Streaming Data Collection with a Private Sketch-based Protocol

Ying Li, Xiaodong Lee, Botao Peng, Themis Palpanas, Jingan Xue

***Abstract*—Data stream collection is critical to analyze service conditions and detect anomalies in time, especially in Internet of Things. However, it may undermine the individual privacy. Local differential privacy (LDP) has recently become a popular privacy-preserving technique protecting users' privacy. However, most of them are still limited to the assumption of one-item collection, resulting in poor utility when extended to the multi-item collection from a very large domain. This paper proposes a private streaming data collection framework, PSF, which takes advantage of sketches. Combining the proposed background information and a decode-first collection-side workflow, the framework improves the utility by reducing the errors introduced by the sketching algorithm and the privacy budget utilization when collecting multiple items. We analytically prove the superior accuracy and privacy characteristics of PSF. In order to support specific computing tasks, we build two private protocols based on PSF, PrivSketch and PrivSketch+, aiming at frequency estimation and mean estimation, respectively. We demonstrate the utility of PrivSketch and PrivSketch+ theoretically, and also evaluate them experimentally. Our evaluation, with several diverse synthetic and real datasets, demonstrates that PrivSketch is 1-3 orders of magnitude better than the competitors in terms of utility in both frequency estimation and frequent item estimation, while being up to ∼100x faster. PrivSketch+ performs ∼4 orders of magnitude better than advanced solutions, such as Piecewise Mechanism (PM) and Hybrid Mechanism (HM), under a limited privacy budget.**

***Index Terms*—LDP, Sketch, Frequency estimation, Mean estimation.**

I. Introduction

Motivation. Privacy protection issues in data stream collection have attracted attention, typically in Internet of Things (IoT) [1]–[3]. The data stream collection has become widespread for the purpose of analysis and services [4]–[6]. For instance, by gathering passive DNS traffic, security service providers can figure out domains related with domain generation algorithms (DGA) that are usually used in botnets, and then attach them to blacklists to block access to these domains [7]. Although such an approach results in more convenient and secure services for users, it may undermine their privacy [2], [3]. For example, domain names in the DNS traffic reflect the individuals' network activities, including the websites that the users visit, which can expose the users' daily routines and interests. Given the increased awareness regarding privacy protection, several regulations and laws have been promulgated, such as the GDPR [8] in the European Union and the Personal Information Protection Law [9] in China, which further restrict data collection without privacy protection.

Local Differential Privacy (LDP) has emerged as a widely adopted technique for preserving individual privacy during data collection. Data is locally perturbed by users before being transmitted, eliminating the need for trust in the collector. Without knowing the actual values of individual users, the collector can obtain approximate statistics, mitigating the risk of privacy breaches. In LDP, a parameter denoted as ϵ is used to quantify the amount of perturbation applied to the data. This parameter plays a crucial role in determining the level of privacy protection and the utility achieved by the privacy-preserving algorithm. LDP does not rely on any trust, leading several technology companies (such as Apple [10], Google [11], Microsoft [12]) to adopt it in their applications.

Utility Problem. Several research have dedicated efforts to developing LDP solutions for private data collection. However, these approaches often encounter limitations when applied to the collection of data streams, due to the following reasons: (i) The data stream is heterogeneous among different users. The heterogeneity refers to the varying size of data items generated by different users, also known as data length. Some works [12]–[14] assume to collect only one item in an interval, which is inconsistent with the real-world situation in data stream collection. Some works [15]–[17] represent the data stream as set-valued data and unify the data size by Padding and Sampling [15], where the unified size L needs to be predefined or estimated. However, the stream generated in different intervals by one user can also be heterogeneous, which means the predefined or estimated L in a collection is not universal and additional estimation is required for each collection, thus affecting the efficiency. (ii) The domain of items in data streams is often vast, such as URLs and IP addresses. The large cardinality of these domains results in high computation and

This work was supported in part by National Natural Science Foundation of China under Grant No. 62202450; in part by Huawei under Grant No. TC20201119008; in part by Postdoctoral Exchange Program No. YJ20210185. *(Corresponding authors: Xiaodong Lee, Botao Peng)*

Ying Li is with the Lab for Internet Infrastructure, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: liying18z@ict.ac.cn).

Xiaodong Lee is with the Lab for Internet Infrastructure, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the Fuxi Institution, Beijing 100192, China (e-mail: XL@ict.ac.cn).

Botao Peng is with the Lab for Internet Infrastructure, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: pengbotao@ict.ac.cn).

Themis Palpanas is with LIPADE, Université Paris Cité, Paris 75006, France, and also with the French University Institute (IUF), Paris 75006, France (e-mail: themis@mi.parisdescartes.fr).

Jingan Xue is with Huawei Technologies, Shenzhen 518129, China (e-mail: xuejingan@huawei.com).

communication expenses, along with increased perturbation errors. Although several existing research have focused on frequency estimation [16], [17] and mean estimation [14] for multi-dimensional data collection, these approaches are still not suitable for large-cardinality domains.

The sketching technique is among the most widely used to provide an efficient data structure for streaming data processing, which can also play an important role in designing a practical LDP protocol for data stream collection. The Private Count-Mean Sketch (PCMS) algorithm, introduced by Apple [10], demonstrates the effectiveness of sketches in reducing the large cardinality under LDP. This is particularly useful in scenarios where only a small fraction of items from a vast domain are accessed by individual users, i.e., the data of each user is sparse. In addition, sketches have the additional advantage of unifying the user data length, thus avoiding the extra cost introduced by Padding and Sampling [15]. However, the PCMS algorithm is limited to data collection for single items. When extending it to support data stream collection, several challenges arise: (i) The sketching algorithm under LDP introduces extra errors which can not be disregarded. The collision errors in sketches increase with the size of each user's dataset because the probability of data hashed to the same position positively correlates with the size of dataset [18]. When the collector collects perturbed sketches from users under LDP, all these sketches are aggregated directly, similar to encode all data into a single sketch, resulting in a rise in collisions. Therefore, a solution is required to lessen collisions without resorting to giant sketches, which would raise communication costs. (ii) Without an upper bound on the amount of data generated by users, the number of different counters between two users can be as large as the size of sketches. To preserve user-level privacy, it is necessary to allocate the privacy budget among the counters in the sketch. However, this allocation may introduce significant inaccuracies and compromise the utility of the collected data. So, to reduce estimation errors, efficient privacy budget utilization is required.

Our solution. To meet the above challenges, we introduce PSF, a privacy-preserving sketch-based framework for data stream collection, aimed at achieving high utility. PSF introduces a novel approach to encoding multiple items into a single sketch, without unifying the size of items of each user beforehand. An innovative workflow is introduced for the collector-side under LDP, where the perturbed sketch is decoded before aggregation and calibration. The workflow is distinct from a conventional LDP protocol that aggregates and calibrates first, thereby avoiding collision errors when directly aggregating all perturbed local sketches. Furthermore, PSF leverages the ordering matrix taken from the original sketch, in order to protect each user's sketch privacy while allowing the collector to extract the minimum index information (cf. proof in Section IV-D). This is the first effort to use background information to improve utility, which effectively mitigates the errors when estimating the minumum. Additionally, PSF transmits reasonably accurate information with a constrained budget for privacy by utilizing the sampling technique to enhance the information utilization in the sketch. As a result, the error arising from uniformly allocating the privacy budget is reduced. Overall, PSF enhances the utility of the private data stream collection. Based on our framework, we propose PrivSketch for the problem of frequency estimation, and PrivSketch+ for the problem of mean estimation, in the context of data streams. We demonstrate the effectiveness of our designs both theoretically and experimentally. Our private streaming data collection protocols used for frequency and mean estimation form the fundamental technology for data analysis. They can further aid in detecting anomalies, monitoring traffic patterns, and identifying data trends. As a result, these protocols hold great potential for widespread application in various monitoring scenarios across different domains, such as network security, transportation systems.

Contributions. A summary of our contributions[1] is presented following.

- We design a novel private data collection framework, PSF, for streaming data collection, which utilizes sketching techniques. PSF is the first sketch-based privacy-preserving framework that considers the errors introduced by the sketching algorithm. It adopts a different strategy from traditional LDP: it decodes before aggregating, based on the analysis of these errors. Additionally, to reduce the perturbation errors and improve the utility, PSF leverages background information to keep minimum index and employs the sampling technique to improve information utilization. Based on PSF, we propose PrivSketch and PrivSketch+, in order to perform frequency and mean estimation, respectively, in data streams.
- We demonstrate that the utilization of the ordering matrix as background information in PSF does not compromise the privacy of individuals, through a rigorous analysis and proof (cf. Section IV-D). Our work introduces a novel concept of the indistinguishable input set, wherein the collector is unable to differentiate any two values contained in the set. Our results show that by incorporating appropriate additional background information, the utility of LDP algorithms can be enhanced without compromising the privacy of the users.
- We conducted comprehensive experiments to show the effectiveness of our proposed solutions, PrivSketch and PrivSketch+, on both synthetic and real datasets. In our experiments, these solutions are compared with state-of-art algorithms. Results demonstrate that PrivSketch exhibits significantly improved utility compared to existing algorithms for frequency estimation and frequent item estimation, with accuracy enhancements ranging from 1 to 3 orders of magnitude. Furthermore, the speed of PrivSketch is observed up to $\sim$100x faster. When privacy-preserving requirements are strict (i.e., privacy budget is small), PrivSketch+ performs about 4 orders of magnitude better than its advanced mean estimation competitors, Piecewise Mechanism (PM) and Hybrid Mechanism (HM).

II. Background and Problem Statement

A. Local Differential Privacy

Differential privacy (DP) [19] is a solid privacy-preserving technology, of which privacy protection is quantified by precise mathematical proofs. DP assumes that the third-party

[1] A preliminary version of this work has appeared elsewhere [1].

collector is trustworthy, i.e., the collector processes original data from users without disclosing users' data or using them for malicious activities. However, third-party collectors in the real world are sometimes linked to data breaches, due to attacks or internal malicious behavior. Therefore, LDP [20] was proposed, which does not rely on a trusted collector. Under LDP, the data is perturbed before sending to the collector and the original data remains locally. Its formal definition follows:

Definition 1 (ϵ-Local Differential Privacy [20]): A randomized algorithm $\mathcal{M}$ satisfies ϵ-local differential privacy ($\epsilon > 0$), if and only if for any two input tuples $x, x' \in \mathcal{D}$ and output y, then $\frac{\Pr[\mathcal{M}(x)=y]}{\Pr[\mathcal{M}(x')=y]} \leq e^{\epsilon}$.

Intuitively, the collector cannot infer the value of the original data after perturbation, which means the input by the user becomes indistinguishable from the collector. The degree of indistinguishability is determined by the value of the privacy budget ϵ. When $\epsilon = 0$, the original value is entirely indistinguishable. The perturbation makes the data totally deviate from the original value, with no utility. When ϵ becomes larger, the original data become easier to distinguish, and the higher the utility. One of the main properties of LDP is the sequential composition mechanism, which can be used to decompose a complex LDP algorithm into a sequence of simple algorithms.

Theorem 1 (Sequential Composition Mechanism [21]): Assume a randomized algorithm $\mathcal{M}$ consists of a sequence of randomized algorithms $\mathcal{M}_i (1 \leq i \leq t)$. When for each $i, \mathcal{M}_i$ satisfies ϵ_i-LDP, $\mathcal{M}$ satisfies $\sum_{i=1}^{t} \epsilon_i$-LDP.

Theorem 1 indicates that the problem of collecting data streams can be split into multiple sub-problems of collect statistics about only one item in the stream.

Randomized Response (RR) [11], [22]. One of the fundamental mechanisms for achieving LDP is RR. It allows users not always provide the truth, i.e., original value. Specifically, in the case of binary values, users can answer truthfully with a certain probability p, or respond with the opposite value with a probability $q = 1 - p$. To make RR satisfy ϵ-LDP:

$$\frac{\max \Pr[\mathcal{M}(x) = y]}{\min \Pr[\mathcal{M}(x') = y]} = \frac{p}{1-p} = e^{\epsilon}, \tag{1}$$

therefore $p = \frac{e^{\epsilon}}{1+e^{\epsilon}}$. Denote the percentage of 1 received by the collector as f. To obtain the estimation of the true percentage of input $x = 1$, according to the perturbation probability, the collector can calibrate f as $\frac{f}{2p-1} + \frac{1-p}{2p-1}$.

Piecewise Mechanism (PM) [14]. This is an advanced perturbation mechanism for mean estimation. In frequency estimation, the frequency of each user is a binary value. Different from this, in mean estimation, the value of each user is numerical and within a specific range, which is assumed to be $[-1, 1]$. PM confines the perturbed range of the value and divides the range into three "pieces". It perturbs the value to the "piece" which is close to the original value with a high probability, in order to minimize variance. Formally, the Probability Density Function (PDF) of perturbed value y is

$$pdf(\mathcal{M}(x) = y) = \begin{cases} p, & y \in [\ell(x), r(x)], \\ \frac{p}{e^{\epsilon}}, & y \in [-C, \ell(x)) \cup (r(x), C], \end{cases} \tag{2}$$

where

$$\begin{aligned} p &= \frac{e^{\epsilon} - e^{\epsilon/2}}{2e^{\epsilon/2} + 2}, \\ C &= \frac{e^{\epsilon} + e^{\epsilon/2}}{e^{\epsilon} - e^{\epsilon/2}}, \\ \ell(x) &= \frac{C+1}{2} x - \frac{C-1}{2}, \\ r(x) &= \ell(x) + C - 1. \end{aligned}$$

Thus, the perturbed value belongs to $[-C, C]$, where $[l(x), r(x)]$ is the center "piece" close to the original value x, and $[-C, \ell(x))$ and $(r(x), C]$ are the other two "pieces" further away from x. The perturbed value y is sampled from $[l(x), r(x)]$ with a high probability p, and from $[-C, \ell(x))$ and $(r(x), C]$ with a low probability $\frac{p}{e^{\epsilon}}$.

