2019 IEEE 35th International Conference on Data Engineering (ICDE)

Collecting Preference Rankings under Local Differential Privacy

Jianyu Yang [†1], Xiang Cheng [†2*], Sen Su [†3], Rui Chen [‡], Qiyu Ren [†4], Yuhan Liu [†5]
[†]State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing, China
{jyyang[1], chengxiang[2], susen[3], qyren[4], liuyuhan[5]}@bupt.edu.cn
[‡]Samsung Research America, Mountain View, USA
rui.chen1@samsung.com

***Abstract*—In this paper, we initiate the study of collecting preference rankings under local differential privacy. The key technical challenge comes from the fact that the number of possible rankings increases factorially in the number of items to rank. In practical settings, this number could be large, leading to excessive injected noise. To solve this problem, we present a novel approach called SAFARI. The general idea is to collect a set of distributions over small domains which are carefully chosen based on the riffle independent model to approximate the overall distribution of users' rankings, and then generate a synthetic ranking dataset from the obtained distributions. By working on small domains instead of a large domain, SAFARI can significantly reduce the magnitude of added noise. Extensive experiments on real datasets confirm the effectiveness of SAFARI.**

I. Introduction

Preference ranking is one of the most common representations of personal data, which places different items into a total order based on individual opinions about their relative quality. With the deep penetration of the Internet and mobile devices, collecting and analyzing users' rankings are essential for service providers to provide better user experience and generate new revenue opportunities. However, users' preference rankings in many applications (e.g., political preference) are highly sensitive. Without proper privacy protection mechanisms, it either puts individual privacy in jeopardy or hampers business operations due to users' unwillingness to share their true rankings.

Local differential privacy (LDP) is the state-of-the-art privacy model for protecting individual privacy in the process of data collection. In LDP, each user locally perturbs his/her real data before sending it to the collector. In this paper, we, for the first time, investigate the problem of collecting preference rankings under local differential privacy. A straightforward solution is to consider each user's ranking as a categorical value in the domain that consists of all possible rankings, directly collect the overall distribution of users' rankings by existing LDP methods such as RAPPOR [1], and synthesize rankings from the obtained distribution. Nevertheless, this simple solution suffers from excessive noise and thus low data utility due to the large number of possible rankings.

To address this problem, we propose a novel approach, called **S**ampling R**A**ndomizer **F**or Multiple **A**ttributes with **R**iffle **I**ndependent Model (SAFARI), for collecting rankings under local differential privacy. Its main idea is to collect a set of distributions over small domains that are carefully chosen based on the riffle independent model to approximate the overall distribution of users' rankings, and then generate a synthetic ranking dataset from the obtained distributions. Since SAFARI works on small domains instead of a large domain, it can significantly reduce the magnitude of added noise.

The main contributions of this work are summarized as follows:

- We formulate the problem of preference ranking collection under local differential privacy and present a novel approach SAFARI based on the riffle independent model. While being proposed for collecting rankings, our idea is also applicable to other types of data.
- We design two transformation rules to instruct users to transform their data into multi-attribute data in which each attribute has a small domain. We further propose a new LDP method called SAFA for frequency estimation over multiple attributes that have small domains.
- We conduct extensive experiments over a real dataset to demonstrate the effectiveness of SAFARI.

II. Preliminaries and Problem Statement

A. Preference Ranking

Given an item set $\mathcal{X} = \{x_1, x_2, \ldots, x_d\}$, a ranking of $\mathcal{X}$ is an ordered list which contains all d items in $\mathcal{X}$. A typical type of rankings is preference ranking, which places different items into a total order based on individual opinions about their relative quality. We denote a preference ranking by $\sigma = \langle \sigma(1), \sigma(2), \ldots, \sigma(d) \rangle$ where $\sigma(j) = x_k$ means that x_k's rank under σ is j. We define the function $\sigma^{-1}(x_k) = j$ to retrieve x_k's rank j under σ.

