot v}{6} \right) + v^2$.

When the sampling probability p_y of all subspaces is equal, it is equivalent to selecting a subspace in a completely random way. Based on p_y, the probability of p_{s_y} can be obtained as follows:

$$p_{s_y} = \frac{1}{1+r} \cdot \frac{1}{C_{v'}^{y} \cdot C_{L-y}^{v'-y}} \tag{4-13}$$

It can be found that the first half of p_{s_y} is the random probability of the subspace, and the second half is the completely random probability of all the candidate sets whose intersection is y in v' subspace, which is very reasonable. In the intuitive sense, first we randomly select a subspace, then randomly select a candidate itemset from this subspace, and $ErrorMSE_{RS}$ reaches the upper bound.

4.6. Randomly select candidate set algorithm I (RS_Direct)

Table 6. RS_Direct algorithm.

Input: user's set-valued data $t \in D, \lvert t \rvert \le m$, domain $I = \{a_1, \cdots a_d\}$, privacy budget ε_3, output set-valued data length k
Output: the perturbation $t', \lvert t' \rvert = k$
1: $t' = \varnothing, d = \lvert I \rvert$
2: ▷ Make the length of t equal
3: $\hat{t} = t, \hat{I} = \{a_1, \cdots a_d\} \bigcup \{a_{d+1}, \cdots a_{d+m}\}$
4: if $m < \lvert t \rvert$:
5: $\hat{t} \leftarrow$ Randomly select m items from t
6: else:
7: for $i = 0; i < m - \lvert t \rvert; i = i + 1$ do:
8: $\hat{t} = \hat{t} \bigcup a_{d+i+1}$ // padding items are needed to be appended to achieve symmetry
9: end for
10: ▷ Randomize $\hat{t}$
11: $\Omega = \underbrace{e^{\frac{-\varepsilon_3}{2} \cdot 0} \cdot C_d^k}_{*} + \underbrace{\sum_{inter=1}^{k} \left(e^{\frac{-\varepsilon_3}{2} \cdot (k-inter)} \cdot \left(C_m^{inter} \cdot C_d^{k-inter} \right) \right)}_{**}$ // Ω is probability normalizer
12: $r = uniform\_random(0.0, 1.0)$ // generates a uniform random variable $r \in [0.0, 1.0)$
13: $inter = 0$
14: $p = e^{\frac{-\varepsilon_3}{2} \cdot 0} \cdot \frac{C_d^k}{\Omega}$
15: while $p < r$ do:
16: $inter = inter + 1$
17: $p = p + \frac{e^{\frac{-\varepsilon_3}{2} \cdot (k-inter)} \cdot \left(C_m^{inter} \cdot C_{d-1}^{k-1-inter} \right)}{\Omega}$ // given the actual probabilities that t' lies in each parts
18: end while
19: $t' = t' \bigcup sample(\hat{t}, inter)$ // the retained true data, *inters* items are sampled from the set-valued data $\hat{t}$
// the noisy data, the remaining *k–inter*s items are sampled from the rest of the padded domain $\hat{I} - \hat{t}$
20: $t' = t' \bigcup sample(\hat{I} - \hat{t}, k - inter)$
21: return t'

For RS_Direct based on the LDP privacy model we input the user's set-valued data *t*, item

domain I, privacy budget ε_3, and the length of the output set-valued data k. The output is the perturbation set-valued data t', and the length is $k = |t'|$. First, RS_Direct preprocesses t to get $\hat{t}$, with $|\hat{t}| = m$, and then we select a t' randomly from all candidate sets. The probability is:

$$\exp\left(\frac{-\varepsilon_3 \cdot dist(t,t')}{2}\right) \Big/ \Omega \tag{4-14}$$

where Ω is the probability generalization factor, and is defined in Eq (4-15). The RS_Direct algorithm is shown in Table 6.

$$\Omega = \underbrace{e^{\frac{-\varepsilon_3 \cdot 0}{2}} \cdot C_d^k}_{*} + \underbrace{\sum_{inter=1}^{k} \left(e^{\frac{-\varepsilon_3 \cdot (k-inter)}{2}} \cdot \left(C_m^{inter} \cdot C_d^{k-inter} \right) \right)}_{**} \tag{4-15}$$

Part * of Eq (4-15) represents the subspace with intersection of 0, while the part ** represents a subspace with the intersection size greater than 0. Because its sample space size is C_{d+m}^k, when d and m are large, the running time and the sample space are exponential. Therefore, RS_Direct divides the sample space of the candidate itemset into $k + 1$ sample subspaces. For all the candidate itemset in each sample subspace, the size *inter* of intersection with t is the same, and its range is [0, k], as shown in Table 7. In the RS_Direct method, steps 2–9 make the set-valued data t all of length m, and the added items are selected from the set $\{a_{d+1}, \cdots a_{d+m}\}$. Step 12 generates probability r based on $uniform\_random(0.0,1.0)$, and steps 12–18 determine the sample subspace based on r and p_{inter}. In steps 19 and 20, the number of items randomly selected from $\hat{t}$ is *inter*, and the number of items extracted from $\hat{I} - \hat{t}$ is *k-inter*. After splicing, the final data is returned to the user and sent to the server by the user. The time complexity of the RS_Direct method is linearly related to the size of the itemset domain.

Table 7. Subspace and corresponding sampling probability.

Sample subspace	Probability of each subspace
$\|s \cap t_c\| = 0$, $inter = 0$	
$\|s \cap t_c\| = 1$, $inter = 1$	$p_{inter} = \frac{e^{\frac{-\varepsilon_3 \cdot (k-inter)}{2}} \cdot \left(C_m^{inter} \cdot C_{d-1}^{k-1-inter} \right)}{\Omega}$
…	
$\|s \cap t_c\| = k$, $inter = k$	

Note: the candidate set in each subspace is the same size as the intersection of t

4.7. Randomly select candidate set algorithm II (RS_VP)

The advantage of the RS_VP algorithm over the RS_Direct algorithm is that it does not need to specify the equal length m_c for each category c. First, we use VP_LP or VP_EM to disturb the set-valued data value count v in each category c for the user to get v'. Then the data is processed to have equal lengths. The RS_VP algorithm is shown in Table 8. RS_VP obtains m_c in steps 1 and 2, and the subsequent steps are the same as in RS_Direct.

The set-valued data from user can be represented as triples $\langle c, b, v \rangle$, and the data under category c is t_c. The category perturbation algorithm (CP), value count perturbation algorithm (VP), and

randomly select candidate set algorithm (RS) are used to disturb the user's data under category c to obtain $\langle c,b',v'\rangle$ and t_c'. The SetLDP method is proposed to deal with all the categories of the data, and all t_c' are combined to form a complete set-valued data which is sent to the data center.

Table 8. Random response candidate set algorithm II (RS_VP).

Input: the set-valued data t_c under category c, privacy budget ε_2, ε_3, the length of the output set-valued data k
Output: the disturbed data $t_c', \left\|t_c'\right\| = k$ under category c
1: $v' = VP\_EM\left(v, L_c, \varepsilon_2\right)$
2: $m_c \leftarrow v'$
3: $t_c' \leftarrow RS\_Direct\left(t_c, m_c, \varepsilon_3, k\right)$
4: return t_c'

Table 9. SetLDP algorithm.

Input: user's set-valued data t, privacy budget $\varepsilon_1, \varepsilon_2, \varepsilon_3$
Output: the disturbed data t'
1: $t' = []$
2: for j=1 to d do:
3: $c \leftarrow c_j$
4: Init $\langle c,b,v\rangle$, t_c, $L_c = \left\|IC_c\right\|$, $CandIC_c$
5: $\langle c,b',v\rangle \leftarrow CP\left(\langle c,b,v\rangle, \varepsilon_1\right)$
6: If $b' = 1$ then
7: $\langle c,b',v'\rangle \leftarrow VP\_EM\left(\langle c,b',v\rangle, L_c, \varepsilon_2\right)$
8: If $v' > 0$ then
9: $t_c' \leftarrow RS\left(\langle c,\hat{b},v'\rangle, t_c, CandIC_c, \varepsilon_3\right)$
10: $t'.append(t_c')$
11: end if
12: end if
13: end for
14: return t'

The procedure of the SetLDP method is shown in Figure 4, and the three algorithms are executed sequentially. There are two special cases of the SetLDP. When the number of category $d = |I|$ is the maximum, i.e., each item has an independent category, and the random response to the category is equivalent to the random response to each item; When $d = 1$, all items belong to one category. Random response is made to the only one item category. If the result is 0, all the data will be eliminated. When the result is 1, random response will be made from one candidate set of the whole item set domain I, when the size of I increases, it is hard to achieve.