Hybrid Mechanism (HM) [14]. Prior to PM, Duchi et al. [23] and Nguyên et al. [24] proposed to discretize the numerical value x to 1 or -1 (with a probability $\frac{1+x}{2}$ and $\frac{1-x}{2}$ respectively), and then use RR to solve the problem of mean estimation. Under a small privacy budget ϵ, Duchi et al's method has a smaller variance, even though in most cases it is larger than that of PM. Therefore, HM proposes to combine the advantages of these two solutions. When $\epsilon \leq 0.61$, the solution proposed by Duchi et al is adopted. When $\epsilon > 0.61$, PM is chosen with a probability of $1 - e^{-\epsilon/2}$, and Duchi et al's solution is chosen with a probability of $e^{-\epsilon/2}$.

B. Sketching

Data streams are often sparse, which include only a small fraction of the items from a large domain. Thus, to count the occurrences of items, an efficient data structure is required. A typical solution is to compress items to a smaller domain, e.g., with sketching [25], which usually uses the matrix of size $K \times M$ to store the statistics of streams. Sketching is a two-phase process. In the update phase, items from the large domain (size d) are mapped into a smaller domain (size M) using K hash functions. The corresponding counters are updated accordingly. In the query phase, the items' counts are estimated using the counters associated with the original items.

Count-Min Sketch (CMS) [18] is one of the efficient sketching technique. Formally, CMS is represented by a matrix X of K rows and M columns, serving as counters. Each row i is associated with one hash function H_i, which maps items from the domain $\{1, \ldots, d\}$ to $\{1, \ldots, M\}$. Note these hash functions are pairwise-independent. When any item x from $[d]$ is observed in the stream, the following updates are performed: $X_{k,H_k(x)} = X_{k,H_k(x)} + 1, \forall 1 \leq k \leq K$. When querying the count $c(x)$ of the item x, its estimation $\widetilde{c}(x)$ is [18]:

$$\widetilde{c}(x) = \min_{1 \leq k \leq K} X_{k,H_k(x)}. \tag{3}$$

Private Count-Mean Sketch (PCMS-Mean) [10] is an LDP algorithm proposed by Apple for obtaining private data counts. It utilizes a Count-Mean Sketch matrix X to store the counts of data items. To protect users' privacy, each user generates updated data locally and perturbs them. Then, the collector only receives the perturbed data. Specifically, K hash functions are shared among users and the collector. For each item x in the data stream, the user chooses one of K hash functions, denoted by H_k, to encode the update. After encoding, the

kth row of the sketch is updated, where only $X_{k,H_k(x)} = 1$, and the remaining positions are -1. The updated row $\hat{X}_k$ is sent to the collector after the client uses the random response mechanism to independently perturb each position in X_k to achieve the LDP. Subsequently, the collector sums up the locally perturbed row $\hat{X}_k$ indexed by the same k, then, obtains an aggregated matrix of the same size of the original matrix X. Finally, the item frequency is estimated by summing the corresponding counters across the rows of the aggregated matrix and averaging the sum. PCMS assumes only one item is collected for each user at intervals. Thus, the selected rows of different users differ at most 2 positions. When using RR to protect the privacy, based on Theorem 1, the perturbation probability is $\frac{e^{\epsilon/2}}{1+e^{\epsilon/2}}$.

C. Problem Statement

The focus of this paper is on the problems of frequency and mean estimation for data streams. The problem involves a collector that is curious but honest, and a set of users U of size n. Each user, U_i, generates a streaming data, which is presented as a set of items of length $L^{(i)}(L^{(i)} \geq 0)$ and denoted by $S^{(i)} = \{S_1^{(i)}, S_2^{(i)}, \ldots\}, |S^{(i)}| = L^{(i)}$. Each item $S_\ell^{(i)}(0 \leq \ell \leq L^{(i)})$ is discrete value and drawn from a large domain $\mathcal{D}$ of size $|\mathcal{D}| = d$, that is, $S_\ell^{(i)} \in \mathcal{D}$. The defined streaming data refers to continuously generated data, such as DNS queries, website clickstreams, and travel records [13].

Frequency Estimation. The item frequency means the ratio of the number of users who possess the item to the total number of users. Formally, $f(x)$ represents the frequency of item x ($x \in \mathcal{D}$), defined as:

$$f(x) = \frac{1}{n}|\{i|\exists \ell, 0 \leq \ell \leq L^{(i)}, S_\ell^{(i)} = x\}|. \tag{4}$$

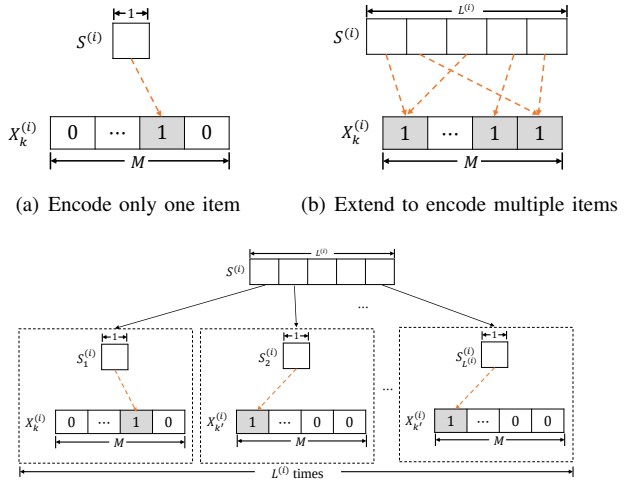
Mean Estimation. The mean estimation for each item $x \in \mathcal{D}$ is to estimate the average number of x possessed by n users. Formally, the mean is defined as follows:

$$v(x) = \frac{1}{n}\sum_{i=1}^{n}|\{l|0 \leq \ell \leq L^{(i)}, S_\ell^{(i)} = x\}|. \tag{5}$$

A summary of symbols is presented in Table I, which are listed along with their corresponding meanings.

TABLE I
NOTATION.

Symbol	Description
U	the user set
n	the size of the user set, $\|U\| = n$
U_i	the ith user
$S^{(i)}$	the set of data that U_i possess
$L^{(i)}$	the number of data that U_i possess, $\|S^{(i)}\| = L^{(i)}$
$S_l^{(i)}$	the lth data that U_i possess, $0 \leq \ell \leq L^{(i)}$
$\mathcal{D}$	the domain of data
d	the size of domain, $\|\mathcal{D}\| = d$
x	a value from $\mathcal{D}$, $x \in \mathcal{D}$
$f(x)$	the frequency of x
$v(x)$	the mean value of x
$\mathcal{M}$	the randomized algorithm used in LDP
$\mathbb{T}$	the computing task
$\Psi_\mathcal{M}, \Phi_\mathcal{M}$	the perturbed and calibrated algorithms used in $\mathcal{M}$
$U_\mathbb{T}, Q_\mathbb{T}$	a pair of encoding and decoding algorithms for $\mathbb{T}$
$\mathcal{G}$	the ordering matrix generation algorithm
$H_1, H_2, \ldots, H_K$	hash functions used by the sketching algorithm
M	the domain size of hash functions
$X^{(i)}$	the original sketch representing $S^{(i)}$
$\hat{X}^{(i)}$	the perturbed sketch of $X^{(i)}$
$I^{(i)}$	the index of selected counter $\hat{X}_{k,m}^{(i)}$
$O^{(i)}$	the ordering matrix of $X^{(i)}$
$c(x)$	the actual count of x
C_{max}	the upper bound of $c(x)$
$\tilde{c}(x)$	the count of x estimated by querying CMS
$Q(x)$	the perturbed count of x by querying perturbed sketches



(a) Encode only one item (b) Extend to encode multiple items

(c) Execute multiple times

Fig. 1. Solutions based on PCMS-Mean. (a) Existing PCMS-Mean (each user produces one item). When the user produces multiple items, (b) extends PCMS-Mean to encode multi-items, and (c) just executes PCMS-Mean multiple times.

III. Baseline Solutions

We describe a naive solution to our problem obtained by extending PCMS-Mean [10] (see Section II-B). Recall that PCMS-Mean uses a matrix $X^{(i)}$ of size $K \times M$ to encode data, and a randomized response algorithm to perturb a selected row $X_k^{(i)}$ in the sketch matrix. It assumes each user only possesses one item, $L^{(i)} = 1$. In our problem, where $L^{(i)} \geq 0$, PCMS-Mean can be extended through multi-time executions using one-item encoding and one-time execution using multi-item encoding, as two naive solutions shown in Fig. 1.

The first solution is to execute the original PCMS-Mean multiple times, denoted by nPCMS-Mean. To keep the indistinguishability of different users, nPCMS-Mean unifies each user's data length as L by Padding and Sampling first and then performs PCMS-Mean L times. However, this solution requires multiple transmissions, and the estimation error will accumulate, resulting in low accuracy [15]. Besides, nPCMS-Mean allocates the privacy budget ϵ between L times. Thus, the perturbed probability p is set to $\frac{e^{\epsilon/2L}}{1+e^{\epsilon/2L}}$, because there are two positions to be protected each time as explained in Section II-B. Dividing the ϵ by L is a naive way to apply the sequential composition to satisfy LDP, resulting in further deterioration of the utility.

Another solution, denoted Multi-PCMS-Mean, is to encode multiple items using one sketch in PCMS-Mean. Since there is no limitation on the data length of each user and all the data will be encoded in one sketch, the updated positions

in the selected row X_k are between 0-M. Therefore, the difference between any two selected rows is at most M instead of 2. To protect the privacy of each position under ϵ-LDP, Multi-PCMS-Mean utilizes a naive way that ensures the perturbation for each position should satisfy $\frac{\epsilon}{M}$-LDP. Thus, according to Theorem 1, Multi-PCMS-Mean satisfies ϵ-LDP. By using Equation (1) (and replacing ϵ with ϵ/M), the perturbed probability p is set to $\frac{e^{\epsilon/M}}{1+e^{\epsilon/M}}$ in Multi-CMS-Mean. A large M can incur a low privacy budget allocated to each position, which means heavy perturbation and poor utility.

In terms of communication overhead, nPCMS-Mean costs L times of Multi-PCMS-Mean, because its multi-execution using one-item encoding leads to transferring perturbed row of size M with times of the unified data length L. It is possible that $2L \leq M$ makes the perturbation error in nPCMS-Mean smaller than in Multi-PCMS-Mean. However, collecting multiple times could accumulate perturbation errors and estimation errors, resulting in the worse performance of nPCMS-Mean, as shown in Section VII-A. Although Multi-PCMS-Mean is a better solution than nPCMS-Mean, the irrational allocation of the privacy budget makes it performs poorly, especially when the data length (i.e., L) and the size of hash dimension (i.e., M) is large. Furthermore, it is worth noting that the sketching algorithm under LDP introduces errors that cannot be ignored. As the large domain is encoded into a much smaller sketch, the probability of hash collisions are large [10]. When the number of users increases, since all their data are encoded into the same sketch, the probability of the collisions happening becomes higher and the sketching errors grow larger. The sketching algorithm itself also affects the accuracy of estimations. Different algorithms have varying properties and characteristics that can influence the estimation errors. Research [26] shows, compared to Count-Mean Sketch, Count-Min Sketch has a smaller estimation error. Therefore, we propose that LDP algorithms take advantage of the Count-Min Sketch and use the privacy budget efficiently to address the problem described in Section II-C.

IV. Private Sketch-based Framework

We propose *Private Sketch-based Framework (PSF)*, which combines CMS with a randomized mechanism $\mathcal{M}$ in LDP, for streaming data collection. Unlike a naive approach that we call PCMS-Min, which directly replaces the sketching algorithm in PCMS-Mean with the Count-Min Sketch, PSF proposes a novel decode-first workflow for the collector (cf. Section IV-B) to reduce collisions in sketches, and utilizes the ordering matrix (cf. Section IV-C) to decide on the minimum index to use for its estimations. In order to enhance the utility, PSF also incorporates a sampling technique to provide more useful information under the strict perturbation.

The procedure of PSF is shown in Fig. 2. PSF consists of four components: encoder, perturber, decoder, and calibrator. For each user, the encoder encodes multiple items from the original data stream using the Count-Min Sketch first. Then, the perturber samples one of the counters in the sketch and perturbs the sampled counter. Subsequently, the perturber computes the counter orders of the original sketch $X^{(i)}$ and generates the ordering matrix $O^{(i)}$, which is sent with the perturbed count $\hat{X}^{(i)}_{k,m}$ and the sampled index $I^{(i)} = (k, m)$ to the collector. For the collector, the decoder recovers the value from $[1, m]$ back to $[1, d]$ and infers the perturbed count for item $x \in \mathcal{D}$ based on the $\hat{X}^{(i)}_{k,m}$ received, which corresponds to the query procedure in a sketching algorithm. Specifically, the decoder calculates the minimum index for each item x based on the ordering matrix $O^{(i)}$. If there is an item x' satisfying $H_k(x') = m$, where (k, m) is the index of the sampled perturbed counter $\hat{X}^{(i)}_{k,m}$, the decoder updates its corresponding counters. Afterwards, the calibrator aggregates the count calculated from each user's data and calibrates the perturbation error to accurately estimate the items' frequency.

