Privacy-Preserving Social Media Data Publishing for Personalized Ranking-Based Recommendation

Dingqi Yang, Bingqing Qu, and Philippe Cudré-Mauroux

Abstract—Personalized recommendation is crucial to help users find pertinent information. It often relies on a large collection of user data, in particular users' online activity (e.g., tagging/rating/checking-in) on social media, to mine user preference. However, releasing such user activity data makes users vulnerable to inference attacks, as private data (e.g., gender) can often be inferred from the users' activity data. In this paper, we proposed PrivRank, a customizable and continuous privacy-preserving social media data publishing framework protecting users against inference attacks while enabling personalized ranking-based recommendations. Its key idea is to continuously obfuscate user activity data such that the privacy leakage of user-specified private data is minimized under a given data distortion budget, which bounds the ranking loss incurred from the data obfuscation process in order to preserve the utility of the data for enabling recommendations. An empirical evaluation on both synthetic and real-world datasets shows that our framework can efficiently provide effective and continuous protection of user-specified private data, while still preserving the utility of the obfuscated data for personalized ranking-based recommendation. Compared to state-of-the-art approaches, PrivRank achieves both a better privacy protection and a higher utility in all the ranking-based recommendation use cases we tested.

Index Terms—Privacy-preserving data publishing, Customized privacy protection, Personalization, Ranking-based recommendation, Social media, Location based social networks

1 Introduction

Developing effective recommendation engines is critical in the era of Big Data in order to provide pertinent information to the users. To deliver high-quality and personalized recommendations, online services such as e-commerce applications typically rely on a large collection of user data, particularly user activity data on social media, such as tagging/rating records, comments, check-ins, or other types of user activity data. In practice, many users are willing to release the data (or data streams) about their online activities on social media to a service provider in exchange for getting high-quality personalized recommendations. In this paper, we refer to such user activity data as *public data*. However, they often consider part of the data from their social media profile as private, such as gender, income level, political view, or social contacts. In the following, we refer to those data as *private data*. Although users may refuse to release private data, the inherent correlation between public and private data often causes serious privacy leakage. For example, one's political affiliation can be inferred from her rating of TV shows [1]; one's gender can be inferred from her activities on location-based social networks [2]. These studies show that private data often suffers from *inference attacks* [3], where an adversary analyzes a user's public data to illegitimately gain knowledge about her private data. It is thus crucial to protect user private data when releasing public data to recommendation engines.

To tackle this problem, privacy-preserving data publishing has been widely studied [4]. Its basic idea is to provide protection on the private data by distorting the public data before its publication, at the expense of a loss of utility of the public data in the latter processing stages. For the use case of recommendation engines, utility refers to the personalization performance based on the distorted public data, i.e., *whether the recommendation engines can accurately predict the individual's preference based on the obfuscated data*. There is an intrinsic trade-off between privacy and personalization. On one hand, more distortion of public data leads to better privacy protection, as it makes it harder for adversaries to infer private data. On the other hand, it also incurs a higher loss in utility, as highly distorted public data prevents recommendation engines from accurately predicting users' real preferences.

To apply privacy-preserving data publishing techniques in the case of social media based recommendation, one immediate strategy is to obfuscate user public data on the user side before being sent to social media. However, such an approach is unrealistic as it hinders key benefits for users. In real-world use cases, social media provides users with a social sharing platform, where they can interact with their friends by intentionally sharing their comments/ratings on items, blogs, photos, videos, or even their real-time locations. For example, when a user watched a good movie and wants to share her high rating on it with her friends, she does not want the rating to be obfuscate in any sense.