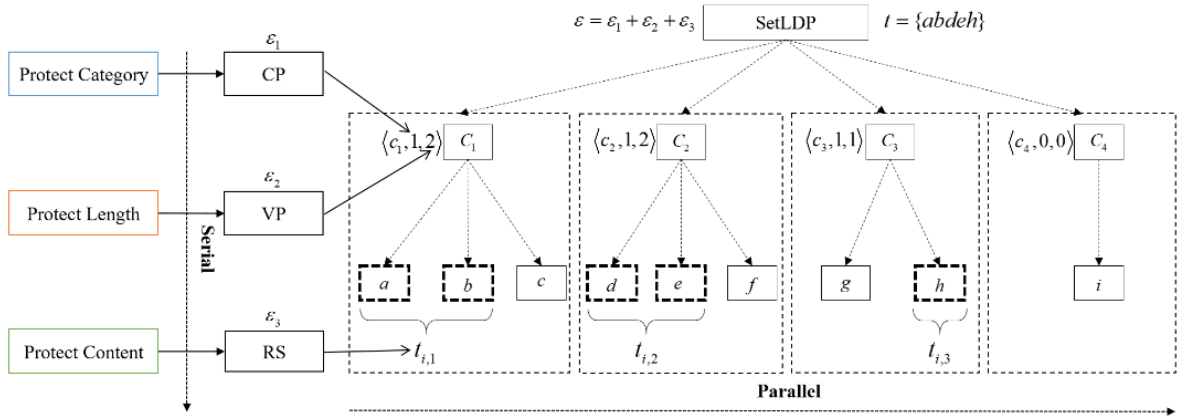

Figure 4. Illustration of SetLDP.

Theorem 4.8. SetLDP satisfies ε-local differential privacy, $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3$.

In the SetLDP method, the user runs CP, VP, and RS algorithms in sequence for a specific category *c*. CP is used to protect the category privacy of set-valued data, VP is used to protect the length of data, and RS is used to protect the content of data. There is a total of *d* categories that need to be run separately for *d* times; they are run in parallel. Therefore, based on the combination theory of local differential privacy, it can be concluded that step 7 in Theorem 4.8 can also use the VP_LP algorithm. A detailed description of the SetLDP method is presented in Table 9. The fourth step of the SetLDP method is to initialize, and the fifth step is the random response for the category. If $b' = 0$, which will be directly ignored and no subsequent operation will be carried out. If $b' = 1$ ($b = 1 \to b' = 1$ or $b = 0 \to b' = 1$), then steps 7–10 of the algorithm are run.

5. Theory analysis

5.1. Item support count distribution estimation and error boundary for RS_Direct

Inspired by [24], we derive the support count distribution estimation and error boundary of items. Because different utility functions are used in this paper, the TPR and FPR here are totally different from those in [24], and the principle of the whole derivation process is also different. Note that the whole derivation is based on one category *c*, and the total error boundary needs to be accumulated for all categories.

The estimation of the item support count is an important task in set-valued data collection and an important indicator of privacy protection data collection method. The item support count is defined as the number of set-valued data containing the item, $P_a = \#\{t_i | a \in t_i, i \in [1, n]\}$. Considering the item a_i and the set-valued data *t*, if $a_i \in t$, the received data t' also contains the item a_i, the true positive rate (*TPR*) as defined below:

$$TPR = \frac{e^{\frac{-\varepsilon_3}{2}\cdot(k-1)} \cdot \left(C_d^{k-1}\right) + \sum_{inter=2}^{k} \left(e^{\frac{-\varepsilon_3}{2}\cdot(k-inter)} \cdot \left(C_{m-1}^{inter-1} \cdot C_d^{k-inter}\right)\right)}{\Omega} \tag{5-1}$$

Otherwise, if $a_i \notin t$, the received data t' also contains the item a_i, the false positive rate (FPR) as defined below:

$$FPR = \frac{e^{\frac{-\varepsilon_3}{2}\cdot 0}\cdot C_{d-1}^{k-1} + \sum_{inter=1}^{k-1}\left(e^{\frac{-\varepsilon_3}{2}\cdot(k-inter)}\cdot\left(C_m^{inter}\cdot C_{d-1}^{k-1-inter}\right)\right)}{\Omega} \tag{5-2}$$

With the items $a_i \in I$ and set-valued dataset $D=\{t_1,t_2,t_3,...,t_n\}$ containing n users, the received dataset is $D'=\{t_1',t_2',t_3',...,t_n'\}$. If the item's true distribution is $P_a=\{P_{a_1},P_{a_2},P_{a_3},...,P_{a_d}\}$, then the expected frequency of a_i in t' is:

$$E\left[F_{a_i}\right]=E\left[\#\left\{t_i'|a_i\in t_i'\right\}\right]=n\cdot P_{a_i}\cdot TPR+n\cdot\left(1-P_{a_i}\right)\cdot FPR \tag{5-3}$$

Here, the first part $n\cdot P_{a_i}\cdot TPR$ means that the true a_i is retained, and the second part $n\cdot\left(1-P_{a_i}\right)\cdot FPR$ is noise. Therefore, P_{a_i} can be estimated as follows:

$$\tilde{P}_{a_i}=\frac{1}{n}\cdot\frac{F_{a_i}-n\cdot FPR}{TPR-FPR} \tag{5-4}$$

$\tilde{P}_a$ is an unbiased estimation for P_a. The proof is shown in Eq (5-5).

$$\begin{aligned} E\left[\tilde{P}_a\right] &= E\left[\frac{1}{n}\cdot\frac{F_a-n\cdot FPR}{TPR-FPR}\right]=\frac{1}{n\cdot\left(TPR-FPR\right)}\cdot E\left[F_a\right]-\frac{FPR}{TPR-FPR} \\ &= \frac{1}{n\cdot\left(TPR-FPR\right)}\cdot\left\{n\cdot P_a\cdot TPR+n\cdot\left(1-P_a\right)\cdot FPR\right\}-\frac{FPR}{TPR-FPR} \\ &= P_a \end{aligned} \tag{5-5}$$

Table 10. Frequency estimation algorithm.

Input: Data received by the collector $D'=\{t_1',t_2',t_3',...,t_n'\}$
Output: Estimation of frequency distribution of items $P_{a_i}=\{P_{a_1},P_{a_2},P_{a_3},...,P_{a_d}\}$
1: $\tilde{P}_a=\{0\}^{\lvert I\rvert}$, $F_a=\{0\}^{\lvert I\rvert}$
2: for $t'\in D'$ do:
3: for $a_i\in t'$ do:
4: $F_{a_i}=F_{a_i}+1$ // Count the frequency of each item from D'
5: end for
6: end for
7: for $i=1$ *to* $\lvert I\rvert$ do:
8: $\tilde{P}_{a_i}=\frac{1}{n}\cdot\frac{F_{a_i}-n\cdot FPR}{TPR-FPR}$
9: end for
10: return $\tilde{P}_a$

From D' we can get F_{a_i}, and we use the frequency estimation algorithm proposed in [24] to get F_{a_i}, which is showed in Table 10, and then further deduce $\tilde{P}_{a_i}$ based on the TPR and FPR, for item a_i, the random variable $\tilde{P}_a$ in Eq (5-4) is the linear transformation of F_a, and F_a is the sum of n Bernoulli random variables. Therefore, $n\cdot P_a$ is the number of successes with probability TPR in Bernoulli experiment. $n-n\cdot P_a$ is the number of successes with probability FPR. The variance of

the repeated n independent Bernoulli experiment is $Var(X) = E(X^2) - E(X)^2 = n \cdot p \cdot (1-p)$, where p is the success probability. The variance of F_a is:

$$Var(F_a) = n \cdot P_a \cdot TPR \cdot (1\text{-}TPR) + (n - n \cdot P_a) \cdot FPR \cdot (1\text{-}FPR) \tag{5-6}$$

Because random variable $\tilde{P}_a$ is a linear transformation of F_a, and according to the linear properties of the variance of the discrete random variables, we set a and b as constants:

$$Var(a \cdot X + b) = a^2 \cdot Var(x) \tag{5-7}$$

The linear transformation of $\tilde{P}_a$ can be reorganized to

$$\tilde{P}_a = \frac{1}{n \cdot (TPR - FPR)} \cdot F_a - \frac{FPR}{TPR - FPR} \tag{5-8}$$

The variance of $\tilde{P}_a$ is:

$$\begin{aligned} Var(\tilde{P}_a) &= \frac{n \cdot P_a \cdot TPR \cdot (1\text{-}TPR) + (n - n \cdot P_a) \cdot FPR \cdot (1\text{-}FPR)}{n^2 \cdot (TPR - FPR)^2} \\ &= \frac{P_a \cdot TPR \cdot (1\text{-}TPR) + (1 - P_a) \cdot FPR \cdot (1\text{-}FPR)}{n \cdot (TPR - FPR)^2} \end{aligned} \tag{5-9}$$

The MSE of item distribution estimation is as follows:

$$\begin{aligned} E\left[\left(\tilde{P}_a\text{-}P_a\right)^2\right] &= E\left[\left(\tilde{P}_a\text{-}E\left[\tilde{P}_a\right] + E\left[\tilde{P}_a\right]\text{-}P_a\right)^2\right] \\ &= E\left[\left(\tilde{P}_a\text{-}E\left[\tilde{P}_a\right]\right)^2\right] + E\left[E\left[\tilde{P}_a\right] - P_a\right]^2 + 2E\left[\left(\tilde{P}_a - E\left[\tilde{P}_a\right]\right) \cdot \left(E\left[\tilde{P}_a\right] - P_a\right)\right] \\ &= Var(\tilde{P}_a) + \left(E\left[\tilde{P}_a\right] - P_a\right)^2 \end{aligned} \tag{5-10}$$

Equation (5-5) proves that $\tilde{P}_a$ is an unbiased estimation for P_a. Therefore,

$$MSE(\tilde{P}_a) = E\left[\left(\tilde{P}_a\text{-}P_a\right)^2\right] = Var(\tilde{P}_a) \tag{5-11}$$

Then, the total MSE deviation is:

$$\begin{aligned} Error\ Bound &= \sum_{i \in [1,d]} E\left[\left(\tilde{P}_{a_i} - P_{a_i}\right)^2\right] = \sum_{i \in [1,d]} Var(\tilde{P}_{a_i}) \\ &= \frac{\sum_{i \in [1,d]} P_{a_i} \cdot TPR \cdot (1\text{-}TPR) + \left(d - \sum_{i \in [1,d]} P_{a_i}\right) \cdot FPR \cdot (1\text{-}FPR)}{n \cdot (TPR - FPR)^2} \end{aligned} \tag{5-12}$$

5.2. Item support count distribution estimation and error boundary for RS_VP

The difference between RS_VP and RS_Direct is that RS_VP does not directly specify the equal length m, but randomly generates m based on the count perturbation algorithm VP. Because of this difference in m, the estimation of the item support count distribution and the error boundary are derived in a different manner. RS_Direct processes the set-valued data of all users equally under all

categories, that is, all users are divided into the same group and have the same *m* and *k*.

