A Practical Framework for Privacy-Preserving Data Analytics

Liyue Fan[*]
Integrated Media Systems Center
University of Southern California
Los Angeles, CA, USA
liyuefan@usc.edu

Hongxia Jin
Samsung R&D Research Center
San Jose, CA, USA
hongxia@acm.org

ABSTRACT

The availability of an increasing amount of user generated data is transformative to our society. We enjoy the benefits of analyzing big data for public interest, such as disease outbreak detection and traffic control, as well as for commercial interests, such as smart grid and product recommendation. However, the large collection of user generated data contains unique patterns and can be used to re-identify individuals, which has been exemplified by the AOL search log release incident. In this paper, we propose a practical framework for data analytics, while providing *differential privacy* guarantees to individual data contributors. Our framework generates differentially private aggregates which can be used to perform data mining and recommendation tasks. To alleviate the high perturbation errors introduced by the differential privacy mechanism, we present two methods with different sampling techniques to draw a subset of individual data for analysis. Empirical studies with real-world data sets show that our solutions enable accurate data analytics on a small fraction of the input data, reducing user privacy risk and data storage requirement without compromising the analysis results.

Categories and Subject Descriptors

H.2.7 [**Database Management**]: Database Administration—*Security, integrity, and protection*; H.2.8 [**Database Management**]: Database Applications—*Data mining*

Keywords

Data Analytics, Differential Privacy, Sampling

1. INTRODUCTION

We live in the age of big data. With an increasing number of people, devices, and sensors connected with digital networks, individual data now can be largely collected and analyzed to understand important phenomena. One example is Google Flu Trends [1], a service that estimates flu activity by aggregating individual search

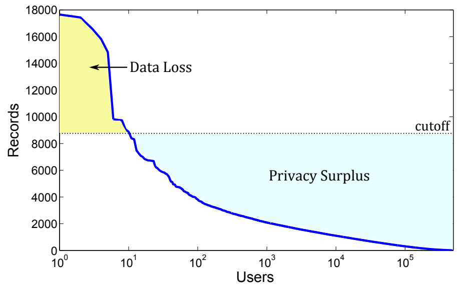


Figure 1: Record Distribution of *Netflix* Users

queries. In the retail market, individual purchase histories are used by recommendation tools to learn trends and patterns. Performing analytics on private data is clearly beneficial, such as early detection of disease and recommendation services. However, user concerns rise from a privacy perspective, with sharing an increasing amount of information regarding their health, location, service usage, and online activities.

As a matter of fact, the uniqueness of each user is increased by the big collection of individual data. The AOL data release in 2006 is an unfortunate example of privacy catastrophe [1], in which the search logs of an innocent citizen were quickly identified by a newspaper journalist. A recent study by de Montjoye et al. [9] concludes that human mobility patterns are highly unique and four spatio-temporal points are enough to uniquely identify 95% of the individuals. In order to protect users from re-identification attacks, their private data must be transformed prior to release for analysis.

The current state-of-the-art paradigm for privacy-preserving data analysis is *differential privacy* [10], which allows un-trusted parties to access private data through aggregate queries. The aggregate statistics are perturbed by a randomized algorithm, such that the output remains roughly the same even if any user is added or removed in the input data. Differential privacy provides a strong guarantee: given the output statistics, an adversary will not be able to infer whether any user is present in the input database. However, this indistinguishability can be only achieved at high perturbation cost. Intuitively, the more data a user contributes to the analysis process, the more perturbation noise is needed to hide his/her presence. In some cases, a user could generate an *unbounded* amount of data, such as purchase or check-in history, the addition or removal of which may result in unlimited impact on the output.

The challenge of enforcing differential privacy is that it incurs a surplus of privacy cost, i.e. high perturbation error, being designed to protect each user according to the highest possible data contribution. In reality, only a very small number of users generate large amount of personal data, while the rest contribute little data each. As shown in Figure 1, out of 500K users from Netflix prize com-

[*]work done while interning with Samsung.
[1]http://www.google.org/flutrends/

WWW 2015, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3469-3/15/05.
http://dx.doi.org/10.1145/2736277.2741122.

petition [2], only 1 user generated around 17K data records, while the majority of users generated much less personal data, less than 2K data records each. If a upper bound is imposed on individual user data contribution, the surplus of privacy, e.g. high perturbation noise, can be reduced at the cost of data loss, i.e. part of data from those users who contributed more than the threshold.

To limit individual data contribution, some strategies have been adopted by several works [16][25]. The authors of [16] used the first d search queries submitted by each user, and the work in [25] reduced the number of items contained in each transaction to l with "smart" truncation. However, there has been no discussion on the choice of the bounds, i.e., d and l. Furthermore, the choice of actual user records (or items in a single transaction) remains non-trivial, for generic applications.

With a rigorous privacy notion, we consider how to analyze individually contributed data to gain a deep understanding of service usage and behavior patterns, for various application domains. We would like to understand the impacts of privacy and data loss on the resulting data analytics, and design algorithms to draw private data accordingly. Example data analytical questions are: "Which places do people visit on Thursdays?" and "What are the most popular movies with female watchers under age 25?" We formally define the tasks as database queries and details are provided in Section 3.

Contributions. In this paper, we address the problem of differentially private data analytics, where each user could contribute a large number of records. We propose a generic framework to generate analysis results on a sampled database, and study two sampling methods as well as the sampling factor in order to achieve a balance between data loss and privacy surplus. We summarize the contributions of this paper as follows:

(**1**) We propose a generic, sampling-based framework for an important class of data analytical tasks: top-K mining and context-aware recommendation. We consider the problem of releasing a set of *count* queries regarding the domain-specific items of interest as well as customizable predicates to answer deep, analytical questions. The count queries are perturbed prior to release such that they satisfy differential privacy.

(**2**) We design two algorithms that draw a sample of user records from the raw database and generate analysis results on the sampled data. The SRA algorithm randomly samples up to l records per user. The HPA algorithm selects up to l records from each user that are most useful for the specific analytical tasks. The utility of each record can be customized based on the actual application domain. We outline each sampling method and provide pseudo code for easy implementation.

(**3**) We provide analysis on the accuracy of random sampling, i.e. Mean Squared Error of released counts, with respect to the sampling factor l. We conclude that the optimal l value is positively correlated the privacy constraint. We show that performing record sampling on individual user's data does not inflict extra privacy leakage. We formally prove that both sampling algorithms satisfy differential privacy.

(**4**) We conduct extensive empirical studies with various real-world data sets. We compare our approaches with existing differentially private mechanisms and evaluate the accuracy of released count data with three utility metrics. The experimental results show that although performed on a small sampled database, our methods provide comparable performance to the best existing approaches in MSE and KL-divergence, and superior performance in top-K discovery and context-aware recommendation tasks. The HPA algorithm yields higher precision, while the SRA algorithm preserves well the distributional properties in released data. We believe that our privacy-preserving framework will enable data analytics for a variety of services, reducing user privacy cost and data storage requirement without compromising output utility.

The rest of the paper is organized as follows: Section 2 briefly surveys the related works on privacy-preserving data publishing and analytics. Section 3 defines the problem and privacy notion. Section 4 presents the technical details of the proposed framework and two sampling algorithms. Theoretical results of privacy guarantees are provided in Section 5. Section 6 describes the data set and presents a set of empirical studies. Finally, Section 7 concludes the paper and states possible directions for future work.

2. RELATED WORKS

A plethora of differentially private techniques have been developed since the introduction of ϵ-differential privacy in [12]. Here we briefly review the most recent, relevant works to our problem.

Differential Privacy. Dwork et al. [12] first proposed ϵ-differential privacy and established the Laplace mechanism to perturb aggregate queries to guarantee differential privacy. Since then, two variants have been proposed and adopted by many works as relaxations of ϵ-differential privacy. The (ϵ, δ)-probabilistic differential privacy [19] achieves ϵ-differential privacy with high probability, i.e. $\geq (1-\delta)$. The (ϵ, δ)-indistinguishability [11, 21] relaxes the bound of ϵ-differential privacy by introducing an additive term δ. Our work adopts the strict definition of ϵ-differential privacy and the Laplace mechanism to release numeric data for analysis.

Data Publication Techniques. A plethora of works have been proposed to publish sanitized data with differential privacy. To list a few representatives among them, there is histogram publication for range queries [7], for a given workload [24], and for sparse data [8]. The majority of data publication methods consider settings where each user contributes only one record, or affects only one released *count*. In contrast, we focus on those services where each individual may contribute a large number of records and could even have unbounded influence on the released *count* queries.

