SCIENCE CHINA
Information Sciences

• **Supplementary File** •

APRS: a privacy-preserving location-aware recommender system based on differentially private histogram

Sheng GAO[1*], Xindi MA[2, 3], Jianming ZHU[1] & Jianfeng MA[2, 3]

[1]*School of Information, Central University of Finance and Economics, Beijing* 100081*, China;*
[2]*School of Computer Science and Technology, Xidian University, Xi'an* 710071*, China;*
[3]*School of Cyber Engineering, Xidian University, Xi'an* 710071*, China*

Appendix A Preliminaries

In this section, we present some preliminaries that server as the basis of APRS. For convenience, the notations used in this paper are listed in Tabel A1.

Table A1 Definitions and notations in APRS

Symbol	Definition
I	the public set of items
C	the public category set
$\mathbf{C}$	the public item-category correlation matrix
rad_I	the radius of AOI
rad_R	the radius of AOR
x_u	user u_q's current location
H_r	user u_q's aggregated histogram of raw history data
$\widehat{H_r}$	user u_q's perturbed aggregated histogram before clustering
$\widetilde{H}$	user u_q's perturbed aggregated histogram

Appendix A.1 Histogram

Let I be the public set of items which can be reviewed, $|I| = m$, and C be the public category set, $|C| = n$. In LBS recommendation, each location item is associated with a subset of categories, which is represented by a public correlation matrix $\mathbf{C}$ of size $m \times n$. If the entry $c_{i,j}$ in $\mathbf{C}$ is 1, we consider the item i is associated with category j. For each category $j \in C$, its count is the number of item $i \in I$ with $c_{ij} = 1$, $i \in [1, m], j \in [1, n]$. The histogram H over the categories C consists of a set of bins: $H = \{H_1, H_2, \ldots, H_n\}$, where each bin H_j is of a count of its corresponding category $j \in C$.

Appendix A.2 Differential Privacy

The privacy-preserving mechanism in our APRS is on the basis of differential privacy [1], which can not only provide strong privacy protection but also resist any background knowledge attack from adversaries. Informally, an algorithm $\mathcal{A}$ is differentially private if the output is indistinguishable to any particular record in the dataset.

* Corresponding author (email: sgao@cufe.edu.cn)

Definition 1 (ε-Differential Privacy [1]). Let $\varepsilon > 0$ be the privacy budget. A randomized algorithm $\mathcal{A}$ is ε-differentially private if for all data sets D_1 and D_2 differing on at most one element, i.e., $d(D_1, D_2) = 1$, and all $\mathcal{S} \in Range(\mathcal{A})$,

$$Pr[\mathcal{A}(D_1) \in \mathcal{S}] \leqslant exp(\varepsilon) Pr[\mathcal{A}(D_2) \in \mathcal{S}]$$

Privacy budget $\varepsilon > 0$ is a small constant, which specifies the desired privacy level. The smaller of ε, the stronger of privacy preservation, leading to more limit on the influence of items. Typically, ε is small (e.g., $\varepsilon \leqslant 1$).

To achieve the differential privacy, there are two well-established techniques: the Laplace mechanism [2] and the exponential mechanism [3], which are both based on the concept of global sensitivity [2] to compute over a dataset.

Definition 2 (Global Sensitivity [2]). The global sensitivity of a function $f : \mathcal{D} \to \mathbb{R}^n$, denoted by $S(f)$ is defined as the largest ℓ_1-norm difference $\| f(D_1) - f(D_2) \|_1$, where D_1 and D_2 are neighboring datasets that differ at one element. More formally,

$$S(f) = \max_{D_1, D_2 : \|D_1 - D_2\| = 1} \|f(D_1) - f(D_2)\|_1$$

We mainly use the Laplace mechanism in our solution. Intuitively, we should generate some properly Laplace noise to mask the influence of items.

Definition 3 (Laplace Mechanism [2]). For function $f : \mathcal{D} \to \mathbb{R}^n$, the Laplace mechanism is achieved by computing $f(D) + \eta$, where $\eta \in \mathbb{R}^n$ is a random vector of independent variables and η_i is generated from the Laplace distribution with parameter $S(f)/\varepsilon$. That is:

$$Pr[\eta_i = z] \propto exp(-z \cdot \varepsilon / S(f))$$

In this paper, we mainly achieve the differentially private histogram. Given two neighboring histograms H_1 and H_2, we represent the global sensitivity of function f as $S(f) = \max_{H_1, H_2} \|f(H_1) - f(H_2)\|_1$. So given a histogram $H \to \mathbb{R}^n$, a function f, and privacy parameter ε, the mechanism $\mathcal{A}$ can achieve ε-differential privacy in the following way:

$$\mathcal{A}(H) = f(H) + < \mathrm{Lap}_1(\frac{S(f)}{\varepsilon}), \ldots, \mathrm{Lap}_n(\frac{S(f)}{\varepsilon}) >$$

