mathematics

MDPI

Article

A Differential Privacy Framework with Adjustable Efficiency–Utility Trade-Offs for Data Collection

Jongwook Kim [1] and Sae-Hong Cho [2,*]

[1] Department of Computer Science, Sangmyung University, Seoul 03016, Republic of Korea; jkim@smu.ac.kr
[2] School of Computer Engineering, Hansung University, Seoul 02876, Republic of Korea
* Correspondence: chosh@hansung.ac.kr

Abstract: The widespread use of mobile devices has led to the continuous collection of vast amounts of user-generated data, supporting data-driven decisions across a variety of fields. However, the growing volume of these data raises significant privacy concerns, especially when they include personal information vulnerable to misuse. Differential privacy (DP) has emerged as a prominent solution to these concerns, enabling the collection of user-generated data for data-driven decision-making while protecting user privacy. Despite their strengths, existing DP-based data collection frameworks are often faced with a trade-off between the utility of the data and the computational overhead. To address these challenges, we propose the differentially private fractional coverage model (DPFCM), a DP-based framework that adaptively balances data utility and computational overhead according to the requirements of data-driven decisions. DPFCM introduces two parameters, α and β, which control the fractions of collected data elements and user data, respectively, to ensure both data diversity and representative user coverage. In addition, we propose two probability-based methods for effectively determining the minimum data each user should provide to satisfy the DPFCM requirements. Experimental results on real-world datasets validate the effectiveness of DPFCM, demonstrating its high data utility and computational efficiency, especially for applications requiring real-time decision-making.

Keywords: differential privacy; data utility; computation overhead; data-driven decision

MSC: 68P27

Received: 29 January 2025
Revised: 21 February 2025
Accepted: 27 February 2025
Published: 28 February 2025

Citation: Kim, J.; Cho, S.-H. A Differential Privacy Framework with Adjustable Efficiency–Utility Trade-Offs for Data Collection. *Mathematics* **2025**, *13*, 812. https://doi.org/10.3390/math13050812

1. Introduction

The widespread use of mobile devices has led to the continuous generation and collection of a vast amount of diverse data. When such user-generated data are collected, they can be used to make data-driven decisions across a wide range of applications. These data enable improved decision-making processes, allowing systems to respond to real-time information and improve services in a variety of areas, including transportation, healthcare, retail, and urban planning [1–4]. For example, in transportation, real-time data from users allow digital map applications to monitor traffic patterns and suggest alternative routes [5]. In healthcare, wearable devices collect data on users' physical activity, heart rate, and other health metrics [6]. Healthcare providers then use this information to remotely monitor patients and make personalized recommendations. This approach promotes preventive care, ultimately leading to better patient outcomes and a more responsive healthcare system.

As the collection of data from users has become more prevalent, concerns about data privacy have increased. Including sensitive information in these datasets increases the risk of privacy breaches, as unauthorized access or misuse can have serious consequences for

individuals [7,8]. In response to these challenges, significant efforts have been made to protect user privacy through various techniques. Among these, differential privacy (DP) [9] has emerged as a leading standard. DP adds controlled noise to real datasets, allowing analysts to derive insights without exposing individual data points, thereby protecting personal information.

There are two primary frameworks for collecting sensitive data from users using DP: local differential privacy (LDP) [10,11] and distributed differential privacy (DDP) [12,13]. Both frameworks are designed to operate in environments with untrusted servers, adding noise to the data locally, before they are transmitted to a central server. This approach protects individual privacy by ensuring that the raw data are never directly exposed. LDP has the advantage of relatively low computational overhead; however, the added noise can result in significant data distortion, which can reduce the utility of the collected data. In contrast, DDP provides greater data utility but requires significantly more computational resources, largely due to the use of complex cryptographic techniques.

The utility requirements of collected data vary significantly based on the specific objectives and operational needs of each application. Some applications require highly accurate, comprehensive datasets for accurate analysis and reliable results. For these applications, data completeness is critical to delivering results that satisfy strict performance criteria. In contrast, other applications are more flexible in their data needs, capable of extracting meaningful insights even from partial or incomplete datasets. For these applications, identifying general patterns or trends in real-time is often more important than perfect accuracy, allowing them to perform effectively even with incomplete data.

This variability in utility requirements plays a crucial role in the design of data collection and privacy mechanisms. Applications that do not require exhaustive datasets can benefit from approaches that emphasize efficiency over complete data collection. Motivated by this need, we propose a novel DP-based data collection framework that exploits the adequacy of partial data collection for certain applications, particularly those that require efficiency for real-time decision-making. In particular, the contribution of this paper can be summarized as follows:

- We introduce the differentially private fractional coverage model (DPFCM), which is designed to meet the needs of applications that can operate effectively with partial data collection. DPFCM specifies two parameters, α and β, which are determined by the specific purpose of the application. (α, β)-DPFCM aims to collect at least a fraction α of the total data elements, with the requirement that for each of these collected elements, data are also collected from at least a fraction β of the users. This method guarantees that both the breadth of data elements and the depth of user representation are maintained, supporting robust data utility even when only partial data are collected.
- We propose two different probability-based approaches that effectively determine the minimum number of data elements the server should collect from each user to satisfy the requirements of an (α, β)-DPFCM. These approaches establish precise lower bounds on data collection, ensuring that the utility requirements of the model are satisfied while optimizing for efficiency.
- Finally, we validate the effectiveness of our proposed framework through experiments on real-world datasets, demonstrating that DPFCM achieves high data utility with reduced data collection requirements. Our results show that DPFCM maintains high data utility and computational efficiency, confirming its practical value in real-world applications.

The rest of this paper is structured as follows: Section 2 reviews the related work, and Section 3 presents the necessary background information. Section 4 formally defines

the problem addressed in this paper and discusses existing approaches. Section 5 details the proposed (α, β)-DPFCM framework. Section 6 evaluates the proposed approach through experiments conducted on real-world datasets and Section 7 presents the conclusions of the paper.

2. Related Work

DP has been widely used to protect sensitive data in various data collection scenarios. One of the most representative models is LDP, in which each user independently perturbs his or her data prior to transmission. Depending on the type of sensitive data being collected, existing LDP mechanisms can be broadly divided into two groups: methods designed for categorical data and those tailored for numerical data. For categorical data, the randomized response technique is often used to ensure privacy while allowing accurate frequency estimation. RAPPOR [10] was developed to collect user data, such as the default home page of their browser, in Google Chrome while maintaining privacy. The frequency estimation scheme proposed by Bassily and Smith [14] builds on randomized response techniques, optimizing communication efficiency by encoding user responses into a compact bit representation before transmission. Optimal local hashing [11] further improves frequency estimation under LDP by using a hash-based encoding scheme. For numerical data, various perturbation mechanisms, including Laplace, Gaussian, and staircase noise, are commonly used. Among them, the staircase mechanism [15] is recognized as an optimal noise-adding method, achieving lower expected error than the Laplace mechanism in specific scenarios. Although LDP ensures privacy by adding noise at the user level, it often results in significant utility loss due to the high noise required to meet privacy guarantees.

DDP-based data collection is primarily categorized into two main approaches. The first category integrates DP with secure aggregation to enable privacy-preserving data collection in distributed environments [12,13]. Early works in this category focused on secure data collection in distributed systems. For instance, Lyu et al. [16] applied DP with secure aggregation to collect smart grid data in a fog computing architecture. More recently, this approach has received considerable attention in federated learning, where DP is used to protect local model updates prior to aggregation [17]. Truex et al. [18] leveraged homomorphic encryption alongside DP to securely collect local learning parameters from participating users in federated learning. Much of the research in this area has been dedicated to improving scalability and computational efficiency for large-scale learning [19,20]. Bell et al. [20] introduced an efficient secure aggregation method for federated learning, where the overhead scales logarithmically with the number of participating users. We note that the proposed framework in this paper is general enough to incorporate such advanced secure aggregation techniques.

The second main approach to DDP-based data collection utilizes the shuffle model [13,21], where an additional untrusted server, known as the shuffler, is introduced between users and the data collector. Users independently randomize their data before sending it to the shuffler, which then aggregates and randomly reorders the data before forwarding it to the data collector. The shuffle model has been widely explored, leading to various enhancements in aggregation and privacy amplification techniques [22–24]. However, a major limitation is its reliance on an untrusted shuffler, which introduces deployment challenges and potential vulnerabilities. In many real-world applications, the availability and trustworthiness of a shuffler cannot be guaranteed, making the model impractical for certain data collection scenarios.

Geo-indistinguishability (Geo-Ind) extends DP with a distance-based privacy metric [25–28]. The most commonly used mechanisms in Geo-Ind for data collection are the planar Laplace mechanism and the perturbation function-based mechanism. The Laplace

mechanism perturbs a user's true location by adding noise from a 2D Laplace distribution before sending it to the server [25]. In contrast, the perturbation function-based mechanism pre-calculates an obfuscation function on the server and distributes it to users, who then apply the received function to perturb their data before uploading it to the server [29–32]. Unlike DP, which enforces a consistent privacy guarantee regardless of data values, Geo-Ind allows privacy loss to increase with distance, making it vulnerable to attacks that exploit spatial correlations. Consequently, this approach is unsuitable for direct application to our problem, where strong and consistent privacy protection is required over all data points.