A. Overview

Formally, there is a pair of encoding and decoding algorithms $\langle U_{\mathbb{T}}, Q_{\mathbb{T}} \rangle$ for different tasks $\mathbb{T}$, a pair of algorithms $\langle \Psi_{\mathcal{M}}, \Phi_{\mathcal{M}} \rangle$ for randomized mechanisms $\mathcal{M}$, and an ordering matrix generation algorithm $\mathcal{G}$.

- $\langle U_{\mathbb{T}}, Q_{\mathbb{T}} \rangle$ are algorithms used to encode streaming data $S^{(i)}$ from the original domain $\mathcal{D}$ into Count-Min Sketch $X^{(i)}$, and restore each item count $c(x)(x \in \mathcal{D})$ from perturbed sketch $\hat{X}^{(i)}$ to the original domain. The encoding and decoding processes correspond to the pair of update and query algorithms in CMS (cf. Section II-B). We define it as follows: $X^{(i)} = U_{\mathbb{T}}(S^{(i)}), c^{(i)}(x) = Q_{\mathbb{T}}(\hat{X}^{(i)}, x)$. Note the design of encoding/decoding algorithms is slightly different on different computing tasks, and the details are discussed in Section V-A and Section VI-A.
- $\langle \Psi_{\mathcal{M}}, \Phi_{\mathcal{M}} \rangle$ are algorithms used to perturb and calibrate the count in a randomized mechanism $\mathcal{M}$ (i.e. RR, PM).
- $\mathcal{G}$ is an algorithm used to generate the ordering matrix $O^{(i)}$ that records the ordering of counters in original sketches $X^{(i)}$; $\mathcal{G}$ is designed to keep the estimation unbiased (we discuss $\mathcal{G}$ in detail in Section IV-C).

The PSF can be described as follows, which consists of two phases:

- **User-side Phase:** According to the computing task $\mathbb{T}$ (i.e., frequency estimation, or mean estimation), each user transforms their original data streams into a sketch using $U_{\mathbb{T}}$. Then, the user utilizes $\Psi_{\mathcal{M}}$ to perturb the sampled counter with privacy budget ϵ at position (k, m) in CMS. The process can be defined as:
$$\Psi_{\mathcal{M}(\epsilon)}(U_{\mathbb{T}}(\cdot), k, m).$$
At the same time, the user calculates the ordering matrix using (the matrix generation algorithm) $\mathcal{G}$ based on the sketch, that is:
$$\mathcal{G}(U_{\mathbb{T}}(\cdot)).$$
Each user needs to execute the above processes locally.
- **Collector-side Phase:** After collecting the perturbed counter and ordering matrices of all users, the collector first decodes the counts using $Q_{\mathbb{T}}$, and then performs calibration and estimation using $\Phi_{\mathcal{M}(\epsilon)}$, defined as follows:
$$\Phi_{\mathcal{M}(\epsilon)}(Q_{\mathbb{T}}(\cdot)) \cdot K \cdot M,$$
where $K \times M$ is the size of the sketch, and the factor of $K \cdot M$ is used for calibration due to sampling.

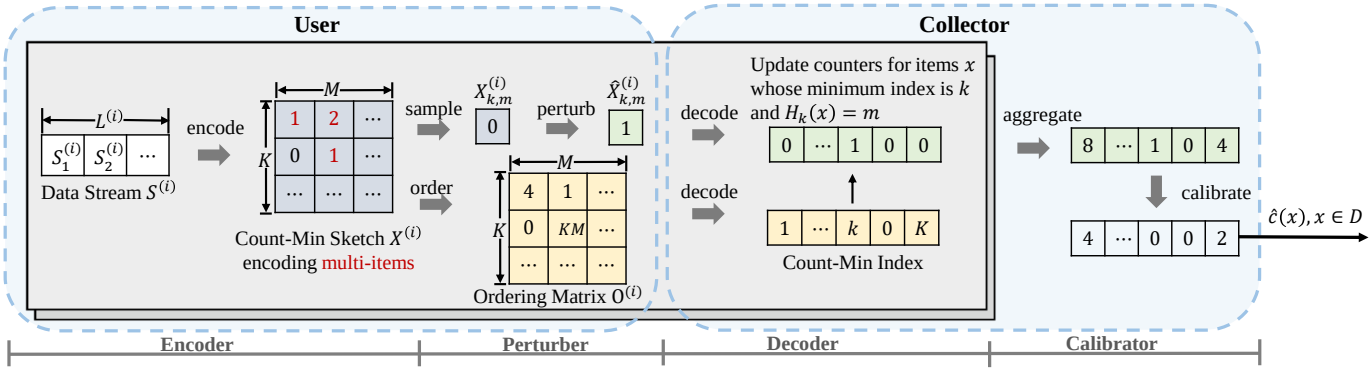


Fig. 2. Overview of PSF.

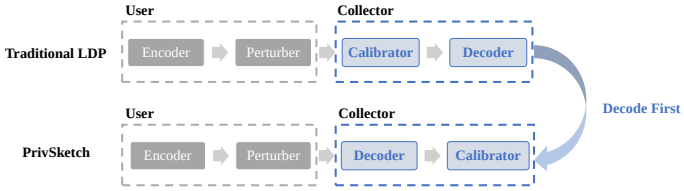


Fig. 3. Different workflows between PrivSketch with traditional protocols.

The framework is shown in Algorithm 1. It has one part for users (lines 1-5) and the other for the collector (lines 6-11). We elaborate on the details in the following sections.

B. Decoding-First Collector-Side Workflow

An important characteristic of PSF is the decoding-first feature on the collector side, which plays a crucial role in mitigating collisions in the private sketching protocol. Unlike many LDP protocols that overlook the decoding algorithm implemented by the collector, PSF recognizes its importance and includes it as an important part of the procedure. As shown in Fig. 3, the user side includes an encoder and a perturbator, and the collector side includes a calibrator and a decoder.

Considering sketching procedures in the naive protocol, PCMS-Min (as the same as traditional LDP protocols), collisions can occur when encoding data streams and calibrating counts. When encoding data streams, as the user U_i possesses more items, there is a higher probability that more items are hashed into the same position, thus, the more collisions there will be. A good choice of the sketching parameters can reduce errors caused by this collision. When calibrating counts, the calibrator aggregates perturbed sketches from users, thus resulting in more collisions, which has the same effect of encoding all user data into a single sketch. To eliminate the possibility of collisions during the calibration step, PSF adopts a novel collector-side workflow where the perturbed data are decoded first before calibrating. The calibrator does not sum the sketches anymore and just aggregates the perturbed counts without collisions. To show the effectiveness of our decoding-first workflow, the theoretical proof is presented as follows.

Theorem 2: For estimating the count of a value $x \in \mathcal{D}$ using Count-Min Sketch, $\min_k \sum_{i=1}^{n} X^{(i)}_{k,H_k(x)}$ represents the results of aggregating sketches before decoding, $\sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)}$ represents the results of decoding sketches before aggregating, the following formula holds:

$$\min_k \sum_{i=1}^{n} X^{(i)}_{k,H_k(x)} \geq \sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)} \geq c(x) \tag{6}$$

where $c(x)$ represents the true count of x.

Proof: For each user U_i and any $1 \leq k \leq K$, $X^{(i)}_{k,H_k(x)}$ reflects the count of both x and $x'(x' \neq x)$, which are hashed into the same position with x.

$$X^{(i)}_{k,H_k(x)} = c^{(i)}(x) + \sum\{c^{(i)}(x') | x' \in S^{(i)}, H_k(x) = H_k(x')\}.$$

For the minimum index k where $X^{(i)}_{k,H_k(x)}$ is minimal, the equation above holds. As a result, $\sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)} = c(x) + \sum c(x') \geq c(x)$. Moreover, $X^{(i)}_{k,H_k(x)} \geq \min_k X^{(i)}_{k,H_k(x)}$ always holds. Thus, $\sum_{i=1}^{n} X^{(i)}_{k,H_k(x)} \geq \sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)}$, $1 \leq k \leq K$. Considering $\min_k$ is one of the case that belongs to $[1, K]$, we can conclude that $\min_k \sum_{i=1}^{n} X^{(i)}_{k,H_k(x)} \geq \sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)}$. ■

According to Theorem 2, when the estimation of our private protocol is unbiased (compared with the result of CMS without privacy protection), our decoding-first workflow on the collector side can reduce the errors. We elaborate on the approach to achieve unbiased estimation next.

Algorithm 1 PSF

Require: $\{S^{(1)}, S^{(2)}, \ldots, S^{(n)}\}, \epsilon, K, M, D \subset \mathcal{D}$
1: select a set of hash functions $\mathcal{H} = \{H_1, H_2, \ldots, H_K\}$
2: **for** each $i \in [1, n]$ **do**
3: $\quad I^{(i)}, \hat{X}^{(i)}_{I^{(i)}}, O^{(i)} \leftarrow$ PSF-User$_{\mathcal{M},\mathbb{T}}(S^{(i)}, \epsilon, n, K, M, \mathcal{H})$
4: $\quad$ send $I^{(i)}, \hat{X}^{(i)}, O^{(i)}$ to the collector
5: **end for**
6: set $\mathcal{I} \leftarrow \{I^{(1)}, I^{(2)}, \ldots, I^{(n)}\}$
7: set $\hat{\mathcal{X}} \leftarrow \{\hat{X}^{(1)}_{I^{(1)}}, \hat{X}^{(2)}_{I^{(2)}}, \ldots, \hat{X}^{(n)}_{I^{(3)}}\}$
8: set $\mathcal{O} \leftarrow \{O^{(1)}, O^{(2)}, \ldots, O^{(n)}\}$
9: **for** each $x \in D$ **do**
10: $\quad \hat{c}(x) \leftarrow$ PSF-Collector$_{\mathcal{M},\mathbb{T}}(x, \epsilon, n, M, \mathcal{H}, \mathcal{I}, \hat{\mathcal{X}}, \mathcal{O})$
11: **end for**
12: **return** $\{\hat{c}(x) | x \in D\}$

C. Ordering Matrix Generation

In PSF, the randomized response mechanism introduces variability to the minimum index of the perturbed count, thereby impeding unbiased estimation. For the collector, the estimation of each item x is obtained from perturbed CMS of each user U_i, i.e., $\hat{X}^{(i)}$. Denote the count of item x as $c(x)$ and the function used to calibrate $c(x)$ as h. Thus, the estimated count $\hat{c}(x)$ can be represented as $h(\sum_{i=1}^{n} \min_k \hat{X}^{(i)}_{k,H_k(x)})$. To ensure that the randomized mechanism utilized in PSF does not introduce any bias in the estimation, the expectation of the estimation on perturbed matrix (i.e., $\hat{c}(x)$) should be same as that of the estimation based on the original matrix (i.e., $\widetilde{c}(x) = \sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)}$). That is,

$$\mathbb{E}[\hat{c}(x)] = \mathbb{E}[h(\sum_{i=1}^{n} \min_k \hat{X}^{(i)}_{k,H_k(x)})] = \widetilde{c}(x) = \sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)}.$$

Denote the row indices of the minimum count for x in the perturbed and original sketches k' and k, respectively. Due to the randomization, these two row indices may be different. Without loss of generality, the opposite perturbed value of $X^{(i)}_{k',H'_k(x)}$ is represented as $aX^{(i)}_{k',H'_k(x)} + b$. When $k \neq k'$,

$$\begin{aligned}\mathbb{E}[\hat{X}^{(i)}_{k',H'_k(x)}] &= pX^{(i)}_{k',H'_k(x)} + q(aX^{(i)}_{k',H'_k(x)} + b) \\ &= (p + aq)X^{(i)}_{k',H'_k(x)} + bq \\ &\neq (p + aq) \min_k X^{(i)}_{k,H_k(x)} + bq,\end{aligned}$$

where p and q represent the probability of keeping the original value and flipping to the opposite value, respectively. It is challenging to construct a function h to transform the inequality into an equality, due to the varying gap depending on data distribution. To address this issue, the ordering matrix is proposed, which is designed to find accurate indices of the minimum and keep the frequency estimation unbiased.

The ordering matrix is generated at the user end, for providing accurate indices for minimum in CMS. It is a matrix with the same size as local sketch for each user U_i, denoted by $O^{(i)}$, consisting of count orders of corresponding counters in $X^{(i)}$. Firstly, we divide counters into different groups $G^{(i)}_v$ according to its count v. As a result, $G^{(i)}_v$ includes a set of counters $\{(k,m)|X^{(i)}_{k,m} = v\}$ and its length is denoted by $|G^{(i)}_v| = g_v$. Secondly, we bind each group G_v with its order range $R^{(i)}_v = [\sum_{v'<v} g_{v'}, \sum_{v'\leq v} g_{v'} + g_v]$. Thirdly, for each counter $X^{(i)}_{k,m}$ in $G^{(i)}_v$, we perform random sampling without replacement from the set of possible orders $R^{(i)}_v$. Finally, $O^{(i)}_{k,m}$ is updated with the sampled order value, denoted by $r^{(i)}_{k,m}$, i.e., $O^{(i)}_{k,m} = r^{(i)}_{k,m}$. As a result, the index of the minimum order in $O^{(i)}$ is same as that of the index of the minimum value in $X^{(i)}$, which is helpful for keeping the estimation unbiased.