Appendix A.3 Geo-Indistinguishability

The location privacy definition used in our APRS is based on a generalized variant of differential privacy that can be defined on an arbitrary set of secrets χ, equipped with a metric d_χ [4]. The distinguishability level between the secrets x and x' can be expressed by the distance $d_\chi(x, x')$. Let $\mathcal{Z}$ be the obfuscated values reported to recommender server and let $\mathcal{P}(\mathcal{Z})$ denote the probability measures over $\mathcal{Z}$. The multiplicative distance $d_\mathcal{P}$ on $\mathcal{P}(\mathcal{Z})$ is defined as:

$$d_\mathcal{P}(\mu_1, \mu_2) = sup_{Z \in \mathcal{Z}} \left| \ln \frac{\mu_1(Z)}{\mu_2(Z)} \right|$$

where $\mu_1(Z)$ and $\mu_2(Z)$ are the posterior probabilities that the reported locations belong to the set $Z \in \mathcal{Z}$ when users' locations are x and x'. From the above equation, we can conclude that $d_\mathcal{P}(\mu_1, \mu_2)$ is small when μ_1, μ_2 assign similar probabilities to each reported value.

The perturbed mechanism can be designed as a probabilistic function $\mathcal{A} : \chi \to \mathcal{P}(\mathcal{Z})$, which generates a probability distribution $\mathcal{A}(x)$ for each secret x over the reported values $\mathcal{Z}$. The generalized variant of differential privacy, called d_χ-privacy, can be defined as follows [5]:

Definition 4. (d_χ-privacy). A mechanism $\mathcal{A} : \chi \to \mathcal{P}(\mathcal{Z})$ satisfies d_χ-privacy if:

$$d_\mathcal{P}(\mathcal{A}(x), \mathcal{A}(x')) \leqslant d_\mathcal{X}(x, x'), \forall x, x' \in \mathcal{X}$$

or equivalently $\mathcal{A}(x)(Z) \leqslant exp(d_\mathcal{X}(x, x'))\mathcal{A}(x')(Z)$, $\forall x, x' \in \mathcal{X}$, $Z \subseteq \mathcal{Z}$.

Given different d_χ, we can obtain different privacy notions. Generally, we can also scale this metric by the privacy budget ϵ, represented as ϵd_χ . For the location privacy involved in this paper, the secrets χ and reported values $\mathcal{Z}$ are all the sets of locations, while $\mathcal{A}$ is an obfuscation mechanism. So we can obtain ϵd_2-privacy when using the Euclidean metric d_2, a natural notion of location privacy called geo-indistinguishability in [5]. If for any radius d_2, a mechanism makes the user enjoy ϵd_2-privacy within d_2, we claim that the mechanism satisfies ϵ-geo-indistinguishability.

Appendix A.4 Accuracy and Utility Metrics

Similar to [5], if the probability of the area of interest(AOI) to be fully contained in the area of retrieval(AOR) is bounded from below by a confidence factor c, we also consider that is (c, rad_I)-accurate, where rad_I is the radius of AOI. Given a privacy parameter ε and accuracy parameter (c, rad_I), our goal is to obtain an LBS recommendation $(\mathcal{A}, rad_R)$ satisfying both ε-geo-indistinguishability and (c, rad_I)-accuracy.

In this manner, we will introduce the notion of (α, δ)-usefulness to measure the utility of location privacy preservation in our APRS, which was introduced in [6]. If a location perturbing mechanism $\mathcal{A}$ is (α, δ)-useful for each location $x \in \chi$ and sanitized location $z = \mathcal{A}(x)$, then $Pr[d(x, z) \leqslant \alpha] \geqslant \delta$.

Additionally, we use the similar utility in [7] to measure user's perturbed aggregated history data. To be more specific, the expected Mean Absolute Error (MAE) are used to measure the deviation between user's raw and perturbed category aggregates, which is formally defined as following [7].

$$MAE(H, \widetilde{H}) = E[\frac{1}{n} \sum\nolimits_{j=1}^{n} |H_j - \widetilde{H_j}|]$$

where H and $\widetilde{H}$ are user's raw and perturbed category aggregates, respectively.

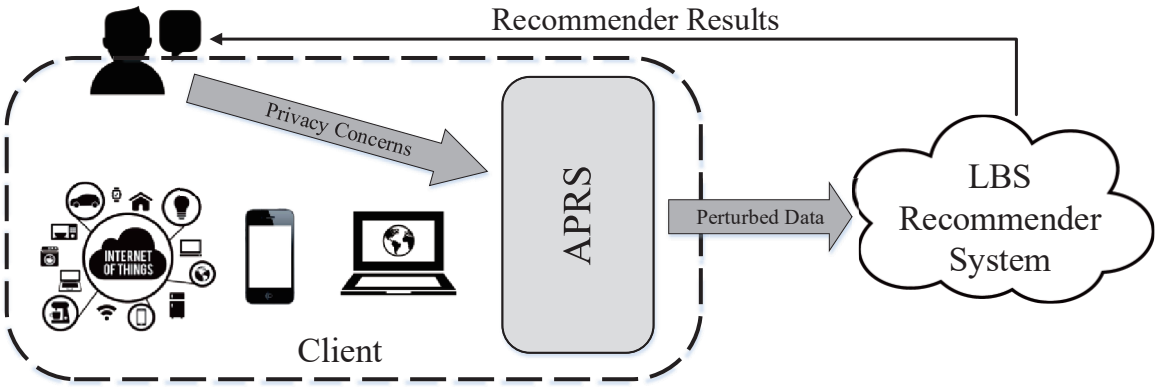


Figure A1 The system model of APRS

Appendix A.5 System Model

Our APRS is designed to protect user's location and history data privacy without changing the existed recommender system. Before describing APRS, we formally present the definition of a general location-aware recommender system as follows.