Bounding Individual Contribution. Here we review works established in a similar problem setting, i.e. where individual data contribution is high, i.e. high *global sensitivity*. The work of Nissim et al. [21] proposed *smooth sensitivity*, which measures individual impact on the output statistics in the neighborhood of the database instance. They showed that smooth sensitivity allows a smaller amount of perturbation noise injected to released statistics. However, it does not guarantee ϵ-differential privacy. Proserpio et al. [22] recently proposed to generalize ϵ-DP definition to weighted datasets, and scale down the weights of data records to reduce sensitivity. Rastogi and Nath [23], Fan and Xiong [13] and Chan et. al [5] studied the problem of sharing time series of counts with differential privacy, where the maximum individual contribution is T, the number of time points. The authors of [23] proposed to preserve only k discrete Fourier coefficients of the original count series. The FAST framework in [13] reduced the sensitivity by sampling M points in a count series and predicts at other points. The work [5] proposed the notion of p-sum to ensure each item in the stream only affects a small number of p-sum's. Two works by Korolova et al. [16] and Hong et al. [14] addressed the differentially private publication of search logs, where each user could contribute a large search history. The work of [16] keeps the first d queries of each user, while the work of [14] explicitly removes those users whose data change the optimal output by more than a certain threshold. Zeng et al. [25] studied frequent itemset mining with differential privacy and truncated each individual transaction to contain up to l items. Recently, Kellaris and Papadopoulos [15]

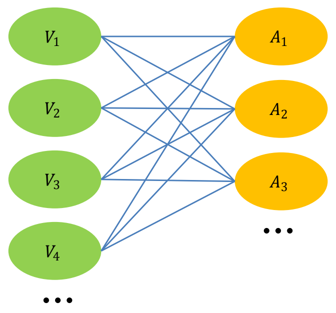


Figure 2: Releasing Counts for Data Analytics

proposed to release non-overlapping count data by grouping similar columns, i.e. items in our definition. In their work, each user is allowed to contribute no more than one records to each column, thus the maximum individual contribution is bounded by the number of columns. However, the binary representation of user data may not truly convey information about each column, i.e. place of interest or product. For example, when the bit for a user and a location is set, we cannot distinguish whether it was an accidental check-in or the user went there many times due to personal preference.

Sampling Differential Privacy. There have been a few works which studied the relationship between sampling and differential privacy. Chaudhuri and Mishra [6] first showed that the combination of k-anonymity and random sampling can achieve differential privacy with high probability. Li et al. [17] proposed to sample each record with a calibrated probability β and then perform k-anonymity on the sampled data, to achieve (ϵ, δ)-indistinguishability. Both works adopt the random sampling technique which samples a data record with certain probability. However, when applied in our setting, no guarantee is provided on bounding the individual data in the sampled database.

Our Competitors. After reviewing existing differentially private technique, we identify three works that allow high individual contribution, release aggregate statistics, and satisfy ϵ-differential privacy. The first is a straight-forward application of Laplace perturbation mechanism [12] to each released count, denoted as `LPA`. The second is the Fourier transform based algorithm from [23], which can be adapted to share count vectors, denoted as `DFT`. The third is `GS`, which is the best method proposed in [15].

3. PRELIMINARIES

3.1 Problem Formulation

Suppose a database D contains data records contributed by a set of n users about m items of interest. Each item could represent in reality a place or a product. Each record in dataset D is a tuple $(rID, uID, vID, attrA)$, where rID is the record ID, uID corresponds to the user who contributed this record, vID is the item which the record is about, and $attrA$ represents contextual/additional information regarding this record. In reality, various information is often available in the actual database, such as transaction time, user ratings and reviews, and user demographic information. In our problem setting, $attrA$ can be an attribute, e.g. *dayOfWeek*, or a set of attributes, e.g. (*Gender*, *Age*), which can be customized to offer deep insight in a specific application domain. Let h denote the number of possible $attrA$ values.

To be more concrete, we select two analytic tasks, i.e. *top-K discovery* and *context-aware recommendation*, to illustrate the usability of our solutions. The first task answers questions as "What are the most popular places in city X?", while the second task aims more specifically at "Recommend good places to visit on a Tuesday!". Moreover, we consider the problem of performing the above tasks on released *count* data. As in Figure 2, each V_i's represents an item of interest, e.g. a restaurant, and each A_j represents a value of the context, e.g."Monday". For each V_i, the number of records containing V_i is released. For each edge connecting V_i and A_j, the number of records containing V_i and A_j is released. As a result, top-K discovery can be performed on the item counts and context-aware recommendation on the edge counts connected to any context A_j. We formally state the problem to investigate below.

DEFINITION 1 (ITEM COUNTS). *For each item V_i in database D, release*

$$c_i(D) \leftarrow \text{select} * \text{from } D \quad \text{where } vID = V_i \tag{1}$$

DEFINITION 2 (EDGE COUNTS). *For each edge connecting item V_i and attribute A_j, release*

$$c_{i,j}(D) \leftarrow \text{select} * \text{from } D \quad \text{where } vID = V_i \text{ and } attrA = A_j \tag{2}$$

PROBLEM 1 (PRIVATE DATA ANALYTICS). *Given database D and privacy parameter ϵ, release a sanitized version of item counts and edge counts, such that the released data satisfies ϵ-differential privacy.*

Note that the problem definition, i.e. the counting queries to release, can be customized according to the analytical task to perform. For instance, to understand the correlation between items, the bipartite graph in Figure 2 can be adapted as follows: A_* nodes will be replaced by items, i.e. V_* nodes; and each edge (V_i, V_j) represents "the number of times that V_j is purchased/watched/visited by users who also purchase/watch/visit V_i". Similarly, those counts can be released privately with slight adaption of our proposed solutions below.

3.2 Privacy Definition

The privacy guarantee provided by our work is *differential privacy* [4]. Simply put, a mechanism is differentially private if its outcome is not significantly affected by the removal or addition of any user. An adversary thus learns approximately the same information about any individual, irrespective of his/her presence or absence in the original database.

DEFINITION 3 (ϵ-DIFFERENTIAL PRIVACY). A non-interactive privacy mechanism $\mathcal{A} : \mathcal{D} \rightarrow \mathcal{T}$ satisfies ϵ-differential privacy if for any neighboring databases D_1 and D_2, and for any set $\widetilde{D} \in \mathcal{T}$,

$$Pr[\mathcal{A}(D_1) = \widetilde{D}] \leq e^{\epsilon} \cdot Pr[\mathcal{A}(D_2) = \widetilde{D}] \tag{3}$$

where the probability is taken over the randomness of $\mathcal{A}$.

The privacy parameter ϵ, also called the *privacy budget* [20], specifies the degree of privacy offered. Intuitively, a lower value of ϵ implies stronger privacy guarantee and a larger perturbation noise, and a higher value of ϵ implies a weaker guarantee while possibly achieving higher accuracy. The neighboring databases D_1 and D_2 differ on at most one user.

Laplace Mechanism. Dwork et al. [12] show that ϵ-differential privacy can be achieved by adding i.i.d. noise to query result $q(D)$:

$$\tilde{q}(D) = q(D) + (\tilde{N}_1, \ldots, \tilde{N}_z)^{\intercal} \tag{4}$$

$$\tilde{N}_i \sim Lap(0, \frac{GS(q)}{\epsilon}) \text{ for } i = 1, \ldots, z \tag{5}$$

Symbol	Description
$D/\mathcal{D}$	Input database / Domain of all databases
T_k	Set of records contributed by user u_k in D
$D_R/\mathcal{D_R}$	SRA sampled database / Domain of D_R
$D_G/\mathcal{D_G}$	HPA sampled database / Domain of D_G
$D_E/\mathcal{D_E}$	HPA sampled database / Domain of D_E
$\mathbf{q_1}/\tilde{\mathbf{q}}_1$	Query of all item counts / Noisy output of $\mathbf{q_1}$
$\mathbf{q_2}/\tilde{\mathbf{q}}_2$	Query of all edge counts / Noisy output of $\mathbf{q_2}$
$\mathbf{p}/\tilde{\mathbf{p}}$	Popularity vector for all items / Estimation of $\mathbf{p}$
M	Max records per user allowed in D
l	Max records per user allowed in D_R and D_G
d	Max records per user allowed in D_E

Table 1: Summary of notations

where z represents the dimension of $q(D)$. The magnitude of $\tilde{N}$ conforms to a Laplace distribution with 0 mean and $GS(q)/\epsilon$ scale, where $GS(q)$ represents the *global sensitivity* [12] of the query q. The global sensitivity is the maximum L1 distance between the results of q from any two neighboring databases. Formally, it is defined as follows:

$$GS(q) = \max_{D_1, D_2} ||q(D_1) - q(D_2)||_1 \ . \tag{6}$$

Sensitivity Analysis. Let M denote the maximum number of records any user could contribute and $\mathcal{D}$ denote the domain of database D. Let $\mathbf{q_1} = \{c_1, \ldots, c_m\}$ output the item counts for every V_i. Let $\mathbf{q_2} = \{c_{1,1}, c_{1,2}, \ldots, c_{m,h}\}$ output the edge counts for every V_i and A_j. The following lemmas establish the global sensitivity of $\mathbf{q_1}$ and $\mathbf{q_2}$, in order to protect the privacy of each individual user. The proof is quite straightforward thus omitted here for brevity.