Example 1. *An example of ordering matrix generation is shown in Fig. 4. To simply, assume in the original matrix $X^{(i)}$, all counters are either* 0 *(i.e., item does not appear in this client) or* 1 *(i.e., item appears in this client). When the counters are arranged in ascending order, the orders* $[0, 3]$ *correspond to the counters with value* 0*, and orders* $[4, 8]$ *correspond to those with value* 1*. The order of counters with the same value is shuffled, to avoid the collector finding the rule and guessing the true value of counters; this does not affect the correctness of the computation of the minimum values. Finally, the indices of counters with their corresponding orders are sent to the collector without the original value.*

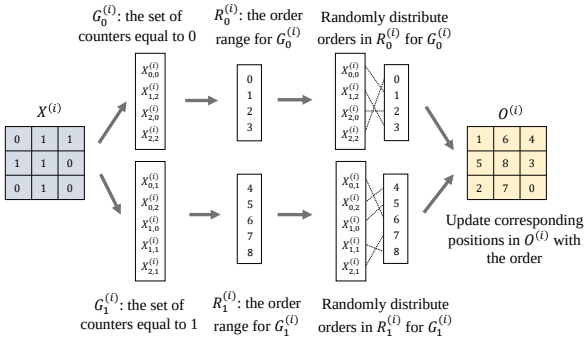


Fig. 4. The process of generating the ordering matrix.

We analyze how the ordering matrix affects privacy next.

D. Privacy Analysis

Considering the collector-side protocol without the ordering matrix, the user only sends a sampled perturbed counter $\hat{X}^{(i)}_{k,m}$ to the collector. When the user perturbs the original counter $X^{(i)}_{k,m}$ to the opposite value with a randomized mechanism $\mathcal{M}$ that satisfies ϵ-LDP, the privacy is guaranteed. However, in PSF, the user sends the perturbed counter $\hat{X}^{(i)}_{k,m}$ along with the corresponding ordering matrix $O^{(i)}$, which may expose individual private information to the collector. Thus, it is important to analyze the potential influence of $O^{(i)}$ on privacy.

The ordering matrix $O^{(i)}$ can be utilized to exclude some possible inputs for the collector, but the collector still cannot distinguish some inputs. Assuming values of all counters are either 1 or 0, if two positions satisfy the order $O^{(i)}_{k,m} \leq O^{(i)}_{k',m'}$, it is not possible for $X^{(i)}_{k,m} = 1$ and $X^{(i)}_{k',m'} = 0$ to be correct simultaneously. This implies that the ordering matrix provides background information about the counters in the original sketch. From the collector's perspective, the information reduces some possible cases of input sketches. Specifically, the cases where $X^{(i)}_{k,m} = 1$ and $X^{(i)}_{k',m'} = 0$ are excluded, resulting in only three remaining possibilities (cf. Fig. 5). To quantify the effect, the concept of *indistinguishable input set* is introduced. Formally, let T denote the indistinguishable input set, which means any two inputs from T is indistinguishable for the collector. For classical LDP protocol, by its definition, for any two input x, x' from the possible input set $\mathcal{D}$ and output y, $\frac{\Pr[\mathcal{M}(x)=y]}{\Pr[\mathcal{M}(x')=y]} \leq e^{\epsilon}$ is satisfied. Therefore, its indistinguishable input set T includes all values in $\mathcal{D}$. The presence of background information, such as the ordering matrix, may reduce the size of the indistinguishable set. However, it does not affect the fundamental principle of LDP and the remaining inputs in the smaller indistinguishable set still adhere to the principle of indistinguishability. Because LDP ensures that the privacy of individual inputs is protected, irrespective of any additional knowledge available to an adversary. We show in Theorem 3 that the background information affects the size of the indistinguishable set.

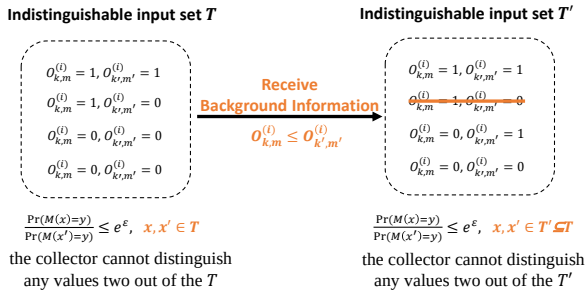


Fig. 5. Example of the effect of background information on indistinguishable input set.

Theorem 3: Consider a mechanism $\mathcal{M}$ that satisfies ϵ-LDP, its indistinguishable input set T, and any two inputs x, x'. When the collector receives any output y, along with the background information I, there exists an indistinguishable set $T' \subseteq T$ satisfying the following inequality: $\frac{\Pr[M(x)=y]}{\Pr[M(x')=y]} \le e^{\epsilon}, \quad x, x' \in T'$.

Proof: For any I, T can be divided into two parts, T_+ and T_-. The former represents the inputs that are consistent with the information I, i.e., the possible inputs when I is true. The latter includes the inputs that contradict the information I, that is, the impossible inputs when I is true. Based on I, the collector can infer that the original input belongs to $T_+ (\subseteq T)$. For any two inputs $x, x' \in T_+$, x, x' is also in T. Therefore, following the definition of ϵ-LDP, $\frac{\Pr[M(x)=y]}{\Pr[M(x')=y]} \le e^{\epsilon}$ is satisfied and any two input $x, x' \in T'$ is distinguishable. ■

The indistinguishable input set T' computed by the ordering matrix $\mathcal{O}$ is enough to protect the privacy of users. In PSF, the user data are encoded in the sketch $X^{(i)}$, thus, to maintain the privacy of users' data is to protect the sketch $X^{(i)}$. The collector should be prevented from obtaining exact counts in the original sketch. Intuitively, the collector only receives a perturbed value and its order, thus, still lacks enough information to infer the true value of each counter. The collector may be able to narrow down the potential sketches to a smaller set $T' \in T$ other than the whole set of possible input sketches. However, the collector still cannot infer the exact values of any counter in the original sketch. In the simplest situation where the possible value for each counter is 1 or 0 thus only two groups for counters, G_1 and G_0, and $g_1 + g_0 = KM$, there are $KM+1$ possible inputs in terms of the different sizes of each group. Thus, there is no counter whose value is always the same in different possible inputs. Sometimes, there are some constraints in the sketching algorithm. For example, it is impossible that $g_1 = 1, 2, 3$, because when there is an item occurred, for each $k \in [1, K]$, $\exists (k, m) \in G_1, m \in [1, M]$. Nevertheless, $\{0\}^{KM}, \{1\}^{KM} \in T'$ always holds. Thus, there are still multiple possible original values for each counter. The collector has no way to know which of the possible values is the true value. If there are more than two possible values for counters, the situation becomes more complicated, creating more obstacles for inferring the original value. The limited ordering information in the ordering matrix has no deterministic information regarding the sketch value. As a result, the original values of the counters cannot be inferred and the user privacy is effectively protected.

V. Frequency Estimation Solution: PrivSketch

In this section, we describe our protocol, PrivSketch, which derives from PSF and is designed for frequency estimation. To showcase its practicality, we conduct a thorough analysis in terms of the privacy and utility.

A. Protocol

The design of the encoding algorithm $U_{\mathbb{T}}$. Different from the traditional CMS, in PrivSketch, each user is required to maintain local records indicating whether a specific item x appears. This modification is made to achieve the objective of obtaining the item frequency rather than their counts. As a result, each update performed by the encoding algorithm $U_{\mathbb{T}}$ is an logical OR operation, rather than an integer addition. Assuming the initial value of each position $X_{k,m}$ is *False* (i.e., 0), there are two cases for updating. When x is the first value hashed to $X_{k,m}$, the update is $X_{k,m} = X_{k,m} \vee True = False \vee True = True$. When other values x' are hashed to the same position, the update is $X_{k,m} = X_{k,m} \vee True = True \vee True = True$. Thus, each counter in X has a value of 1 (i.e., $True$) or 0 (i.e., $False$).

The choice of the randomized mechanism $\mathcal{M}$. Since only the sampled counter needs to be perturbed, PrivSketch directly chooses RR as $\mathcal{M}$ for randomization:

$$\hat{X}^{(i)}_{k,m} = \begin{cases} 2X^{(i)}_{k,m} - 1, & \text{w.p. } \frac{e^{\epsilon}}{e^{\epsilon}+1} \\ -2X^{(i)}_{k,m} + 1, & \text{w.p. } \frac{1}{e^{\epsilon}+1} \end{cases} \tag{7}$$

The original value (0 or 1) of counters in $X^{(i)}_{k,m}$ are perturbed to 1 or -1 randomly. The details of the protocols follow:

User-side protocol (Algorithm 2). It consists of an encoder (lines 1-4) and a perturber (lines 5-13). In the encoder, each user first encodes the input sequence $S^{(i)}$ into a CMS matrix $X^{(i)}$ (lines 2-3). Each encoding updates the value of the corresponding position of items in $X^{(i)}$ to $True$ (i.e., 1). Then, the perturber computes the ordering matrix (line 5). It also uniformly samples k and m from $[1, K]$ and $[1, M]$ and perturbs the sampled counter $X^{(i)}_{k,m}$ using RR (as in Equation (7)). Lastly, instead of transmitting the entire sketch matrix, only the selected counter along its index and the ordering matrix are communicated after the perturbation process.

Collector-side Protocol (Algorithm 3). To estimate the frequency of the item x, the first step is to find the minimum counter after perturbation related to the item. The index of the minimum is computed by comparing the orders of corresponding counters in the ordering matrix (line 4). Subsequently, the calibrator utilizes the perturbed minimum value to estimate the frequency by calibrating its perturbation error (line 9). We present the privacy and utility proof for PrivSketch next.

B. Privacy and Utility Proof

Privacy Guarantee. We have proved that a PSF combined with a $\epsilon-$LDP perturbation mechanism $\mathcal{M}$ is privacy-preserving (cf. Section IV-D). Therefore, we should prove that the RR used in PrivSketch satisfies $\epsilon-$LDP. In PrivSketch,

Algorithm 2 User-side protocol in PrivSketch

Require: $S^{(i)}, \epsilon, n, K, M, \mathcal{H}$
1: initialize a sketch $X^{(i)} \leftarrow \{0\}^{K\times M}$
2: **for** each $\ell \in [1, L^{(i)}]$,each $k \in [1, K]$ **do**
3: $\quad X^{(i)}_{k,H_k(s^{(i)}_\ell)} = 1$
4: **end for**
5: generate the ordering matrix $O^{(i)}$
6: uniformly sample k and m from $[1, K]$ and $[1, M]$ respectively
7: $I^{(i)} \leftarrow (k, m)$
8: then, uniformly sample r from $[0, 1]$
9: **if** $r < \frac{1}{e^\epsilon+1}$ **then**
10: $\quad \hat{X}^{(i)}_{I^{(i)}} = -2X^{(i)}_{I^{(i)}} + 1$
11: **else**
12: $\quad \hat{X}^{(i)}_{I^{(i)}} = 2X^{(i)}_{I^{(i)}} - 1$
13: **end if**
14: **return** $I^{(i)}, \hat{X}^{(i)}_{I^{(i)}}, O^{(i)}$

Algorithm 3 Collector-side protocol in PrivSketch

Require: $x, \epsilon, n, M, \mathcal{H}, \mathcal{I}, \hat{\mathcal{X}}, \mathcal{O}$
1: select a set of hash functions $\mathcal{H} = \{H_1, H_2, \ldots, H_K\}$
2: $Q(x) \leftarrow 0$
3: **for** each $i \in [1, n]$ **do**
4: $\quad k_{\min} \leftarrow \arg\min_k O^{(i)}_{k,H_k(x)}$
5: $\quad$ **if** $k_{\min} = k^{(i)}$ and $H_{k_{\min}}(x) = m^{(i)}$ **then**
6: $\quad\quad Q(x) \leftarrow Q(x) + \hat{X}^{(i)}_{k_{\min},H_{k_{\min}}(x)}$
7: $\quad$ **end if**
8: **end for**
9: $\hat{f}(x) \leftarrow \frac{KM}{2}(\frac{e^\epsilon+1}{e^\epsilon-1}\frac{Q(x)}{n} + 1)$
10: **return** $\hat{f}(x)$

$X_{k,m}$ keep its value with probability $\frac{e^\epsilon}{e^\epsilon+1}$ and turn to the opposite value with probability $\frac{1}{e^\epsilon+1}$. Then,

$$\frac{\Pr[\mathcal{M}(x) = y]}{\Pr[\mathcal{M}(x') = y]} \le \frac{\max \Pr[\mathcal{M}(x) = y]}{\min \Pr[\mathcal{M}(x') = y]} = \frac{\frac{e^\epsilon}{e^\epsilon+1}}{\frac{1}{e^\epsilon+1}} = e^\epsilon.$$

The mechanism satisfies $\epsilon-$LDP. In order to show the effect of the sampling technique on utility, we first analyze the utility of PrivSketch without sampling. That is, the user sends the whole matrix X to the collector after perturbation, and thus, each position should be protected. Privacy budget ϵ is equally distributed to each location and the perturbation probability of each location becomes $\frac{1}{e^{\epsilon/KM}+1}$. Its utility is as follows:

Theorem 4: Let $Q(x)$ denote the perturbed counts for each value in $\mathcal{D}$ inferred from perturbed sketches. $\hat{f}(x) = \frac{1}{2}(\frac{e^{\epsilon/KM}+1}{e^{\epsilon/KM}-1}\frac{Q(x)}{n} + 1)$ is an unbiased estimation of $\widetilde{f}(x) = \frac{1}{n}\sum_{i=1}^{n} \min_k X^{(i)}_{k,H_k(x)}$ which is the frequency inferred from the original count-min sketch. Furthermore, the variance of $\hat{f}(x)$ is $\frac{e^{\epsilon/KM}}{n(e^{\epsilon/KM}-1)^2}$.