Definition 5. Given the location x_u of user u_q and the requested area radius r_I of POIs, a *location-aware recommender system* returns recommended items located in u_q's interesting area defined by $AOI(x_u, r_I)$, by taking into account u_q's access history d_u and a public item-category correlation matrix $\mathbf{C}$.

To protect the privacy of such a recommendation under an untrusted recommender environment, the privacy information such as x_u and d_u should be kept in secret. Before describing the details, we derive the basic components in our APRS as follows (see Fig.A1):

• **User Client:** User client collects user private data, including location and history footprint, from a variety of user's devices (e.g., smartphone, pad, laptop, etc.). As a module, APRS resides on user's hub device and perturbs the private data based on the perturbed mechanisms. At the beginning of the recommendation, user will first perturb the private location and history footprint and send the perturbed data to the untrusted recommender server.

• **LBS Recommender Server:** LBS recommender server owns all the information of public items and their corresponding categories. In this paper, we do not make any changes to the recommender server to ensure the practical of APRS. When user requests the recommender service, the LBS recommender server can process the requests and send the recommender results to user using the existed recommender algorithms.

Notably, during the procedure, the recommender user always does not disclosure his secret current location and history footprint to the recommender server, and only the perturbed location and history data will be sent. So the privacy of user is not leaked when he uses the recommendation service.

Appendix A.6 Threat Model

In APRS, malicious attackers may exist around the users and steal information during recommendation. Firstly, we consider the LBS recommender server to be curious-but-honest. During the recommendation process, the recommender server may be curious about user's privacy information, such as current location and history footprints. So the recommender server may strictly follow the recommender algorithms but also violate and disclose user's sensitive and private information. Secondly, external adversaries are also interested in the information delivered by the user. Specially, when user data is sent from device to the recommender server, attackers can eavesdrop the transmission channel, therefore user private data may need to be perturbed during the transmission.

Appendix A.7 Design Goals

To be a privacy-preserving location-aware recommender system, APRS should satisfy requirements as follows.

• **Quality of Service(QoS).** One of APRS goals is to hide the privacy and disclose enough useful information for the recommender service. So the proposed mechanism should guarantee the utility of the perturbed data and minimize the effect of sanitized noise on recommender results.

• **Privacy Preservation.** Another goal of our APRS is to achieve the privacy preservation, which is embodied in the following aspects.

1. *Location privacy.* We treat user current location as private data since it reveals user's trace trajectories and may potentially suffer physical threats. It requires that the location privacy cannot be revealed to other entities during the recommendation.

2. *History data privacy.* Since the history data represent user's profiles and preferences, directly revealing one's history data would leak his profile privacy. It requires that user's history footprint cannot be revealed during the recommender process.

Appendix B A greedy clustering algorithm

We introduce a greedy clustering algorithm to get the optimal cluster set. During the clustering process, we iteratively judge whether to put the next bin $\widehat{H_{rj}}$ into current cluster S_i. If adding $\widehat{H_{rj}}$ into S_i results in a lower error, we will merge $\widehat{H_{rj}}$ into S_i; otherwise, a new cluster will be created.

When $\widehat{H_{rj}}$ is added into S_i, the resulting error is the error of the new cluster $S_i \cup \widehat{H_{rj}}$ and we have the error as following:

$$err(S_i \cup \widehat{H_{rj}}) = \sum\nolimits_{\widehat{H_{rk}} \in S_i \cup \widehat{H_{rj}}} (\widehat{H_{rk}} - \overline{S_i \cup \widehat{H_{rj}}})^2 + \frac{2}{(|S_i|+1)(\varepsilon_{2b})^2}$$

When $\widehat{H_{rj}}$ is not added into S_i, the error is the sum of $err(S_i)$ and $err(\widehat{H_{rj}})$. It is easy to obtain $err(S_i)$ from the above equation. However, we need more efforts to computer the error of $\widehat{H_{rj}}$.

While $\widehat{H_{rj}}$ is not added into S_i, there are $n-j+1$ possible clusters in which $\widehat{H_{rj}}$ may put. So $err(\widehat{H_{rj}})$ is a variable which is dependent on the actual cluster containing $\widehat{H_{rj}}$ and is not known now. However, if we obtain the minimum of $err(\widehat{H_{rj}})$, denoted by $err^*(\widehat{H_{rj}})$, we can also decide whether to add $\widehat{H_{rj}}$ into S_i. If $\widehat{H_{rj}}$ locates in potential cluster $S_{j,l}$ which contains $\{\widehat{H_{rj}}, \widehat{H_{rj+1}}, \ldots, \widehat{H_{rl}}\}$, $j \leqslant l \leqslant n$, we can obtain the error for $\widehat{H_{rj}}$ as $err^*(\widehat{H_{rj}}) = min_l\{(\widehat{H_{rj}} - \overline{S_{j,l}})^2 + \frac{2}{|S_{j,l}|^2(\varepsilon_{2b})^2}\}$.