3. Preliminary

DP is a mathematical framework that provides probabilistic privacy protection even against attackers with arbitrary background knowledge [9]. It guarantees that an attacker cannot confidently identify an individual's inclusion in a dataset. Formally, an algorithm $\mathcal{A}$ satisfies (ϵ, δ)-DP if, for any two neighboring datasets D_1 and D_2 differing by one record, and for any output O of $\mathcal{A}$, the following probability condition holds:

$$Pr[\mathcal{A}(D_1) = O] \leq e^{\epsilon} \times Pr[\mathcal{A}(D_2) = O] + \delta. \tag{1}$$

Here, the privacy budget, ϵ, controls the privacy strength.

The Gaussian mechanism is commonly used to achieve (ϵ, δ)-DP. For a given function f, the Gaussian mechanism $\mathcal{A}$ produces a differentially private output by adding random noise drawn from a Gaussian distribution with mean 0 and variance σ^2 [9].

$$\mathcal{A}(D) = f(D) + \mathcal{N}(0, \sigma^2) \tag{2}$$

The standard deviation σ is calculated as $\sigma = \frac{\Delta f \cdot \sqrt{2\ln(1.25/\delta)}}{\epsilon}$, where Δf (global sensitivity) is the maximum change in the output of the function f when a single record in the dataset is modified.

Local Differential Privacy (LDP) In traditional DP settings, a trusted server collects original data from individuals, applies noise to the aggregated data, and shares the data in a privacy-preserving manner. However, real-world scenarios do not always allow for such a trusted server. LDP addresses this limitation by allowing each data owner to independently apply noise to their data before sharing it with an untrusted server [10,11]. Formally, a randomized algorithm $\mathcal{A}$ satisfies (ϵ, δ)-LDP if, for any two data values v_a and v_b, and any output O of $\mathcal{A}$, the following condition holds:

$$Pr[\mathcal{A}(v_a) = O] \leq e^{\epsilon} \times Pr[\mathcal{A}(v_b) = O] + \delta. \tag{3}$$

The above equation ensures that the aggregator cannot confidently determine whether the output of $\mathcal{A}$ was generated from v_a or v_b, and thus guarantees the privacy of the individual's input. This property prevents the inference of sensitive information by ensuring that any two inputs, v_a and v_b, produce outputs that are statistically indistinguishable within the bounds defined by the privacy parameters.

Distribute Differential Privacy (DDP) Like LDP, DDP enables privacy in data collection without relying on a trusted central server. In DDP, each user i adds independent noise to their data x_i, resulting in locally randomized data $x_i + \mathcal{N}(0, \frac{\sigma^2}{n})$. Here, $\mathcal{N}(0, \frac{\sigma^2}{n})$ represents noise drawn from a Gaussian distribution with mean 0 and variance $\frac{\sigma^2}{n}$, where n is the total number of users. When all n users' contributions are aggregated by the untrusted server, the result is the following:

$$\sum_{i=1}^{n}\left(x_i + \mathcal{N}\left(0, \frac{\sigma^2}{n}\right)\right) = \sum_{i=1}^{n} x_i + \mathcal{N}(0, \sigma^2), \tag{4}$$

where $\mathcal{N}(0, \sigma^2)$ represents the combined noise. Thus, DP is satisfied for the aggregated result, $\sum_{i=1}^{n} x_i$, rather than for each individual user's data. By adding noise collaboratively across all users, the model ensures that the aggregated result satisfies (ϵ, δ)-DP for the entire dataset while minimizing the noise that each individual must add. However, since individual data do not satisfy DP, computationally intensive cryptographic techniques are necessary to ensure that individual contributions remain protected during the aggregation process.

4. Problem Definition and Baseline Approaches

4.1. Problem Definition

In this section, we formally define the problem addressed in this paper. Let us assume that a set of users is defined as $U = \{u_1, u_2, \cdots, u_n\}$. Let $E = \{e_1, e_2, \cdots, e_m\}$ be the set of all possible data elements. Each user $u_t \in U$ has a set of data elements, represented by $D_t = \{(e_k, f_{t,k}) \mid k = 1, 2, \ldots, m\}$, where $f_{t,k}$ denotes the value associated with data element e_k for user u_t. In many real-world applications, the number of data elements m is large, and thus $f_{t,k}$ is often zero, resulting in a sparse representation of D_t. For example, in location-based services, each e_k might represent a unique point of interest (PoI), and $f_{t,k}$ might indicate the frequency of visits by user u_t to that PoI. In this case, since there are many PoIs and most users visit only a small number of them, the data are very sparse.

In this paper, we assume that for each $u_t \in U$, the value $f_{t,k}$ corresponds to sensitive data that could reveal user preferences. Therefore, it is necessary to protect this information when sharing it with external parties. The goal of the service provider (i.e., the central server) is to collect the values $(e_k, f_{t,k})$ from all users in U. However, since $f_{t,k}$ represents sensitive information, it is necessary to apply DP to protect the sensitive information of individual users. Specifically, for each data element $e_k \in E$, the service provider aims to compute the aggregate sum $\sum_{t=1}^{n} f_{t,k}$ across all users, while ensuring that each user's privacy is preserved.

4.2. Baseline Approaches

Algorithm 1 represents the baseline approach using LDP to the problem addressed in this paper. This pseudocode corresponds to the procedure for the t-th participating user, which is applied identically to all other users. Algorithm 1 ensures privacy by perturbing each user's non-zero data values before reporting them to the central server. We assume that reporting only non-zero data does not significantly reveal sensitive information, as the sparsity of real-world datasets often limits the risk of identification from missing values alone. Each user initializes an empty list, R_t, and iterates through all data elements, adding Gaussian noise, $\mathcal{N}(0, \sigma^2)$, to non-zero values $f_{t,k}$ (line 2–6). The noisy values are stored in R_t, which is then sent to the server.

Algorithm 1 Baseline Approach Based on LDP (Each Participating User Processing)

1: Initialize R_t as an empty list
2: **for** each $k \in \{1, 2, \ldots, m\}$ **do**
3: **if** $f_{t,k} > 0$ **then**
4: Append $(e_k, f_{t,k} + \mathcal{N}(0, \sigma^2))$ to R_t
5: **end if**
6: **end for**
7: Report R_t to the central server

In Algorithm 1, each user independently perturbs their data to satisfy (ϵ,δ)-DP locally. However, since the goal of this work is to compute the sum of all users' data for each element, adding noise locally at the user level leads to excessive noise in the aggregated result. For example, consider a data element e_l where h users have non-zero values. The aggregated result at the server would contain noise $\mathcal{N}(0, h\sigma^2)$, which increases with the number of contributing users h. Instead, adding the noise directly to the aggregated sum would only require $\mathcal{N}(0, \sigma^2)$, significantly improving the utility of the result while still satisfying (ϵ,δ)-DP.

Algorithm 2 presents a DDP-based baseline approach to the problem addressed in this paper. In this approach, each participating user iterates through all data elements, adding noise from a Gaussian distribution, $\mathcal{N}(0, \frac{\sigma^2}{n})$, where n is the total number of participating users (line 3). The noisy values are then encrypted using a threshold-based homomorphic encryption scheme (line 4). These encrypted noisy values are appended to R_t, which is then sent to the server for decryption and aggregation.

Algorithm 2 Baseline Approach Based on DDP (Each Participating User Processing)

1: Initialize R_t as an empty list
2: **for** each $k \in \{1, 2, \ldots, m\}$ **do**
3: $\quad noise\_f_{t,k} \leftarrow f_{t,k} + \mathcal{N}(0, \frac{\sigma^2}{n})$
4: $\quad enc\_noise\_f_{t,k} = Enc_{pk}(noise\_f_{t,k})$
5: $\quad$ Append $(e_k, enc\_noise\_f_{t,k})$ to R_t
6: **end for**
7: Upload R_t to the central server

Although in Algorithm 2, the noise added by each user does not satisfy ϵ-DP for individual local data, the aggregated noise across all users ensures that the final aggregated result satisfies (ϵ,δ)-DP [12,13]. For instance, for the k-th data element, the aggregated value across all users is computed as $\sum_{t=1}^{n}(f_{t,k} + \mathcal{N}(0, \frac{\sigma^2}{n})) = \sum_{t=1}^{n} f_{t,k} + n \cdot \mathcal{N}(0, \frac{\sigma^2}{n}) = \sum_{t=1}^{n} f_{t,k} + \mathcal{N}(0, \sigma^2)$, which satisfies the ($\epsilon$,$\delta$)-DP for the aggregated data. As a result, compared to the LDP-based solution, the DDP-based approach achieves better data utility in the aggregated results. However, in Algorithm 2, since ϵ-DP is not satisfied for each user's individual local data, it is essential to ensure that the server can only access the aggregated results and cannot access individual user data. This requires cryptographic techniques such as homomorphic encryption [17,18] and multi-party aggregation [19,20], which allow the server to process only the aggregated results while preventing access to individual user contributions. However, these techniques are widely recognized for their computational overheads, which can significantly increase overall system complexity.