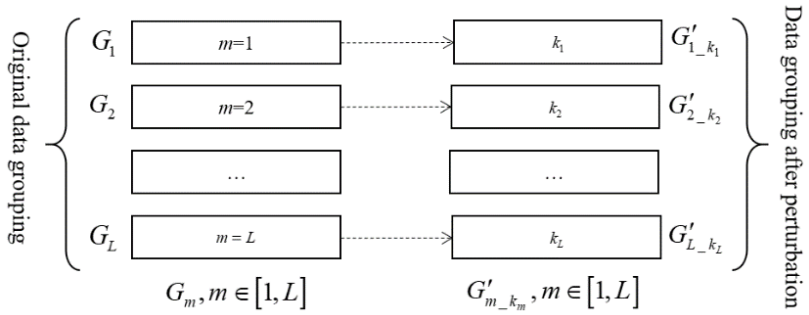

Figure 5. Data grouping.

Conversely, RS_VP groups users based on the same category and each group has the same *m* and *k*. Users with the same data length *m* are grouped into the same group G_m; there are *L* groups in total, which is the size of the itemset domain under category *c*. Since the lengths of set-valued data in the same group of users is equal, the same *k* value will be generated, and the data obtained after each group of users is disturbed will also be in the same group G'_{m_k}. This is illustrated in Figure 5.

Finally, the total number of groups is *L*, the data in each group has the same *m* and *k*, and the length of the candidate set of the random response is *k*. The frequency distribution of the items can be estimated for each specific group G'_{m_k}, and the estimation process is exactly the same as explained in section V.A. We set $n_{m\_k_m} = |G_m| = |G'_{m\_k_m}|$, and the MSE of group G'_{m_k} is:

$$Error\ Bound_{m\_k_m} = \frac{\sum\limits_{i\in[1,L_c]} P_{a_i} \cdot TPR \cdot (1\text{-}TPR) + \left(L - \sum\limits_{i\in[1,L_c]} P_{a_i}\right) \cdot FPR \cdot (1\text{-}FPR)}{n_{m\_k_m} \cdot (TPR - FPR)^2} \tag{5-13}$$

Then, the MSE under one category is the average of the MSE of all user groups:

$$Error\ Bound = \sum_{i=1}^{m} Error_{i\_k_i} \Big/ L_c \tag{5-14}$$

Similarly, the total of MSE of a user is the sum of MSEs for all categories.

6. Experimental results and analysis

In this section, we evaluate the SetLDP in an experimental environment. The experimental scheme is shown in Table 11.

The algorithms used for comparisons are RAPPOR [14], PrivSet [24], and BRR [26]. PrivSet is divided into original PrivSet_NG (NG for No Group), which is applied to the whole itemset domain, and PrivSet_G (G for Group), which is applied to the grouped itemset domain (i.e., it includes the category level). BRR is excellent for random response of set-valued data based on the local differential privacy model. Its core idea is to randomly extract an item from the set-valued data and send it to the server after a random response. This means that the server will only receive one bit of data.

Table 11. Experiment scheme.

Name	Parameter	Validation algorithm	Comparison algorithm	Metric
verification	ε_1	CP	RAPPOR [14]	Relative Error: true count and estimated count for category (Figures 6 and 7)
	ε_2	VP_EM	VP_LP	MSE and Absolute Error: true count and estimated count for the set-valued data (Figures 8 and 9)
	$\varepsilon_2 = 0.1, \varepsilon_3$			
	The size of the domain for group Average length	RS_Direct	RS_Direct +VP_EM	Average relative error: intersection length (Figures 10 and 11)
comparison	$(d,m), \varepsilon_3$	RS_Direct _NG RS_Direct_G RS_VP_G	PrivSet_NG [24] PrivSet_G [24] BRR_NG [26]	k value and error boundary of the item support count estimation (Tables 12 and 13)
application	$(d,m), \varepsilon_3$	RS_Direct _NG RS_Direct_G RS_VP_G	PrivSet_NG [24] PrivSet_G [24]	Total mean square error of support count estimation for items (Figure 12)
	$\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3$	SetLDP= CP+VP_EM+RS_VP	None	Total mean square error of support count estimation for items (Figure 13)

The overall experimental scheme consists of four parts:

(1) Different privacy parameters are examined for CP, VP_LP, VP_EM, and RS_Direct. CP is compared with RAPPOR. Since VP and RS are only based on the basic idea of random response, we just verify the effectiveness of each independent algorithm and conduct no comparison. For VP, there are two different methods (VP_LP and VP_EM), and we compare them with each other in the experiment. For RS, we compare the pure RS_Direct (without VP) with RS_Direct +VP_EM (with VP).

(2) Without considering the category privacy of the item (i.e., without running the CP algorithm), for *_NG (with category) and *_G (without category) methods, we compare the utility between RS, PrivSet, and BRR. The considered index is the output data length k value and the error boundary of item support count estimation. The * means a different specific algorithm is used.

(3) Without considering the category privacy of the item (i.e., without running the CP algorithm), the utility of the VP + RS algorithm is examined. We mainly focus on the privacy protection for the data length and the content. In the experiment, the distribution estimation of the support count of items is compared with that of other algorithms (*_NG and *_G algorithms).

(4) With the category privacy protection of the item, experiments with CP + VP_EM + RS_VP (SetLDP) are conducted. We analyze the distribution estimation of the support count of items by SetLDP under the influence of different privacy parameters. Since there is no related algorithm for comparison, only the influence of privacy parameters on the distribution estimation of support count is analyzed.

6.1. Experimental setup

For comparison with PrivSet, the dataset used in the experiment is the same as that in the experimental environment in PrivSet, and the set-valued dataset generated by simulation is as close as possible to the real dataset. The number of users in the dataset is 10,000, the item domain size d is 4–100, the maximum length of the set-valued data m is [2,16], and the range of privacy parameters is 0.01–3.0. For different parameter combinations, 1000 simulation experiments are conducted and the average value is obtained. The generation of the set-valued dataset is as follows. The items in the

itemset field are divided into G groups (each group is a category), and each category has its corresponding itemset sub-domain IC_c; the length of IC_c is $L_c = |IC_c|$. Each term is randomly selected by probability m/d from the domain I. SetLDP is similar to the PrivSet in that it is also independent of the specific dataset. Therefore, the set-valued dataset generated by simulation is enough to validate the effectiveness of the SetLDP.

The experimental environment is an Intel (R) Core (TM) i7-7660U CPU @ 2.50GHz, 16.00GB memory, 64Bit Win10 operating system, and the programming language is Python3.6.3. There are two main comparative performance metrics: relative error and absolute error. The absolute error is defined as the difference between a numerical result F (such as type count, data length, content intersection length, etc.) of the real dataset and the result F' of the resulting dataset. Meanwhile, the relative error is defined as the ratio of absolute error to the true numerical result F. Similar to PrivSet, the probability simplex method proposed in [29] is used to optimize the estimation of the item support count distribution $\tilde{P}_a$.

Absolute Error:

$$absolute\ error = |F - F'| \tag{6-1}$$

Relative Error:

$$relative\ error = \frac{|F - F'|}{F} \tag{6-2}$$

6.2. Experimental setup analysis of category perturbation algorithm - CP

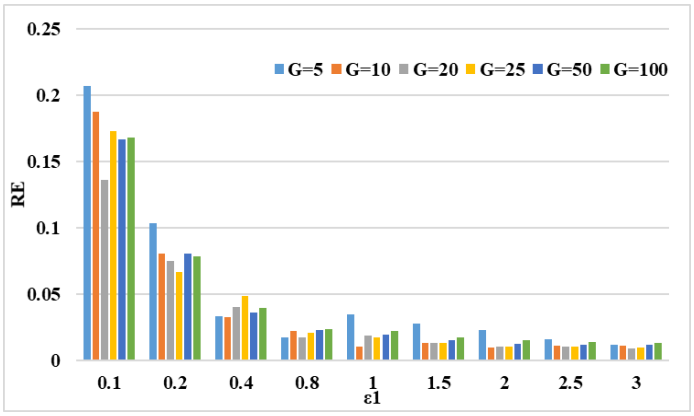

Figure 6. Algorithm CP and its parameter ε_1.

The performance metrics estimated for the CP algorithm are the relative error (Figure 6) and percentile (Figure 7) of the frequency estimation of the category. The relative error of category frequency estimation varies with ε_1, as shown in Figures 6 and 7. The overall effect is excellent: when ε_1 is 0.1, RE is 0.2, and with the growth of the privacy parameter, RE decreases. This is consistent with the theory, because when privacy parameter ε_1 increases, the privacy decreases and the utility increases. However, if the privacy parameters ε_1 have the same value, different categories of numbers have little impact on the results, which shows that CP has good adaptability to different-length categories.