As it is inappropriate to obfuscate user public data before being sent to social media, an alternative solution is to protect user privacy *when releasing their public data from social media to any other third-party services*. Specifically, many third-party services for social media require access to user activity data (or data streams) in order to provide them with

Authors are with the Department of Informatics at the University of Fribourg, Switzerland, E-mail: {dingqi.yang, bingqing.qu, philippe.cudre-mauroux}@unifr.ch
Manuscript received xxx; revised xxx.

personalized recommendations. In addition to such public data, these services may require optional access to users' profiles. While some privacy-conscious users want to keep certain data from their profiles (e.g., gender) as private, other non privacy-conscious users may not care about the same type of private data and choose to release them. Subsequently, an adversary could illegitimately infer the private data of the privacy-conscious users, by learning the correlation between the public and the private data from the non privacy-conscious users. Therefore, it is indispensable to provide privacy protection when releasing user public data from social media.

In this paper, we study the problem of privacy-preserving publishing of user social media data by considering both the specific requirements of user privacy on social media and the data utility for enabling high-quality personalized recommendation. Towards this goal, we face the following three challenges. First, since users often have different privacy concerns [5], a specific type of data (e.g., gender) may be considered as private by some users, while other users may prefer to consider it as public in order to get better personalized services. Therefore, the first challenge is to provide users with *customizable privacy protection*, i.e., to protect user-specified private data only. Second, when subscribing to third-party services, users often allow the service providers to access not only their historical public data, but also their future public data as a data stream. Although the obfuscated historical public data can efficiently reduce privacy leakage, the continuous release of user activity feed will incrementally increase such leakage (see Figure 7 for details). Therefore, the second challenge is to provide *continuous privacy protection over user activity data streams*. Third, we consider the case of ranking-based (or top-N) recommendation, which is more practical and has been widely adopted in practice by many e-commerce platforms [6]. As ranking-based recommendation algorithms mainly leverage the ranking of items for preference prediction, they are sensitive to the ranking loss incurred from the data obfuscation process. However, the computation of ranking losses often implies a high cost that is super-linear in the number of items used for recommendation [7]. Therefore, the third challenge is to *efficiently bound ranking loss* in data obfuscation.

Aiming at overcoming the above challenges, we propose PrivRank, a customizable and continuous privacy-preserving data publishing framework protect users against inference attacks while enabling personalized ranking-based recommendation. It provides continuous protection of user-specified private data against inference attacks by obfuscating both the historical and streaming user activity data before releasing them, while still preserving the utility of the published data for enabling personalized ranking-based recommendation by efficiently limiting the pairwise ranking loss incurred from data obfuscation. Our main contributions are summarized as follows:

- First, considering the use case of recommendation based on social media data, we identify a privacy-preserving data publishing problem by analyzing the specific privacy requirements and users' benefits of social media.
- Second, we propose a customizable and continuous data obfuscation framework for user activity data on social media. The key idea is to measure the privacy leakage of *user-specified private data* from public data based on mutual information, and then to obfuscate public data such that the privacy leakage is minimized under a given data distortion budget, which can ensure the utility of the released data. To handle the real-world use case of third-party services built on top of social media, our framework considers both historical and online user activity data:
  - *Historical data publishing*: When a user subscribes to a third-party service for the first time, the service provider has access to the user's entire historical public data. To obfuscate the user's historical data, we minimize the privacy leakage from her historical data by obfuscating her data using data from another user whose historical data is similar but with less privacy leakage.
  - *Online data publishing*: After the user subscribed to third-party services, the service provider also has real-time access to her future public data stream. Due to efficiency considerations, online data publishing should be performed based on incoming data instances only (e.g., a rating/tagging/checking-in activity on an item), without accessing the user's historical data. Therefore, we minimize the privacy leakage from individual activity data instance by obfuscating the data stream on-the-fly.
- Third, to guarantee the utility of the obfuscated data for enabling personalized ranking-based recommendation, we measure and bound the data distortion using a pairwise ranking loss metric, i.e., the Kendall-τ rank distance [8]. To efficiently incorporate such ranking loss, we propose a bootstrap sampling process to fast approximate the Kendall-τ distance.
- Finally, we conduct an extensive empirical evaluation of PrivRank. The results show that PrivRank can continuously provide customized protection of user-specified private data, while the obfuscated data can still be exploited to enable high-quality personalized ranking-based recommendation.