5. Proposed Approach

Although the DDP-based approach presents a promising solution by reducing noise relative to the LDP-based approach, it is not directly applicable to the problem addressed in this paper. In particular, two key challenges prevent the straightforward application of DDP to our problem:

- Impracticality of Applying DDP to All Data. The DDP-based baseline approach in Algorithm 2 is highly inefficient because it requires cryptographic techniques to be applied to all m data elements across n users. In real-world scenarios, where m (the number of data elements) is extremely large, the computational overhead of cryptographic techniques becomes prohibitive, especially when n is also large. The complexity of these techniques scales with both the number of users and the size of the dataset [17,20]. As a result, applying DDP to all data elements in scenarios with a large number of users is highly inefficient due to the significant computational cost.

- Non-Zero Data Variability. An alternative solution is to apply DDP only to data elements with non-zero values, similar to the LDP-based approach in Algorithm 1. In this approach, each user i perturbs its data by adding independent noise $\mathcal{N}(0, \frac{\sigma^2}{h})$, where h represents the total number of users contributing to the aggregation for a specific data element. However, this solution is not feasible for sparse datasets, as each user typically has a different set of non-zero data elements. Since h depends on the number of users contributing non-zero values for a given data element $e_k \in E$, the server cannot determine h for each element without knowing the users' individual non-zero data. Consequently, the noise variance $\mathcal{N}(0, \frac{\sigma^2}{h})$ cannot be accurately calibrated, resulting in the failure to satisfy (ϵ,δ)-DP globally.

These challenges highlight the limitations of directly applying DDP to our problem, and the need for an alternative approach. In this section, we propose the (α, β)-DPFCM framework, which builds on DDP to balance data utility and efficiency while adapting to the specific requirements of the application.

5.1. Definition of (α, β)-DPFCM

In this subsection, we formally define the proposed (α, β)-DPFCM.

Definition 1 ((α, β)-DPFCM)**.** *A data collection process satisfies (α, β)-DPFCM if it satisfies the following two conditions for the given set of users $U = \{u_1, u_2, \cdots, u_n\}$ and data elements $E = \{e_1, e_2, \cdots, e_m\}$:*

$$|E_c| \geq \alpha \cdot m, \tag{5}$$

where $E_c \subseteq E$ is the set of collected data elements, and $0 \leq \alpha \leq 1$, and

$$\forall e_k \in E_c, |U_k| \geq \beta \cdot n, \tag{6}$$

where $U_k \subseteq U$ is the set of users contributing to data element e_k, and $0 \leq \beta \leq 1$.

The (α, β)-DPFCM framework introduces two key parameters, α and β, which control the breadth and depth of data collection, respectively. α specifies the minimum fraction of the total data elements that must be collected to ensure sufficient diversity in the dataset. For example, if $\alpha = 0.9$ and the total number of items is $m = 1000$, at least 900 items must be included in the collection process. On the other hand, β determines the minimum fraction of users who must contribute data for each collected element, ensuring reliable user representation. For instance, if $\beta = 0.6$ and the total number of users $n = 500$, at least 300 users must provide data for each selected element.

Figure 1 shows an example of $(0.8, 0.8)$-DPFCM applied to a scenario with five users and 10 data elements. This example satisfies the $(0.8, 0.8)$-DPFCM requirements because the set $E_c = \{e_1, e_2, e_4, e_5, e_6, e_7, e_8, e_{10}\}$ consists of data elements for which at least four users contribute their data to the server, thereby meeting the specified thresholds for both α and β. We note that by adjusting α and β, the framework can balance the trade-off between the diversity of data elements and the robustness of user contributions, supporting efficient and utility-driven data collection in real-world applications.

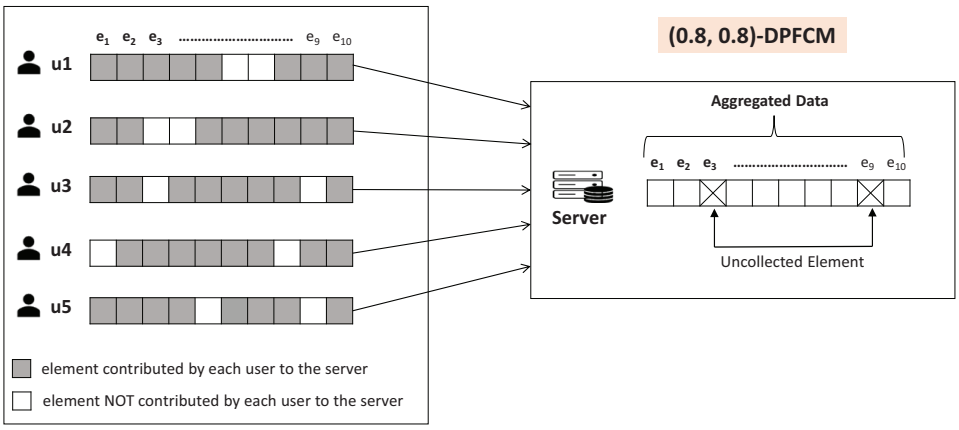

Figure 1. An example of $(0.8, 0.8)$-DPFCM with 5 users and 10 data elements.

5.2. Overview of (α, β)-DPFCM Framework

Figure 2 shows an overview of the proposed (α, β)-DPFCM framework, which consists of three phases.

- Computing minimum user contributions: The data collection server calculates the minimum number of data elements, τ, that each user must contribute to the server to satisfy the requirements of the (α, β)-DPFCM framework, and then distributes it to all users (Section 5.3).
- Contributing data using DDP: Each user employs a DDP-based mechanism to report at least τ data elements to the server (Section 5.4).
- Secure aggregation: The server aggregates encrypted contributions for each data element, verifies that the threshold β is satisfied, and securely decrypts the noisy values for qualifying elements (Section 5.5).

In the next subsections, we will present the detailed descriptions of each step in the proposed (α, β)-DPFCM framework.

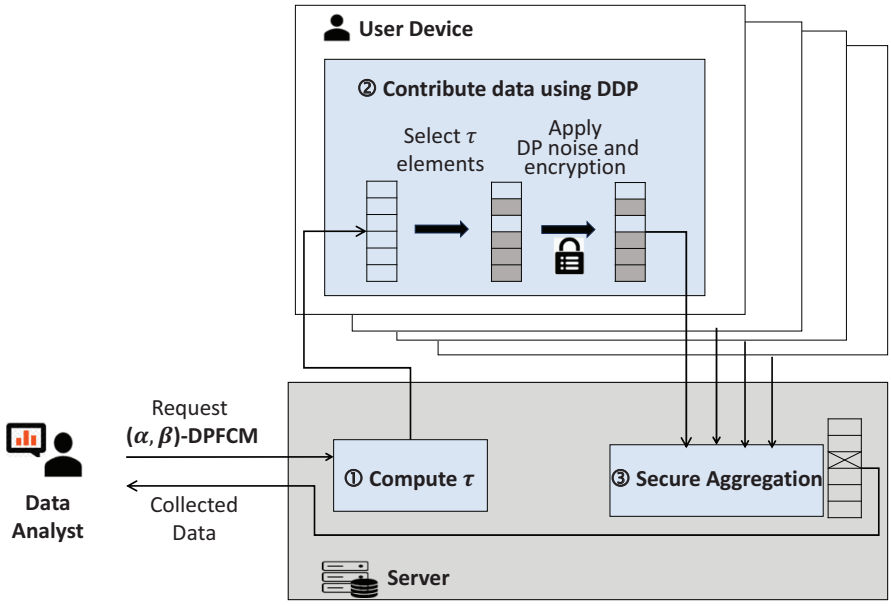

Figure 2. An overview of the proposed (α, β)-DPFCM framework.

5.3. Computing Minimum User Contributions for (α, β)-DPFCM

The requirement of the (α, β)-DPFCM cannot be satisfied if each user sends only its non-zero data elements to the server, as in the LDP-based approach described in

Algorithm 1. To address this, it is necessary to determine the minimum number of data elements that each user must contribute to the central server in order to satisfy the requirement of the (α, β)-DPFCM. In this subsection, we propose two different probability-based approaches.

5.3.1. Binomial Model-Based Approach

To determine the minimum number of data elements each user must send to satisfy (α, β)-DPFCM, we represent the selection process using a binomial distribution. Let us assume that each user randomly selects τ data elements from a total of m available data elements. The probability that any specific data element e_t is selected by a user is given by $p_t = \frac{\tau}{m}$.