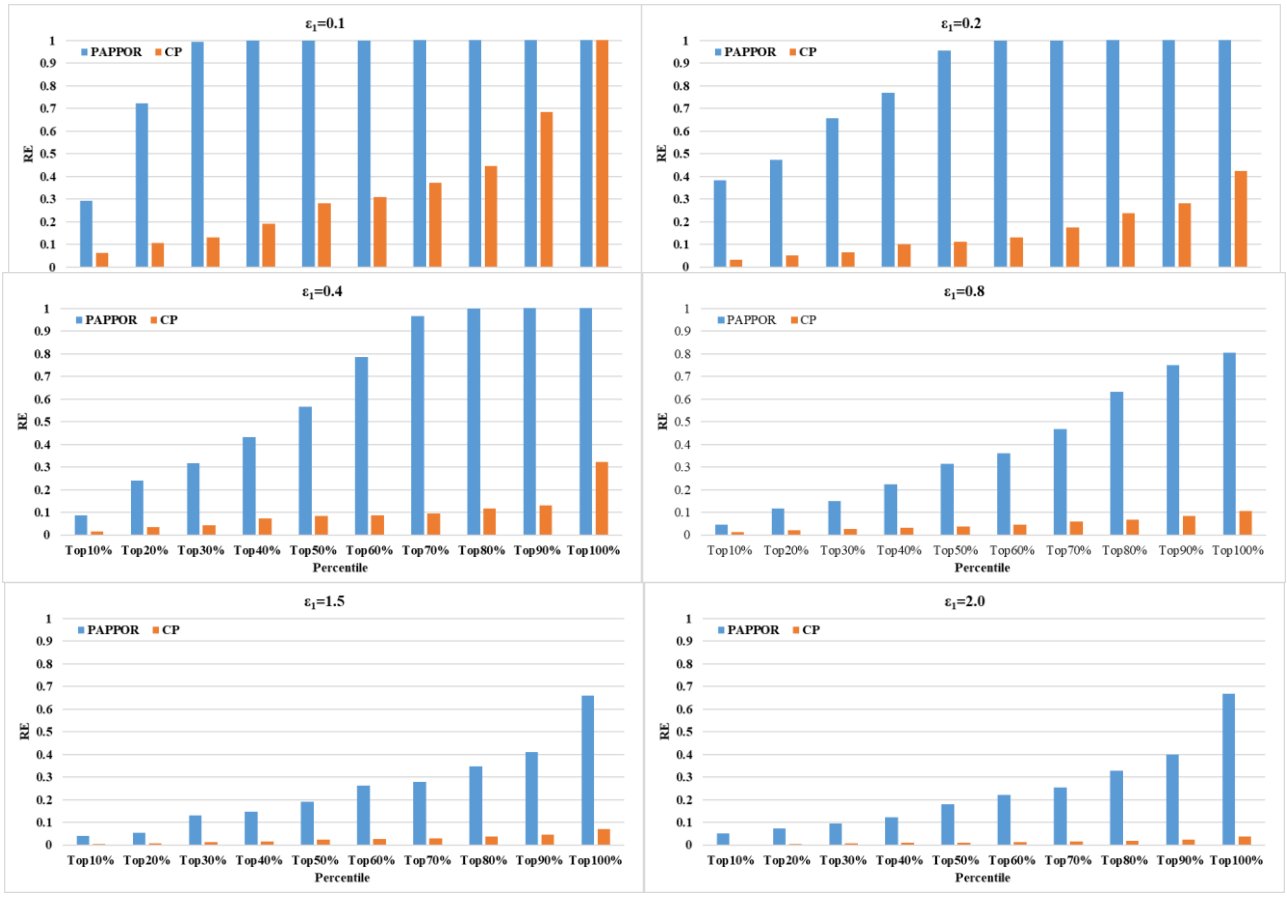

Figure 7. Frequency estimations of CP and PAPPOR along with ε.

Figure 7 compares the utility between CP and PAPPOR. We define the number of categories as G = 32, and the itemset domain size of each category is 6. The privacy parameter ε_1 takes 6 different values, and the metric is the percentile. We set the parameter bloom filter length h of PAPPOR to 64, the number of hash functions to 2, and the number of cohorts to 2. We set the probability value f according to the original privacy parameter ε_1, p = 0.5, and q = 0.75. It can be seen from Figure 7 that the CP algorithm is better than the PAPPOR algorithm, and both of them show a trend consistent with the theory of the change of privacy parameters. When ε_1 = 2.0, the relative error of the percentile of the CP algorithm is very low, but the privacy is low.

6.3. *Analysis of value perturbation algorithm - VP*

The effectiveness of VP_LP and VP_EM is estimated in this section. The metrics are MSE and ABS of the value count. The utility is analyzed for different groups (categories) G and different privacy parameters ε_2, as shown in Figures 8 and 9. In those figures, d is the total itemset domain size, set to 100, G is the group number (categories), and L is the size of the itemset sub-domain of each group. It can be found that the two algorithms are greatly affected by the group number G, that is, when G is larger and L is smaller, the value count of the items under each category is relatively smaller. Especially in VP_EM, the smaller the L and the smaller the mean square error, which is consistent with the theory. In addition, VP_LP is greatly affected by privacy parameter ε_2, while

VP_EM is not; this is mainly related to the limited countable sample space.

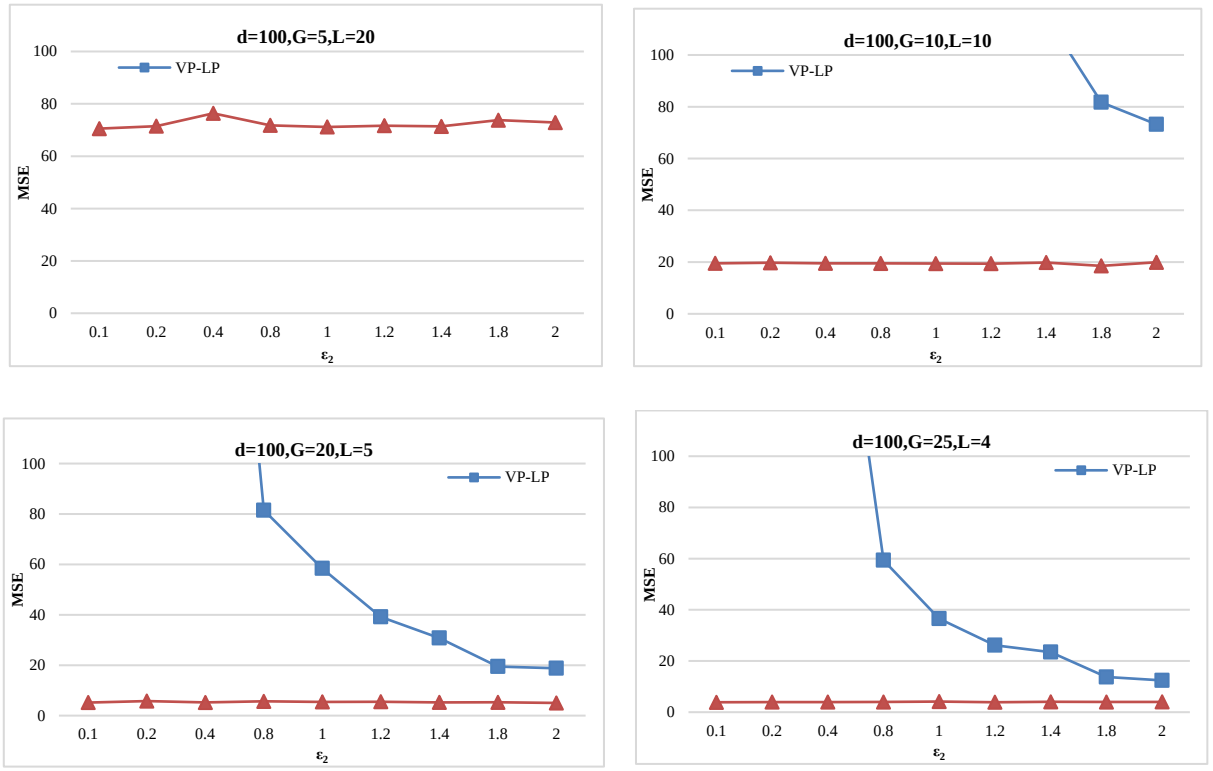

Figure 8. MSE of VP.