In the following, we create a prefix sum array $\mathbf{B}$ to efficiently calculate $\overline{S_{j,l}}$, where $\mathbf{B} \in \mathbb{R}^n$ and $\mathbf{B}_i = \sum_{k=1}^{i} \widehat{H_{rk}}$. Then we can obtain $\overline{S_{j,l}} = \frac{\mathbf{B}_l - \mathbf{B}_{j-1}}{l-j+1}$. After that, we can use the monotonicity of $(\widehat{H_{rj}} - \overline{S_{j,l}})^2$ and $\frac{2}{|S_{j,l}|^2(\varepsilon_{2b})^2}$ to calculate $err^*(\widehat{H_{rj}})$: with the growth of l, $(\widehat{H_{rj}} - \overline{S_{j,l}})^2$ monotonically increase, while $\frac{2}{|S_{j,l}|^2(\varepsilon_{2b})^2}$ monotonically decreases.

Considering $S_{j,l}, j \leqslant l \leqslant n$, with the ascending order of his sizes, if $(\widehat{H_{rj}} - \overline{S_{j,l+1}})^2 - (\widehat{H_{rj}} - \overline{S_{j,l}})^2 \geqslant \frac{2}{|S_{j,l}|^2(\varepsilon_{2b})^2} - \frac{2}{(n-j+1)^2(\varepsilon_{2b})^2}$, we obtain that $err^*(\widehat{H_{rj}}) = (\widehat{H_{rj}} - \overline{S_{j,l}})^2 + \frac{2}{|S_{j,l}|^2(\varepsilon_{2b})^2}$. The reason for our conclusion is that if the increase of $(\widehat{H_{rj}} - \overline{S_{j,l}})^2$ between $S_{j,l+1}$ and $S_{j,l}$ exceeds the maximum decrease of $\frac{2}{|S_{j,l}|^2(\varepsilon_{2b})^2}$, the minimum point has been missed.

For better illustration of our proposed greedy clustering algorithm, we show an example as follows.

Example 1. Let $\widehat{H_r} = \{1, 1, 4, 4, 5, 13\}$ be the sorted noisy histogram. Given the current cluster $S_1 = \{1, 1\}$ and $\varepsilon_{2b} = 0.4$, we now judge whether to put $\widehat{H_{r3}} = 4$ into S_1. Firstly, we can computer $err(S_1 \cup \widehat{H_{r3}}) = (1 + 1 + 4) + 25/6 = 61/6$ and $err(S_1) = 25/4$. Because of $err(S_1 \cup \widehat{H_{r3}}) > err(S_1)$, so the sanitized $\widehat{H_{r3}}$ cannot be put into S_1. Then we create the second cluster $S_2 = \{4, 4\}$ and consider whether to add $\widehat{H_{r5}}$ to S_2. To calculate $err^*(\widehat{H_{r5}})$, we consider 2 potential clusters, and obtain $err^*(\widehat{H_{r5}}) = 0 + 25/2 = 25/2$ when the potential cluster that contains $\widehat{H_{r5}}$ is $\{5\}$. Since $err(S_2) = 25/4$ and $err(S_2 \cup \widehat{H_{r5}}) = 29/6$, so $err(S_2 \cup \widehat{H_{r5}}) < err(S_2) + err^*(\widehat{H_{r5}})$ and we set the cluster $S_2 = \{4, 4, 5\}$. Finally, we set $S_3 = \{13\}$.

After the process described above, the users can get the sanitized location and history data. Then the users will send the perturbed information to recommender server and request the recommender service.

Appendix C Privacy and Utility Analysis

In this section, we theoretically analyze the privacy preservation and utility which APRS satisfies.

Appendix C.1 Privacy of APRS

We now proceed to show that our APRS described above is ε-differential privacy. The privacy of APRS depends on that of its components. In the following, we will show that the location perturbation and history data perturbation are ε_1-differential privacy and ε_2-differential privacy, respectively.

Theorem 1. Based on location and history data perturbation, APRS satisfies ε-differential privacy.

Proof. To prove the privacy preservation of APRS, we just to prove that the location and history data perturbation are both differential privacy for the corresponding privacy budget. As it is proved in [5], the location perturbing mechanism satisfies ε_1-geo-indistinguishability. However, the geo-indistinguishability is also a generalized variant of differential privacy, which is described in Section Appendix A.3. So we realize that the location perturbation satisfies ε_1-differential privacy.

While perturbing the history data, we added the Laplacian noise twice to the aggregated histogram. So we split the history data perturbation into two phases: perturbing raw histogram before sorting (Phase A) and perturbing sorted histogram after clustering (Phase B). Next, we will first prove that Phase A satisfies ε_{2a}-differential privacy. Let D_1, D_2 be neighboring datasets (i.e., $||D_1 - D_2|| = 1$), $\mathcal{A}_1$ be the sanitized mechanism in Phase A, and $f(D_i)$ be the category aggregates of user's private history data D_i. For any $r = \{r_1, \ldots, r_n\} \in \mathit{Range}(\mathcal{A}_1)$, we have analyzed as follows.

$$\frac{\Pr[\mathcal{A}_1(D_1) = r]}{\Pr[\mathcal{A}_1(D_2) = r]} = \prod_{j \in S} \frac{\Pr[\mathcal{A}_1(D_1)(j) = r(j)]}{\Pr[\mathcal{A}_1(D_2)(j) = r(j)]} \geqslant \exp(-\sum_{j \in S} \varepsilon_j |f_j(D_1) - f_j(D_2)|)$$

$$\geqslant exp(-\max_{||D_1-D_2||=1}\sum_{j\in S}\varepsilon_j|f_j(D_1)-f_j(D_2)|) \geqslant exp(-\varepsilon_{2a})$$

The first step is established because of the noises is injected independently on each aggregated histogram bin; the second step is obtained from the introduced Laplace noises and triangle inequality.