Proof: For each user U_i, the counters for the item x in row k of perturbed sketch $\hat{X}$ is denoted by $\hat{X}^{(i)}_{k,H_k(x)}$, which value is determined by $X^{(i)}_{k,H_k(x)}$ (lines 9-13 in Algorithm 2). $X^{(i)}_{k,m}$ uses 1 and 0 to denote whether the corresponding values appears in this client, and $X^{(i)}_{k,m}$ is then used to compute the frequency by the collector. For $X^{(i)}_{k,H_k(x)} = 1$, $\mathbb{E}[\hat{X}^{(i)}_{k,H_k(x)}] = 2p - 1$. For $X^{(i)}_{k,H_k(x)} = 0$, $\mathbb{E}[\hat{X}^{(i)}_{k,H_k(x)}] = 2q - 1$. $Q(x)$, which represents the result by aggregating the perturbed counters at the minimum position $\min_k \hat{X}^{(i)}_{k,H_k(x)}(x)$, equal to $\sum_{i=1}^{n} \min_k \hat{X}^{(i)}_{k,H_k(x)}(x)$, satisfies:

$$\mathbb{E}[Q(x)] = \mathbb{E}[\sum_{i=1}^{n} \min_k \hat{X}^{(i)}_{k,H_k(x)}(x)] = 2(p-q)n\widetilde{f}(x) + (2q-1)n,$$

$$\begin{aligned}\mathrm{Var}[Q(x)] &= \mathrm{Var}[\sum_{i=1}^{n} \min_k \hat{X}^{(i)}_{k,H_k(x)}(x)] \\ &= 4n\{(p+q-1)(p-q)\widetilde{f}(x) + q(1-q)\},\end{aligned}$$

where $n\widetilde{f}(x)$ is the estimated number of users with x in their data streams using the set of original sketch $\mathcal{X}$. In our protocol,

$$p = \frac{e^{\epsilon/KM}}{e^{\epsilon/KM}+1}, q = \frac{1}{e^{\epsilon/KM}+1}. \tag{8}$$

$$\hat{f}(x) = \frac{1}{2}(\frac{e^{\epsilon/KM}+1}{e^{\epsilon/KM}-1}\frac{Q(x)}{n} + 1). \tag{9}$$

Thus, the expectation of $\hat{f}(x)$, can be shown to be equal to $\widetilde{f}(x)$ as follows, which means the estimation is unbiased:

$$\mathbb{E}[\hat{f}(x)] = \frac{1}{2}(\frac{e^{\epsilon/KM}+1}{e^{\epsilon/KM}-1}\frac{\mathbb{E}[Q(x)]}{n} + 1) = \widetilde{f}(x). \tag{10}$$

By Definition (8), $p + q = 1$, $\mathrm{Var}[Q(x)] = 4nq(1-q)$. Thus, the variance of $\hat{f}(x)$ is satisfied:

$$\mathrm{Var}[\hat{f}(x)] = \frac{1}{4n^2}\frac{(e^{\epsilon/KM}+1)^2}{(e^{\epsilon/KM}-1)^2}\mathrm{Var}[Q(x)] = \frac{e^{\epsilon/KM}}{n(e^{\epsilon/KM}-1)^2}. \tag{11}$$

■

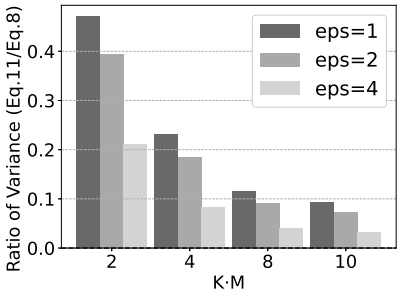


Fig. 6. Sampling effect on variance: variance ratio Eq.(14)/Eq.(11) when varying $K \cdot M$, with $n = 10^6$ and $r = 1$.

Sample the sketches. When the size of the sketch becomes larger, according to Equation (8), the original input becomes more likely to flip to other values, with a probability nearly to $\frac{1}{2}$, which means that the perturbed data are almost randomly chosen. Besides, according to Equation (11), larger values of K and M also lead to a higher variance. The reason is that the limited privacy budget is evenly allocated to each counter, thus, each counter is allocated less privacy budget with a larger size sketch, so that more randomness is introduced. Thus, the perturbed sketches become less informative, making it challenging to estimate the frequency for the collector. To mitigate the perturbation error and improve the efficiency of the sketching information utilization, this section introduces the sampling technique on the user end. In order to provide effective information within a limited privacy budget, a common approach is sampling. During sampling, a subset of counters from the total $K \cdot M$ counters are selected and the privacy budget is distributed across the smaller subset of counters, enabling a more effective privacy budget allocation.

Assuming r counters are sampled, each counter consumes $1/r$ of the privacy budget for perturbation. which is larger than $1/KM$. However, sampling introduces new variables for the estimation, which can be reflected in the variance as follows:

$$\mathrm{Var}[\hat{f}(x)] = \frac{KMe^{\epsilon/r}}{nr(e^{\epsilon/r}-1)^2}. \tag{12}$$

As the number of samples increases, the variance will increase [27]. Therefore, when $r = 1$, the variance is minimal. The perturbation probability in the RR is set to: $p = \frac{e^\epsilon}{e^\epsilon+1}, q = \frac{1}{e^\epsilon+1}$. The expectation and the variance now are:

$$\mathbb{E}[\hat{f}(x)] = \frac{KM}{2}\left(\frac{e^\epsilon+1}{e^\epsilon-1}\frac{\mathbb{E}[Q(x)]}{n}+1\right) = \widetilde{f}(x), \tag{13}$$

$$\mathrm{Var}[\hat{f}(x)] = \frac{KMe^\epsilon}{n(e^\epsilon-1)^2}. \tag{14}$$

Compared to Equation (10), randomly sampling one of the KM counters promotes the privacy budget allocated for each position from ϵ/KM to ϵ and requires to introduce a factor of KM to calibrate the estimation. Moreover, compared to the protocol without the sampling technique, the variance in Equation (14) is directly proportional to K and M, which increases at a slower rate compared to the exponential increase in Equation (11). As $K \cdot M$ increases, the advantage of the PrivSketch sampling technique increases, as shown in Fig. 6.

The error of PrivSketch is influenced not only by the perturbation but also by the collision error inherent in the CMS. As K and M increase, the collision error tends to decrease. Additionally, the collision error is also affected by the data domain size d and its distribution [18]. This renders the task of obtaining the optimal sketching parameters very challenging. Though, we conduct experimental evaluations to assess the impact of varying K and M on frequency estimation in Section VII-B. Besides, in Section VII-A, we demonstrate the effectiveness of the sampling technique in sketches by performing a comparative analysis with traditional PSFO.

VI. Mean Estimation Solution: PrivSketch+

In this section, we present PrivSketch+ for mean estimation. This protocol is a variant of PrivSketch and also follows PSF.

A. Protocol

The design of the encoding/decoding algorithms $U_\mathbb{T}$ and $Q_\mathbb{T}$. In PrivSketch+, each user needs to record locally how many times x appears, i.e., the count of x, similarly to the traditional CMS. Thus, when x is hashed to $X_{k,m}$, the update is $X_{k,m} = X_{k,m} + 1$. The count is an integer larger than or equal to 0. However, for mean estimation under LDP, there is always an assumption that the value is in $[-1, 1]$. Thus, We should transform the count to $[-1, 1]$. Assume that the count has an upper bound C_{max}. To transform it into $[-1, 1]$, the first step is to prune the count exceeding the bound C_{max}. Then, the count in the counter $X_{k,m}$ should be projected to $[-1, 1]$. Denote the projection using $f : [0, C_{max}] \to [-1, 1]$, where $f(X_{k,m}) = 2 \times \frac{X_{k,m}}{C_{max}} - 1$. Correspondingly, in the decoding algorithm $Q_\mathbb{T}$, the collector needs to use the inverse operation $(\hat{X}_{k,m} + 1) \cdot \frac{C_{max}}{2}$ to restore the collected perturbed counts $\hat{X}_{k,m}$ to the original range.

Algorithm 4 User-side protocol in PrivSketch+

Require: $S^{(i)}, \epsilon, n, K, M, \mathcal{H}, C_{max}$
1: initialize a sketch $X^{(i)} \leftarrow \{0\}^{K\times M}$
2: **for** each $\ell \in [1, L^{(i)}]$,each $k \in [1, K]$ **do**
3: $\quad X^{(i)}_{k,H_k(s^{(i)}_\ell)} = X^{(i)}_{k,H_k(s^{(i)}_\ell)} + 1$
4: **end for**
5: generate the ordering matrix $O^{(i)}$
6: uniformly sample k and m from $[1, K]$ and $[1, M]$ respectively
7: $I^{(i)} \leftarrow (k, m)$
8: **if** $X^{(i)}_{I^{(i)}} > C_{max}$ **then**
9: $\quad X^{(i)}_{I^{(i)}} = C_{max}$
10: **end if**
11: $X^{(i)}_{I^{(i)}} = \frac{2X^{(i)}_{I^{(i)}}}{C_{max}} - 1$
12: $C = \frac{e^\epsilon+e^{\epsilon/2}}{e^\epsilon-e^{\epsilon/2}}$
13: $l = \frac{C+1}{2}X_{I^{(i)}} - \frac{C-1}{2}, r = l + C - 1$
14: then, uniformly sample r from $[0, 1]$
15: **if** $r < \frac{e^{\epsilon/2}}{e^{\epsilon/2}+1}$ **then**
16: $\quad$ uniformly sample a value from $[l, r]$ as $\hat{X}^{(i)}_{I^{(i)}}$
17: **else**
18: $\quad$ uniformly sample a value from $[-C, l] \cup [r, C]$ as $\hat{X}^{(i)}_{I^{(i)}}$
19: **end if**
20: **return** $I^{(i)}, \hat{X}^{(i)}_{I^{(i)}}, O^{(i)}$

Algorithm 5 Collector-side protocol in PrivSketch+

Require: $x, \epsilon, n, M, \mathcal{H}, \mathcal{I}, \hat{\mathcal{X}}, \mathcal{O}, C_{max}$
1: select a set of hash functions $\mathcal{H} = \{H_1, H_2, \ldots, H_K\}$
2: $Q(x) \leftarrow 0$
3: **for** each $i \in [1, n]$ **do**
4: $\quad k_{\min} \leftarrow \arg\min_k O^{(i)}_{k,H_k(x)}$
5: $\quad$ **if** $k_{\min} = k^{(i)}$ and $H_{k_{\min}}(x) = m^{(i)}$ **then**
6: $\quad\quad Q(x) \leftarrow Q(x) + (\hat{X}^{(i)}_{k_{\min},H_{k_{\min}}(x)} + 1) \cdot C_{max}/2$
7: $\quad$ **end if**
8: **end for**
9: $\hat{v}(x) \leftarrow \frac{KM}{n}Q(x)$
10: **return** $\hat{v}(x)$

The choice of the randomized mechanism $\mathcal{M}$. PrivSketch+ utilizes the advanced mean estimation protocol PM (cf. Equation (2)) to perturb and estimate the mean of the counts.

In the following paragraphs, we detail the protocols.

User-side protocol (Algorithm 4). Similar to PrivSketch, the user-side protocol consists of an encoder (lines 1-4) and a perturber (lines 5-19). However, this protocol is different in the following three aspects: (i) During encoding, PrivSketch+ utilizes the traditional CMS algorithm (cf. Section II-B), that is, when encoding each item $s^{(i)}_l$ in the sequence $S^{(i)}$, the count of the corresponding position $H_k(s^{(i)}_l)$ for each hash function k in the sketch $X^{(i)}$ is increased by 1 (line 3). (ii) After the encoding, the perturber normalizes the counts into the range $[-1, 1]$. To reduce the calculations, only sampled counters $X^{(i)}_{k,m}$ are processed (lines 8-11). Note C_{max} is a bound that most of the counts will not exceed. Thus, not all counts are necessarily less than C_{max}, and the exceeded counts need to be pruned (lines 8-10). (iii) During perturbation, PrivSketch+ uses PM (cf. Section II-A) instead of RR. Thus, the perturbation value $\hat{X}_{k,m}$ randomly comes from the range $[-C, C]$ (rather than being the opposite value of the original

one with a certain probability).