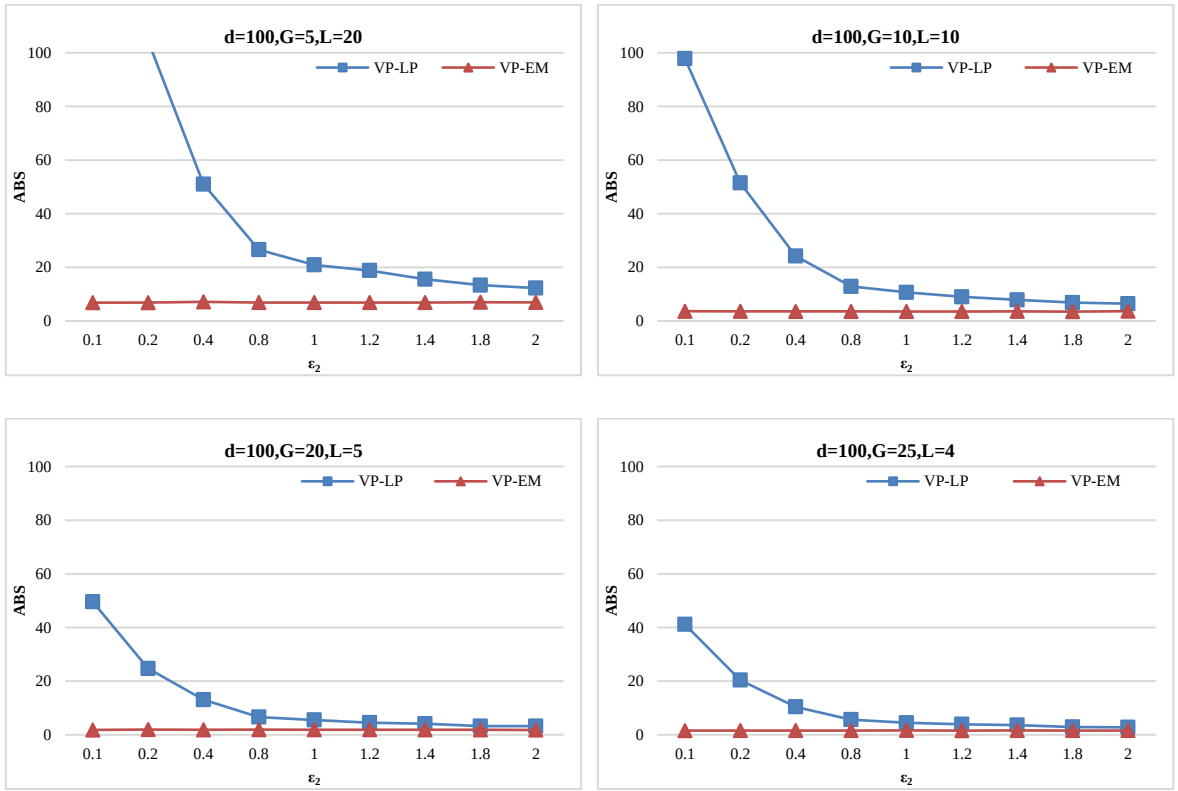

Figure 9. Absolute error of VP.

6.4. Analysis of randomly select candidate set algorithm - RS

The category perturbation algorithm and value perturbation algorithm are mainly used to protect the category of the item and the length of set-valued data; meanwhile, the randomly select algorithm RS of candidate set is mainly used to protect the content of set-valued data. In this section, the effectiveness of the RS algorithm is verified through experiments, as shown in Figures 10 and 11. Figure 10 is the result of algorithm RS_VP + VP_EM: the privacy parameter ε_2 is set to 0.1, and the metric is the MSE of the intersection number of the returned candidate set and the original set-valued data. We conclude that the MSE varies with the average length of the dataset, the size of the set-valued domain of different groups, and the privacy parameters ε_2. As shown in the upper left of Figure 10, the average length 4.5 refers to the average length of all user set-valued data. For group 5-2/3, 5 is the number of the category group (i.e., there are 5 items in each group), and 2/3 is the length of randomly generated set-valued data obtained from the sub-domain of each category group. From Figure 11, we can find two conclusions. Firstly, the MSE of the intersection is less affected by privacy parameter ε_2, which is consistent with the conclusions obtained in Figures 8 and 9. Secondly, it is found from Figure 10 that when the average length of the generated data is the same, the sizes of different domains and the minimum length of randomly generated data have a greater impact on MSE; moreover, the longer the length of randomly generated set-valued data, the smaller the MSE. This is because the longer the generated data, the less the noise in the sample space of the candidate set, the larger the intersection of the random response candidate set and the original data in the candidate set sample space. Therefore, more information is retained.

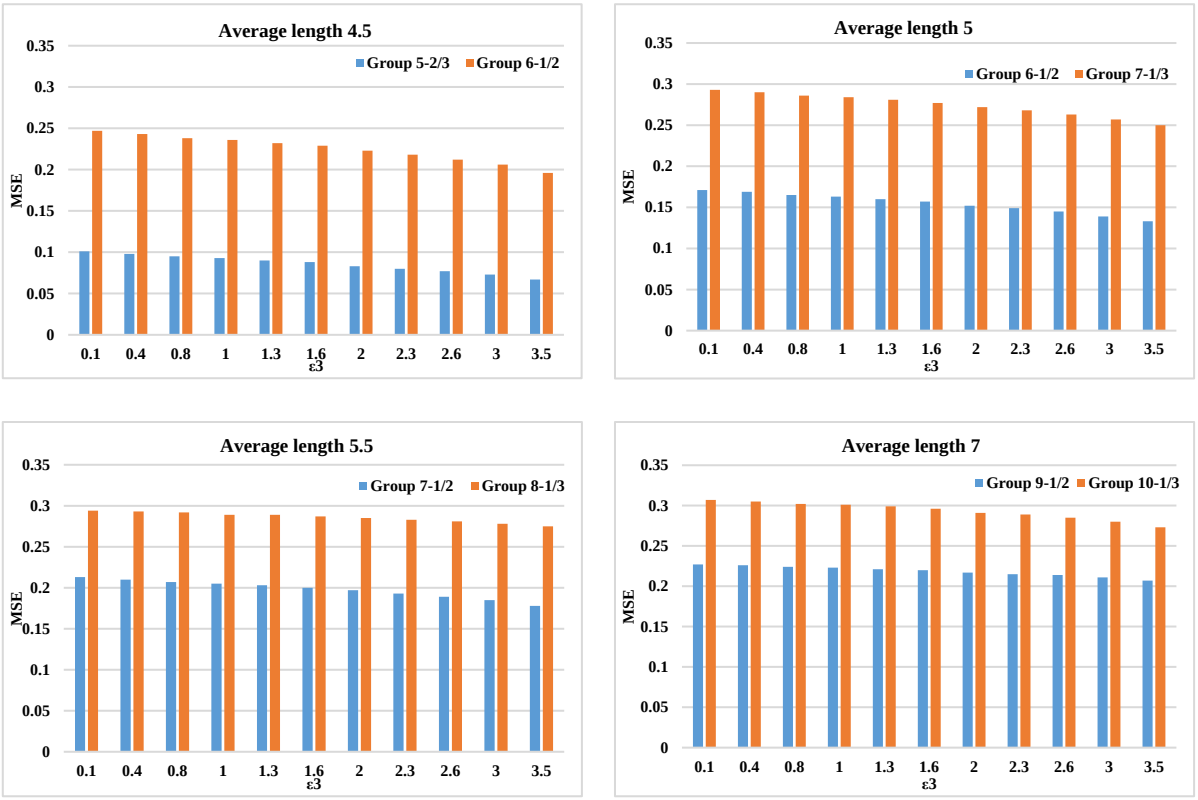

Figure 10. The influence of the average length of set-valued data and the size of the itemset field of a group on MSE.

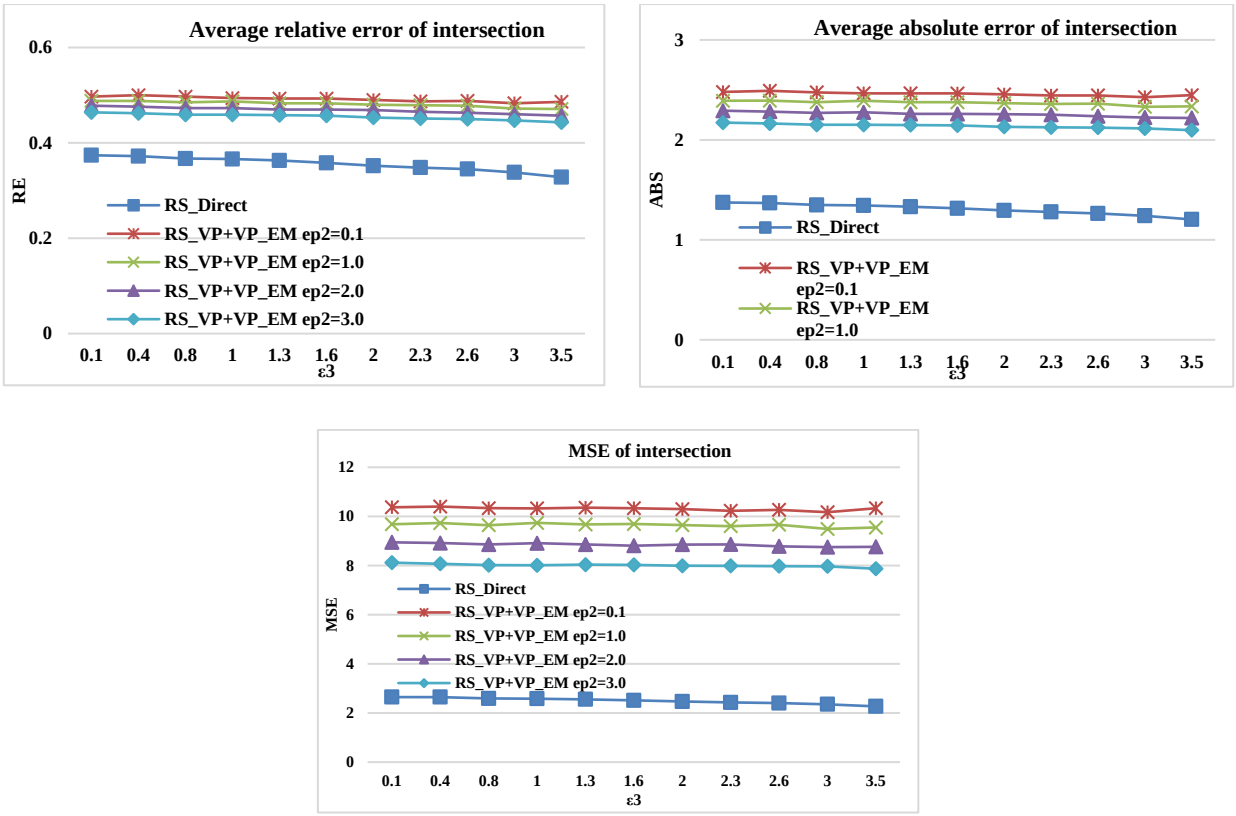

Figure 11. Influence of VP on relative error, absolute error, MSE.