The rest of the paper is organized as follows. We present the related work in Section 2. The preliminaries of our work are presented in Section 3. Afterward, we firstly define our threat model in Section 4, and present our historical and online data publishing methods in Section 5 and 6, respectively. The experimental evaluation is shown in Section 7. We conclude our work in Section 8.

2 Related Work

To protect user privacy when publishing user data, the current practice mainly relies on policies or user agreements, e.g., on the use and storage of the published data [4]. However, this approach cannot guarantee that the users' sensitive information is actually protected from a malicious attacker. Therefore, to provide effective privacy protection when releasing user data, privacy-preserving data publishing has been widely studied. Its key idea is to obfuscate user data such that published data remains useful for some

application scenarios while the individual's privacy is preserved. According to the attacks considered, existing work can be classified into two categories.

The first category is based on heuristic techniques to protect ad-hoc defined user privacy [4]. Specific solutions mainly tackle the privacy threat when attackers are able to link the data owner's identity to a record, or an attribute in the published data. For example, to protect user privacy from identity disclosure, K-anonymity [9] obfuscates the released data so that each record cannot be distinguished from at least k-1 other records. However, since these techniques usually have ad-hoc privacy definitions, they have been proven to be non-universal and can only be successful against limited adversaries [10].

The second category is theory-based and focuses on the uninformative principle [11], i.e., on the fact that the published data should provide attackers with as little additional information as possible beyond background knowledge. Differential privacy [12] is a well-known technique that is known to guarantee user privacy against attackers with arbitrary background knowledge. Information-theoretic privacy protection approaches have also been proposed in that context. They try to quantitatively measure privacy leakage based on various entropy-based metrics such as conditional entropy [10] and mutual information [13], and to design privacy-protection mechanisms based on those measures. Although the concept of differential privacy is stricter (i.e., against attackers with arbitrary background knowledge) than that of information-theoretic approaches, the latter is intuitively more accessible and fits the practical requirements of many application domains [10]. In particular, information theory can provide intuitive guidelines to quantitatively measure the amount of a user's private information that an adversary can learn by observing and analyzing the user's public data (i.e. the privacy leakage of private data from public data).

In this study, we advocate the information-theoretic approach. Specifically, we measure the privacy leakage of user private data from public data based on mutual information, and then obfuscate public data such that the privacy leakage is minimized under a given data distortion budget. In the current literature, existing data obfuscation methods mainly ensure data utility by bounding the data distortion using metrics such as Euclidean distance [14], Squared L^2 distance [15], Hamming distance [1] or Jensen-Shannon distance [2]. They are analogous to limiting the loss of predicting user ratings on items, where the goal is to minimize the overall difference (e.g., mean absolute error) between the predicted ratings and the real ratings from the users. Although minimizing such a rating prediction error is widely adopted by the research community, ranking-based (or top-N) recommendation is more practical and is actually adopted by many e-commercial platforms [6]. Specifically, different from rating prediction that tries to infer how users rate items, ranking-based recommendation tries to determine a ranked list of items for the user, where the top items are most likely to be appealing to her. However, we argue that bounding data distortion using traditional metrics is not optimal for ranking-based recommendation, whose goal is to minimize the ranking difference (e.g., pairwise ranking loss or mean average precision) between the predicted ranking list and the actual list from the users. Therefore, different from existing methods that bound data distortion using non-ranking-based measures, our approach considers bounding the ranking loss incurred from the data obfuscation process using the Kendall-τ rank distance [8] to preserve the utility of the published data for personalized ranking-based recommendation. In addition, as the computation of ranking losses often implies a high cost that is super-linear in the number of items for recommendation [7], we develop a bootstrap sampling process to fast approximate the Kendall-τ distance.