*Corresponding author

2375-026X/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDE.2019.00151

B. Riffle Independent Model

The riffle independent model [2, 3] is a well-established probabilistic graphical model used to model rankings. Formally, its construction includes the following two phases.

1. Structure learning. In this phase, it constructs the structure of the riffle independent model, named K-thin chain. A K-thin chain denoted by $\mathcal{T}$ is a binary tree, which consists of leaf item sets and internal item sets. We denote leaf item sets in $\mathcal{T}$ by $\mathcal{L} = \{l_j | 1 \leq j \leq |\mathcal{L}|\}$ and internal item sets by $\mathcal{G} = \{g_j | 1 \leq j \leq |\mathcal{G}|\}$. The size of each leaf item set is no more than the constant K.

Specifically, it first computes the tripletwise mutual information of each possible triplet of distinct items. The definition of tripletwise mutual information is given below.

Definition 1 (Tripletwise Mutual Information)**.** *Given any triplet of distinct items, $(x_{k_1}, x_{k_2}, x_{k_3})$, its tripletwise mutual information is:*

$$\mathcal{I}(x_{k_1}, x_{k_2}, x_{k_3}) = \sum_{\varphi_{k_1}} \sum_{\lambda_{k_2,k_3}} \Pr[\varphi_{k_1}, \lambda_{k_2,k_3}] \log \frac{\Pr[\varphi_{k_1}, \lambda_{k_2,k_3}]}{\Pr[\varphi_{k_1}] \Pr[\lambda_{k_2,k_3}]},$$

where $\varphi_{k_1} = \sigma^{-1}(x_{k_1})$ and λ_{k_2,k_3} is a binary variable. In particular, $\lambda_{k_2,k_3} = 0$ represents $\sigma^{-1}(x_{k_2}) < \sigma^{-1}(x_{k_3})$ and $\lambda_{k_2,k_3} = 1$ represents $\sigma^{-1}(x_{k_2}) > \sigma^{-1}(x_{k_3})$.

Then, by optimizing an objective function of tripletwise mutual information (i.e., the Anchors method [3]), it hierarchically splits the original item set into disjoint subsets whose items have strong correlations to obtain the K-thin chain.

2. Parameter learning. In this phase, it computes the distributions over item sets in the K-thin chain $\mathcal{T}$. The distributions over leaf item sets are relative ranking distributions, while those over internal item sets are interleaving distributions. With these two types of low-dimensional distributions, it can approximate the overall distribution of rankings.

C. Problem Statement

Consider an item set $\mathcal{X} = \{x_1, x_2, \ldots, x_d\}$. Let n be the total number of users, and u_i $(1 \leq i \leq n)$ denote the i-th user. Each user u_i possesses a preference ranking σ_i of the d items in $\mathcal{X}$. Our goal is to design an approach to enable an untrusted data collector to collect the distribution of rankings from the n users and generate a synthetic ranking dataset that approximates the users' original rankings while satisfying local differential privacy. In this paper, we assume that all users use the same privacy budget ε for ease of presentation and analysis.

III. Safari

A. Overview of SAFARI

SAFARI consists of the following phases:

Phase 1. Each user transforms his/her ranking to provide information about the distributions required to compute the tripletwise mutual information by a transformation rule, called Rule I. The details of Rule I are given in Section III-B.

Phase 2. The data collector first invokes a new locally differentially private method called SAFA, using privacy budget $\varepsilon/2$, to collect those distributions from users' transformed data generated in Phase 1. The details of SAFA are described in Section III-D. Then the data collector computes all tripletwise mutual information and constructs a K-thin chain $\mathcal{T}$ (K is a small value set by the data collector). Finally, the data collector broadcasts $\mathcal{T}$ to all users.

Phase 3. Each user transforms his/her ranking to provide information about the relative ranking distributions and interleaving distributions over $\mathcal{T}$ by another transformation rule, called Rule II, which is discussed in detail in Section III-C.