Collector-side protocol (Algorithm 5). Same as PrivSketch, the collector first obtains the minimum index based on the ordering matrix $\mathcal{O}$, then updates the counts of items with the minimum value located at (k, m), and finally calibrates the counts. Note when aggregating the counts, the collector should restore them from $[-1, 1]$ to its original range (line 6), which is a reverse operation of the projection in encoding.

B. Privacy and Utility Proof

Privacy Guarantee. Consider the worst case of the indistinguishability of two inputs, where the difference of the probability of two different inputs x and x' perturbed to the same value y is maximized. In the worst case, $y \in [l(x), r(x)]$ but $y \notin [l(x'), r(x')]$. For $y \in [l(x), r(x)]$, y is uniformly sampled from $[l(x), r(x)]$ with perturbation probability $\frac{e^{\epsilon/2}}{e^{\epsilon/2}+1}$ and sampling probability $\frac{1}{r(x)-l(x)}$. For $y \notin [l(x'), r(x')]$, y is uniformly sampled from $[-C, l(x')) \cup (r(x'), C]$ with perturbation probability $\frac{1}{e^{\epsilon/2}+1}$ and sampling probability $\frac{1}{C-r(x')+l(x')-(-C)}$. Then, the worst case is:

$$\frac{\max \Pr[\mathcal{M}(x) = y]}{\min \Pr[\mathcal{M}(x') = y]} = \frac{\frac{e^{\epsilon/2}}{e^{\epsilon/2}+1} \cdot \frac{1}{r(x)-l(x)}}{\frac{1}{e^{\epsilon/2}+1} \cdot \frac{1}{C-r(x)+l(x)-(-C)}} = e^{\epsilon}, \quad (15)$$

where $C, l(x), r(x)$ refer to Equation (2). Thus, we can conclude that for any inputs x and x', $\frac{\Pr[\mathcal{M}(x)=y]}{\Pr[\mathcal{M}(x')=y]} \leq e^{\epsilon}$.

Theorem 5: Let $Q(x)$ denote the perturbed counters for each value in $\mathcal{D}$. $\hat{v}(x) = KM \cdot Q(x)$ is an unbiased estimation of $\widetilde{v}(x) = \frac{C_{max}}{2n} \sum_{i=1}^{n} (\min_k X^{(i)}_{k,H_k(x)} + 1)$, which is the mean inferred from the original count-min sketch. Furthermore, the worst case of variance of $\hat{v}(x)$ is $\frac{4KMe^{\epsilon/2}}{3(e^{\epsilon/2}-1)^2}$.

Proof: In PrivSketch+, each counter $X^{(i)}_{k,m}$ is perturbed using PM, which satisfies $\epsilon-$LDP and achieves unbiased estimation. Thus, $E[\hat{X}^{(i)}_{k,m}] = X^{(i)}_{k,m}$. Since the ordering matrix guarantees that the index of the minimum value remains unchanged, $E[\min \hat{X}^{(i)}_{k,H_k(x)}] = \min X^{(i)}_{k,H_k(x)}$. Thus, when aggregating the perturbed counter for x:

$$\begin{aligned} E[Q(x)] &= \frac{1}{KM} E[\frac{1}{n} \sum_{i=1}^{n} (\min_k \hat{X}^{(i)}_{k,H_k(x)} + 1) \cdot C_{max}/2] \\ &= \frac{C_{max}}{2nKM} \sum_{i=1}^{n} (E[\min_k \hat{X}^{(i)}_{k,H_k(x)}] + 1) \\ &= \frac{C_{max}}{2nKM} \sum_{i=1}^{n} (\min_k X^{(i)}_{k,H_k(x)} + 1) = \frac{\widetilde{v}(x)}{KM}, \end{aligned}$$

where the sampling technique results in the aggregated count being only $\frac{1}{KM}$ of the mean $\widetilde{v}(x)$ estimated by the original sketches. For $\hat{v}(x) = KM \cdot Q(x)$, a factor KM calibrates the bias to obtained an unbiased mean estimation of $Q(x)$. Moreover, $\mathrm{Var}[\hat{X}^{(i)}_{k,H_k(x)}] = \frac{1}{e^{\epsilon/2}-1}(X^{(i)}_{k,H_k(x)})^2 + \frac{e^{\epsilon/2}+3}{3(e^{\epsilon/2}-1)^2}$. Due to $X^{(i)}_{k,H_k(x)} \in [-1, 1]$, the worst case of variance is $\frac{4e^{\epsilon/2}}{3(e^{\epsilon/2}-1)^2}$ and the variance of $\widetilde{v}(x)$ is:

$$\mathrm{Var}[\widetilde{v}(x)] = \frac{KMe^{\epsilon/2}}{3n^2(e^{\epsilon/2}-1)^2}.$$

■

TABLE II
DATASETS CHARACTERISTICS

Dataset	n	d	max	min	P_{90}
Kosarak	990002	41270	2498	1	15
AOL	521693	1632788	61932	1	62
Dataset1	100000	100000	117	1	78
Dataset2	10000	100000	123	1	80
Dataset3	1000000	100000	134	1	84
Dataset4	100000	20000	112	1	73
Dataset5	100000	40000	107	1	72
Dataset6	100000	60000	110	1	74
Dataset7	100000	80000	109	1	75

VII. EXPERIMENTAL EVALUATION

In this section, we conduct a comprehensive evaluation of the utility and running time of PrivSketch and PrivSketch+ on both synthetic and real datasets. We also analyze the impact of key parameters on their performance. To provide a thorough evaluation, we compare PrivSketch to the state-of-the-art PCMS-Mean [10], and PSFO [28] based on OLH [29] (denoted as PS-OLH) for frequency estimation. Additionally, we compare PrivSketch with SVIM [28], a two-phase heavy hitter discovery protocol, for identifying frequent items. We compare PrivSketch+ with the advanced mean estimation protocols PM [14] and HM [14]. We also compare PrivSketch+ with other sketch-based techniques combined with sampling, i.e., PM-PCMS-Mean extended by PCMS-Mean [10], which utilizes the same mean estimation protocol PM as PrivSketch+ to show the effectiveness of min estimation. Note that for all mean estimation solutions, a sampling algorithm is used.

Environment. We implement all LDP protocols in Python and conduct experiments on a server with 2 Intel Xeon 3206R Processors and 32G RAM running Centos. We repeat each experiment 10 times and report the average results

Datasets. We conduct experiments on 7 synthetic datasets and 2 real datasets with varying distributions and parameters. We adopt real datasets, Kosarak and AOL, which are widely employed in differential privacy studies [13], [14], to evaluate the effectiveness and utility of our algorithm. Furthermore, the inclusion of synthetic datasets enable us to figure out the algorithm's behavior under different parameters, facilitating a comprehensive evaluation. Table II shows the dataset details, including the number of users, the size of domains that items belong to, and the min, max, and top 90 percent of the users' input size.

• **Synthetic Datasets**: We generate two groups of synthetic datasets following Zipf distribution that the real data stream often conforms to. For the first group, we fix the domain size $d = 10^5$ and vary the number of users $n = 10^4$, $n = 10^5$ and $n = 10^6$. For the other group, we fix the number of users $n = 10^5$ and range the domain size d from $20,000$ to $100,000$ in increments of $20,000$.

• **Kosarak** [30]: This dataset contains click-stream data from a Hungarian online news portal, involving nearly 1M users and 40K items. Each line in the dataset includes a set of integers representing the clicked items an anonymized user accessed. We treat these values as a sequence, and the items appearing in all sequences constitute the large domain.

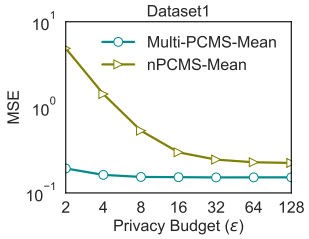


Fig. 7. Two naive solutions.

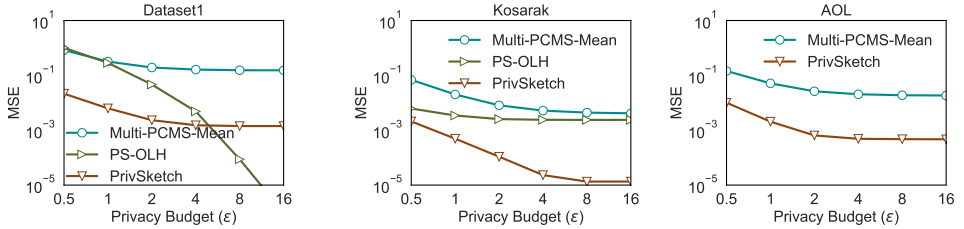


Fig. 8. Experimental results for frequency estimations.

• **AOL** [31]: This dataset contains search queries of users on AOL between March 1 and May 31, 2016, with corresponding URLs clicked by them. After removing users with no clicks, the dataset includes more than 500K users with 1.6 million distinct URLs.

Parameters. We set the following default sketching parameters for all sketches in our experiments. The number of hash functions K in the sketch is set to 4, and each hash function's hash domain size M is set to 128. The default privacy budget ϵ is 3, which is within the privacy budget range used in the experiments of existing LDP research [15], [26], [32].

Evaluation Measures. We use the following measures, including running time.

• **Mean Squared Error (MSE)**. We evaluate the accuracy of frequency and mean estimation using MSE. For frequency estimation, the MSE is $\frac{1}{d}\sum_{x\in\mathcal{D}}(\hat{f}(x)-f(x))^2$, where $f(x)$ is x's true frequency. For frequency estimation, the MSE is $\frac{1}{d}\sum_{x\in\mathcal{D}}(\hat{v}(x)-v(x))^2$, where $v(x)$ is x's true mean value.

• **Variance (Var)**. The Var measures the estimation error of the top-k frequency items: $\frac{1}{|C_e\cap C_t|}\sum_{x\in C_e\cap C_t}(n\hat{f}(x)-nf(x))^2$.

• **Normalized Cumulative Rank (NCR)**. We use NCR to measure the estimation accuracy of frequent items. NCR measures how many top-k items are identified by the protocol with a quality function $q(.)$, i.e., $\sum_{x\in C_e} q(x)/\sum_{x'\in C_t} q(x')$, C_t and C_e represents the true top-k items and the estimated top-k items respectively. For $x\in C_t$ with a rank i, $q(x)=k+1-i$. For $x\notin C_t$, $q(x)=0$.

A. Comparison with Advanced Protocols

Experiments on Frequency Estimation: We compare our protocol to two advanced solutions: (i) a sketch-based solution, Multi-PCMS-Mean, which is an extended version of PCMS-Mean [10] for multi-item collection. Section III discussed two naive solutions. However, the utility of the multi-time execution with one-item encoding is always worse than the one-time execution with multi-item encoding, as shown in Fig. 7, especially under a small privacy budget. Thus, to compare with the existing protocols in a meaningful way when measuring the MSE, we extend the existing PCMS-Mean to one-time execution with multi-item encoding. (ii) a non-sketch-based solution, PS-OLH, which is an advanced PSFO [28]. Padding and sampling is a popular method to ensure that each user contributes the same number of items. Based on this method and randomized mechanisms, PSFO [28] defines a frequency oracle. When the domain size is large, i.e., $d\geq 3e^{\epsilon}+2$, the optimal local hash (OLH) [28] is the best randomized mechanism for frequency estimation [29]. Therefore, we combine Padding and Sampling with OLH, denoted by PS-OLH. Note the privacy budget is only used to estimate frequency. We assume the distribution of user input length is known and the padding length l is set to the 90th percentile of the user input [15] (avoiding using the privacy budget for estimating l).

We evaluate the MSE of frequency estimation under different privacy budgets, varying from 0.5 to 16, on synthetic and real datasets. The results (cf. Fig. 8) show the better utility of PrivSketch than the other two competitors, especially when the privacy budget is small. This finding suggests the stronger privacy protection ability of PrivSketch, and its capacity to encode more information under a limited privacy budget. To prove the practicality of PrivSketch, we also conduct experiments on two real datasets, Kosorak and AOL. We observe that PrivSketch is always superior, and each LDP protocol performs similarly on the real and synthetic datasets. However, compared to synthetic datasets, a lower MSE is achieved on Kosorak and AOL. As shown in Fig. 10, we analyze the distribution of different datasets, including Kosorak, AOL, and synthetic Dataset1. The data distribution in Kosorak and AOL is more skewed than in synthetic datasets, which favors sketching techniques, especially the Count-Min Sketch.

Experiments on Frequency Item Estimation: Since frequency estimation is often used to find frequent items, we evaluate PrivSketch's performance in this task. We compare PrivSketch with SVIM [28], an advanced multi-phase protocol following LDPMiner [15] for large-domain data collection. As shown in Fig. 9, for top-k frequency estimation, the error of PrivSketch is lower than SVIM, especially for larger k. In frequent item identification, PrivSketch performs better, or very close to SVIM. The above results are expected, since the primary focus of PrivSketch is on estimating the frequency of items, instead of identifying frequent items.