We then model the number of users selecting a specific data element e_t as a random variable X. Since each user's selection is independent of others, X follows a binomial distribution,

$$X \sim \text{Binomial}(n, p_t), \tag{7}$$

where n is the total number of users and p_t is the probability of a user selecting e_t. This distribution effectively captures the likelihood of a specific data element being selected by a given number of users.

The probability that the number of users selecting e_t is greater than or equal to the threshold specified in Equation (6) is represented as $\Pr(X \geq \beta \cdot n)$. Moreover, according to the condition in Equation (5), at least a fraction α of the total data elements must be collected. This implies that α can be interpreted as the confidence level required to satisfy $\Pr(X \geq \beta \cdot n)$. This condition can be expressed using the cumulative distribution function (CDF) of the binomial distribution:

$$\begin{aligned} &\Pr(X \geq \beta \cdot n) = 1 - \Pr(X \leq \beta \cdot n - 1) \geq \alpha \\ &\Rightarrow \Pr(X \leq \beta \cdot n - 1) \leq 1 - \alpha \end{aligned} \tag{8}$$

Finally, we compute the cumulative probability $\Pr(X \leq \beta \cdot n - 1)$ using the CDF of the binomial distribution as follows:

$$\Pr(X \leq \beta \cdot n - 1) = \sum_{k=0}^{\beta \cdot n - 1} \binom{n}{k} p_t^k (1 - p_t)^{n-k} = \sum_{k=0}^{\beta \cdot n - 1} \binom{n}{k} (\frac{\tau}{m})^k (1 - \frac{\tau}{m})^{n-k} \leq 1 - \alpha \tag{9}$$

We need to compute the minimum value of τ that satisfies the above equation, which provides a probabilistic guarantee that at least $\alpha \cdot m$ data elements are selected, each with contributions from at least $\beta \cdot n$ users.

5.3.2. Chernoff Bound-Based Approach

The next approach for computing the minimum τ to satisfy (α, β)-DPFCM is to use the Chernoff bound. The Chernoff bound provides an exponential bound on the tail probabilities of a random variable and is particularly useful for bounding the sum of independent random variables. For a random variable X with expectation $\mathbb{E}[X]$, the lower bound of the Chernoff bound is defined as follows:

$$P(X \leq (1 - \gamma)\mathbb{E}[X]) \leq \exp\left(-\frac{\gamma^2}{2}\mathbb{E}[X]\right), \quad \text{for } 0 < \gamma \leq 1. \tag{10}$$

This bound provides an exponential decay rate for the probability that X is less than $(1 - \gamma)\mathbb{E}[X]$. By applying this Chernoff lower bound, we can determine the minimum threshold τ such that the probability of X being greater than or equal to $\beta \cdot n$ is at least α.

As in the case of Section 5.3.1, let us assume that each user randomly selects τ data elements from a total of m available data elements. Thus, the probability that any specific data element e_t is selected is $p_t = \frac{\tau}{m}$. We model the number of users selecting a specific data element e_t as a random variable X. The expected value of X, denoted as $\mathbb{E}[X]$, is computed as follows:

$$\mathbb{E}[X] = n \cdot p_t = n \cdot \frac{\tau}{m} \tag{11}$$

To satisfy the (α, β)-DPFCM requirements, we need to ensure that the probability of X being greater than or equal to $\beta \cdot n$ is at least α. This requirement can be rewritten as follows:

$$\Pr(X \geq \beta \cdot n) \geq \alpha \Rightarrow \Pr(X \leq \beta \cdot n) \leq 1 - \alpha. \tag{12}$$

To apply the Chernoff bound in this context, we define γ such that $\gamma = 1 - \frac{\beta \cdot n}{\mathbb{E}[X]}$. Substituting this into the Chernoff bound in Equation (10), we obtain

$$\Pr(X \leq \beta \cdot n) \leq \exp\left(-\frac{\gamma^2}{2}\mathbb{E}[X]\right). \tag{13}$$

By rearranging the terms, the inequality becomes

$$\exp\left(-\frac{\gamma^2}{2}\mathbb{E}[X]\right) \leq 1 - \alpha. \tag{14}$$

By taking the natural logarithm on both sides, we derive

$$\frac{\gamma^2 \mathbb{E}[X]}{2} \geq -\ln(1 - \alpha). \tag{15}$$

By substituting $\mathbb{E}[X]$ and γ, the inequality becomes

$$\frac{\left(1 - \frac{\beta \cdot m}{\tau}\right)^2 \cdot n \cdot \frac{\tau}{m}}{2} \geq -\ln(1 - \alpha). \tag{16}$$

To further simplify, we multiply both sides of the equation by $\frac{2m}{n}$:

$$\left(1 - \frac{\beta \cdot m}{\tau}\right)^2 \cdot \tau \geq \frac{-2m \ln(1 - \alpha)}{n}. \tag{17}$$

We need to compute the minimum value of τ that satisfies the above inequality, thereby ensuring that the (α, β)-DPFCM requirements are satisfied.

The choice between the binomial model-based approach and the Chernoff bound-based approach determines how the minimum required user contribution τ is calculated. The binomial approach tends to produce a lower τ, optimizing efficiency by reducing computational overhead. However, this efficiency comes with a trade-off. That is, there is a small probability that the (α, β)-DPFCM condition may not be fully satisfied. In contrast, the Chernoff approach slightly overestimates τ, which guarantees that the (α, β)-DPFCM condition is always satisfied, but leads to a higher computational cost due to the increased number of reported data elements. The experimental results in Section 6 confirm this trade-off, demonstrating that while the binomial model-based approach minimizes computational overhead, the Chernoff bound-based approach is preferable when strict satisfaction of (α, β)-DPFCM is the priority.

5.3.3. Algorithm for Computing Minimum User Contribution

Since deriving a closed-form solution for Equations (9) and (17) is not feasible, we use an iterative method to compute the minimum τ efficiently. Algorithm 3 provides the pseudocode for this iterative approach. The algorithm begins by initializing τ to the smallest possible value, $\lceil \beta \cdot m \rceil$. The algorithm iteratively increments τ by one and checks whether the specified condition (Binomial or Chernoff) is satisfied based on the chosen method. The `BinomialCondition` procedure checks whether the condition specified in Equation (9) is satisfied (lines 16–19), while the `ChernoffCondition` procedure checks whether the Chernoff bound-based inequality in Equation (17) holds (lines 20–24). The algorithm continues until a valid τ is found.

Algorithm 3 Pseudocode for Computing Minimum τ

```
1:  procedure ComputeTau(n, m, α, β, method)
2:      τ ← ⌈β · m⌉
3:      while τ ≤ m do
4:          if method = "binomial" then
5:              if BinomialCondition(τ, n, m, α, β) then
6:                  return τ
7:              end if
8:          else if method = "chernoff" then
9:              if ChernoffCondition(τ, n, m, α, β) then
10:                 return τ
11:             end if
12:         end if
13:         τ ← τ + 1
14:     end while
15: end procedure
16: procedure BinomialCondition(τ, n, m, α, β)
17:     P ← BinomialCDF(β · n − 1, n, τ/m)
18:     return P ≤ 1 − α
19: end procedure
20: procedure ChernoffCondition(τ, n, m, α, β)
21:     LHS ← (1 − β·m/τ)^2 · τ
22:     RHS ← −2·m·ln(1−α)/n
23:     return LHS ≥ RHS
24: end procedure
```

By restricting each user to uploading τ data elements, as determined by Algorithm 3, we effectively address the two key challenges previously identified: the impracticality of applying DDP to all data and non-zero data variability. To address the impracticality of applying DDP to all data, the proposed framework allows each user to upload only a subset of data elements, significantly reducing computational overhead. By selecting and reporting only a subset of data elements rather than the entire dataset, the computational overhead is significantly reduced, enabling scalability for large datasets. For non-zero data variability, the proposed framework ensures that each user sends at least τ data elements to the server. The value of τ is determined using either the proposed binomial model-based approach or the Chernoff bounds-based approach. By ensuring that each user contributes τ data elements, the (α, β)-DPFCM condition is satisfied, enabling DDP-based data collection regardless of variations in non-zero data across users.

5.4. Contributing Data Using DDP

After calculating the minimum τ, the server distributes this value to all participating users. Each user is then required to select at least τ of data elements from their available

dataset. These selected data elements are then sent back to the server. Algorithm 4 provides the pseudocode for the user-side processing of the proposed (α, β)-DPFCM algorithm. This algorithm is based on the principles of DDP, but introduces a novel approach to reduce the computational overhead associated with DDP-based schemes. Unlike existing DDP methods, which require users to report all data elements to the server for our problem, the proposed algorithm allows each user to report only a subset of data elements, specified by τ. By limiting the number of reported elements to τ, the algorithm significantly reduces the computational overhead associated with DDP.