LEMMA 1 (ITEM COUNTS SENSITIVITY). *The global sensitivity of* $\mathbf{q_1} : \mathcal{D} \to \mathbb{R}^m$ *is* M, *i.e.*

$$GS(\mathbf{q_1}) = M. \tag{7}$$

LEMMA 2 (EDGE COUNTS SENSITIVITY). *The global sensitivity of* $\mathbf{q_2} : \mathcal{D} \to \mathbb{R}^{mh}$ *is* M, *i.e.*

$$GS(\mathbf{q_2}) = M. \tag{8}$$

Composition.. The composition properties of differential privacy provide privacy guarantees for a sequence of computations, which can be applied to mechanisms that require multiple steps.

THEOREM 1 (SEQUENTIAL COMPOSITION [20]). Let $\mathcal{A}_i$ each provide ϵ_i-differential privacy. A sequence of $\mathcal{A}_i(D)$ over the dataset D provides $(\sum_i \epsilon_i)$-differential privacy.

4. PROPOSED SOLUTIONS

Below we describe two sampling-based solutions to privacy-preserving data analytics. The notations used in the problem definition and our proposed solutions are summarized in Table 1.

4.1 Simple Random Algorithm (SRA)

Our first solution has been inspired by the fact that the maximum number of records contributed by each user, i.e. M, could be rather large in real applications. For example, the *Netflix* user who contributed the most data submitted $17,000$ reviews, as shown in Table 4. In fact, a user could contribute as many records as the domain size, i.e. m, as in the total number of movies on Netflix. As a result of the large magnitude of M, a very high perturbation noise is required to provide differential privacy, according to the Laplace mechanism. Furthermore, the number of records contributed by each user can be unbounded for many applications, as a user could repeatedly check in at the same location or purchase the same product. In that case, M may not be known without breaching individual user privacy.

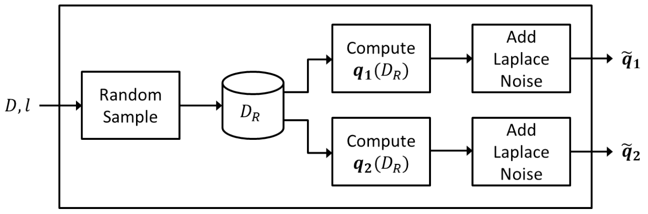


Figure 3: Outline of SRA Algorithm

Algorithm 1 Simple Random Algorithm (SRA)

Input: raw dataset D, sampling factor l, privacy budget ϵ
Output: sanitized answer $\tilde{q}_1$ and $\tilde{q}_2$

```
/* Simple Random Sampling */
1: $D_R \leftarrow \emptyset$
2: for k = 1, ..., n do
3:     $T_k \leftarrow \sigma_{uID=u_k}(D)$   /* $T_k$: records of user $u_k$ */
4:     if $|T_k| \leq l$ do
5:         $D_R \leftarrow D_R \bigcup T_k$
6:     else do
7:         $T'_k \leftarrow$ random sample $l$ records from $T_k$
8:         $D_R \leftarrow D_R \bigcup T'_k$
/* Generate Private Item Counts */
9: $\mathbf{q_1}(D_R) \leftarrow$ compute count $c_i(D_R)$ for every $i$
10: Output $\tilde{\mathbf{q}}_1(D_R) = \mathbf{q_1}(D_R) + \langle Lap(\frac{l}{\epsilon_1})\rangle^m$
/* Generate Private Edge Counts */
11: $\mathbf{q_2}(D_R) \leftarrow$ compute count $c_{i,j}(D_R)$ for every $i, j$
12: Output $\tilde{\mathbf{q}}_2(D_R) = \mathbf{q_2}(D_R) + \langle Lap(\frac{l}{\epsilon_2})\rangle^{mh}$
```

In order to mitigate the effect of very large or unbounded individual data contribution, we propose to sample the raw input dataset D and allow up to l records per user in the sampled database. Therefore, the individual contribution to the sampled database is bounded by the fixed constant l. The aggregate statistics will be generated from the sampled data and then perturbed correspondingly in order to guarantee differential privacy. The sampling technique used in our solution is simple random sampling without replacement, after which our solution is named. An outline of the algorithm is provided in Figure 3.

Given the input database D and a pre-defined sampling factor l, the SRA method generates a sampled database D_R by random sampling without replacement at most l records for each user in input database D. The sampled database D_R could be different every time the algorithm is run, due to the randomness of sampling. However, it is guaranteed that for every possible sample D_R, any user could have no more than l records. The following lemma establishes the sensitivity of $\mathbf{q_1}$ and $\mathbf{q_2}$ under such constraint.

LEMMA 3 (SAMPLE SENSITIVITY). *In the domain of* D_R, *it holds that* $GS(\mathbf{q_1}) = l$ *and* $GS(\mathbf{q_2}) = l$.

Subsequently, the SRA method computes the query answers to $\mathbf{q_1}$ and $\mathbf{q_2}$ from the sampled database D_R, where all individual count queries c_i and $c_{i,j}$ are evaluated based on the data records in D_R. According to the Laplace mechanism, it is sufficient to add perturbation noise from $Lap(\frac{l}{\epsilon_1})$ to each item count $c_i(D_R)$ to guarantee ϵ_1-differential privacy. Similarly, adding perturbation noise from $Lap(\frac{l}{\epsilon_2})$ to each edge count $c_{i,j}(D_R)$ guarantees ϵ_2-differential privacy. The pseudocode of SRA method is provided in Algorithm 1.

rID	uID	vID	Day-Of-Week
r_1	Alice	Gym	Monday
r_2	Alice	Mary's house	Tuesday
r_3	Alice	de Young Museum	Friday
r_4	Alice	Golden Gate Bridge	Saturday

Table 2: Example Check-in Records

To sum up, SRA injects low Laplace noise into released query results, due to reduced sensitivity in the sampled database. However, the accuracy of released query results is affected by only using D_R, a subset of the input data D. Intuitively, the more we sample from each user, the closer $\mathbf{q_1}(D_R)$ and $\mathbf{q_2}(D_R)$ are to the true results $\mathbf{q_1}(D)$ and $\mathbf{q_2}(D)$, respectively, at the cost of a higher Laplace perturbation error to achieve differential privacy. Below we formally analyze the trade-off between accuracy and privacy for query $\mathbf{q_1}$ to study the optimal choice of l. Similar analysis can be conducted for query $\mathbf{q_2}$ and is thus omitted here for brevity.

DEFINITION 4 (MEAN SQUARED ERROR). *Let $\tilde{c}_i$ denote the noisy count released by $\tilde{\mathbf{q}}_\mathbf{1}(D_R)$ and c_i denote the real count computed by $\mathbf{q_1}(D)$, for each item V_i. The Mean Squared Error of the noisy count $\tilde{c}_i$ is defined as follows:*

$$MSE(\tilde{c}_i) = Var(\tilde{c}_i) + (Bias(\tilde{c}_i, c_i))^2 \; . \tag{9}$$

THEOREM 2. *Given D_R is a simple random sample of D and $\tilde{\mathbf{q}}_\mathbf{1}(D_R) = \mathbf{q_1}(D_R) + \langle Lap(\frac{l}{\epsilon_1})\rangle^m$, the value of l that minimizes MSE is a monotonically increasing function of ϵ_1^2.*

PROOF. *See Appendix A.* □

The above theorem provides a guideline to choose the l value given the privacy budget ϵ_1: when the privacy budget is higher, we can afford to use more private data to overcome the error due to data loss; When privacy budget is limited, a small number of data records should be taken from each user to reduce the perturbation error by the differential privacy mechanism.

4.2 Hand-Picked Algorithm (HPA)

Observing that a majority of data analytical tasks depend on "popular" places or products, such as in traffic analysis and recommendation services, data related to popular items should preferably be preserved in the sample database. In other words, some records generated by one user might be more useful for data analytics than the rest. The following example illustrates the concept of record "usefulness".