Figure 11 mainly estimate the effect of VP on RS. The algorithm in the experiment is RS_VP+VP_EM, and the comparison algorithm is RS_Direct (where the value count is not disturbed by the VP). The metrics are the average relative error, average absolute error, and mean square error of the intersection number. The size of the sub-domain for each category is 8, and the average length of the data is 5. We observe that RS_Direct is significantly better than RS_VP+VP_EM because VP_EM disrupts the value count and causes information loss. Although utility is affected by privacy parameters ε_2 and ε_3, the effect is small, which is consistent with the experimental results in Figures 8–10.

6.5. *Analysis of k value and error boundary of frequency distribution estimation of the item*

k is the length of set-valued data generated by the RS method, and the possible range of k value is [1, d]. Since the estimation of the item support count is the core of the whole algorithm, its error boundary is key: the smaller the value, the better. If d and m are determined, the error boundary is only related to k, and it can be minimized by traversing [1, d] to find the k value. The key of RS is to determine the k value and the minimum value of the error boundary of the support count of the term. We compare the error boundary of the k value and support count estimation in RS and PrivSet with different parameters through experiments. Because the BRR algorithm only selects one term randomly, the k value is meaningless; thus, it is only necessary to compare the error boundary. The experimental results are shown in Tables 12 and 13. Table 12 compares BRR_NG, PrivSet_NG, and RS_Direct_NG when the data is not grouped; meanwhile, Table 13 compares PrivSet_G, RS_Direct_NG, RS_Direct_G, and RS_VP_G when the data is grouped. Here, the Group means the category in all the experiments, every item belongs to a category (a group).

Table 12. *k* Value and the error boundary of and frequency distribution estimation (BRR, PrivSet, RS_Direct).

(d, m)	Method	$\varepsilon_3 = 0.01$		$\varepsilon_3 = 0.1$		$\varepsilon_3 = 0.4$		$\varepsilon_3 = 1$		$\varepsilon_3 = 2$	
		k	Error Bound	k	Error Bound	k	Error Bound	k	Error Bound	k	Error Bound
(16,8)	BRR_NG	-	61439998	-	614398	-	38398	-	6142	-	1534
	PrivSet_NG	1	5501702	1	53460	1	3086	1	457	1	127
	RS_Direct _NG	12	3526666	12	35266	12	2204	11	350	10	85
(32,8)	BRR_NG	-	102399997	-	1023997	-	63997	-	10237	-	2557
	PrivSet_NG	2	11996243	2	117231	2	6907	1	948	1	192
	RS_Direct _NG	20	6084008	20	60848	19	3796	17	601	14	145
(32,16)	BRR_NG	-	491519996	-	4915196	-	307196	-	49148	-	12284
	PrivSet_NG	1	22485227	1	218502	1	12624	1	1879	1	531
	RS_Direct _NG	24	7363333	24	73633	23	4597	22	731	20	179
(64,8)	BRR_NG	-	184319994	-	1843194	-	115194	-	18426	-	4602
	PrivSet_NG	4	25296086	4	248035	3	14606	2	2007	1	359
	RS_Direct _NG	36	11202249	35	112011	33	6984	29	1103	23	263
(64,16)	BRR_NG	-	819199993	-	8191993	-	511993	-	81913	-	20473
	PrivSet_NG	2	48925531	2	477949	2	28134	1	3852	1	791
	RS_Direct _NG	40	12482015	39	124817	38	7788	34	1234	29	298
(64,32)	BRR_NG	-	3932159992	-	39321592	-	2457592	-	393208	-	98296
	PrivSet_NG	1	90897749	1	883327	1	51057	1	7619	1	2167
	RS_Direct _NG	48	15041666	48	150416	46	9391	44	1493	40	365
(128,32)	BRR_NG	-	6553599987	-	65535987	-	4095987	-	655347	-	163827
	PrivSet_NG	2	197575436	2	1929764	2	113547	1	15531	1	3208
	RS_Direct _NG	80	25281030	79	252783	75	15772	68	2500	58	605
(128,64)	BRR_NG	-	31457279984	-	314572784	-	19660784	-	3145712	-	786416
	PrivSet_NG	1	365504678	1	3551948	1	205346	1	30681	1	8757
	RS_Direct _NG	96	30400833	95	303988	93	18979	88	3019	81	739
(128,96)	BRR_NG	-	82575359982	-	825753581	-	51609581	-	8257517	-	2064365
	PrivSet_NG	1	498814919	1	4932308	1	302681	1	50957	1	17045
	RS_Direct _NG	112	35520699	112	355192	110	22181	108	3532	105	868
(256,64)	BRR_NG	-	52428799973	-	524287973	-	32767973	-	5242853	-	1310693
	PrivSet_NG	8	212041690	8	2081414	6	121963	4	16745	2	3029
	RS_Direct _NG	136	43200725	133	431929	124	26924	106	4244	80	1007

A few points can be seen from Table 12. (i) When the values of (*d*, *m*) are the same, with the growth of privacy parameters, the error boundaries are all decreasing, which is consistent with the theoretical inference. (ii) The *k*-values of RS and PrivSet will decrease with the increase in privacy parameters because there is less noise. Furthermore, the *k*-values of PrivSet are very small, and eventually become 1 with the increase in privacy parameters because PrivSet is mainly used for the support count estimation of a single item, which is equivalent to the support count estimation of 1-frequent itemset. Conversely, it is important that the *k* value of RS is generally larger; this is because the RS method adopts different utility functions and keeps more information. (iii) When (*d*, *m*) is smaller, the error bound of the PrivSet method is smaller than that of the RS method, but the SetLDP method is better than BRR. When (*d*, *m*) is increased to a certain extent (e.g., $d = 16$, $m = 8$), the error bounds of the SetLDP method are smaller than those of the PrivSet method.

From Table 13, it can be seen that the RS_*_* proposed in this paper is better than PrivSet_* and BRR_*.

Table 13. *k* Value and the error boundary of frequency distribution estimation (PrivSet, RS_Direct, RS_VP).

(d, m)	Method	$\varepsilon_3 = 0.01$		$\varepsilon_3 = 0.1$		$\varepsilon_3 = 0.4$		$\varepsilon_3 = 1$		$\varepsilon_3 = 2$	
		k	Error Bound	k	Error Bound	k	Error Bound	k	Error Bound	k	Error Bound
(32,16) G = 2	PrivSet_G	2	11003404	2	106921	2	6172	2	914	2	254
	RS_Direct _NG	24	7363333	24	73633	23	4597	22	731	20	179
	RS_Direct_G	24	7053333	24	70533	24	4408	22	699	20	170
	RS_VP_G	20.0	5942629	20.0	59389	19.25	3705	17.25	586	14.5	140
(64,16) G = 2	PrivSet_G	4	23992485	4	234463	4	13814	2	1895	2	384
	RS_Direct _NG	40	12482015	39	124817	38	7788	34	1234	29	298
	RS_Direct_G	40	12168015	40	121695	38	7591	34	1202	28	290
	RS_VP_G	36.0	11052418	36.0	110494	33.75	6889	29.25	1087	23.0	259
(64,32) G = 4	PrivSet_G	4	22006807		213842	4	12345	4	1829	4	508
	RS_Direct _NG	48	15041666	48	150416	46	9391	44	1493	40	365
	RS_Direct_G	48	14106666	48	141066	44	8816	44	1398	40	341
	RS_VP_G	40.0	11885259	40.0	118777	38.5	7409	34.5	1171	29.0	281
(128,32) G = 4	PrivSet_G	8	47984971	8	468926	8	27627	4	3790	4	768
	RS_Direct _NG	80	25281030	79	252783	75	15772	68	2500	58	605
	RS_Direct_G	80	24336030	80	243390	76	15182	68	2403	56	580
	RS_VP_G	72.0	22104836	72.0	220987	67.5	13777	58.5	2174	46.0	517
(128,64) G = 8	PrivSet_G	8	44013615	8	427684	8	24690	8	3657	8	1017
	RS_Direct _NG	96	30400833	95	303988	93	18979	88	3019	81	739
	RS_Direct_G	96	28213332	96	282132	96	17632	88	2797	80	681
	RS_VP_G	80.0	23770518	80.0	237555	77.0	14819	69.0	2343	58.0	562
(128,96) G = 8	PrivSet_G	8	60394583	8	597137	8	36599	8	6121	8	2016
	RS_Direct _NG	112	35520699	112	355192	110	22181	108	3532	105	868
	RS_Direct_G	112	33325696	112	333239	112	20811	104	3314	104	811
	RS_VP_G	88.0	26323828	88.0	263112	86.0	16417	79.3	2602	70.7	629

6.6. *Analysis of MSE of item support count distribution estimation*

The distribution estimation of the item support count is one of the most important applications of set-valued data collection, which can be used for 1-frequent itemset mining, hot commodity

analysis, and other tasks. In this section, the effectiveness of the proposed methods RS_NG, RS_G, and RS_VP are compared with those PrivSet_NG and PrivSet_G through experiments, as shown in Figure 12. The effectiveness of the SetLDP (= CP + VP + RS) method is analyzed in Figure 13. *G*, *d*, and *m* respectively represent the number of category groups, the number of itemset sub-domains in each group, and the maximum length of set-valued data. The privacy parameter of VP is set to 0.1, while the abscissa is the privacy parameter of PrivSet and RS. The ordinate is the logarithm after taking the average value of the error bound (Eq (5-12)) of all groups.