Compared to our previous work [2], this paper makes the following improvements: 1) we extend the scope of our privacy-preserving data publishing problem from location based social networks to general social media; 2) we improve the data utility guarantee by explicitly considering the use case of personalized ranking-based recommendation, and re-design the privacy-preserving data publishing framework by bounding ranking loss; 3) we discuss and compare different types of ranking losses and select Kendall-τ distance, and propose a bootstrap-sampling process for its fast approximation; 4) we re-design and conduct new experiments with two ranking-based recommendation use cases to show the effectiveness of our framework and its superiority over our previous work [2] for enabling ranking-based recommendations; 5) we conduct a thorough scalability study with synthetic datasets, and show that PrivRank can scale up to large datasets.

3 Preliminaries

3.1 System Workflow

Figure 1 illustrates the end-to-end workflow of our system. PrivRank is implemented as a supplementary module to existing social media platforms, in order to let user enjoy high-quality personalized recommendations from third-party services under a customized privacy guarantee.

1) When users interact with each other via a social media service, they voluntarily share their activity data, particularly the tagging/rating/checking-in activities which massively implies their preference.
2) When a user wants to subscribe to third-party services, she typically needs to give them access to such kind of activity data. Specifically, right after the user's subscription, third-party services can immediately access the user's historical activity data. Before releasing her such data and according to the user's own criteria, the *historical data publishing* module obfuscates her historical activity data to protect user-specified private data against inference attacks. Afterward, when the user continuously report her activity on the social media, the *online data publishing* module obfuscates each activity (e.g., adding a tag to a photo, rating a movie or checking in at a POI) from her activity streams before sending to third-party services. All data obfuscation is performed with the utility guarantee for personalized ranking-based recommendation by limiting the ranking loss incurred from data obfuscation.
3) Despite receiving obfuscated public data, the third-party services can still provide high-quality personalized ranking-based recommendation to the users.

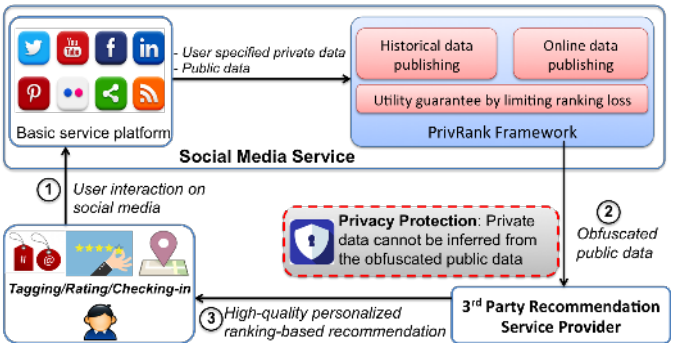


Fig. 1. System workflow for privacy-preserving publishing of social media data: 1) Users report their activity (i.e., public data) on social media services; 2) PrivRank publishes the obfuscated public data to third-party service providers; 3) The third-party service providers can still deliver high-quality personalized ranking-based recommendation to users.

Our system workflow is beneficial to all the involved entities. First, a user still shares her actual activities with her friends on a social media platform, while now enjoying high-quality personalized recommendations from a third-party service under a customized privacy guarantee, as only obfuscated user activity data are released from the social media platform to the third-party service. Second, the third-party service may attract more users (in particular privacy-conscious users), when providing high-quality recommendation services with privacy protection. Third, the social media platform may also boost its business by attracting more users and more third-party services by providing privacy protection data publishing. In addition, another advantage of our framework lies on its easy integration with the existing social media platform, where the latter does not need to be changed significantly (as shown in Figure 1). Therefore, these benefits could incentivize both the social media platform and the third-party service to implement PrivRank.