EXAMPLE 1. *Table 2 illustrates Alice's check-in records in the raw database. Among the 4 places Alice has been, de Young Museum and the Golden Gate Bridge are places of interest and attract a large number of visitors. On the other hand, gym and Mary's house are local and personal to Alice and may not interest other people. Therefore we consider r_3 and r_4 more useful than r_1 and r_2 for data analytics. However, r_1 and r_2 may be chosen by SRA over r_3 and r_4, due to the simple random sampling procedure.*

From Example 1, it can be seen that r_3 and r_4 should be picked by the sampling procedure over r_1 and r_2, in order to generate meaningful recommendation results. Therefore, we define the following popularity-base utility score for each private data record and propose to preserve records with highest scores for each user.

DEFINITION 5 (UTILITY SCORE). *Given record r and $r.vID = V_i$, the utility score of r is defined as follows:*

$$score(r) = p_i \tag{10}$$

where p_i represents the underlying popularity of item V_i.

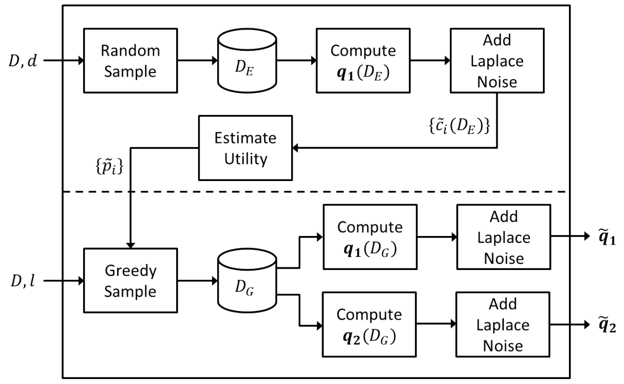


Figure 4: Outline of HPA Algorithm

rID	uID	vID	Day-Of-Week	Utility
r_4	Alice	Golden Gate Bridge	Saturday	0.2
r_3	Alice	de Young Museum	Friday	0.1
r_1	Alice	Gym	Monday	0.001
r_2	Alice	Mary's house	Tuesday	0.0001

Table 3: Example Check-in Records Sorted by Utility Score

Note that the record utility can be defined in other ways according to the target analytical questions. Our choice of the popularity-based measure is motivated by the tasks of discovering popular places or products, as well as the fact that popular items are less personal/sensitive to individual users.

In order to maximize the utility of the sampled database, we propose to greedily pick up to l records with highest utility scores for each user. Note that a user's records with the same score will have equal chance to be picked. The outline of HPA is provided in Figure 4. Below we describe (1) private estimation of record utility and (2) greedy sampling procedure.

Popularity Estimation. For each item V_i, the popularity p_i represents the probability of any record r having $r.vID = V_i$, which is often estimated by the relative frequency of such records. However, the estimation of p_i's from the private user data must not violate the privacy guarantee. We present our privacy-preserving utility estimation in Algorithm 2 from Line 1 to Line 7, which is outlined in the upper half of Figure 4.

The utility estimation is also conducted on a sampled database D_E with sampling factor d. D_E is obtained by randomly choosing up to d records per user from the raw database D. We adopt randomly sampling here because we do not have prior knowledge about the database at this point. The query $\mathbf{q_1}$ is computed based on D_E and each count is perturbed with Laplace noise from $Lap(\frac{d}{\epsilon_0})$. The perturbed counts $\{\tilde{c}_i(D_E)\}$ are used to estimate the popularity for each item V_i by the following normalization:

$$\tilde{p}_i = \frac{max(\tilde{c}_i(D_E), 0)}{\sum_{i=1}^{m} max(\tilde{c}_i(D_E), 0)} \; . \tag{11}$$

Since the Laplace perturbation noise is a random variable and therefore could be negative, we replace the negative counts with 0's in computing item popularity. The resulting $\tilde{p}_i$ is used to estimate the utility score of each record r with $r.vID = V_i$.

The following lemma establishes the sensitivity of $\mathbf{q_1}$ and $\mathbf{q_2}$ where each user can contribute up to d records. The proof is straightforward and is thus omitted.

LEMMA 4. *In the domain of D_E, it holds that $GS(\mathbf{q_1}) = d$.*

Greedy Sampling. The greedy sampling procedure hand-picks up to l records with highest utility scores among each user's data in

Algorithm 2 Hand-Picked Algorithm (HPA)

Input: raw dataset D, sampling factor l
Output: sanitized answer $\tilde{q}_1$ and $\tilde{q}_2$

```
  /* Popularity Estimation */
    /* Random Sample */
 1: $D_E \leftarrow \emptyset$
 2: for k = 1, ..., n do
 3:    $T_k \leftarrow \sigma_{uID=u_k}(D)$   /* $T_k$: records of user $u_k$ */
 4:    Random sample $d$ record from $T_k$, add to $D_E$
    /* Generate Private Item Counts */
 5: $\mathbf{q_1}(D_E) \leftarrow$ compute count $c_i(D_E)$ for every $i$
 6: $\tilde{\mathbf{q}}_\mathbf{1}(D_E) = \mathbf{q_1}(D_E) + \langle Lap(\frac{d}{\epsilon_0})\rangle^m$
    /* Estimate Popularity */
 7: $\tilde{\mathbf{p}} \leftarrow$ normalize histogram $\tilde{\mathbf{q}}_\mathbf{1}(D_E)$
  /* Greedy Sampling */
 8: $D_G \leftarrow \emptyset$
 9: for k = 1, ..., n do
10:    $T_k \leftarrow \sigma_{uID=u_k}(D)$
11:    if $|T_k| \le l$ do
12:        $D_G \leftarrow D_G \bigcup T_k$
13:    else do
14:        for record $r$ in $T_k$ do
15:           assign $score(r) = \tilde{p}_i$ iff $r.vID = V_i$
16:        $T'_k \leftarrow$ pick $l$ records with highest scores from $T_k$
17:        $D_G \leftarrow D_G \bigcup T'_k$
  /* Generate Private Item Counts */
18: $\mathbf{q_1}(D_G) \leftarrow$ compute count $c_i(D_G)$ for every $i$
19: Output $\tilde{\mathbf{q}}_\mathbf{1}(D_G) = \mathbf{q_1}(D_G) + \langle Lap(\frac{l}{\epsilon_1})\rangle^m$
  /* Generate Private Edge Counts */
20: $\mathbf{q_2}(D_G) \leftarrow$ compute count $c_{i,j}(D_G)$ for every $i$, $j$
21: Output $\tilde{\mathbf{q}}_\mathbf{2}(D_G) = \mathbf{q_2}(D_G) + \langle Lap(\frac{l}{\epsilon_2})\rangle^{mh}$
```

D. The pseudo code is provided in Algorithm 2 from Line 8 to Line 17. Table 3 illustrates Alice's records sorted by utility score. Since "Gym" and "Mary's House" do not interest greater public, their scores are likely to be much lower than "Golden Gate Bridge" and "de Young Museum". Then the top l records on the sorted list will be put in the sampled database D_G. This step is performed on every user's data in the raw database D.

LEMMA 5. *In the domain of D_G, it holds that $GS(\mathbf{q_1}) = l$ and $GS(\mathbf{q_2}) = l$.*

After the greedy sampling step, the results to $\mathbf{q_1}$ and $\mathbf{q_2}$ will be computed on the sampled database D_G. Each individual item count and edge count will be perturbed by Laplace noise from $Lap(\frac{l}{\epsilon_1})$ and $Lap(\frac{l}{\epsilon_2})$, respectively. We will provide proof of privacy guarantee in the next section.

The advantage of HPA is that it greedily picks the most valuable data records from each user, without increasing the sample data size, i.e. l records per user. The utility of each data record is estimated privately from the overall data distribution. Records with high utility have higher chance to be picked by greedy sampling. Since the sampled data greatly depends on the relative usefulness among each user's records, it is difficult to analyze the accuracy of released counts. We will empirically evaluate the effectiveness of this approach in Section 6.

5. PRIVACY GUARANTEE

In this section, we prove that both SRA and HPA algorithms are differentially private. We begin with the following lemma, which states that record sampling on each user does not inflict differential privacy breach.