Similarly, we can also prove that Phase B satisfies ε_{2b}-differential privacy. Using the sequential composition property [8] stated below, we learn that Phase A and Phase B together satisfy $(\varepsilon_{2a}+\varepsilon_{2b})$-differential privacy. And all the process of APRS satisfies $(\varepsilon_1+\varepsilon_{2a}+\varepsilon_{2b})$-differential privacy, that is ε-differential privacy. ■

Appendix C.2 Utility analysis

Then, we analyze the utility provided by APRS. In APRS, we will show the utilities which are achieved by location perturbation and history data perturbation, respectively.

As indicated in Section Appendix A.4, we introduce the notion of (α,δ)-usefulness to describe the utility of location perturbation mechanism. Therefore, given a confidence factor δ, we can get the utility of location perturbation in APRS is $(C_{\varepsilon_1}^{-1}(\delta),\delta)$-useful.

Let H_{rj} be the j-th bin in the raw histogram, S_i be the i-th clustering after perturbation, so we can obtain the following equation as the utility for history data perturbation:

$$\begin{aligned}
MAE(H_r,\widetilde{H}) &= E[\frac{1}{n}\sum_{j=1}^{n}|H_{rj}-\widetilde{H_j}|] = E[\frac{1}{n}\sum_{S_i\in S}\sum_{j=1}^{n}|H_{rj}-\frac{\sum_{\widehat{H_{rj}}\in S_i}\widehat{H_{rj}}}{|S_i|}-\frac{Lap(1/\varepsilon_{2b})}{|S_i|}|] \\
&\leqslant E[\frac{1}{n}\sum_{S_i\in S}\sum_{j=1}^{n}|H_{rj}-\frac{\sum_{\widehat{H_{rj}}\in S_i}\widehat{H_{rj}}}{|S_i|}|]+E[\frac{1}{n}\sum_{S_i\in S}\frac{Lap(1/\varepsilon_{2b})}{|S_i|}] \\
&= E[\frac{1}{n}\sum_{S_i\in S}\sum_{j=1}^{n}|H_{rj}-\frac{\sum_{\widehat{H_{rj}}\in S_i}(H_{rj}+Lap(1/\varepsilon_{2a}))}{|S_i|}|] \\
&\leqslant E[\frac{1}{n}\sum_{S_i\in S}\sum_{j=1}^{n}|H_{rj}-\frac{\sum_{\widehat{H_{rj}}\in S_i}H_{rj}}{|S_i|}|]+E[\frac{1}{n}\sum_{S_i\in S}\sum_{j=1}^{n}\frac{\sum_{\widehat{H_{rj}}\in S_i}Lap(1/\varepsilon_{2a})}{|S_i|}] \\
&= E[\frac{1}{n}\sum_{S_i\in S}\sum_{j=1}^{n}|H_{rj}-\frac{\sum_{\widehat{H_{rj}}\in S_i}H_{rj}}{|S_i|}|]
\end{aligned}$$

So the utility of history data perturbation is obtained as above.

Appendix D Experimental Evaluation

In this section, we depict a series of experimental results of APRS conducted over a real-world dataset, which indicate that APRS can effectively and efficiently fulfill the design goals described in Section Appendix A.7. APRS was implemented using Java and the experiments were conducted on a machine with a 3.2 GHz CPU and 8GB of RAM. To simulate the existed LBS recommender system, we conduct the classic recommendation algorithm, collaborative filtering [9], using GraphLab[1)]. Then we run each experiment 10 times and report the average result. While evaluating the recommendation accuracy, we use stochastic gradient decent algorithm (SGD) for collaborative filtering.

Dataset. We adopted a real business rating data provided by RecSys Challenge 2013[2)], in which Yelp reviews, business items, users, and checkins are collected at Phoenix, AZ metropolitan area. The number of item categories is 22 and we use all the reviews in training dataset, which owns 229,907 reviews from 43,873 users on 11,537 items.

Appendix D.1 Location Perturbation

While perturbing user's location, we first define a radius r which we wish to be protected, that we assume is 500 meters, and then set the privacy budget as ε_1. This means that taken two points on the radius of 500 meters, their probability of being the observable of the same secret differ at most by $exp(\varepsilon_1)$, which is the definition of geo-indistinguishability, and even less the more we take them closer to the secret. When requesting the recommender service, we choose a large confidence factor for utility, that is 0.9.