Algorithm 4 Pseudocode for the User-Side Processing of (α, β)-DPFCM

1: Initialize R_t as an empty list
2: **for** each $k \in \{1, 2, \ldots, m\}$ **do**
3: **if** $f_{t,k} > 0$ **then**
4: $noise\_f_{t,k} \leftarrow f_{t,k} + \mathcal{N}(0, \frac{\sigma^2}{\beta \cdot n})$
5: Append $(e_k, Enc_{pk}(noise\_f_{t,k}))$ to R_t
6: **end if**
7: **end for**
8: **while** size$(R_t) < \tau$ **do**
9: Randomly select i that is not in R_t
10: $noise\_f_{t,i} \leftarrow f_{t,i} + \mathcal{N}(0, \frac{\sigma^2}{\beta \cdot n})$
11: Append $(e_i, Enc_{pk}(noise\_f_{t,i}))$ to R_t
12: **end while**
13: Upload R_t to the central server

The algorithm begins by initializing an empty list, R_t, and then iterates over all data elements, adding Gaussian noise with variance $\mathcal{N}(0, \frac{\sigma^2}{\beta \cdot n})$ to non-zero values (line 4). The noisy values are encrypted using a homomorphic encryption scheme with a given threshold, such as that given in [33] (line 5). The noisy and encrypted values are appended to R_t. If the size of R_t is less than the required threshold τ, the user randomly selects additional indices whose corresponding data element value is zero, applies the same noise addition and encryption steps, and appends the results to R_t (lines 8–12). Finally, the encrypted list R_t is sent to the server for aggregation.

5.5. Secure Aggregation of User Contributions

After receiving the encrypted data from users, the server aggregates the contributions for each data element, checks whether the number of contributions meets the threshold β, and securely decrypts the noisy aggregated values for elements that qualify. Algorithm 5 provides the pseudocode for the server-side processing of the proposed (α, β)-DPFCM algorithm. Upon receiving $R_1, R_2, \cdots, R_n$ from n users, the server first aggregates the data for each data element (line 1). Let assume that $\{(U_k, \{Enc_{pk}(noise\_f_{t,k})\}_{t \in U_k})\}_{k=1}^{m}$ represents the aggregated results for all data elements, where $U_k \subseteq U$ denotes the set of users who uploaded $(e_k, Enc_{pk}(f_{t,k}))$ to the server.

The server then uses the homomorphic properties of the encryption scheme to securely aggregate these encrypted values. If the number of users who submitted encrypted noisy values for a specific data element k satisfies the condition $|U_k| \geq \beta$, the server computes the encrypted aggregated noisy value $Enc_{pk}(noise\_f_k)$ by multiplying the encrypted values provided by all contributing users in U_k, leveraging the additive homomorphic property of the encryption scheme (line 5). Next, the server initiates the threshold decryption process (line 6–10). It randomly selects a subset of users, $U_{dec} \subseteq U_k$, such that $|U_{dec}| = \beta$, to collaboratively decrypt the aggregated value. The server queries each user in U_{dec} with $Enc_{pk}(noise\_f_k)$. Each user computes a partial decryption of the encrypted value using

their share of the private key and sends the result back to the server. The server then combines the partial decryptions to compute the final aggregated noisy value, $noise\_f_k$. Finally, the server stores the result $(e_k, noise\_f_k)$ in the result set, $ResultSet$, and repeats the process for all data elements that satisfy the condition $|U_k| \geq \beta$.

Algorithm 5 Pseudocode for the Sever-Side Processing of (α, β)-DPFCM

$\{(U_k, \{Enc_{pk}(noise\_f_{t,k})\}_{t \in U_k})\}_{k=1}^{m}$ are the aggregated results
Initialize an empty set $ResultSet \leftarrow \emptyset$
for $k \leftarrow 1$ to m **do**
    if $|U_k| \geq \beta$ **then**
        $Enc_{pk}(noise\_f_k) \leftarrow \prod_{t \in U_k} Enc_{pk}(noise\_f_{t,k})$
        Select $U_{dec} \subseteq U_k$ such that $|U_{dec}| = \beta$
        for $u_j \in U_{dec}$ **do**
            Query u_j with $Enc_{pk}(noise\_f_k)$
            Receive partial decryption of $noise\_f_k$ from u_j
        end for
        Compute $noise\_f_k$ from partial decryptions
    end if
    $ResultSet \leftarrow ResultSet \cup \{(e_k, noise\_f_k)\}$
end for

5.6. Analysis of Effect of α and β on τ

In the (α, β)-DPFCM framework, the parameters α and β play a critical role in determining the minimum number of data elements, τ, that each user must contribute. The value of τ has a direct impact on the overall efficiency of the framework, affecting both user-side processing, as described in Algorithm 4, and server-side processing, as described in Algorithm 5. A higher value of τ increases the utility of the collected dataset. However, this comes at the cost of higher computational requirements. On the other hand, a smaller τ reduces the computational overhead, improving the scalability and efficiency of the system. However, it may compromise the utility of the collected data. Thus, in this subsection, we analyze the effect of α and β on τ based on the previously proposed binomial model-based and Chernoff bound-based approaches.

5.6.1. Effect of α on τ

The parameter α specifies the fraction of data elements that must meet the contribution threshold of at least $\beta \cdot n$ users. A higher α implies stricter requirements as a larger proportion of data elements must satisfy this condition.

- Binomial Model-Based Approach: Using the condition from Equation (9), an increase in α decreases $1 - \alpha$, tightening the inequality. To satisfy this tighter condition, the cumulative probability $\Pr(X \leq \beta \cdot n - 1)$ must decrease. This requires an increase in τ, as increasing τ increases the probability that more users will select each data element, thus shifting the probability mass of the binomial towards higher values of X.
- Chernoff Bound-Based Approach: From Equation (17), an increase in α results in a larger $-\ln(1 - \alpha)$ on the right-hand side. To maintain the inequality, τ must be increased to ensure that the left-hand side remains greater than or equal to the right-hand side. Specifically, a larger τ compensates by increasing both the quadratic term $\left(1 - \frac{\beta \cdot m}{\tau}\right)^2$ and the overall product with τ.

In summary, for both approaches, as α increases, τ must also increase to satisfy the stricter condition that a larger proportion of data elements meet the required contribution threshold.

5.6.2. Effect of β on τ

The parameter β represents the fraction of users, $\beta \cdot n$, that must contribute to each data element for it to qualify. A higher β increases the required number of contributions for each data element.

- Binomial Model-Based Approach: From Equation (9), the inequality involves a summation up to $\beta \cdot n - 1$. Increasing β raises this upper limit, requiring the binomial probability mass to shift toward higher values of X. To achieve this, τ must increase, as a higher τ ensures that more users contribute to each data element, thereby meeting the increased threshold.
- Chernoff Bound-Based Approach: From Equation (17), an increase in β raises the term $\frac{\beta \cdot m}{\tau}$, which reduces the factor $\left(1 - \frac{\beta \cdot m}{\tau}\right)^2$ on the left-hand side. To restore balance, τ must be increased to ensure that the left side satisfies the inequality.

Therefore, for both approaches, as β increases, τ must also increase to satisfy the stricter condition requiring more user contributions for each data element.

5.6.3. Discussion on Selecting Appropriate Values for α and β

In the (α, β)-DPFCM framework, the parameters α and β significantly influence the minimum value of τ, which determines the number of data elements that each user must contribute. If either α or β increases, τ must also increase to satisfy the stricter requirements imposed by these parameters. Higher values of α demand that a larger proportion of data elements meet the contribution threshold, while higher values of β require more users to contribute to each data element. As a result, increasing α and β increases the utility of the collected dataset by collecting more comprehensive and representative data.

However, this improvement in utility comes at a cost. An increased value of τ directly results in higher computational overhead due to the DDP mechanism. Users must process and transmit more data elements, and the server must handle more contributions, increasing the complexity of secure aggregation and decryption. Therefore, the balance between utility and efficiency becomes critical.

The proposed approach allows for dynamic adjustment of the efficiency–utility trade-offs for data collection by carefully selecting the values of α and β based on the intended data analysis objectives and available computational resources. For applications that prioritize high utility, larger values of α and β can be chosen to ensure a more comprehensive and representative dataset. On the other hand, for scenarios where efficiency and scalability are critical due to resource constraints, smaller values of α and β can be chosen to minimize the computational overhead, thereby optimizing the use of available resources.

6. Experiments

In this section, we evaluate the proposed scheme using real datasets. We first describe the experimental setup and then present a discussion of the results.

6.1. Experimental Setup

Datasets: In this study, we evaluate the performance of the proposed method using real-world data to demonstrate its practical applicability. The T-Drive dataset [34], which consists of GPS-based driving records of 10,357 taxis operating in Beijing, was used for our experiments. To generate a PoI dataset from the T-Drive data, we employed the following process: First, the entire geographic region was divided into 10,000 equally sized zones, and the location of each taxi was mapped to its corresponding zone. Second, in order to focus on areas of significant activity, the 1000 most frequently visited zones were identified and considered as PoIs. The number of visits to each zone was used as the value associated

with the PoI. Finally, to simulate different dataset sizes, we randomly selected two subsets of taxis from the 10,357 available taxis, consisting of 1000 and 3000 taxis, respectively. In this setup, each taxi was considered as a user, the PoIs were treated as a set of data elements (E), and the frequency of each taxi's visits to a PoI was used as the value ($f_{t,k}$) associated with the data element $e_k \in E$.