Dataset	*Gowalla*	*Foursquare*	*Netflix*	*MovieLens*
Users	12,579	45,289	480,189	6,040
Items	15,200	17,967	17,700	3,706
$\lvert D\rvert$	739,600	1,276,988	100,480,507	1,000,209
max $\lvert T_k\rvert$	14,380	1,303	17,000	2,314
avg $\lvert T_k\rvert$	58.8	60.2	209.2	165.6
min $\lvert T_k\rvert$	1	1	1	20

Table 4: Data Sets Statistics

LEMMA 6. *Let $\mathcal{A}$ be an ϵ-differentially private algorithm and $\mathcal{S}$ be a record sampling procedure which is performed on each user individually. $\mathcal{A} \circ \mathcal{S}$ is also ϵ-differentially private.*

PROOF. *See Appendix B.* □

THEOREM 3. *SRA satisfies $(\epsilon_1 + \epsilon_2)$-differential privacy.*

PROOF. *Let $S_{rand,l}$ denote the random sampling procedure in SRA. $S_{rand,l}$ is therefore a function that takes an raw database and outputs a sampled database, i.e. $S_{rand,l} : \mathcal{D} \rightarrow \mathcal{D}_\mathcal{R}$.*

According to the Laplace mechanism and Lemma 3, $\tilde{\mathbf{q}}_\mathbf{1} : \mathcal{D}_\mathcal{R} \rightarrow \mathbb{R}^m$ is ϵ_1-differentially private. By the above Lemma 6, the item counts by SRA, i.e. $\tilde{\mathbf{q}}_\mathbf{1} \circ S_{rand,l} : \mathcal{D} \rightarrow \mathbb{R}^m$ is ϵ_1-differentially private. Similarly, $\tilde{\mathbf{q}}_\mathbf{2} : \mathcal{D}_\mathcal{R} \rightarrow \mathbb{R}^{mh}$ is ϵ_2-differentially private. The edge counts by SRA, i.e. $\tilde{\mathbf{q}}_\mathbf{2} \circ S_{rand,l} : \mathcal{D} \rightarrow \mathbb{R}^{mh}$ is also ϵ_2-differentially private. Therefore, the overall SRA computation satisfies $(\epsilon_1 + \epsilon_2)$-differential privacy by Theorem 1. □

THEOREM 4. *HPA satisfies $(\epsilon_0 + \epsilon_1 + \epsilon_2)$-differential privacy.*

PROOF. *Let $S_{rand,d}$ denote the random sampling procedure in HPA for popularity estimation, i.e. $S_{rand,d} : \mathcal{D} \rightarrow \mathcal{D}_\mathcal{E}$. Let $S_{grd,l}$ denote the greedy sampling procedure, i.e. $S_{grd,l} : \mathcal{D} \rightarrow \mathcal{D}_\mathcal{G}$.*

According to the Laplace mechanism and Lemma 4, $\tilde{\mathbf{q}}_\mathbf{1} : \mathcal{D}_\mathcal{E} \rightarrow \mathbb{R}^m$ is ϵ_0-differentially private. By Lemma 6, the HPA popularity estimation step, i.e. $\tilde{\mathbf{q}}_\mathbf{1} \circ S_{rand,d} : \mathcal{D} \rightarrow \mathbb{R}^m$ is ϵ_0-differentially private. Similarly, we can prove that the HPA item counts $\tilde{\mathbf{q}}_\mathbf{1} \circ S_{grd,l} : \mathcal{D} \rightarrow \mathbb{R}^m$ is ϵ_1-differentially private, and the HPA edge counts $\tilde{\mathbf{q}}_\mathbf{2} \circ S_{grd,l} : \mathcal{D} \rightarrow \mathbb{R}^{mh}$ is ϵ_2-differentially private. Therefore, by Theorem 1, the overall HPA satisfies $(\epsilon_0 + \epsilon_1 + \epsilon_2)$-differential privacy. □

6. EXPERIMENTS

Here we present a set of empirical studies. We compare our solutions SRA and HPA with three existing approaches: 1) LPA, the baseline method that injects Laplace perturbation noise to each count, 2) DFT, the Discrete Fourier Transform based algorithm proposed in [23], applied to a vector of counts, and 3) GS, the best method with grouping and smoothing proposed in [15], applied to count histograms. Given the overall privacy budget ϵ, we set $\epsilon_{1,2} = \frac{\epsilon}{2}$ for SRA method, and $\epsilon_0 = \frac{\epsilon}{10}$ and $\epsilon_{1,2} = 0.45\epsilon$ for HPA method. Without speculating about the optimal privacy allocation, we set ϵ_0 to a small fraction of ϵ, because it is used to protect a small sample of private data for utility score estimation. To achieve the same privacy guarantee, we apply LPA, DFT, and GS to item counts and edge counts separately, with privacy budget $\frac{\epsilon}{2}$ for each application.

Data sets. We conducted our empirical studies with four real-world data sets referred to as *Gowalla*, *Foursquare*, *Netflix*, and *MovieLens*, each named after its data source. The first two data sets consist of location check-in records. *Gowalla* is collected among users based in Austin from Gowalla location-based social network by

Berjani and Strufe [3] between June and October 2010. Similarly, *Foursquare* is collected from Foursquare by Long et al. [18] between February and July 2012. In these two data sets, each record contains a user, a location, and a check-in time-stamp. Since a user can check-in at one location many times, the check-in data sets can represent a class of services which value the returning behavior, such as buying or browsing. The other two data sets consist of movie ratings, where a movie may not be rated more than once by a user. *Netflix* is the training data set for the Netflix Prize competition. *MovieLens* is collected from users of MovieLens website [2]. Each rating corresponds to a user, a movie, a rating score, and a time-stamp. Moreover, *MovieLens* also provides user demographic information, such as gender, age, occupation, and zipcode. The properties of the data sets are summarized in Table 4. Note that the minimum individual contribution in *MovieLens* is 20, as opposed to 1 for other data sets. This is because *MovieLens* was initially collected for personalized recommendation, thus users with fewer than 20 records were excluded from the published data set.

Setup. We implemented our SRA and HPA methods, as well as the baseline LPA and DFT in Java.We obtained Jave code of GS from the authors of [15]. All experiments were run on a 2.9GHz Intel Core i7 PC with 8GB RAM. Each setting was run 20 times and the average result was reported.

The default settings of parameters are summarized below: the overall privacy $\epsilon = 1$, the sampling parameter d for HPA popularity estimation $d = \min |T_k|$, the sampling parameter $l = 10$ for *Gowalla*, *Foursquare*, and *Netflix* and $l = 30$ for *MovieLens*. Our choice of parameter settings is guided by analytical results and minimal knowledge about the data sets and thus might not be optimal. For LPA and DFT, we set M to be equal to $\max |T_k|$. However, this value may not be known *a priori*. Strictly speaking, M is unbounded for check-in applications. In this sense, we *overestimate* the performance of LPA and DFT.

6.1 HPA-Private Popularity Estimation

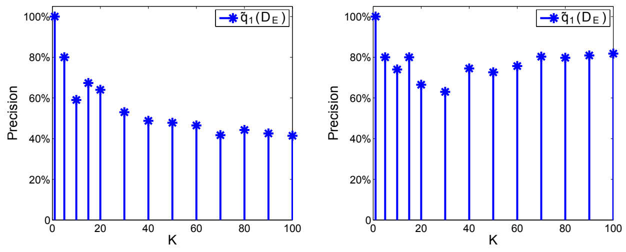
We first examine the private popularity estimation step of HPA method regarding the ability to discover top-K popular items from the noisy counts $\tilde{\mathbf{q}}_1(D_E)$. Recall that D_E is generated by randomly sampling d records per user and the output of $\mathbf{q}_1(D_E)$ is then perturbed with noise from $Lap(\frac{d}{\epsilon_0})$ to guarantee privacy. Given a small privacy budget ϵ_0, it is only meaningful to choose a small d value for accuracy, according to Theorem 2. Therefore, we set d equal to the minimum individual contribution, i.e. $\min |T_k|$, in every data set.

In this experiment, we sort all items according to $\tilde{\mathbf{q}}_1(D_E)$ output and K items with highest noisy counts are evaluated against the ground truth discovered from the raw data set. Figure 5 reports the precision results with various K values on *Foursquare* and *Netflix* data. As can be seen, from the output of $\tilde{\mathbf{q}}_1(D_E)$, we are able to discover more than 60% of top-20 popular locations in *Foursquare* and 70% top-20 popular movies on *Netflix*. When looking at $K = 100$, the output of $\tilde{\mathbf{q}}_1(D_E)$ captures 40% of the real popular locations and almost 80% popular movies. We conclude that HPA popularity estimation provides a solid step stone for subsequent greedy sampling, at very small cost of individual data as well as privacy.

6.2 Impact of Sampling Factor l

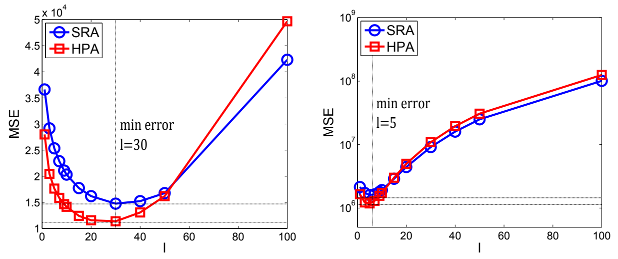
Here we look at the upper bound l of individual data contribution required by our solutions and study its impact on the accuracy of $\tilde{\mathbf{q}}_1$ and $\tilde{\mathbf{q}}_2$ output. Mean Squared Error(MSE) is adopted as the metric for accuracy and is calculated between the noisy output by our methods and the true results of $\mathbf{q}_1$ and $\mathbf{q}_2$ from the raw input data D. We ran our SRA and HPA methods varying the value of l, in order to generate sampled database D_R and D_G with different sizes. Figure 6(a) summarizes the results from *Foursquare* data for item counts, i.e. $\tilde{\mathbf{q}}_1$, and Figure 6(b) for edge counts, i.e. $\tilde{\mathbf{q}}_2$.