The evaluation results is shown in Fig. D1. During the simulation, we measure the radius of retrieval area (radR) and the number of businesses located in AOR varying with the privacy budget ε_1. As shown in Fig. D1, the radius of retrieval area is decreasing quickly when ε_1 varying from 0.1 to 0.5. And then the radius will fall more gentle. The curve proves that a less ε_1 will bring more noisy for the secret location and thus affect the system availability. That result can also be proved by the trend of the number of businesses located in AOR. When the radius is large, there are a lot of businesses located in AOR. So the recommender system will consume more resources to retrieve the recommender results. Varying with ε_1, our APRS will generate a smaller radR to request the service and the recommender server will compute the results from less

1) http://select.cs.cmu.edu/code/graphlab/pmf.html
2) https://www.kaggle.com/c/yelp-recsys-2013

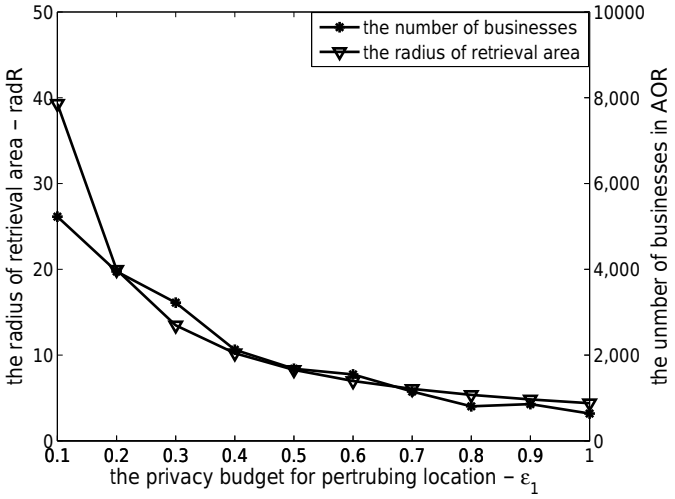


Figure D1 The evaluation results of location perturbation

businesses with less resources. However, another challenge is that a large ε_1 will also increase the risk of location privacy disclosure. So according the simulation results in Fig. D1, we can set $\varepsilon_1 = 0.5$ as the optimal privacy budget for location perturbation.

Appendix D.2 History Data Perturbation

While evaluating the history data perturbation in APRS, we mainly focus on analyzing the perturbation quality and the efficiency of recommendation of APRS. To measure the quality, we will send the perturbed history data to GraphLab and analyze the recommender results in following two aspects:

• *Perturbed Category Aggregates Quality:* We use the expected MAE metric discussed in Section Appendix A.4 to measure the perturbed category aggregates quality.

• *Recommendation Accuracy:* The MAE Loss between the recommender results using raw and perturbed data is considered, which is defined as following:

$$\Gamma = \frac{\sum_{u=1}^{U}\sum_{i=1}^{m}|Rec_p^{ui} - GT^{ui}|}{\sum_{u=1}^{U}\sum_{i=1}^{m}|Rec_r^{ui} - GT^{ui}|} - 1$$

where U is the number of users, m is the number of items, Rec_r^{ui}, Rec_p^{ui} are the predicted recommender results by user u for item i using user raw data and obfuscated data, and GT^{ui} is the raw rating information which is identified as the ground-truth from user u to item i.

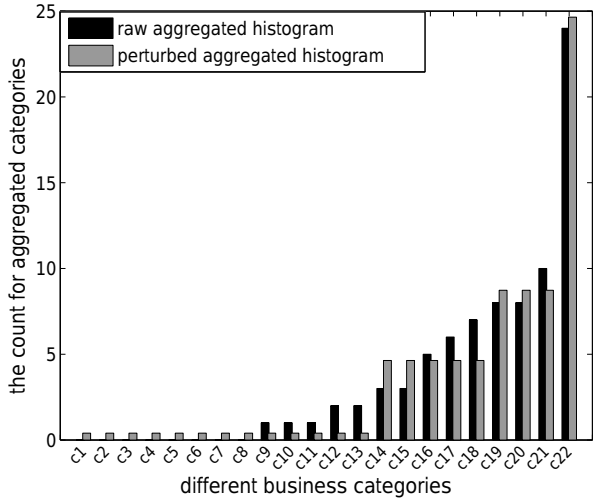
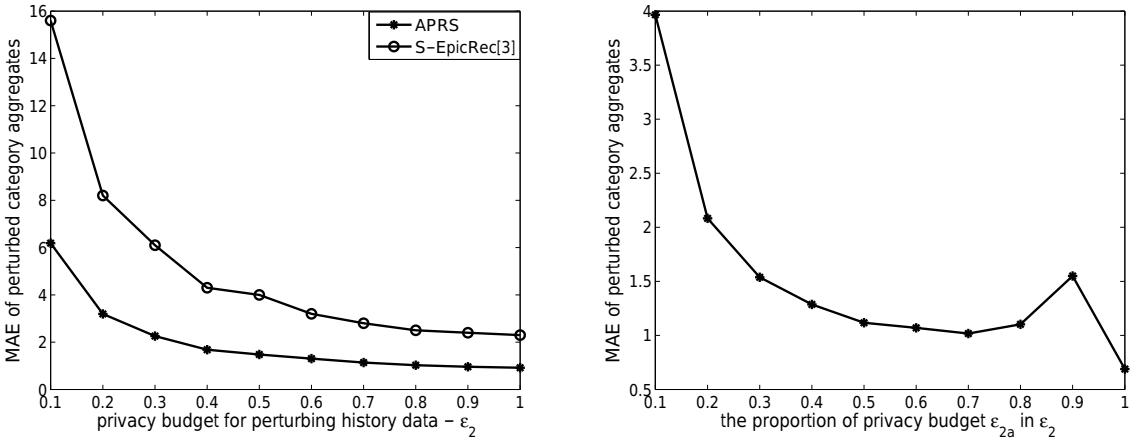