Experiments on Mean Estimation: We compare the errors of mean estimation of PrivSketch+ to two advanced solutions, PM and HM, under privacy budgets varying from 0.5 to 16. Fig. 11 shows on synthetic datasets, PrivSketch+ always performs better than PM and HM, especially under small privacy budgets. We observe similar performance on the real dataset AOL. These results imply that our approach, the solution of compressing data from large domains into a sketch and then sampling, has better accuracy than direct sampling (PM and HM). In the PM and HM, only the count of the selected item is sampled. In the PrivSketch+, the sampled information is

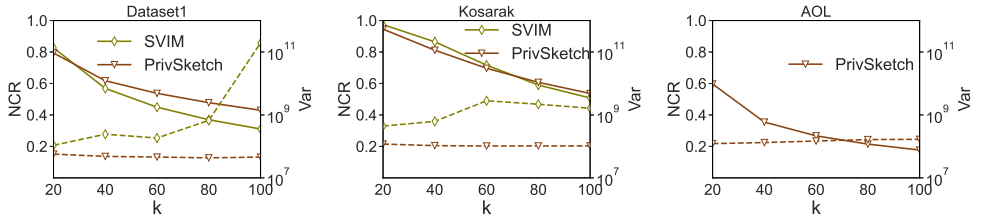


Fig. 9. VAR and NCR when varying parameter k.

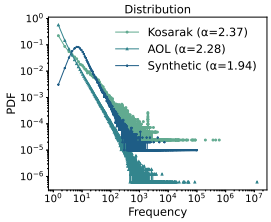


Fig. 10. Data distributions.

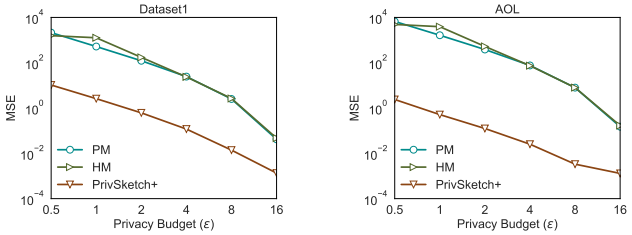


Fig. 11. Experimental results for mean estimations.

from a selected position in the sketch, which includes counts related to multiple items. Thus, the latter contains more useful information and results in higher accuracy.

B. Experiments with Different Parameters

In this section, we perform a comparative analysis of PrivSketch and PrivSketch+ against other sketch-based solutions, to present the effect of our design under different parameters. For frequency estimation, we compare not only PrivSketch to the extension of PCMS-Mean but also its min-estimation variant (referred to as Multi-PCMS-Min). Additionally, a middle version of PrivSketch without sampling (referred to as PrivSketch-noSmp) is also compared. These comparisons aim to demonstrate the superior utility of min estimation and the effectiveness of our decode-first and sampling design. For mean estimation, we compare PrivSketch+ to a simpler design called PM-PCMS-Mean. This alternative design utilizes sampling but lacks the min-estimation and the decode-first design. By making this comparison, we can assess the advantages and utility of the min-estimation and decode-first design incorporated in PrivSketch+.

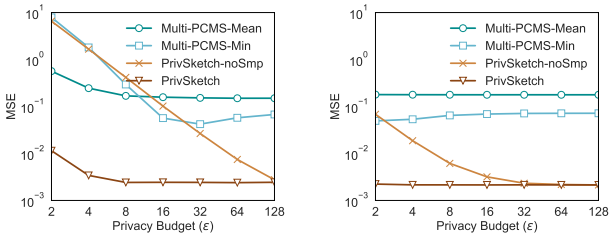
Utility with different number of users. In this experiment, we evaluate the estimation errors with increasing privacy budget under different numbers of users, using Dataset2 with 10^4 users and Dataset3 with 10^6 users. It should be noted the privacy budget range we set in the experiment is $[2, 128]$, which has far exceeded the budget used in practice [2], which is to showcase the effect of our designs.

For frequency estimation, Fig. 12 shows the superior utility of PrivSketch, which is not affected by n (similar to Multi-PCMS-Mean). Without considering the constraints of the privacy budget, PrivSketch with no sampling is also superior to the other two naive LDP protocols. This demonstrates our decode-first workflow and the ordering matrix contributes to mitigating collision errors and improving the accuracy of minimum estimation. However, PrivSketch-noSmp, like the Multi-PCMS-Min algorithm, has a large error under a small privacy budget, especially when n is also small. With the increase of n, the convergence rates of PrivSketch-noSmp are accelerated. As a result, it performs significantly better than the comparison algorithm when $\epsilon > 4$ with $n = 10^6$. As the privacy budget increases, different LDP protocols converge to a stable MSE, where PrivSketch $\approx$ PrivSketch-noSmp $>$ Multi-PCMS-Min $>$ Multi-PCMS-Mean. In practice, the privacy budget is usually less than 5. Under this constraint, except for PrivSketch, the other three protocols have too large errors to be used. For PrivSketch, the utility is guaranteed even if there are only 10^4 users.

For mean estimation, PrivSketch+ performs the best in most cases. The domain compression effect of sketching for mean estimation is also demonstrated. As shown in Fig. 13, the two sketch-based solutions, PrivSketch+ and PM-PCMS-Mean, are always better than the two non-sketch-based solutions, PM and HM. However, PrivSketch+ is sometimes slightly worse than PM-PCMS-Mean when the privacy budget is small. When n increases, this lag disappears. This is because of the error introduced by sampling. In PM-PCMS-Mean, the sampled count $X_{k,m}$ provides valid information for estimated counts of all items $x \in \mathcal{D}$ and $H_k(x) = m$. In PrivSketch+, the sampled count only supports valid information for an item if and only if the item satisfies more stringent requirements: it should hold that $H_k(x) = m$, and that $\arg\min_{k'} X_{k',H'_k(x)} = k$. That is, if the counter sampled by PrivSketch+ is not the minimum value of some items that are hashed to that position, it does not provide any useful information for the mean estimation of these items. Therefore, on a dataset with the same d and n, the amount of data collected by PrivSketch+ to support a specific item x is smaller than that of PM-Mean-PCMS, and the perturbation variance is larger. As n increases, i.e., on a dataset with $n = 10^6$, the variance of PrivSketch+ becomes smaller, and the effect of PrivSketch+ shows its advantages, performing always better than PM-PCMS-Mean.

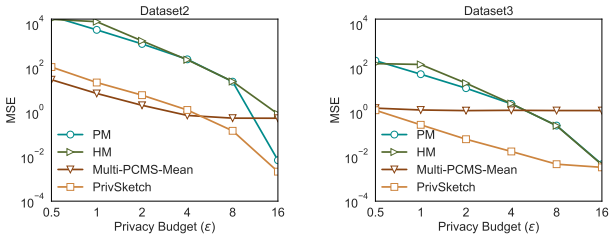
Impact of the parameters of the sketch. In the sketching technique, the original data will be encoded into a $K \times M$ matrix using K hash functions of size M. K and M are key parameters that affect the accuracy of the sketching technique. In Fig. 14, we evaluate the effects of these two parameters using a dataset with parameters $n = 10^5$, $d = 10^5$ under $\epsilon = 3$. Fig. 14(a) shows the effect of K by fixing $K = 4$ and varying $M \in \{4, 8, 16, 32, \ldots, 1024\}$. The results demonstrate

[2]Previous work [14], [26], [29] has for the most part used privacy budgets smaller than 5.



(a) $n = 10^4$ (b) $n = 10^6$

Fig. 12. MSE of frequency estimation with different n.



(a) $n = 10^4$ (b) $n = 10^6$

Fig. 13. MSE of mean estimation with different n.

that PrivSketch consistently exhibits superior utility compared to the other three protocols across different values of M. As expected, increasing the size of the hash vector (i.e., M) can reduce the estimation error. However, an interesting observation is that beyond a certain point, increasing M does not continue to decrease the error but actually leads to an increase. This can be attributed to the impact of M on two types of errors in the LDP protocol based on sketching techniques. For the Multi-PCMS-Mean, Multi-PCMS-Min, and PrivSketch-noSmp, M affects both the collision and perturbation probability. When M increases, the collision probability decreases, but the perturbation probability increases. Similarly, for PrivSketch, M influences both the sampling and collision errors. Therefore, an optimal M exists that minimizes the MSE, as shown in the figure. We also vary the number of hash functions $K \in \{2, 4, 8, 16, 32, \ldots, 256\}$ and fix the hash vector size M to 32 to conduct experiments for evaluating the influence of K. The MSE is plotted in Fig. 14(b). We observed a similar result to the previous experiment, which shows that the performance of PrivSketch is the best. However, different LDP protocols are affected differently by changes in K. The effect of K on Multi-PCMS-Min, PrivSketch-noSmp and PrivSketch protocols are the same as M, which impacts both errors caused by collision and perturbation or sampling. For the Multi-PCMS-Mean protocol, although each user only chooses one of the K hash functions to encode data, the effect is eliminated by summing their corresponding K counters during calibration. Thus, changes of K do not affect the MSE.

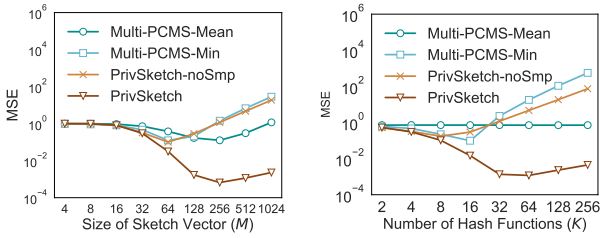
Impact of the size of the domain. We perform the experiment on the second group of synthetic datasets with the following settings: K, M and ϵ are set to their default values, n is fixed at 10^5, and the domain size d is varied. The results are presented in Fig. 15, showing that the four protocols exhibit only a slight change in MSE as d increases. Theoretically, when the domain size increases, more distinct items are introduced, resulting in a greater probability of multiple items being hashed to the same counter. The collision probability increases, consequently leading to a larger estimation error. However, the impact of domain size variation on the performance of the protocols can be relatively small, considering the sparse nature of the items held by each user and the limited changes in the distribution of the number of items per user. This confirms the effectiveness of the sketching technique as a domain reduction and encoding method for large-domain data collection.

Moreover, in Fig. 15(b), we also evaluate the impact of varying d for solutions that directly sample, i.e., PM and HM. We can see that PrivSketch+ always has an advantage for mean estimation. Similar to frequency estimation, a change in the domain size d does not affect the accuracy of the estimates of the sketch-based solutions, PrivSketch+ and PM-PCMS-Mean. Yet, for non-sketch-based solutions, i.e. PM and HM, an increase in d results in an increase of MSE, which translates to worse utility. This is a direct consequence of sampling: each user samples one item from the domain of size d, thus, when n is fixed and d increases, the amount of information collected about each item in the domain reduces. Thus, the variance becomes larger and the estimation less inaccurate.

C. Evaluation of running time

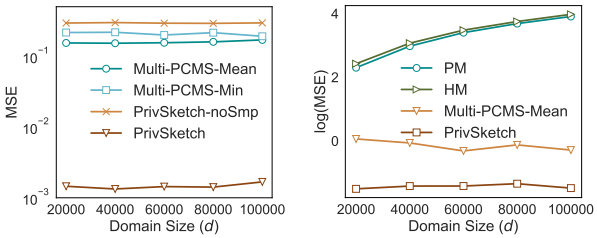
Compared with the traditional protocols, PrivSketch employs the ordering matrix, which introduces additional computations. However, Fig. 16 shows PrivSketch still maintains a user-side running time of less than $0.01s$, despite performing the calculation of the ordering matrix. Experimental results also show that PrivSketch has a significantly faster execution time compared to PS-OLH and SVIM, approximately 100 times faster. However, PrivSketch is slower than Multi-PCMS-Mean, which achieves high speed at the cost of larger MSEs (in Fig. 8). PS-OLH, SVIM and PrivSketch are worse than the other sketch-based solutions in terms of running time. Because these three solutions need to restore the estimated items to the original domain for each user on the collector side, resulting in a complexity of $\mathcal{O}(nd)$. Yet, as shown in Fig. 8, the MSEs of sketch-based solutions are worse, resulting in inaccurate estimations. The long running time of PS-OLH, SVIM, and PrivSketch is the sacrificed time of reducing domain cardinality to gain high utility. Compared with PS-OLH and SVIM, which perform best under a small privacy budget except for PrivSketch, the running time of PrivSketch is about 100 times faster. This is because each user employs local hash functions in PS-OLH, resulting in n times hash function calculations on the collector side. In PrivSketch, each user shares the same hash functions, which are only calculated once. We omit experimental results for PS-OLH and SVIM over AOL, because they need more than 10 days to compute, making them cumbersome to use in practice.

The running time of PrivSketch+ is shown in Table III. The client time is about 0.01s. The server time is similar to PrivSketch and is related to the data set size n and domain size d. Even though PrivSketch+ is slower than PM and HM, we note that (similar to the PS-OLH and SVIM solutions for frequency estimation) PrivSketch+ trades running time



(a) vary M (K=4) (b) vary K (M=32)

Fig. 14. MSE of frequency estimation with different K, M.



(a) frequency estimation (b) mean estimation

Fig. 15. MSE with different d

(a) Frequency estimation (b) Frequent item estimation

Fig. 16. Comparison of running times.

TABLE III
RUNNING TIME FOR MEAN ESTIMATION

Dataset	Solution	Client Time [s]	Server Time [s]
Synthetic	PM	0.00009	0.23
	HM	0.00008	0.20
	PM-PCMS-Mean	0.00048	2.14
	PrivSketch+	0.00218	509.13
AOL	PM	0.01902	6.25
	HM	0.01045	3.69
	PM-PCMS-Mean	0.00182	28.719
	PrivSketch+	0.01308	52039.91

efficiency for much better accuracy (up to ∼4 orders of magnitude more accurate than PM and HM; cf. Fig. 11).