[2]http://movielens.org



(a) *Foursquare* (b) *Netflix*

Figure 5: Estimation of Item Popularity by HPA



(a) MSE of $\tilde{\mathbf{q}}_1$ (b) MSE of $\tilde{\mathbf{q}}_2$

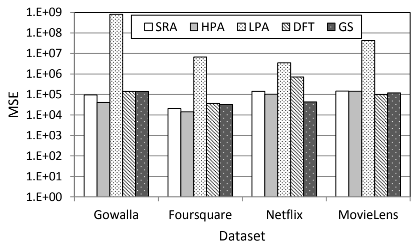
Figure 6: Impact of l with *Foursquare* Data Set

In both figures of Figure 6, when l value increases, the MSE of the noisy output by our methods first drops as sampled database gets larger. For example, we observe a decreasing trend of MSE as l is raised to 30 in Figure 6(a) and as l is raised to 5 in Figure 6(b). Beyond these two points, when further increasing l, the MSE grows due to the perturbation noise from $Lap(\frac{l}{\epsilon_1})$. Clearly, there is a trade-off between sample data size and the perturbation error. The optimal value of l depends on actual data distribution and the privacy parameter ϵ_1, according to Theorem 2. This set of results show that both SRA and HPA achieve minimum MSE with relatively small l values, i.e. $l = 30$ for $\tilde{\mathbf{q}}_1$ and $l = 5$ for $\tilde{\mathbf{q}}_2$. Our findings in Theorem 2 are confirmed and we conclude that choosing a small upper bound l on individual data contribution is beneficial especially when privacy budget is limited.

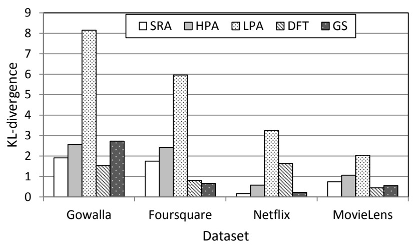
6.3 Comparison of Methods

Here we compare our SRA and HPA methods with existing approaches, i.e. LPA, DFT, and GS on all data sets. The utility of item counts and edge counts released by all private mechanisms are evaluated with three metrics. Note that for *Gowalla*, *Foursquare*, and *Netflix* data, each edge connects an item with a *day-of-week*, from "Monday" to "Sunday". For *MovieLens* data set, each edge connects a movie with a (*Gender*, *Age*) pair. The domain of *Gender* is {"M", "F"} and the domain of *Age* is {"Under 25", "25-34", "Above 34"}. Below we review the results regarding the released item counts and edge counts, for each utility metric.

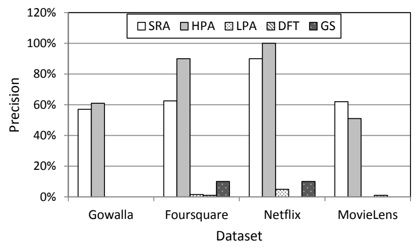
Mean Squared Error (MSE). This metric provides a generic utility comparison of different methods on the released counts. Figure 7(a) and Figure 8(a) summarize the MSE results for item counts and edge counts, respectively. As can be seen, the baseline LPA yields the highest error in both item counts and edge counts. The



(a) Mean Squared Error

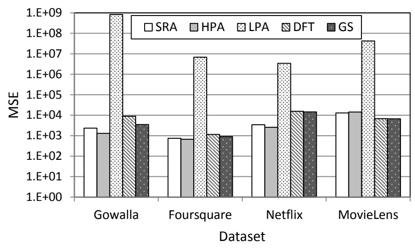


(b) KL-Divergence

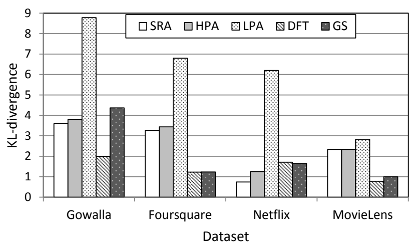


(c) Top-10 Precision

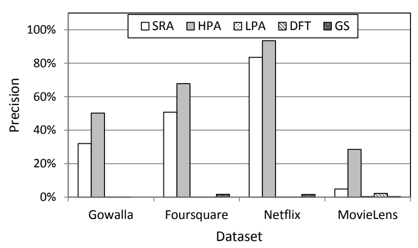
Figure 7: Utility of Released Item Counts



(a) Mean Squared Error



(b) KL-Divergence



(c) Average Top-10 Precision

Figure 8: Utility of Released Edge Counts

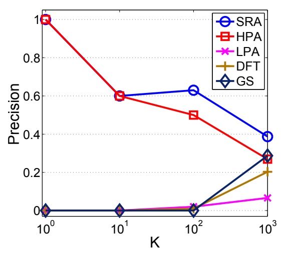
GS method, as studied in the original work [15], is no worse than DFT in every case except for *MovieLens* item counts. Our methods SRA and HPA provide the lowest MSE error except in three cases, i.e. *Netflix* item counts and *MovieLens* item/edge counts. This can be interpreted by the high average user contribution in these two data sets, where our methods inflict more data loss by limiting individual data in the sampled database.

KL-divergence. The KL-divergence is a common metric widely used to measure the distance between two probability distributions. In this set of experiments, we consider the item/edge counts as data record distributions over the domain of items/edges. Both the released counts and original counts are normalized to simulate probability distributions. Note that prior to that, zero or negative counts are replaced with 0.01 for continuity without generating many false positives. We compute the KL-divergence of the released distribution with respect to the original distribution for each query and present the results in Figure 7(b) and Figure 8(b).

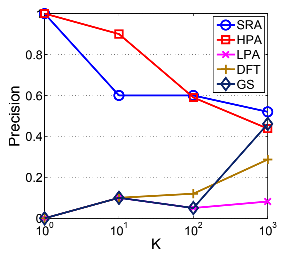
The released distributions by LPA are further from original data distributions than those of other methods for every data set. As expected, DFT and GS preserve the count distributions well in general, because: 1) the DFT method is designed to capture major trends in data series, and 2) the GS method generates smooth distributions by grouping similar columns. However, in several cases, those two methods fail to provide similar distributions, e.g. on *Gowalla* and *Netflix* data. We believe that their performance depends on the actual data distribution, i.e. whether significant trend or near-uniform grouping exists and can be well extracted. On the other hand, our solutions SRA and HPA provide comparable performance to the best existing methods, although not optimized to preserve distributional similarities. Furthermore, SRA constantly outperforms HPA in approximating the true distributions, thanks to the nature of simple random sampling technique.

Top-K Discovery. In this set of experiments, we examine the quality of top-10 discovery retrieved by all privacy-preserving mechanisms. For item counts, the top-10 popular items are evaluated. For edge counts, the top-10 popular items associated with each attribute value are evaluated and the average precision is reported, to simulate discoveries for each day-of-week and each user demographic group. In Figure 7(c), we observe that existing methods fail to preserve the 10 most popular items in any dataset. The reason is the baseline LPA suffers from high perturbation error, and DFT and GS yield over-smoothed released counts and thus cannot distinguish the most popular items from those ranked next to them. When K is large enough, we will see that their performance in top-K discovery slowly recovers in a subsequent experiment. On the other hand, our methods SRA and HPA greatly outperform existing approaches and HPA even achieves 100% precision for *Netflix* data. Similarly, our methods show superior performance in Figure 8(c), with the absolute precision slightly dropped due to sparser data distributions. Overall, HPA outperforms SRA by preserving user records with high popularity scores. The only exception where SRA is better than HPA is in finding the top-10 most popular movies on *MovieLens*. The reason is that those users who contribute less than 20 records were excluded from the data set and no movies were preferred by the majority of the rest users. As for finding top-10 movies for each demographic group, HPA greatly improves over SRA, since users within a demographic group show similar interests.