Figure D2 The perturbed aggregated category histogram

First of all, we carry out the analysis on a perturbed aggregated category histogram compared with the corresponding raw one. As shown in Fig. D2, our APRS divides all the categories into four groups and each category in same group has the same aggregated count. If the user disclosures his raw history data, the attacker can not only get the user's trajectory privacy, but also analyze user's private preference information. However, after perturbation, the attacker cannot tell whether the user prefers these categories. Hence, although the user sends his history data which is perturbed by APRS to the recommender server, it also cannot get user's preference after aggregating the perturbed data.

In the following, we will analyze the perturbed category aggregates quality by varying the privacy budget for perturbing the history data. We compare our APRS with the S-EpicRec scheme [10] when the total privacy budget ε_2 is varying. And then, given a certain ε_2, we analyze the MAE of perturbed category aggregates by varying the proportion of ε_{2a} in ε_2. As shown in Fig. D3(a), we plot the MAE by varying ε_2. Comparing with S-EpicRec scheme, we find that our APRS owns a lower MAE while using the same privacy budget. The reason is that our APRS introduce a greedy clustering strategy and it confirms that our clustering strategy indeed introduce less noise for the aggregated categories. Since S-EpicRec just adds the Laplace noise for each category, it will generate m noises for the aggregated categories and result in a higher MAE. Additionally, the simulation results also show that the MAE of perturbed category aggregates reduce dramatically at the beginning and then maintains relatively stable. So we can find that $\varepsilon_2 = 0.4$ is the optimal privacy budget for history data perturbation. Even if we allocate more privacy budget to the perturbation, but that will not bring a lower MAE and may lead to the disclosure of privacy.

Then, we also plot the MAE by varying the proportion of privacy budget ε_{2a} in ε_2 in Fig. D3(b). Given a certain ε_2, we find that the MAE decreases dramatically before ε_{2a} reaches to 0.7. And then the MAE will increase reversely when $\varepsilon_{2a} = 0.8$ and 0.9. The reason is that the total noise comes from two parts: $Lap(\varepsilon_{2a})$ and $Lap(\varepsilon_{2b})$, $\varepsilon_{2b} = \varepsilon_2 - \varepsilon_{2a}$. While we allocate less budget for ε_{2a}, more noise will be added to the raw categories. However, with the varying of ε_{2a}, the noise added to raw data will decrease and the reduced noise exceeds the increased noise for perturbing the sorted histogram. So the MAE will clearly decrease before ε_{2a} reaches to 0.7. After that, ε_{2b} will be allocated less and more noise will be generated for perturbing the sorted histogram. When the increased noise exceeds the reduced noise for perturbing raw categories, the MAE for category aggregates will increase again. It should be noted that when we assign all ε_2 to ε_{2a}, the MAE will reach the lowest point. The reason is that we only perturb the raw aggregated categories before sorting ($\varepsilon_{2a} = \varepsilon_2$) and less noise will be generated to perturb the categories than any other points.



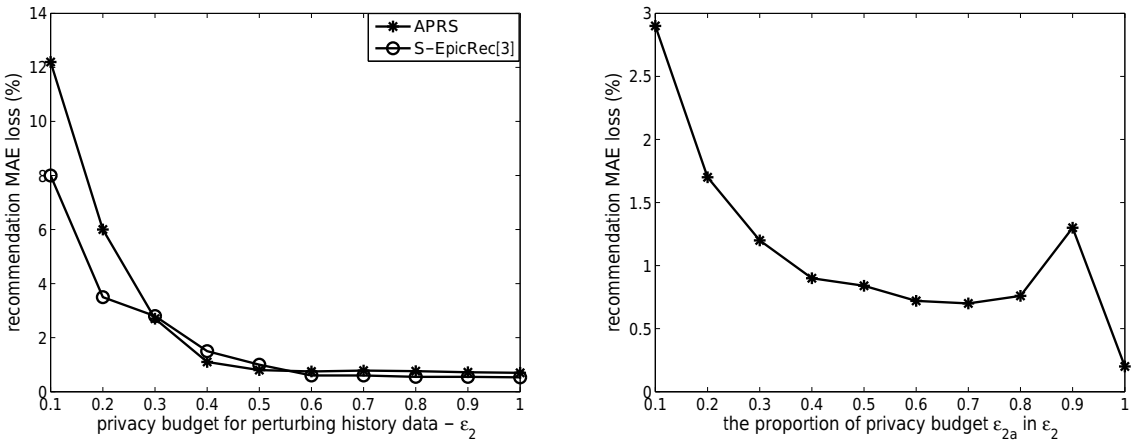
(a) MAE by varying privacy budget ε_2 (b) MAE by varying the proportion of ε_{2a} in ε_2