Phase 4. The data collector invokes SAFA, using privacy budget $\varepsilon/2$, to collect the relative ranking distributions and interleaving distributions over $\mathcal{T}$ from users' transformed data generated in Phase 3.

Phase 5. The data collector synthesizes n rankings independently from the constructed riffle independent model to generate a ranking dataset. To synthesize a ranking from the riffle independent model, the data collector first samples a relative ranking for each leaf item set in $\mathcal{T}$, and then samples an interleaving for each internal item set in $\mathcal{T}$. Finally, with sampled interleavings, the data collector interleaves these relative rankings to a full ranking according to $\mathcal{T}$ in a bottom-up manner.

B. Design of Rule I

According to the definition of tripletwise mutual information shown in Section II-B, to compute the tripletwise mutual information for every possible triplet $(x_{k_1}, x_{k_2}, x_{k_3})$ of distinct items, the data collector needs to collect three types of distributions, i.e., $\mathcal{S}_1 = \{\Pr[\varphi_{k_1}] | 1 \leq k_1 \leq d\}$, $\mathcal{S}_2 = \{\Pr[\lambda_{k_2,k_3}] | 1 \leq k_2, k_3 \leq d\}$ and $\mathcal{S}_3 = \{\Pr[\varphi_{k_1}, \lambda_{k_2,k_3}] | 1 \leq k_1, k_2, k_3 \leq d\}$.

Since $\Pr[\varphi_{k_1}] = \sum_{\lambda_{k_2,k_3}} \Pr[\varphi_{k_1}, \lambda_{k_2,k_3}]$ and $\Pr[\lambda_{k_2,k_3}] = \sum_{\varphi_{k_1}} \Pr[\varphi_{k_1}, \lambda_{k_2,k_3}]$, the data collector only needs to collect the distributions in $\mathcal{S}_3$, from which the distributions in $\mathcal{S}_1$ and $\mathcal{S}_2$ can be derived. To achieve this goal, we design Rule I, which instructs users to transform their rankings to provide information about the distributions in $\mathcal{S}_3$. The details of Rule I are given as follows.

Rule I. Initially, each user u_i transforms his/her ranking σ_i into a tuple t_i that contains the values of all the attributes in an attribute set $\mathcal{A} = \{A_j | 1 \leq j \leq d \cdot \binom{d-1}{2}\}$. Here each attribute A_j in $\mathcal{A}$ corresponds to a triplet $(x_{k_1}, x_{k_2}, x_{k_3})$, where k_2 is less than k_3. The domain of A_j is $dom(A_j) = \{(1,0), (1,1), \ldots, (d,0), (d,1)\}$, which consists of $2d$ possible 2-tuples. In a 2-tuple, the first value represents the rank of x_{k_1} and the second value represents the relative rank relation between x_{k_2} and

Algorithm 1 Sampling Randomizer for Multiple Attributes (SAFA)

Input: Users' tuples $\{t_i | 1 \leq i \leq n\}$
Input: Privacy budget ε'
Output: Frequency vector set $\mathcal{Z} = \{\mathbf{z}_j | 1 \leq j \leq |\mathcal{A}|\}$
1: DC initializes all vectors in $\mathcal{Z}$;
2: **for** each user u_i **do**
3: DC draws an index j randomly from $\{1, \ldots, |\mathcal{A}|\}$;
4: DC sends j to u_i ;
5: u_i returns index $\widetilde{k}_i^j = LR(j, t_i, \varepsilon')$ to DC ;
6: DC increases $\mathbf{z}_j[\widetilde{k}_i^j]$ by 1 ;
7: **end for**
8: **for** each j-th vector $\mathbf{z}_j$ in $\mathcal{Z}$ **do**
9: DC sets the probability $p_j = \frac{e^{\varepsilon'}}{e^{\varepsilon'} + |dom(A_j)| - 1}$;
10: DC sets the probability $q_j = \frac{1}{e^{\varepsilon'} + |dom(A_j)| - 1}$;
11: **for** each k-th value $\mathbf{z}_j[k]$ in $\mathbf{z}_j$ **do**
12: DC updates

$$\mathbf{z}_j[k] = \frac{1}{n} \cdot \frac{|\mathcal{A}| \cdot \mathbf{z}_j[k] - nq_j}{p_j - q_j} ;$$