As observed in Figure 12, the metric is the error bound of the distribution estimation of the item support count. The five algorithms all show a downward trend with the increase in privacy parameter. As can be seen in Figure 10, the RS_* algorithm proposed in this paper is better than sPrivSet_* in overall effect. Furthermore, it is found that grouping is better than no grouping. In addition, when *d* and *m* increase, the trend of the RS algorithm is more obvious than that of the PrivSet algorithm.

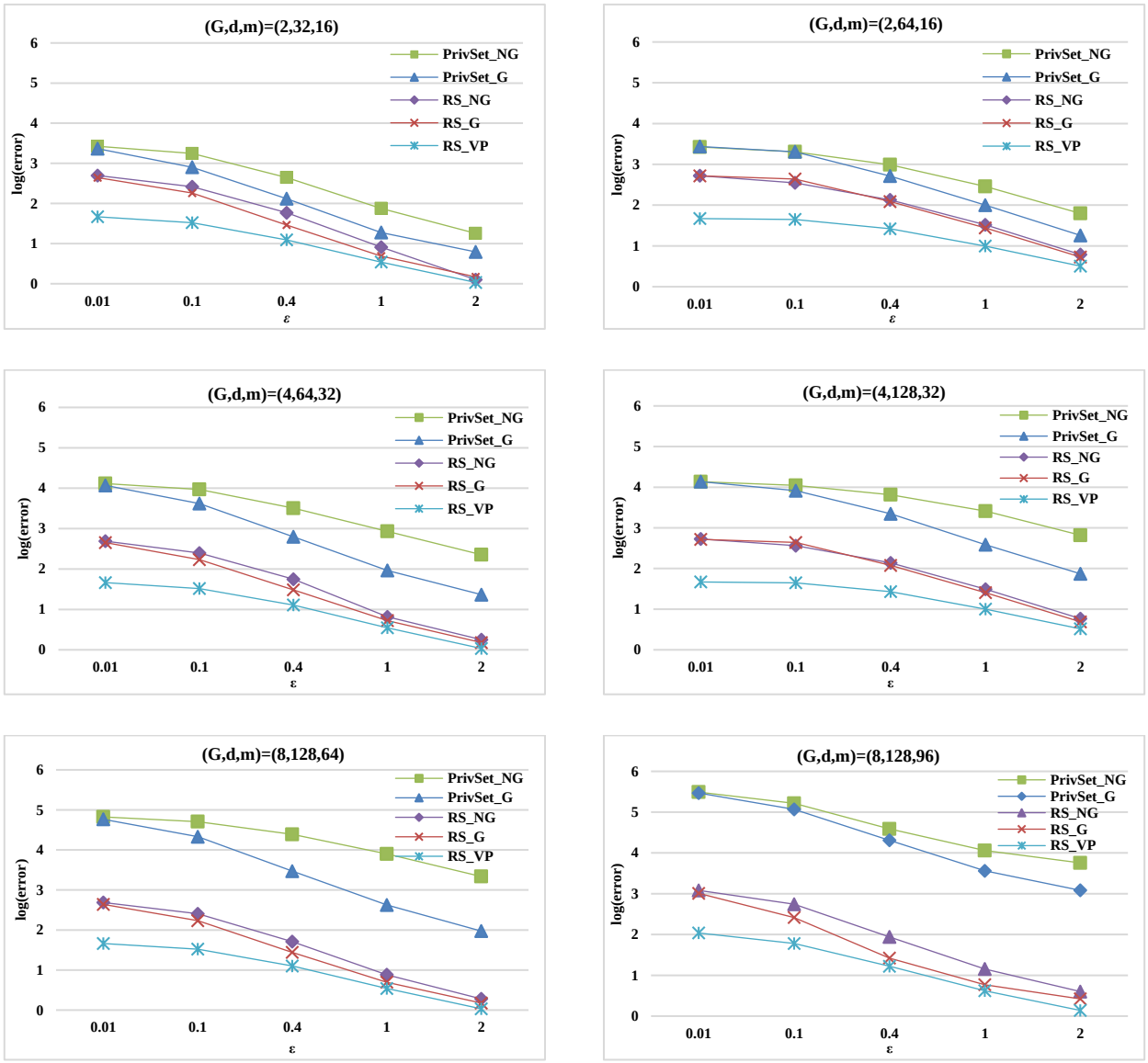

Figure 12. Influence of VP on MSE.

By integrating the CP, VP, and RS proposed in this paper, a set-valued data collection method

based on category level, SetLDP, is formed. Figure 13 verifies the impact of privacy parameters of the three different algorithms on error bounds. In the experiment (g, d, m) = (4,64,32). In the left figure of Figure 13 we set the privacy parameter ε_2 of VP as 1, and the privacy parameter ε_1 of CP as [0.1, 0.5, 1, 2, none]. Among all the value of ε_1, the result with no privacy parameter is the baseline for comparison, and its utility is the best because it has not been disturbed by CP. From the left figure in Fig.6-8, it can be found that different categories have a great influence on utility because the privacy parameter controls the degree of privacy protectionprivate information. The category changes from exists (1) - > does not exist (0). A loss of all the information leads to the item values of users being eliminated, which will affect the results of CP and RS.

Table 14. Differences and improvements obtained by using RS_Direct vs. PrivSet.

Name	Algorithm	Differences and Improvements	Advantages
Utility Function	RS_Direct	Intersection of candidate set s and user's real data t_c under category c $\lvert s \cap t_c \rvert / \lvert t_c \rvert$	better data utility
	PrivSet	Whether there is intersection: $\left[\hat{t} \cap \hat{s} \neq \varnothing\right]$	
Sampling Probability	RS_Direct	$e^{\frac{-\varepsilon_3 \cdot (\lvert s \cap t_c \rvert / \lvert t_c \rvert)}{2}} \Big/ \Omega$	The greater the intersection number, the better the utility
	PrivSet	$e^{\varepsilon \cdot [\hat{t} \cap \hat{s} \neq \varnothing]} \Big/ \Omega$	
A new utility function leads to the normalization factor, the real rate, and the false positive rate			
Ω	RS_Direct	$e^{\frac{-\varepsilon_3}{2} \cdot 0} \cdot C_d^k + \sum_{inter=1}^{k} \left(e^{\frac{-\varepsilon_3 \cdot (k-inter)}{2}} \cdot \left(C_m^{inter} \cdot C_d^{k-inter} \right) \right)$	
	PrivSet	$C_d^k + \exp(\varepsilon) \cdot \left(C_{d+m}^k - C_d^d \right)$	
TPR	RS_Direct	$\dfrac{e^{\frac{-\varepsilon_3 \cdot k}{2}} \cdot \left(C_d^{k-1} \right) + \sum_{inter=2}^{k} \left(e^{\frac{-\varepsilon_3 \cdot (k-inter)}{2}} \cdot \left(C_{m-1}^{inter-1} \cdot C_d^{k-inter} \right) \right)}{\Omega}$	
	PrivSet	$\dfrac{e^{\varepsilon} \cdot C_{d+m-1}^{k-1}}{C_d^k + e^{\varepsilon} \cdot \left(C_{d+m}^k - C_d^k \right)}$	
FPR	RS_Direct	$\dfrac{e^{\frac{-\varepsilon_3}{2} \cdot 0} \cdot C_{d-1}^{k-1} + \sum_{inter=1}^{k-1} \left(e^{\frac{-\varepsilon_3 \cdot (k-inter)}{2}} \cdot \left(C_m^{inter} \cdot C_{d-1}^{k-1-inter} \right) \right)}{\Omega}$	
	PrivSet	$\dfrac{C_{d-1}^{k-1}}{\Omega} + \exp(\varepsilon) \cdot \dfrac{k \cdot \left(C_{d+m}^k - C_d^k \right) - m \cdot C_{d+m-1}^{k-1}}{d \cdot \Omega}$	
Conclusion		Theoretical and experimental results show that RS_Direct is better than PrivSet	

In the right part of Figure 13, the privacy parameter ε_1 of CP is 1, and the privacy parameter ε_2 of VP is [0.1, 0.5, 1, 2, none]. As described above, the result with no privacy parameter is the baseline for comparison. It can be found that the effect of different privacy parameters ε_2 on utility is relatively small, and the interval between curves is relatively narrow.