Baseline: For the proposed (α, β)-DPFCM framework, we implemented two different methods: the binomial model-based approach ($DPFCM_{bi}$) and the Chernoff bound-based approach ($DPFCM_{ch}$). To evaluate the effectiveness of our framework, we conducted a comparative evaluation against the LDP-based method using the staircase mechanism [15] (LDP_{SM}), which introduces less noise into the original data compared to the Gaussian and Laplace mechanisms, as well as the DDP-based approach (DDP) from [18].

Evaluation Metrics: In our experiments, we used mean absolute error (MAE) and Jensen–Shannon divergence (JSD) as evaluation metrics for comparative analysis. MAE is defined as follows:

$$MAE = \frac{1}{m} \times \sum_{k=1}^{m} |af_k - af'_k| \tag{18}$$

Here, af_k represents the true aggregated frequency value for the data element e_k, computed as the sum of all individual contributions over n users, expressed as $af_k = \sum_{t=1}^{n} f_{t,k}$. On the other hand, af'_k denotes the noisy aggregated frequency value computed using the DP mechanism.

Furthermore, JSD is defined as follows:

$$JSD = \frac{D_{KL}(D(P)||D(P')) + D_{KL}(D(P')||D(P))}{2} \tag{19}$$

Here, D_{KL} represents the Kullback–Leibler divergence, P denotes the true probability distribution of the aggregated frequency values derived by normalizing af_k across all data elements, and P' corresponds to the noisy probability distribution obtained by normalizing the DP private noisy aggregated values af'_k.

6.2. Evaluation of τ Computation Methods in (α, β)-DPFCM Framework

Before presenting the experimental results on the utility of the collected data, we first evaluate the effectiveness of the proposed binomial model-based and Chernoff bound-based approaches for computing τ. Figure 3 shows the differences in τ computed using the Chernoff bound-based approach and the binomial model-based approach under varying values of α and β. As shown in the figure, the τ values computed using the Chernoff bound-based approach are consistently higher than those derived from the binomial model-based approach. This is because the Chernoff bound provides a conservative estimate by bounding the tail probabilities with an exponential decay term, providing a stronger guarantee that the desired confidence level α is met. The squared deviation factor, $(1-\gamma)^2$, in the Chernoff inequality amplifies deviations, leading to an overestimation of τ. In contrast, the binomial model-based approach directly computes the exact probabilities using the CDF of the binomial distribution. This results in accurate τ values that are sufficient to meet the requirements without unnecessary overestimation.

Table 1 shows the failure rate of (α, β)-DPFCM requirements for the binomial model-based ($DPFCM_{bi}$) and the Chernoff bound-based ($DPFCM_{ch}$) approaches. The failure rate is defined as the proportion of experiments in which the (α, β)-DPFCM requirement was not satisfied:

$$\text{Failure Rate} = \frac{\text{Number of experiments not satisfying } (\alpha, \beta)\text{-DPFCM requirements}}{\text{Total number of experiments}}.$$

In this experiment, the total number of experiments conducted is 200. The failure rate provides a measure of the effectiveness of each approach in satisfying the (α, β)-DPFCM requirement under varying values of α and β.

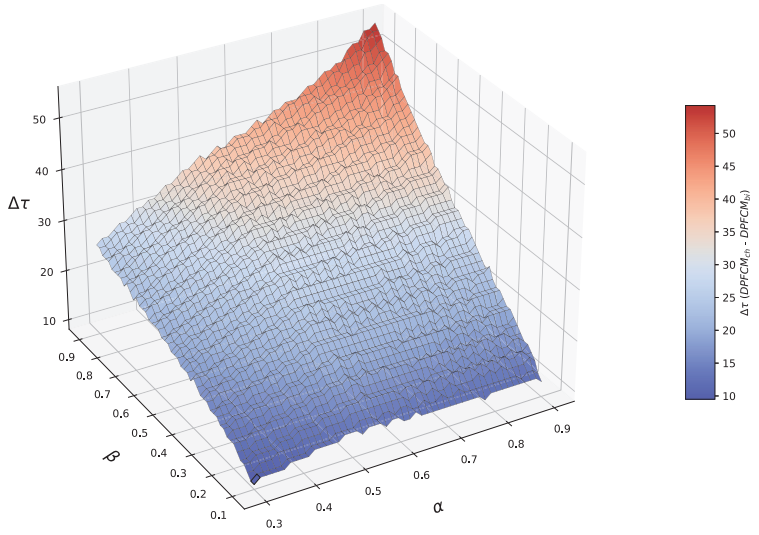

Figure 3. Difference in τ between the Chernoff bound-based approach and the binomial model-based approach.

Table 1. Failure rate of (α, β)-DPFCM requirements for binomial model-based and Chernoff bound-based approaches.

β	$DPFCM_{bi}$					$DPFCM_{ch}$				
	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$
0.1	0	0	0	0	0	0	0	0	0	0
0.2	0.010	0.026	0.028	0.075	0.025	0	0	0	0	0
0.3	0.015	0.035	0.030	0.085	0.080	0	0	0	0	0
0.4	0.035	0.075	0.086	0.095	0.105	0	0	0	0	0
0.5	0.055	0.075	0.090	0.100	0.120	0	0	0	0	0

As shown in Table 1, the $DPFCM_{bi}$ approach has zero failure rates for lower β values. However, it tends to increase the failure rate as α and β increase. On the contrary, the $DPFCM_{ch}$ approach consistently achieves a zero failure rate across all parameter settings, indicating that it satisfies the (α, β)-DPFCM requirement in all experiments, regardless of the values of α and β.

The experimental results in this subsection regarding the computation of τ highlight distinct trade-offs between the two approaches, as explained in Section 5.3. The binomial model-based approach ($DPFCM_{bi}$) utilizes tighter τ values, making it highly effective in reducing the computational overhead of using DDP, as fewer data elements need to be processed by both users and the server. However, this efficiency comes at the cost of occasional failure to satisfy the (α, β)-DPFCM requirement. In contrast, the approach based on the Chernoff bound ($DPFCM_{ch}$) uses slightly overestimated τ values, which results in a slightly higher computational overhead. However, this ensures that the (α, β)-DPFCM requirement is always satisfied. Thus, it is desirable to use $DPFCM_{bi}$ for applications where efficiency and scalability are critical, even with occasional failures in meeting data collection requirements. On the other hand, $DPFCM_{ch}$ is better suited for applications where satisfying data collection requirements is paramount and sufficient computing resources are available.

Figure 4 presents a comparison of the number of data elements sent by each user to the server in the DDP-based approaches (i.e., the proposed $DPFCM_{bi}$ and $DPFCM_{ch}$ methods, and DDP). The experiments were conducted with two different user sizes (1000 and 3000). As shown in the figure, compared to DDP where users must send all data elements to the server, the proposed approaches require users to send only τ selected data elements. This results in a significant reduction in the number of data elements sent to the server, thereby significantly reducing the overhead associated with cryptographic operations in the DDP-based approach. This reduction highlights the efficiency of the proposed framework, particularly in scenarios where minimizing computational overhead is critical.

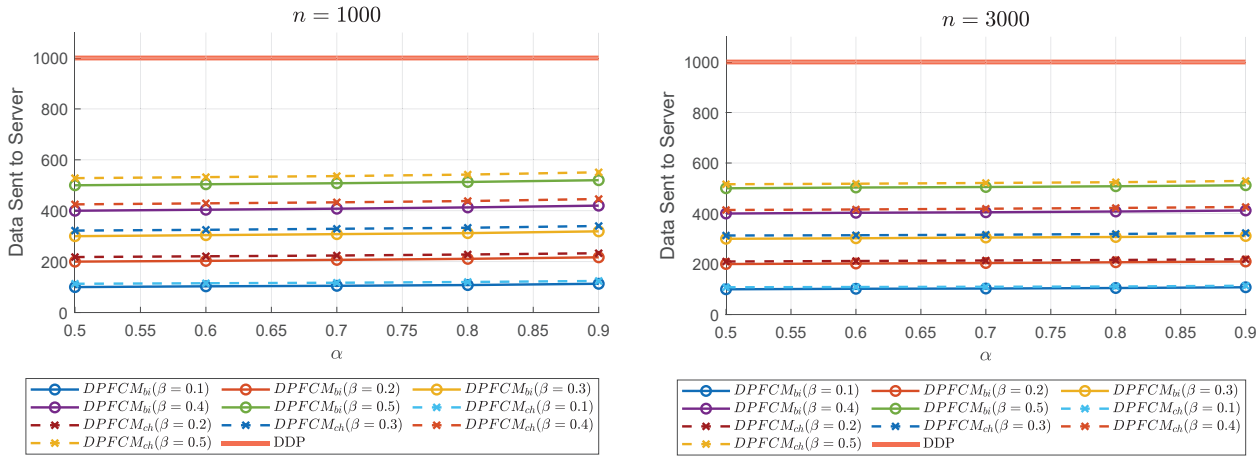

Figure 4. Comparison of data elements sent by each user to server.