VIII. Related Work

Lots of studies have explored LDP protocols for different data types (i.e., numerical data, categorial data, key-value data, itemsets, etc.) and different computing tasks (i.e., frequency estimation, mean estimation, heavy hitter discovery, etc.). In this section, we present a review of representative LDP works that are closely related to our work.

Set-valued Data Collection. Addressing the challenge of varying set sizes among users is crucial when performing frequency estimation on set-valued data. Padding and Sampling [15] is a commonly employed technique to unify the length of sets, as seen in approaches like PSFO [28], PrivSet [16]. Although the wheel mechanism proposed by Wang [17] aims to reduce computational overhead, this work does not specifically focus on the challenges in large domains where the efficiency of data structures becomes crucial. Works on LDP under the large domain, such as LDPMiner [15], SVIM [28], TreeList [33], PEM [32], have concentrated on mining the items that appear frequently, i.e, heavy hitters. These methods take advantage of a multi-phase approach to overcome the challenges posed by the vast domain size. Initially, a portion of the privacy budget is allocated to discover frequent candidates, i.e., potential heavy-hitters. Subsequently, the remaining privacy budget is utilized to refine the estimation process on the identified candidates, achieving accurate estimation. However, data streams typically arrive in a continuous manner, making it impractical to perform an initial phase to discover frequent candidates before estimation.

Frequency estimation with Hash-Encoding Technique. To support large-scale private analytics, hash-based data representations (such as the bloom filters and sketches) are commonly used. Previous works [34], [35] employ sketches to store the heavy hitters in the centralized DP setting. Similarly, Count-Min Sketch is employed in [36] to compute the population distribution of different regions. Without a trusted collector, some studies represent data streams using local sketches, and utilize privacy-enhancement techniques, such as homomorphic encryption and multi-party computation protocols, to aggregate local sketches [37]–[39].

Under LDP settings, RAPPOR [11] adopts Bloom filters as a means to encode data and reduce the data domain. However, Bloom filters in RAPPOR necessitate expensive computations for accurate estimation, such as LASSO regression. Similarly, local hash functions in OLH [29] are employed for compressing data, but involve expensive hash operations. Count-Mean Sketch [10] was proposed as a simpler

estimation solution for estimating the popularity of emojis in iOS. Subsequent works [40], [41] have enhanced its utility by transmitting multiple sketches per user, albeit at the cost of additional communication overhead. When discovering the heavy hitters, Hadamard transform is introduced in Count-Median Sketch [33]. In [26], a comprehensive analysis and comparison of sketch-based LDP protocols are conducted. These protocols utilize sketching algorithms directly for domain reduction during the one-item collection, without considering the extra collision errors of sketches under LDP. In a recent study [42], hash functions were utilized to compute statistics of the k-sparse vector. However, the approach made an assumption regarding the number of items generated by each user.

Mean Estimation for multi-dimensional data. Initially, the Laplace mechanism for mean estimation in centralized DP [19] was extended for LDP [43], where the perturbed data in such solutions are unbounded, and the variance increases with the data domain size d. Duchi et al. [23] propose to bound the perturbed data within a limited range, so as to estimate the mean for multi-dimensional data. For $d-$dimensional input, the solution utilizes the random response mechanism to perturb data in each dimension separately, and outputs a $d-$dimensional perturbation vector. Harmony [24] simplified this mechanism by randomly sampling one dimension from a $d-$dimensional vector for perturbation. It maintains the same error bound as Duchi et al.'s solution, but needs less communication. Subsequently, Wang et al. [14] proposed the PM mechanism to reduce the variance of the Harmony approach. However, PM is designed for collections where the size of data items produced by each user is merely *one*. Wang et al. also proposed two improved mechanisms: HM that combines the advantages of Duchi et al.'s solution, and a variant for multi-dimensional data processing. Even though these solutions simply use sampling to collect multi-dimensional data, the utility is not high for very large domains. A recent study proposed the Square Wave (SW) mechanism [44] that improves PM by combining it with expectation maximization; though, this mechanism aims at estimating the data distribution, not the mean, which is the problem we address in our work.

Variants of LDP. To enhance the utility of LDP, numerous research efforts have been dedicated to optimizing the variants of LDP. One approach involves incorporating additional trust mechanisms into LDP. For example, some studies [45], [46] propose the shuffling mechanism to further randomize the reports submitted by users. And some works [47] combines the strengths of centralized DP and local DP. Another approach (i.e., [48], [49]) relaxes the privacy constraint imposed by LDP by introducing an additional parameter, such as a distance metric between inputs inspired by the geo-indistinguishability [50]. Additionally, other approaches, such as personalized privacy demand [51], [52], and distinct sensitivity of the input data [53], [54]. However, these works do not utilize the background information to enhance utility as we do in this paper.

IX. Conclusions

This paper studies the frequency and mean estimation problems under local differential privacy. We propose PSF, a privacy-preserving streaming data collection framework based on sketches. This framework adopts a decode-first procedure, and introduces the ordering matrix, which does not expose the original value of any counter in the sketch. For frequency and mean estimation, we propose two protocols, PrivSketch and PrivSketch+, based on PSF. We experimentally verify the effectiveness of our two proposed protocols: they outperform existing LDP protocols by 1-4 orders of magnitude and execute up to $\sim$100x faster.

In future work, we plan to extend our protocol to address more statistical problems in data streams. Furthermore, we plan to explore privacy-preserving methods similar to PrivSketch that make use of LDP combined with background knowledge in more complex scenarios.

References

[1] Y. Li, X. Lee, B. Peng, T. Palpanas, and J. Xue, "Privsketch: A private sketch-based frequency estimation protocol for data streams," in *Proceedings of 34th International Conference on Database and Expert Systems Applications*, 2023, pp. 147–163.

[2] J. Wang, H. Han *et al.*, "Multiple strategies differential privacy on sparse tensor factorization for network traffic analysis in 5G," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, 2021.

[3] S. Xiong, A. D. Sarwate, and N. B. Mandayam, "Network traffic shaping for enhancing privacy in iot systems," *IEEE/ACM Transactions on Networking*, 2022.

[4] Q. Huang and P. P. Lee, "LD-sketch: a distributed sketching design for accurate and scalable anomaly detection in network data streams," in *INFOCOM*, 2014, pp. 1420–1428.

[5] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, 2013.

[6] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[7] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, "Exposure: a passive dns analysis service to detect and report malicious domains," *TISSEC*, vol. 16, no. 4, pp. 1–28, 2014.

[8] E. Commission, "General data protection regulation," 2018. [Online]. Available: https://gdpr-info.eu/

[9] "Personal information protection law of the people's republic of china," 2021. [Online]. Available: http://en.npc.gov.cn.cdurl.cn/2021-12/29/c_694559.htm

[10] D. P. Team, "Learning with privacy at scale," *Apple Machine Learning Journal*, vol. 1, no. 8, 2017.

[11] Ú. Erlingsson, V. Pihur, and A. Korolova, "RAPPOR: randomized aggregatable privacy-preserving ordinal response," in *CCS*, 2014.

[12] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," *NIPS*, vol. 30, 2017.

[13] T. Wang, J. Q. Chen *et al.*, "Continuous release of data streams under both centralized and local differential privacy," in *CCS*, 2021.

[14] N. Wang, X. Xiao *et al.*, "Collecting and analyzing multidimensional data with local differential privacy," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 638–649.

[15] Z. Qin, Y. Yang *et al.*, "Heavy hitter estimation over set-valued data with local differential privacy," in *CCS*, 2016, pp. 192–203.

[16] S. Wang, L. Huang *et al.*, "Privset: set-valued data analyses with locale differential privacy," in *INFOCOM*, 2018, pp. 1088–1096.

[17] S. Wang, Y. Qian *et al.*, "Set-valued data publication with local privacy: Tight error bounds and efficient mechanisms," *PVLDB*, 2020.

[18] G. Cormode and S. Muthukrishnan, "An improved data stream summary: the count-min sketch and its applications," *Journal of Algorithms*, vol. 55, no. 1, pp. 58–75, 2005.

[19] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, 2006, pp. 265–284.

[20] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *FOCS*, 2013, pp. 429–438.

[21] F. D. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *SIGMOD*, 2009, pp. 19–30.

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

[23] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Minimax optimal procedures for locally private estimation," *Journal of the American Statistical Association*, vol. 113, no. 521, pp. 182–201, 2018.

[24] T. T. Nguyên, X. Xiao *et al.*, "Collecting and analyzing data from smart device users with local differential privacy," *arXiv preprint arXiv:1606.05053*, 2016.

[25] G. Cormode and K. Yi, *Small summaries for big data*. Cambridge University Press, 2020.

[26] G. Cormode, S. Maddock, and C. Maple, "Frequency estimation under local differential privacy," *PVLDB*, vol. 14, no. 11, pp. 2046–2058, 2021.

[27] H. H. Arcolezi, J.-F. Couchot, B. Al Bouna, and X. Xiao, "Improving the utility of locally differentially private protocols for longitudinal and multidimensional frequency estimates," *Digital Communications and Networks*, 2022.

[28] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *S&P*, 2018, pp. 127–143.

[29] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *USENIX Security Symposium*, 2017, pp. 729–745.

[30] F. Bodon, "A fast apriori implementation." in *FIMI*, vol. 3, 2003, p. 63.

[31] G. Dudek, "Aol search log," 2007. [Online]. Available: http://www.cim.mcgill.ca/\~{}dudek/206/Logs/AOL/

[32] T. Wang, N. Li, and S. Jha, "Locally differentially private heavy hitter identification," *TDSC*, vol. 18, no. 2, pp. 982–993, 2019.

[33] R. Bassily, K. Nissim, U. Stemmer, and A. Guha Thakurta, "Practical locally private heavy hitters," *NIPS*, vol. 30, 2017.

[34] T.-H. H. Chan, M. Li, E. Shi, and W. Xu, "Differentially private continual monitoring of heavy hitters from distributed streams," in *PoPETs*, 2012, pp. 140–159.

[35] D. Mir, S. Muthukrishnan, A. Nikolov, and R. N. Wright, "Pan-private algorithms via statistics on sketches," in *PODS*, 2011, pp. 37–48.

[36] J. Zhang, Q. Yang *et al.*, "A differential privacy based probabilistic mechanism for mobility datasets releasing," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 201–212, 2021.

[37] L. Melis, G. Danezis, and E. De Cristofaro, "Efficient private statistics with succinct sketches," *arXiv preprint*, 2015.

[38] S. G. Choi, D. Dachman-soled, M. Kulkarni, and A. Yerukhimovich, "Differentially-private multi-party sketching for large-scale statistics," *PoPETs*, vol. 2020, no. 3, pp. 153–174, 2020.

[39] J. Böhler and F. Kerschbaum, "Secure multi-party computation of differentially private heavy hitters," in *CCS*, 2021, pp. 2361–2377.

[40] P. Vepakomma, S. N. Pushpita, and R. Raskar, "DAMS: meta-estimation of private sketch data structures for differentially private COVID-19 contact tracing," Tech. Rep., Tech. Rep., 2021.

[41] C. Piao, Y. Hao, J. Yan, and X. Jiang, "Privacy protection in government data sharing: an improved ldp-based approach," *SOCA*, 2021.

[42] M. Zhou, T. Wang, T. H. Chan, G. Fanti, and E. Shi, "Locally differentially private sparse vector aggregation," in *S&P*, 2022.

[43] T. Wang, X. Zhang, J. Feng, and X. Yang, "A comprehensive survey on local differential privacy toward data statistics and analysis," *Sensors*, vol. 20, no. 24, p. 7030, 2020.

[44] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škoric, "Estimating numerical distributions under local differential privacy," in *SIGMOD*, 2020, pp. 621–635.

[45] A. Cheu, A. Smith, J. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," in *EUROCRYPT*, 2019, pp. 375–403.

[46] Ú. Erlingsson, V. Feldman *et al.*, "Amplification by shuffling: from local to central differential privacy via anonymity," in *SODA*, 2019.

[47] B. Avent, A. Korolova, D. Zeber, T. Hovden, and B. Livshits, "BLENDER: enabling local search with a hybrid differential privacy model," in *USENIX Security*, 2017.

[48] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and A. Pazii, "Metric-based local differential privacy for statistical applications," *arXiv preprint*, 2018.

[49] M. E. Gursoy, A. Tamersoy, S. Truex, W. Wei, and L. Liu, "Secure and utility-aware data collection with condensed local differential privacy," *TDSC*, 2019.

[50] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi, "Broadening the scope of differential privacy using metrics," in *PoPETs*, 2013, pp. 82–102.

[51] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin, "Private spatial data aggregation in the local setting," in *ICDE*, 2016.

[52] N. Yiwen, W. Yang *et al.*, "A utility-optimized framework for personalized private histogram estimation," *IEEE TKDE*, 2018.

[53] T. Murakami and Y. Kawamoto, "Utility-optimized local differential privacy mechanisms for distribution estimation," in *USENIX Security Symposium*, 2019.

[54] X. Gu, M. Li, L. Xiong, and Y. Cao, "Providing input-discriminative protection for local differential privacy," in *ICDE*, 2020, pp. 505–516.