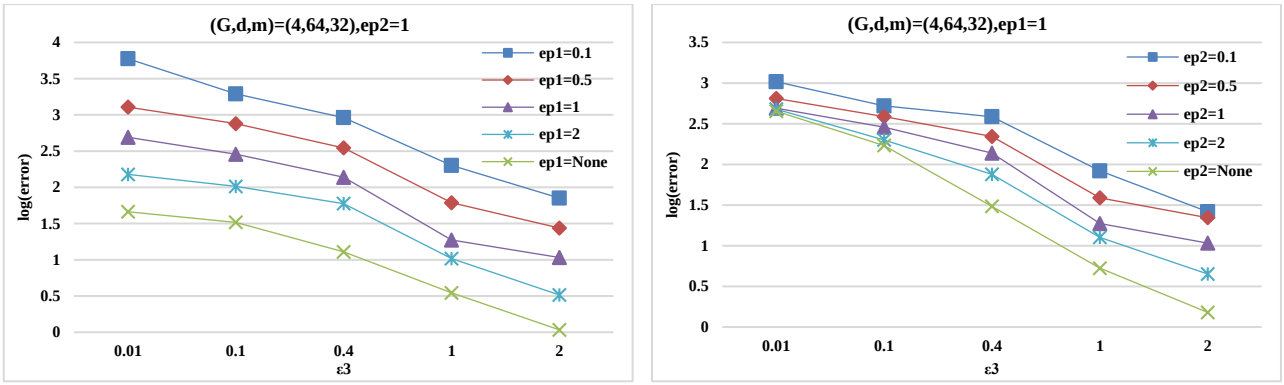

Figure 13. Distribution estimation of item support count.

6.7. Improvement of RS_DIRECT compared with PrivSet

Inspired by [24], the RS_Direct algorithm proposed in this paper improves the PrivSet method as showed in Table 14. The main differences between the two methods are the utility function, TPR, and FPR. Our theoretical analysis and experimental results show that RS_Direct is better than PrivSet overall.

7. Conclusions

Set-valued data is an important data type and has a wide range of application scenarios, such as recommendation systems, shopping analysis, and user behavior analysis. In order to protect the private information such as category, length, and content of set-valued data, we propose a set-valued data privacy-preserving collection method. Based on the local differential privacy model, a new utility function and sampling method are designed, and experimental results show that they are better than other state-of-the-art methods such as PrivSet and BRR. As for future work, some research directions are as follows. 1) We can apply the SetLDP method to the collection of trajectory data privacy protection. 2) From Tables 12 and 13, we can see that the k value of the RS method proposed in this paper is larger than that of PrivSet, which means that more original information is preserved; hence, we can consider frequent itemset mining in the future. 3) Since the CP algorithm disturbs the existence of a category, a lot of information is lost in value counting; inspired by [33], future work can consider optimization after the CP algorithm to reduce the loss of information. 4) Experimental analysis shows that the RS algorithm proposed in this paper is better than the PrivSet algorithm, mainly because its error boundary is lower; however, it is difficult to prove this theoretically because the minimum value of the error boundary is determined by the finite countable k value, which is obtained through experiments, and therefore we should conduct further theoretical analysis on the error boundary.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (61702119); Science and Technology Program of Guangzhou (201804010236); Guangdong Basic and Applied Basic Research Foundation (2019A1515012048); Young creative talents project of Guangdong Provincial Education Department (natural science), 2016KQNCX092.

The authors would like to thank the editor and the anonymous reviewers for their constructive

comments and suggestions, which improve the quality of the paper. We thank LetPub (www.letpub.com) for its linguistic assistance during the preparation of this manuscript.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. Y. He, J. F. Naughton, Anonymization of set-valued data via top-down, local generalization, *Pro. VLDB Endow.*, **2** (2009), 934–945.
2. S. L. Wang, Y. C. Tsai, T. P. Hong, Anonymizing set valued social data, in *Proceedings of 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, IEEE, (2010), 809–812.
3. R. Chen, B. C. M. Fung, N. Mohammed, B. C. Desai, K. Wang, Privacy-preserving trajectory data publishing by local suppression, *Inform. Sci.*, **231** (2013), 83–97.
4. G. Ghinita, Y. Tao, P. Kalnis, On the Anonymization of sparse high-dimensional data, in *Proceedings of the 24nd International Conference on Data Engineering*, IEEE, (2008), 715–724.
5. J. Cao, P. Karras, C. Raïssi, K. L Tan, ρ-uncertainty: inference-proof transaction anonymization, *Proc. VLDB Endowment*, **3** (2010), 1033–1044.
6. M. Terrovitis, N. Mamoulis, P. Kalnis, Privacy-preserving anonymization of set-valued data, *Pro. VLDB Endowment*, **1** (2008), 115–125.
7. Y. Xu, K. Wang, W. C. Fu, P. S. Yu, Anonymizing transaction databases for publication, in *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, (2008), 767–775.
8. L. Sweeney, *k*-Anonymity: A model for protecting privacy, *Int. J. Uncertainty Fuzziness Knowl. Based Syst.*, **5** (2002), 557–570.
9. A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam, *l*-diversity: privacy beyond *k*-anonymity, *ACMT. Knowl. D.*, **1** (2017), 1–3.
10. R. Chen, B. Fung, B. C. Desai, Differentially private trajectory data publication, preprint, arXiv:1112.2020.
11. J. Ouyang, J. Yin, S. P. Liu, Y. B. Liu, An effective differential privacy transaction data publication strategy, *J. Comput. Res. Dev.*, **51** (2014), 2195–2205.
12. J. Ouyang, J. Yin, S. P. Liu, Differential privacy publishing strategy for distributed transaction data, *J. Software*, **26** (2015), 1457–1472.
13. G. Fanti, V. Pihur, U. Erlingsson, Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries, *Proc. Privacy Enhancing Technol.*, **2016** (2016), 41–61.
14. U. Erlingsson, V. Pihur, A. Korolova, RAPPOR: randomized aggregatable privacy–preserving Ordinal Response, in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, (2014), 1054–1067.
15. S. L. Warner, Randomized response: a survey technique for eliminating evasive answer bias, *J. Am. Stat. Assoc.*, **60** (1965), 63–69.
16. C. Sun, Y. Fu, J. Zhou, H. Gao, Personalized privacy-preserving frequent itemset mining using randomized response, *J. Sci. World*, **2014** (2014), 1–10.

17. A. V. Evfimievski, R. Srikant, R. Agrawal, J. E. Gehrke, Privacy preserving mining of association rules, *J. Inform. Syst.*, **29** (2004), 343–364.
18. Q. Ye, X. Meng, M. Zhu, Z. Huo, Survey on local differential privacy, *J. Software*, **7** (2018), 159–183.
19. B. Ding, J. Kulkarni, S. Yekhanin, Collecting telemetry data privately, preprint, arXiv:1712.01524.
20. P. Kairouz, K. Bonawitz, D. Ramage, Discrete distribution estimation under local privacy, in *International Conference on Machine Learning*, (2016), 2436–2444.
21. J. C. Duchi, M. I. Jordan, M. J. Wainwright, Local privacy and minimax bounds: sharp rates for probability estimation, preprint, arXiv:1305.6000.
22. M. Andrés, N. Bordenabe, K. Chatzikokolakis, C. Palamidessi, Geo-indistinguishability: differential privacy for location-based systems, in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, ACM, (2013), 901–914.
23. N. Bordenabe, K. Chatzikokolakis, C. Palamidessi, Optimal geo-indistinguishable mechanisms for location privacy, in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, ACM, (2014), 251–262.
24. S. W. Wang, L. S. Huang, Y. W. Nie, P. Wang, H. Xu, W. Yang, PrivSet: set-valued data analyses with local differential privacy, in *Proceedings of the 2018 IEEE INFOCOM Conference on Computer Communications*, IEEE, (2018), 1088–1096.
25. J. Hsu, S. Khanna, A. Roth, Distributed Private Heavy Hitters, in *International Colloquium on Automata, Languages, and Programming*, Springer, Berlin, Heidelberg, (2012), 461–472.
26. Z. Qin, Y. Yin, T. Yu, I. Khalil, X. K. Xiao, 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*, ACM, (2016), 192–203.
27. R. Bassily, A. Smith, Local, private, efficient protocols for succinct histograms, in *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, ACM, (2015), 127–135.
28. M. E. Gursoy, A. Tamersoy, S. Truex, W. Wei, L. Liu, Secure and utility-aware data collection with condensed local differential privacy, preprint, arXiv:1905.06361.
29. W. Wang, M. A. Carreira-Perpiñán, Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application, preprint, arXiv:1309.1541.
30. C. W. Lin, T. P Hong, H. C. Hsu, Reducing side effects of hiding sensitive itemsets in privacy preserving data mining, *Sci. World J.*, **2014** (2014), 235837.
31. J. C. Lin, P. Fournier-Viger, L. Wu, W. Gan, Y. Djenouri, J. Zhang, PPSF: an open-source privacy-preserving and security mining framework, in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, IEEE, (2018), 1459–1463.
32. X. T. Wang, H. F. Zhu, A novel two-party key agreement protocol with the environment of wearable device using chaotic maps, *Data Sci. Pattern Recognit.*, **3** (2019), 12–23.
33. Q. Q. Ye, H. B. Hu, X. F. Meng, H. D. Zheng, PrivKV: key-value data collection with local differential privacy, in *2019 IEEE Symposium on Security and Privacy (SP)*, IEEE, (2019), 294–308.

AIMS Press