Scalability and Computational Considerations

As demonstrated in the experimental results, (α, β)-DPFCM effectively reduces the number of transmitted data elements compared to DDP-based approaches, enhancing scalability for large-scale applications. However, since secure aggregation is employed to protect user data during collection, it introduces additional computational costs, which increase with the number of participating users.

Recent advances in secure aggregation techniques, particularly in federated learning, have introduced efficient cryptographic protocols that significantly reduce communication complexity and computational overhead while maintaining strong privacy guarantees. Protocols such as those proposed in [19,20] utilize optimized masking techniques and dropout-resistant aggregation mechanisms, ensuring that secure aggregation remains scalable even as the number of users increases. With these protocols, (α, β)-DPFCM remains computationally viable for large-scale applications. In addition, reducing the number of transmitted data elements further mitigates the cryptographic overhead introduced by secure aggregation.

6.3. Evaluation Results on Data Utility

In this subsection, we present the evaluation results for the data utility of the collected data. Figures 5 and 6 illustrate the impact of the privacy budget, ϵ, on MAE and JSD, respectively. The experiments were conducted using two different values of α, specifically 0.6 and 0.9, while varying β from 0.1 to 0.5. The number of users was fixed at 1000, while ϵ was varied from 0.25 to 1.0. Note that in Figures 5 and 6, some results for LDP_{SM} are presented with text annotations on the bars, as the LDP_{SM} results differ significantly from those of the other approaches.

As the privacy budget ϵ decreases, both MAE and JSD increase consistently across all evaluated methods. This behavior reflects the inherent trade-off in DP-based approaches:

achieving stronger privacy guarantees (lower ϵ) necessitates adding more noise to the data, which consequently reduces the utility of the aggregated results. This trade-off is a fundamental characteristic of DP, where greater privacy protection comes at the expense of accuracy in data analysis. Among the evaluated methods, LDP_{SM} shows the worst performance in terms of both MAE and JSD. This is mainly due to the significant noise added at the local level for each user's data, which significantly distorts the aggregated results. The localized noise addition of LDP_{SM}, while enhancing privacy, leads to excessive obfuscation that severely impacts data utility. Given that LDP_{SM} is a more optimized approach compared to the Gaussian and Laplace mechanisms, these results highlight the fundamental limitations of LDP-based methods in data collection. As a result, LDP-based methods are unsuitable for scenarios requiring high data utility, particularly in applications where precise statistical analysis or accurate pattern recognition is essential.

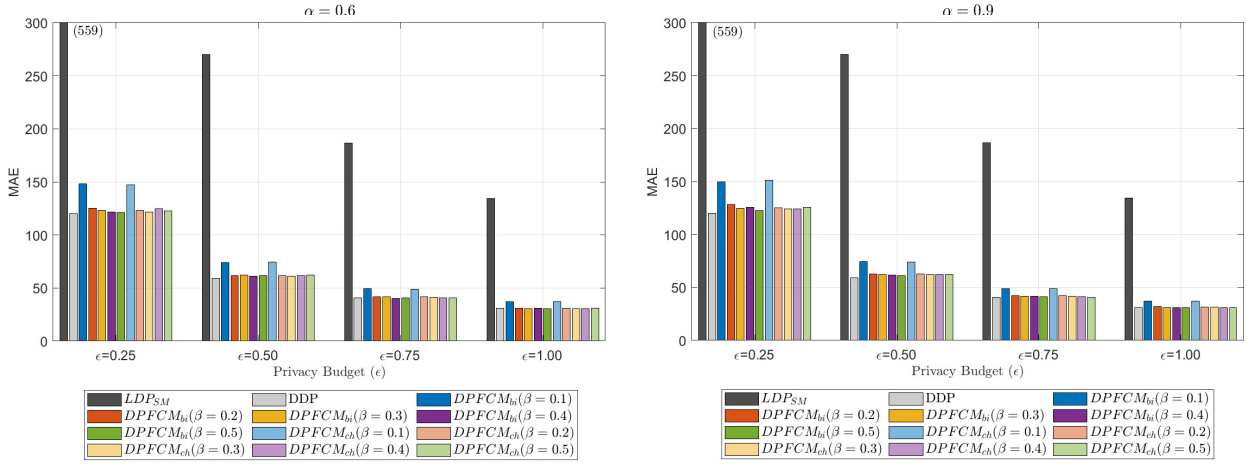

Figure 5. Impact of ϵ and β on MAE.

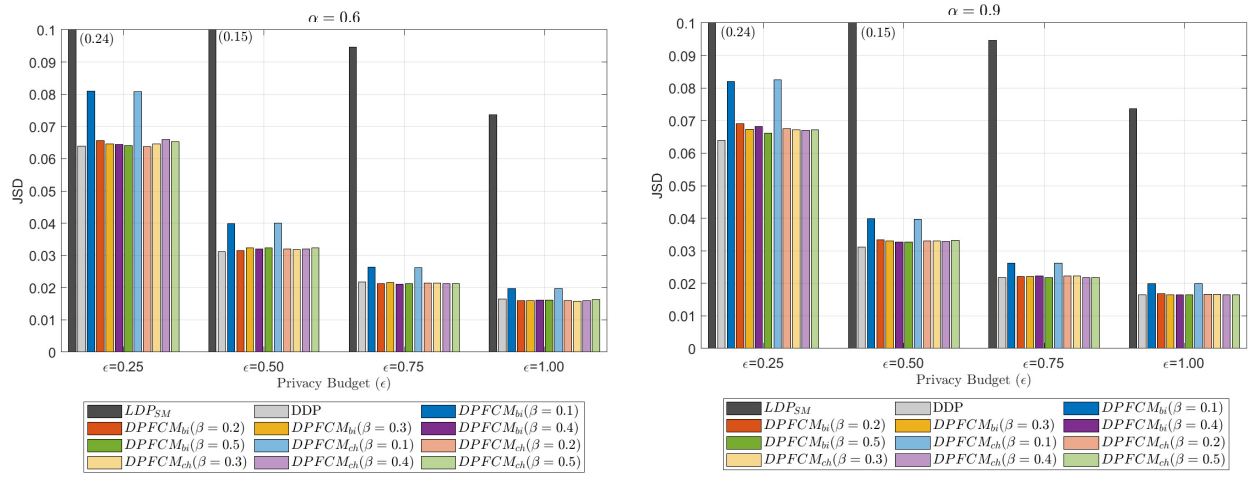

Figure 6. Impact of ϵ and β on JSD.

The proposed (α, β)-DPFCM framework, including $DPFCM_{bi}$ and $DPFCM_{ch}$, significantly outperforms the LDP_{SM} method in both MAE and JSD metrics. Furthermore, the results achieved by the proposed framework are comparable to those of DDP-based approaches, which are characterized by significantly higher computational overhead. This increased overhead in DDP is due to the requirement that each user sends all DP-noised data elements to the server, which then processes the encrypted data for decryption. As shown in the experimental results in Figure 4, the proposed (α, β)-DPFCM framework significantly

reduces the computational burden by requiring users to send only τ selected data elements, while still maintaining utility comparable to DDP-based methods.

Figures 5 and 6 also illustrate the impact of β on the data utility of the collected data. As β increases, both MAE and JSD show a decreasing trend. This is because a higher β ensures that more users contribute to each data element, thereby improving the accuracy in the aggregated results. In particular, there is a significant decrease in both MAE and JSD when β is increased from 0.1 to 0.2. Beyond this point, the improvements in MAE and JSD become marginal. This observation confirms that, even with a small value of β, which significantly reduces the number of data elements transmitted to the server compared to DDP, the proposed (α, β)-DPFCM framework achieves performance comparable to that of DDP.

The experimental results in this subsection verify that the proposed (α, β)-DPFCM framework significantly outperforms LDP_{SM} in terms of data utility, while achieving results comparable to DDP. Furthermore, it significantly reduces the number of data elements transmitted to the server, making (α, β)-DPFCM an effective solution for applications that prioritize both utility and efficiency.

7. Conclusions and Future Work

In this paper, we addressed the challenge of balancing data utility and computational efficiency in privacy-preserving data collection frameworks. Motivated by the variability in data utility requirements across applications, we proposed the (α, β)-DPFCM framework, which provides a flexible solution for applications that can effectively operate with partial data collection. DPFCM introduces two key parameters, α and β, which allow the framework to control the breadth of data elements collected and the depth of user representation, respectively. To satisfy the (α, β)-DPFCM requirements, we developed two probability-based methods—binomial model-based and Chernoff bound-based approaches—that determine the minimum data contribution required from each user.