13: **end for**
14: **end for**
15: **return** $\mathcal{Z}$;

x_{k_3}. More concretely, for the second value, 0 represents $\sigma^{-1}(x_{k_2}) < \sigma^{-1}(x_{k_3})$ while 1 represents $\sigma^{-1}(x_{k_2}) > \sigma^{-1}(x_{k_3})$. Then, for each attribute A_j in $\mathcal{A}$, each user u_i assigns a value to $t_i[A_j]$, which is derived from his/her ranking σ_i.

Note that in SAFARI, the data collector and users use the same Rule I. By estimating the frequency of every possible value of each attribute in $\mathcal{A}$, the data collector can obtain the distributions in $\mathcal{S}_3$, and derive the distributions in $\mathcal{S}_1$ and $\mathcal{S}_2$.

C. Design of Rule II

After the K-thin chain $\mathcal{T}$ is constructed, the data collector needs to collect the distributions over $\mathcal{T}$. To this end, we design Rule II, which instructs users to transform their rankings to provide information about these distributions. The details of Rule II are given below.

Rule II. Each user u_i starts by transforming his/her ranking σ_i into a tuple t_i that contains the values of all the attributes in an attribute set $\mathcal{A}$. Specifically, $\mathcal{A}$ consists of two subsets $\mathcal{A}^{\mathcal{L}}$ and $\mathcal{A}^{\mathcal{G}}$ with $\mathcal{A} = \mathcal{A}^{\mathcal{L}} \cup \mathcal{A}^{\mathcal{G}}$ and $\mathcal{A}^{\mathcal{L}} \cap \mathcal{A}^{\mathcal{G}} = \emptyset$, which are defined as follows.

1) $\mathcal{A}^{\mathcal{L}} = \{A_j | 1 \leq j \leq |\mathcal{L}| - |\mathcal{L}_1|\}$

$\mathcal{A}^{\mathcal{L}}$ corresponds to the leaf item sets $\mathcal{L}$ in $\mathcal{T}$. As a subset of $\mathcal{L}$, $\mathcal{L}_1$ consists of the leaf item sets which contain only one item. Since the relative ranking distribution over each leaf item set in $\mathcal{L}_1$ can be easily derived, it is unnecessary for users to provide information about $\mathcal{L}_1$. Each attribute A_j in $\mathcal{A}^{\mathcal{L}}$ corresponds to a leaf item set l_k in $\mathcal{L} - \mathcal{L}_1$. The domain of A_j consists of all possible relative rankings of l_k. When K is 1, all leaf item sets contain only one item, which leads to $\mathcal{A}^{\mathcal{L}} = \emptyset$.

2) $\mathcal{A}^{\mathcal{G}} = \{A_j | 1 + |\mathcal{A}^{\mathcal{L}}| \leq j \leq |\mathcal{A}^{\mathcal{L}}| + |\mathcal{G}|\}$

Each attribute A_j in $\mathcal{A}^{\mathcal{G}}$ corresponds to an internal item set g_k in $\mathcal{G}$. $dom(A_j)$ consists of all possible interleavings of g_k. For each attribute A_j in $\mathcal{A}$, each user u_i assigns a value to $t_i[A_j]$, which is derived from his/her ranking σ_i.

Note that in SAFARI, the data collector and users apply the same Rule II. By estimating the frequency of every possible value of each attribute in $\mathcal{A}$, the data collector can obtain all the distributions over $\mathcal{T}$.