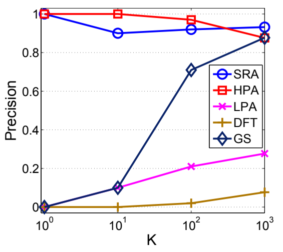
We further look at top-K precision of the released item counts by all methods, with K ranging from 1 to 1000. The results are provided in Figure 9. We can see that the performance of our greedy approach HPA is 100% when $K = 1$ and drops as K increases, since the sampling step only picks a small number of records, i.e. l records, from each user with highest utility score, i.e. item popularity. Our random approach SRA also shows decreasing precision as K increases, due to the data loss caused by sampling. However, the decreasing rate is much slower compared to that of HPA, because records of a user have equal chance to be selected by random sampling. On the contrary, LPA, DFT, and GS show 0% precision when $K = 1$ and higher precision as K increases. We conclude that SRA and HPA can discover the most popular items, superior to existing approaches up to $K = 100$, but do not distinguish less popular items due to lack of information in the sampled database.



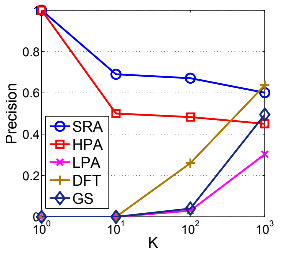
(a) *Gowalla*



(b) *Foursquare*



(c) *Netflix*



(d) *MovieLens*

Figure 9: Comparison of Methods: Top-K Mining

The existing approaches fail to distinguish the most popular items, e.g. top-10, because of perturbation or the smoothing effect of their methods, but might provide good precision for large K, e.g. $K \geq 1000$.

6.4 Additional Benefits

Data Reduction. One beneficial side effect of limiting individual data contribution is the reduction of data storage space by generating analytics from a sampled database. Figure 10 shows the number of records in the sampled databases used by SRA and HPA compared to that of the raw input. As can be seen, the sampled data is much smaller than the raw input for every data set. For *Netflix* data set, our methods perform privacy-preserving analytics and generate useful results on sample databases with less than 5% of the original data, reducing the data storage requirement without compromising the utility of output analytics.

Weekly Distribution. We also examine the sampled database by SRA and HPA by the weekly distribution of data records. The percentage of *Foursquare* check-in records on each day of week is plotted in Figure 11. As is shown, the percentage of Friday, Saturday, and Sunday check-ins is higher in the sampled databases generated by our methods than in the original data set, while the percentage of Monday-Thursday check-ins is lower than the original. Since the majority of the users are occasional users and contribute less than l records, our methods preserve their data completely in the sampled databases. We may infer that the occasional users are more likely to use the check-in service on Friday-Sunday. Moreover, the SRA sampled data is constantly closer to the original data distribution, compared to HPA. We can further infer that users are more likely to check-in popular places on Friday-Sunday.

Movie Recommendation. A example of context-aware, fine-grained recommendation is to suggest items based on the common interest demonstrated among the user group with similar demographics, such as age and gender. We illustrate the top-10 movie recommendation to male users under the age of 25 with released edge counts by our solutions on *MovieLens* data set. The first column in Table 5 shows the top-10 recommended movies using original data, while the second and third columns list movies recommended by our privacy-preserving solutions. We observe that some movies

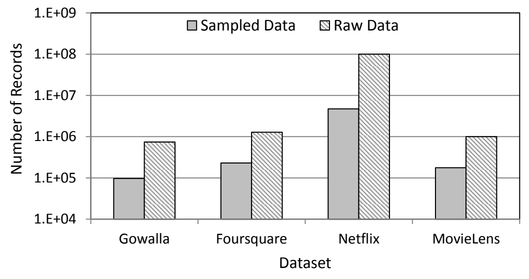


Figure 10: Data Reduction

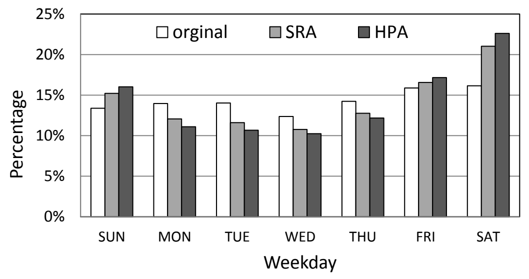


Figure 11: Weekly Distribution with *Foursquare* Data

Top 10 Movies	SRA **Output**	HPA **Output**
American Beauty	Phantasm II	American Beauty
Star Wars VI	Marvin's Room	Star Wars VI
Star Wars V	All Dogs Go to Heaven	Terminator 2
The Matrix	In the Line of Duty 2	Star Wars V
Star Wars IV	Star Wars V	Jurassic Park
Terminator 2	The Slumber Party Massacre III	The Matrix
Saving Private Ryan	The Story of Xinghua	Men in Black
Jurassic Park	American Beauty	The Fugitive
Star Wars I	Shaft	Braveheart
Braveheart	Star Wars I	Saving Private Ryan

Table 5: Movie Recommendations to Male, Under 25.

recommended by SRA may not interest the target audience, such as "Marvin's Room" and "The story of Xinghua". Furthermore, the top movie on SRA list, i.e. "Phantasm II", is a horror movie and not suitable for underage audience. On the other hand, the movies recommended by HPA are quite consistent with the original top-10 except for two movies, i.e. "Men in Black" and "The Fugitive", which may interest the target audience as well. We believe that HPA captures more information by greedy sampling and thus can make better recommendations than SRA, especially when users have very diverse interests.

7. CONCLUSION AND DISCUSSION

We have proposed a practical framework for privacy-preserving data analytics by sampling a fixed number of records from each user. We have presented two solutions, i.e. SRA and HPA, which implement the framework with different sampling techniques. Our solutions do not require the input data be preprocessed, such as removing users with large or little data. The output analysis results are highly accurate for performing top-K discovery and context-aware recommendations, closing the utility gap between no privacy and existing differentially private techniques. Our solutions benefit from sampling techniques that reduce the individual data contribution to a small constant factor, l, and thus reducing the perturbation error inflicted by differential privacy. We provided analysis results about the optimal sampling factor l with respect to the privacy requirement. We formally proved that both mechanisms satisfy ϵ-differential privacy. Empirical studies with real-world data sets confirm that our solutions enable accurate data analytics on a small fraction of the input data, reducing user privacy cost and data storage requirement without compromising utility.

Potential future work may include the design of a hybrid approach between SRA and HPA which could have the benefits of both. For real-time applications, we would like to consider how to dynamically sample user generated data, in order to further reduce the data storage requirement. Another direction is to apply the proposed sampling framework to solving more complex data analytical tasks, which might involve multiple, over-lapping count queries or other statistical queries.

8. ACKNOWLEDGMENTS

We thank the anonymous reviewers for the detailed and helpful comments to the manuscript.

9. REFERENCES

[1] M. Barbaro and T. Zeller. A face is exposed for aol searcher no. 4417749. *The New York Times*, Aug. 2006.

[2] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[3] B. Berjani and T. Strufe. A recommendation system for spots in location-based online social networks. In *Proceedings of the 4th Workshop on Social Network Systems*, SNS '11, pages 4:1–4:6, New York, NY, USA, 2011. ACM.

[4] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618, New York, 2008. ACM.

[5] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, Nov. 2011.

[6] K. Chaudhuri and N. Mishra. When random sampling preserves privacy. In *Proceedings of the 26th annual international conference on Advances in Cryptology*, CRYPTO'06, pages 198–213, Berlin, Heidelberg, 2006. Springer-Verlag.

[7] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 638–649, 2012.

[8] G. Cormode, C. Procopiuc, D. Srivastava, and T. T. L. Tran. Differentially private summaries for sparse data. In *Proceedings of the 15th International Conference on Database Theory*, ICDT '12, pages 299–311, New York, NY, USA, 2012. ACM.

[9] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the Crowd: The privacy bounds of human mobility. *Scientific Reports*, Mar.

[10] C. Dwork. Differential privacy. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *Automata, Languages and Programming*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg, 2006.

[11] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: privacy via distributed noise generation. In *Proceedings of the 24th annual international conference on The Theory and Applications of Cryptographic Techniques*, EUROCRYPT'06, pages 486–503, Berlin, Heidelberg, 2006. Springer-Verlag.

[12] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *In Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284, Heidelberg, 2006. Springer-Verlag.

[13] L. Fan and L. Xiong. An adaptive approach to real-time aggregate monitoring with differential privacy. *Knowledge and Data Engineering, IEEE Transactions on*, 26(9):2094–2106, Sept 2014.

[14] Y. Hong, J. Vaidya, H. Lu, and M. Wu. Differentially private search log sanitization with optimal output utility. In *Proceedings of the 15th International Conference on Extending Database Technology*, EDBT '12, pages 50–61, New York, NY, USA, 2012. ACM.

[15] G. Kellaris and S. Papadopoulos. Practical differential privacy via grouping and smoothing. In *Proceedings of the 39th international conference on Very Large Data Bases*, PVLDB'13, pages 301–312, 2013.