The experimental results validate the practicality and effectiveness of DPFCM. Using real-world datasets, we verified that the proposed framework achieves comparable data utility to DDP, which is known for its high computational cost. At the same time, DPFCM significantly reduces computational overhead by requiring users to contribute only a fraction of their data. This efficiency is particularly valuable for applications requiring real-time decision-making, where responsiveness is critical.

While this work focuses primarily on the efficiency-utility trade-offs in privacy-preserving data collection, an important aspect for future research is the impact of high-frequency collisions on aggregate results. In scenarios where multiple users report the same data elements, such collisions can affect the statistical properties of the collected dataset. Investigating mitigation strategies for such effects will be an important direction for future research.

Author Contributions: Conceptualization, J.K.; Methodology, J.K.; Software, J.K.; Validation, J.K.; Formal analysis, J.K.; Investigation, J.K.; Writing—original draft, J.K.; Writing—review & editing, S.-H.C.; Visualization, J.K.; Project administration, S.-H.C.; Funding acquisition, J.K. and S.-H.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF-2023R1A2C1004919) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00884, Development of Integrated Platform for Untact Collaborative Solution and Blockchain Based Digital Work).

Data Availability Statement: The original data presented in the study are openly available in Kaggle at https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/ (accessed on 1 August 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Rong, C.; Ding, J.; Li, Y. An interdisciplinary survey on origin-destination flows modeling: Theory and techniques. *ACM Comput. Surv.* **2024**, *57*, 1–49. [CrossRef]
2. Behara, K.N.S.; Bhaskar, A.; Chung, E. A DBSCAN-based framework to mine travel patterns from origin-destination matrices: Proof-of-concept on proxy static OD from Brisbane. *Transp. Res. Part Emerg. Technol.* **2021**, *131*, 103370. [CrossRef]
3. Jia, J.S.; Lu, X.; Yuan, Y.; Xu, G.; Jia, J.; Christakis, N.A. Population flow drives spatio-temporal distribution of COVID-19 in China. *Nature* **2020**, *582*, 389–394. [CrossRef]
4. Chen, R.; Li, L.; Ma, Y.; Gong, Y.; Guo, Y.; Ohtsuki, T.; Pan, M. Constructing mobile crowdsourced COVID-19 vulnerability map with geo-indistinguishability. *IEEE Internet Things J.* **2022**, *9*, 17403–17416. [CrossRef]
5. Yu, Z.; Ma, H.; Guo, B.; Yangi, Z. Crowdsensing 2.0. *Commun. ACM* **2021**, *64*, 76–80. [CrossRef]
6. Kim, J.W.; Lim, J.H.; Moon, S.M.; Jang, B. Collecting health lifelog data from smartwatch users in a privacy-preserving manner. *IEEE Trans. Consum. Electron.* **2019**, *65*, 369–378. [CrossRef]
7. Saura, J.R.; Ribeiro-Soriano, D.; Palacios-Marques, D. From user-generated data to data-driven innovation: A research agenda to understand user privacy in digital markets. *Int. J. Inf. Manag.* **2021**, *60*, 102331. [CrossRef]
8. Jiang, H.; Li, J.; Zhao, P.; Zeng, F.; Xiao, Z.; Iyengar, A. Location privacy-preserving mechanisms in location-based services: A comprehensive survey. *Acm Comput. Surv.* **2021**, *54*, 1–36. [CrossRef]
9. Dwork, C. Differential privacy. In Proceedings of the International Colloquium on Automata, Languages, and Programming, Venice, Italy, 12–15 July 2006; pp. 1–12.
10. Erlingsson, U.; Pihur, V.; Korolova, A. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014; pp. 1054–1067.
11. Wang, T.; Blocki, J.; Li, N.; Jha, S. Locally differentially private protocols for frequency estimation. In Proceedings of the USENIX Conference on Security Symposium, Berkeley, CA, USA, 14–16 August 2017.
12. Goryczka, S.; Xiong, L. A comprehensive comparison of multiparty secure additions with differential privacy. *IEEE Trans. Dependable Secur. Comput.* **2015**, *14*, 463–477. [CrossRef] [PubMed]
13. Wei, Y.; Jia, J.; Wu, Y.; Hu, C.; Dong, C.; Liu, Z.; Chen, X.; Peng, Y.; Wang, S. Distributed differential privacy via shuffling versus aggregation: A curious study. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 2501–2516. [CrossRef]
14. Bassily, R.; Smith, A. Local, private, efficient protocols for succinct histograms. In Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing, Portland, OR, USA, 14–17 June 2015.
15. Geng, Q.; Kairouz, P.; Oh, S.; Viswanath, P. The staircase mechanism in differential privacy. *IEEE J. Sel. Top. Signal Process.* **2015**, *9*, 1176–1184. [CrossRef]
16. Lyu, L.; Nandakumar, K.; Rubinstein, B.; Jin, J.; Bedo, J.; Palaniswami, M. PPFA: Privacy preserving fog-enabled aggregation in smart grid. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3733–3744. [CrossRef]
17. Xie, Q.; Jiang, S.; Jiang, L.; Huang, Y.; Zhao, Z.; Khan, S. Efficiency optimization techniques in privacy-preserving federated learning with homomorphic encryption: A brief survey. *IEEE Internet Things J.* **2024**, *11*, 24569–24580. [CrossRef]
18. Truex, S.; Baracaldo, N.; Anwar, A.; Steinke, T.; Ludwig, H.; Zhang, R.; Zhou, Y. A hybrid approach to privacy-preserving federated learning. In Proceedings of the the ACM Workshop on Artificial Intelligence and Security, London, UK, 15 November 2019.
19. Kadhe, S.; Rajaraman, N.; Koyluoglu, O.O.; Ramchandran, K. FastSecAgg: Scalable secure aggregation for privacy-preserving federated learning. *arXiv* **2020**, arXiv:2009.11248.
20. Bell, J.H.; Bonawitz, K.A.; Gascon, A.; Lepoint, T.; Raykova, M. Secure single-server aggregation with (poly)logarithmic overhead. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Virtual, 9–16 November 2020; pp. 1253–1269.
21. Balle, B.; Bell, J.; Gascon, A.; Nissim, K. The privacy blanket of the shuffle model. In Proceedings of the International Cryptology Conference, Santa Barbara, CA, USA, 12–18 August 2019; pp. 638–667.
22. Scott, M.; Cormode, G.; Maple, C. Aggregation and transformation of vector-valued messages in the shuffle model of differential privacy. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 612–627. [CrossRef]
23. Chen, E.; Cao, Y.; Ge, Y. A generalized shuffle framework for privacy amplification: Strengthening privacy guarantees and enhancing utility. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 26–27 February 2024; pp. 11267–11275.

24. Li, K.; Zhang, H.; Liue, Z. A range query scheme for spatial data with shuffled differential privacy. *Mathematics* **2024**, *12*, 1934. [CrossRef]
25. Andres, M.E.; Bordenabe, N.E.; Chatzikokolakis, K.; Palamidessi, C. Geo-indistinguishability: Differential privacy for location-based systems. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013; pp. 901–914.
26. Kim, J.W.; Edemacu, K.; Jang, B. Privacy-preserving mechanisms for location privacy in mobile crowdsensing: A survey. *J. Netw. Comput. Appl.* **2023**, *200*, 103315. [CrossRef]
27. Zhao, Y.; Yuan, D.; Du, J.T.; Chen, J. Geo-Ellipse-Indistinguishability: Community-aware location privacy protection for directional distribution. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 6957–6967. [CrossRef]
28. Fathalizadeh, A.; Moghtadaiee, V.; Alishahi, M. Indoor geo-indistinguishability: Adopting differential privacy for indoor location data protection. *IEEE Trans. Emerg. Top. Comput.* **2023**, *12*, 293–306. [CrossRef]
29. Jin, W.; Xiao, M.; Guo, L.; Yang, L.; Li, M. ULPT: A user-centric location privacy trading framework for mobile crowd sensing. *IEEE Trans. Mob. Comput.* **2022**, *21*, 3789–3806. [CrossRef]
30. Huang, P.; Zhang, X.; Guo, L.; Li, M. Incentivizing crowdsensing-based noise monitoring with differentially-private locations. *IEEE Trans. Mob. Comput.* **2021**, *20*, 519–532. [CrossRef]
31. Zhang, P.; Cheng, X.; Su, S.; Wang, N. Area coverage-based worker recruitment under geo-indistinguishability. *Comput. Netw.* **2022**, *217*, 109340. [CrossRef]
32. Song, S.; Kim, J.W. Adapting geo-indistinguishability for privacy-preserving collection of medical microdata. *Electronics* **2023**, *12*, 2793. [CrossRef]
33. Tian, H.; Zhang, F.; Shao, Y.; Li, B. Secure linear aggregation using decentralized threshold additive homomorphic encryption for federated learning. *arXiv* **2021**, arXiv:2111.10753.
34. T-Drive Trajectory Data Sample. 2018. Available online: https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample (accessed on 1 August 2024).