Algorithm 2 Local Randomizer (LR)

Input: Attribute index j
Input: User u_i's tuple t_i
Input: Privacy budget ε'
Output: Perturbed value index $\widetilde{k}_i^j$
1: Generate a perturbed value index $\widetilde{k}_i^j$ such that

$$\Pr[\widetilde{k}_i^j = k] = \begin{cases} \frac{e^{\varepsilon'}}{e^{\varepsilon'} + |dom(A_j)| - 1}, & if\ k = I(t_i[A_j]) \\ \frac{1}{e^{\varepsilon'} + |dom(A_j)| - 1}, & if\ k \neq I(t_i[A_j]) \end{cases},$$

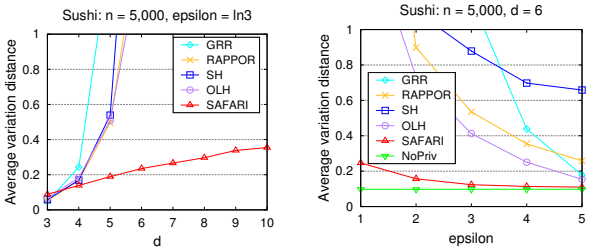
where $k \in \{1, \ldots, |dom(A_j)|\}$.
2: **return** $\widetilde{k}_i^j$;

D. Sampling Randomizer for Multiple Attributes

To collect the distribution information required for constructing the riffle independent model, the data collector needs to estimate the frequency of every possible value of each attribute in users' transformed tuples under LDP.

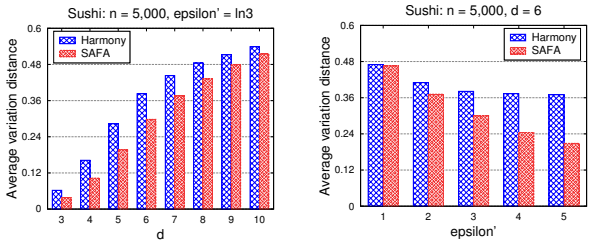
The data collector could use Harmony [4], the state-of-the-art method focusing on 1-way marginal distribution collection over multi-attribute data under LDP, to achieve this task. In particular, for each attribute A_j in $\mathcal{A}$, the data collector projects the domain of A_j into a binary matrix Φ_j of size $|dom(A_j)| \times |dom(A_j)|$. However, we observe that by using either Rule I or Rule II, each of the attributes in the transformed attribute set $\mathcal{A}$ has a small domain whose size is much less than $d!$. For an attribute with a small domain size, especially for a binary attribute, the projection in Harmony will introduce unnecessary noise in the collected data. Inspired by [5], which proves that generalized randomized response (GRR) [6] performs best when estimating frequencies of a small number of values, we propose a new LDP method, called **S**ampling R**A**ndomizer **F**or Multiple **A**ttributes (SAFA), to obtain more accurate frequency estimates for multiple attributes with small domains under LDP.

Algorithm 1 provides the details of SAFA. It takes users' tuples $\{t_i | 1 \leq i \leq n\}$ and privacy budget ε' as inputs, and outputs a vector set $\mathcal{Z}$ of estimated frequencies. For each vector $\mathbf{z}_j$ in $\mathcal{Z}$, it has $|dom(A_j)|$ values. The k-th value in $\mathbf{z}_j$ (i.e., $\mathbf{z}_j[k]$) denotes the frequency of the k-th value in $dom(A_j)$. In Algorithm 1, the data collector (represented by DC in the pseudo code) first initializes all vectors in the set $\mathcal{Z}$ by assigning the values in each vector to zeros. Then, for each user u_i, the data collector draws an index j uniformly at random from $\{1, \ldots, |\mathcal{A}|\}$ and sends j to u_i. After that, u_i perturbs $I(t_i[A_j])$, the index of $t_i[A_j]$ in $dom(A_j)$, as a new value index $\widetilde{k}_i^j$ by



(a) Q_2: varying d (b) Q_2: varying ε

Fig. 1. Performance of SAFARI



(a) $\mathcal{S}_3$: varying d (b) $\mathcal{S}_3$: varying ε'