Figure D3 The perturbed category aggregates quality

In Fig. D4, we plot the recommendation accuracy by varying with the privacy budget. As shown in Fig. D4(a), the recommendation MAE loss decreases rapidly with the increasing of privacy budget ε_2. When we assign more budget ε_2 for histogram perturbation, less noise will be introduced and the recommender results will be more similar with the ground truth. The simulation results also show that our APRS owns a similar recommendation accuracy with S-EpicRec when we assign more privacy budget ε_2 for it. So our APRS can still obtain accurate recommender results after perturbing the aggregated categories. What's more, we also find that the recommendation MAE loss decreases slowly and nearly flat after $\varepsilon_2 = 0.4$. Hence, we find that the simulation result is in good agreement with the conclusion in Fig. D3(a) and consider the $\varepsilon_2 = 0.4$ is the optimal privacy budget for history data perturbation.

In Fig. D4(b), we also plot the recommendation accuracy by varying with the proportion of ε_{2a} in ε_2. The simulation results show that the recommendation accuracy will reduce dramatically at the beginning and then will increase reversely when $\varepsilon_{2a} = 0.8$ and 0.9. The reason is similar with that in Fig. D3(b). If we assign less privacy budget ε_{2a} to perturb the raw histogram before sorting, more noise will be added and the perturbation will be huge. So the recommendation MAE loss is large. With the varying of ε_{2a}, the reduced noise for perturbing raw histogram will exceeds the increased one for perturbing sorted histogram. So the recommendation MAE loss will decrease before ε_{2a} reaches to 0.7. After that, the

increased noise for perturbing sorted histogram will exceeds the reduced one, the recommendation MAE loss will increase again. While $\varepsilon_{2a} = \varepsilon_2$, the privacy budget will be all assigned to perturbing the raw histogram. So less noise will be generated to perturbation and the recommendation MAE loss is lowest.



(a) Recommendation accuracy by varying privacy budget ε_2 (b) Recommendation accuracy by varying the proportion of ε_{2a} in ε_2

Figure D4 Recommendation accuracy evaluation

Finally, we also discuss the efficiency of APRS along with the S-EpicRec scheme. As shown in Fig. D5, the running time always maintains stable by varying the ε and it is no longer than 400ms. The reason is that the varying of ε does not affect the size of categories, so the running time will not be influenced. Additionally, the simulation results also show that our APRS is more efficient than S-EpicRec. Therefore, based on the above analysis, we can convinced that our APRS is sufficiently efficient to perturb user's location and history data while requesting the LBS recommendation.

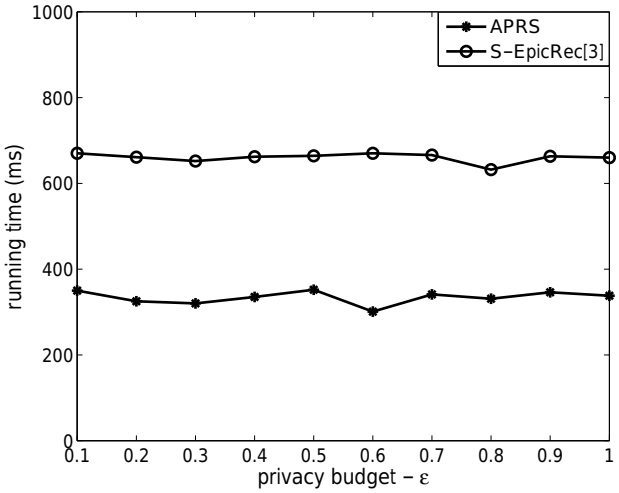


Figure D5 The efficiency evaluation of APRS

References

1 Dwork C. Differential privacy. In: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, Venice, 2006. 1-12

2 Dwork C, McSherry F, Nissim K, et al. Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd Theory of Cryptography Conference, New York, 2006. 265-284

3 McSherry F, Talwar K. Mechanism design via differential privacy. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, Providence, 2007.94-103.

4 Reed J, Pierce B C. Distance makes the types grow stronger: a calculus for differential privacy. In: Proceeding of the 15th ACM SIGPLAN International Conference on Functional Programming, Maryland, 2010. 157-168
5 Andrés M E, Bordenabe N E, Chatzikokolakis K, et al. Geo-indistinguishability: differential privacy for location-based systems. In: Proceedings of the Conference on Computer and Communications Security, Berlin, 2013. 901-914
6 Blum A, Ligett K, Roth A. A learning theory approach to non-interactive database privacy. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, 2008. 609-618
7 Shen Y L, Jin H X. Privacy-preserving personalized recommendation: an instance-based approach via differential privacy. In: Proceedings of the 2014 IEEE International Conference on Data Mining, Shenzhen, 2014.540-549
8 McSherry F. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. Commun. ACM, 2010, 53:89-97
9 Symeonidis P, Nanopoulos A, Papadopoulos A N, et al. Collaborative recommender systems: combining effectiveness and efficiency. Expert Systems with Applications, 2008, 34:2995-3013
10 Shen Y L, Jin H X. Epicrec: Towards practical differentially private framework for personalized recommendation. In: Proceedings of the ACM Conference on Computer and Communications Security, Vienna, 2016. 180-191