[16] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 171–180, 2009.

[17] N. Li, W. Qardaji, and D. Su. On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, pages 32–33, 2012.

[18] X. Long, L. Jin, and J. Joshi. Towards understanding traveler behavior in location-based social networks. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, 2013.

[19] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 277–286, 2008.

[20] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. volume 53, pages 89–97, 2010.

[21] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, STOC '07, pages 75–84, New York, NY, USA, 2007. ACM.

[22] D. Proserpio, S. Goldberg, and F. McSherry. Calibrating data to sensitivity in private data analysis: A platform for differentially-private analysis of weighted datasets. *Proc. VLDB Endow.*, 7(8):637–648, Apr. 2014.

[23] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 735–746, 2010.

[24] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, and Z. Hao. Low-rank mechanism: optimizing batch queries under differential privacy. *Proc. VLDB Endow.*, 5(11):1352–1363, July 2012.

[25] C. Zeng, J. F. Naughton, and J.-Y. Cai. On differentially private frequent itemset mining. *Proc. VLDB Endow.*, 6(1):25–36, Nov. 2012.

APPENDIX

A. PROOF OF THEOREM 2

PROOF. For item V_i, let c_i' denote the true count computed by $\mathbf{q_1}$ from the sample D_R. Therefore, the noisy count $\tilde{c}_i$ is derived by adding a Laplace noise to c_i' as follows:

$$\tilde{c}_i = c_i' + \nu_i \ , \tag{12}$$

$$\nu_i \sim Laplace(0, l/\epsilon_1) \ . \tag{13}$$

The MSE of $\tilde{c}_i$ can be re-written as:

$$\begin{aligned} MSE(\tilde{c}_i) &= Var(c_i' + \nu_i) + (E(c_i' + \nu_i - c_i))^2 \\ &= Var(c_i') + Var(\nu_i) + (E(c_i') - E(c_i))^2 \ . \end{aligned} \tag{14}$$

Note that c_i' and ν_i are mutually independent.

Let p_i denote the popularity of item V_i, i.e. the probability of any record having $vID = V_i$. For simplicity, we assume that users are mutually independent, records are mutually independent, and every user has M records in the raw data set D. To obtain D_R, l records out of M are randomly chosen for each user in D. Thus for any item V_i, c_i' can be represented as the sum of independent random variables:

$$c_i' = \sum_{k=1}^{n} \sum_{r \in T_k} \delta_{r,i} \tag{15}$$

$$\delta_{r,i} = \begin{cases} 1 & \text{if } r.vID = V_i \ \& \ r \in D_R \ , \\ 0 & \text{otherwise} \ . \end{cases} \tag{16}$$

The event of $\delta_{r,i} = 1$ is equivalent to the event of record r is about V_i and r is sampled in D_R by chance:

$$Pr[\delta_{r,i} = 1] = Pr[r.vID = Vi \ \& \ r \in D_R] = p_i \frac{l}{M} \ . \tag{17}$$

Therefore, we can obtain the following expectation and variance for c_i':

$$\begin{aligned} E(c_i') &= \sum_{k=1}^{n} \sum_{r \in T_k} E(\delta_{r,i}) \\ &= \sum_{k=1}^{n} \sum_{r \in T_k} p_i \frac{l}{M} \\ &= nlp_i \end{aligned} \tag{18}$$

$$\begin{aligned} Var(c_i') &= \sum_{k=1}^{n} \sum_{r \in T_k} Var(\delta_{r,i}) \\ &= \sum_{k=1}^{n} \sum_{r \in T_k} p_i \frac{l}{M}(1 - p_i \frac{l}{M}) \\ &= nlp_i(1 - p_i \frac{l}{M}) \end{aligned} \tag{19}$$

Similarly, we can obtain the expectation of c_i:

$$E(c_i) = nMp_i \ . \tag{20}$$

From the above results, we can re-write Equation 14 as follows:

$$MSE(\tilde{c}_i) = nlp_i(1 - p_i \frac{l}{M}) + 2\frac{l^2}{\epsilon_l^2} + (nlp_i - nMp_i)^2 \tag{21}$$

and we can perform the standard least square method to minimize the MSE. The optimal l value is thus:

$$l = \frac{2n^2 p_i^2 M - np_i}{4/\epsilon_1^2 - 2np_i^2/M + 2n^2 p_i^2} \tag{22}$$

We conclude that the optimal l value is a monotonically increasing function of ϵ_1^2. $\square$

B. PROOF OF LEMMA 6

PROOF. By definition of differential privacy, we are to prove that for any neighboring raw databases D_1 and D_2, $\mathcal{A} \circ \mathcal{S}$ satisfies the following inequality for $\widetilde{D} \in Range(\mathcal{A} \circ \mathcal{S})$:

$$Pr[\mathcal{A} \circ \mathcal{S}(D_1) = \widetilde{D}] \leq e^{\epsilon} Pr[\mathcal{A} \circ \mathcal{S}(D_2) = \widetilde{D}] \ . \tag{23}$$

Without loss of generality, we assume D_2 contains one more user than D_1. Let u denote the user that is contained in D_2 but not D_1 and T be user u's set of records in D_2. By definition of neighboring databases, we can rewrite $D_2 = D_1 \oplus T$ [3].

Let $\hat{D}_1$ denote any possible sampling output of $\mathcal{S}(D_1)$. We have:

$$\begin{aligned} & Pr[\mathcal{A} \circ \mathcal{S}(D_1) = \widetilde{D}] \\ &= \sum_{\hat{D}_1} Pr[\mathcal{A} \circ \mathcal{S}(D_1) = \widetilde{D} | \mathcal{S}(D_1) = \hat{D}_1] Pr[\mathcal{S}(D_1) = \hat{D}_1] \\ &= \sum_{\hat{D}_1} Pr[\mathcal{A}(\hat{D}_1) = \widetilde{D}] Pr[\mathcal{S}(D_1) = \hat{D}_1] \end{aligned} \tag{24}$$

Let $\hat{T}$ denote any possible sampling output of $\mathcal{S}(T)$. We note that $\hat{T}$ can take values from the entire domain, in general:

$$\sum_{\hat{T}} Pr[\mathcal{S}(T) = \hat{T}] = 1 \ . \tag{25}$$

Since $\mathcal{S}$ is performed independently on each user, we can derive:

$$\begin{aligned} & Pr[\mathcal{S}(D_1) = \hat{D}_1] \\ &= \sum_{\hat{T}} Pr[\mathcal{S}(D_1) = \hat{D}_1] Pr[\mathcal{S}(T) = \hat{T}] \\ &= \sum_{\hat{T}} Pr[\mathcal{S}(D_1 \oplus T) = \hat{D}_1 \oplus \hat{T}] \ . \end{aligned} \tag{26}$$

Note that since D_1 and T are disjoint, the sampling output on D_1 and T are also independent and disjoint. Therefore,

$$\begin{aligned} & Pr[\mathcal{A} \circ \mathcal{S}(D_1) = \widetilde{D}] \\ &= \sum_{\hat{D}_1} Pr[\mathcal{A}(\hat{D}_1) = \widetilde{D}] \sum_{\hat{T}} Pr[\mathcal{S}(D_1 \oplus T) = \hat{D}_1 \oplus \hat{T}] \\ &= \sum_{\hat{D}_1, \hat{T}} Pr[\mathcal{A}(\hat{D}_1) = \widetilde{D}] Pr[\mathcal{S}(D_1 \oplus T) = \hat{D}_1 \oplus \hat{T}] \\ &\leq \sum_{\hat{D}_1, \hat{T}} e^{\epsilon} Pr[\mathcal{A}(\hat{D}_1 \oplus \hat{T}) = \widetilde{D}] Pr[\mathcal{S}(D_1 \oplus T) = \hat{D}_1 \oplus \hat{T}] \end{aligned} \tag{27}$$

$$= e^{\epsilon} \sum_{\hat{D}_2} Pr[\mathcal{A}(\hat{D}_2) = \widetilde{D}] Pr[\mathcal{S}(D_2) = \hat{D}_2] \tag{28}$$

$$= e^{\epsilon} Pr[\mathcal{A} \circ \mathcal{S}(D_2) = \widetilde{D}] \ . \tag{29}$$

Line 27 is due to the fact that $\mathcal{A}$ is ϵ-differentially private and $\hat{D}_1$ and $\hat{D}_1 \oplus \hat{T}$ are neighboring databases. In line 28 we change notation and let $\hat{D}_2$ represent $\hat{D}_1 \oplus \hat{T}$. The proof is hence complete. $\square$

[3] $\oplus$ is used to denote a co-product, or disjoint union of two databases.