Fig. 2. Effectiveness of SAFA

a local randomizer (LR) based on GRR and returns $\widetilde{k}_i^j$ to the data collector. The details of LR are given in Algorithm 2. With the received $\widetilde{k}_i^j$, the data collector increases $\mathbf{z}_j[\widetilde{k}_i^j]$ by 1. After the data collector finishes the interactions with all users, the values in each vector in $\mathcal{Z}$ are biased, because each user reports the information of one attribute instead of $|\mathcal{A}|$ attributes and uses LR to perform the random perturbation (Line 1 of Algorithm 2). To get unbiased estimates, the data collector performs the following steps. For each j-th vector $\mathbf{z}_j$ in $\mathcal{Z}$, the data collector sets the probability p_j and q_j (Lines 9-10 of Algorithm 1). With p_j and q_j, the data collector updates each value in $\mathbf{z}_j$ (Lines 11-13 of Algorithm 1).

Due to space limitation, we present the privacy and utility guarantees of SAFA in our technical report [7].

IV. Experiments

A. Experimental Settings

We make use of the real ranking dataset Sushi [8] in our experiments. It consists of rankings of 10 kinds of sushi from 5000 voters surveyed across Japan. To experiment with different numbers of items, we generate test datasets with 5,000 users and the number of items ranging from 3 to 10 from the Sushi dataset.

To evaluate the performance of SAFARI, we examine the accuracy of the second-order marginals [2] of the synthetic ranking dataset. For convenience, we denote the set of all the second-order marginals by Q_2. We use total variation distance to measure the accuracy of each noisy marginal and report the average variation distances over all marginals in Q_2. We compare SAFARI against four baseline approaches including GRR [6], RAPPOR [1], SH [9] and OLH [5]. To evaluate the effectiveness of SAFA, we compare it against Harmony [4]. We let the data collector invoke SAFA and Harmony to collect the distributions in $\mathcal{S}_3$ from the users' data transformed by Rule I, and report the average variation distances of the obtained distributions. In our experiments, we set $K = 1$, run each approach 10 times and report the average results.

B. Results

From Fig. 1(a), we observe that GRR, RAPPOR, SH and OLH can perform well when d is less than 4. This is because when d is small, the number of all possible rankings is still manageable. However, when d becomes relatively large, the advantage of SAFARI compared to the baseline approaches becomes much more observable, which confirms its good scalability. In Fig. 1(b), we include a non-private version of SAFARI, denoted by NoPriv, to show the utility loss due to privacy protection. We can see that SAFARI clearly outperforms the baseline approaches and the relative superiority of SAFARI is more pronounced when ε decreases. The reason can be explained as follows. The K-thin chain makes the collected distributions in SAFARI more robust against noise injection, which ensures that the quality of the synthetic data does not degrade significantly as ε decreases. Besides, we notice that SAFARI obtains performance close to that of NoPiv, especially when ε is relatively large. From Fig. 2, we can observe that SAFA outperforms Harmony in our experiments. This is consistent with our analysis in Section III-D that SAFA can improve the accuracy of the collected frequencies of attributes whose domain sizes are relatively small.

V. Conclusion

In this paper, we present SAFARI, a novel approach for collecting preference rankings under local differential privacy. Our results demonstrate the effectiveness of SAFARI. We believe that our work provides an effective means of data collection in many real-world applications while providing local differential privacy guarantee.

Acknowledgment

This work was supported by National Natural Science Foundation of China under Grant 61872045.

References

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

[2] J. Huang and C. Guestrin, "Riffled independence for ranked data," in *NIPS*, 2009.

[3] ——, "Learning hierarchical riffle independent groupings from rankings," in *ICML*, 2010.

[4] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," *CoRR*, vol. abs/1606.05053, 2016.

[5] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *USENIX Security*, 2017.

[6] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," in *NIPS*, 2014.

[7] "Technical report of safari," 2018. [Online]. Available: https://github.com/cheng-lab-at-bupt/SAFARI

[8] http://www.kamishima.net/sushi.

[9] R. Bassily and A. D. Smith, "Local, private, efficient protocols for succinct histograms," in *STOC*, 2015.