3.2 User Preference Modeling from Social Media Data

Users' activities on social media massively implies their preferences. Individual social media services often provide users with a unique feature (or a certain type of items) for interaction, such as photos for Flickr, videos for YouTube, music for Last.fm, and POIs for Foursquare. By interacting with these items on social media (e.g., tagging a photo, rating a video or checking-in at a POI), users explicitly or implicitly express their preferences on those items. In this work, we consider such user activity as public data. Formally, let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of users and items, respectively. A typical representation of such public data from a user u ($u \in \mathcal{U}$) is a vector V^u (of size $|\mathcal{I}|$) that encodes any type of user preference, such as the user's ratings (on a 1-5 scale), her tags/thumbs-ups (in a binary format), or her cumulative number of interactions (e.g., the number of check-ins on POIs). When the user subscribes to a third-party service for the first time, the service provider will have immediate access to the user's (historical) public data vector V^u which contains the user's all historical activities. Afterward, the service provider can also observe the future user activity feed, where each activity (e.g., rating/tagging/checking-in) on item i ($i \in \mathcal{I}$) will be used to update the corresponding element V_i^u of the public data vector.

Items	i_1	i_2	i_3
Original rating: a	1	2	5
Obfuscated rating 1: $\hat{a}_1$	3	4	5
Obfuscated rating 2: $\hat{a}_2$	1	4	3

Fig. 2. A toy example of ranking loss. The real user rating a on three items can be obfuscated to $\hat{a}_1$ or $\hat{a}_2$. While the Euclidean distances for both obfuscations are exactly the same, the ranking loss from the two obfuscations are different. Compared to the ranking list $i_1 < i_2 < i_3$ in the original rating a, the obfuscated rating $\hat{a}_1$ does not incur any ranking loss as we still observe $i_1 < i_2 < i_3$. However, the obfuscated rating $\hat{a}_2$ incurs a certain ranking loss as we find $i_1 < i_3 < i_2$ there.

3.3 Ranking-Based Recommendation

Based on the aforementioned public data vectors, ranking-based recommendation outputs a ranked list of items for a user, where the top items are most likely to be appealing to her. The related algorithms mainly leverage the existing ranking of items in the learning process to predict the missing rank of the items for recommendation [16]. Therefore, ranking-based recommendation algorithms are sensitive to the ranking loss incurred from the data obfuscation process, rather than other types of loss measured by the Euclidean or Squared L^2 distance, for example. Moreover, those traditional data distortion measures are not analogous to ranking loss [16]. Figure 2 shows an example where the same data distortion budget measured by Euclidean distance may imply different ranking losses. Therefore, considering ranking loss incurred from data obfuscation is critical for ranking-based recommendation.

4 Threat Model

In this study, we consider the inference attack [3] as the targeted threat model. As described above, we consider that each user has two types of data: i) *public data* (e.g., her activity data) that she is willing to release for getting personalized recommendations, and ii) *private data* (e.g., gender) that she wants to keep private. We denote public data as $X \in \mathcal{X}$, and private data as $Y \in \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are the sets of values that X and Y can take, respectively. Since Y is often linked to X by their joint probability $p(X, Y)$, an adversary who observes X is able to gain some knowledge about Y. To reduce such privacy leakage, the basic idea is to release a distorted $\hat{X} \in \hat{\mathcal{X}}$ instead of X such that it is hard to infer Y from $\hat{X}$.

4.1 Inference Attack

Inference attacks assume that an adversary has a method q to infer Y, where the adversary always tries to select q such that the cost (e.g., inference error) of using q to infer Y is minimized [13]. Therefore, before observing $\hat{X}$, q can be obtained by solving the following problem:

$$c = \min_{q} E_Y[C(Y, q)] \quad (1)$$

where $C(Y, q)$ is the expected cost function of inferring Y using q. After observing the distorted public data $\hat{X}$, q can be obtained by solving the following problem:

$$\hat{c} = \min_{q} E_{Y|\hat{X}}[C(Y, q)|\hat{X}] \quad (2)$$

The adversary's cost gain after observing $\hat{X}$ is as follows:

$$\Delta C = c - \hat{c}, \quad (3)$$

which measures how much knowledge the adversary gains w.r.t. the inference of Y after observing $\hat{X}$. The idea of privacy protection is to find $\hat{X}$ such that the privacy leakage ΔC is reduced, while the obfuscated $\hat{X}$ can still be used to enable personalized recommendation.

4.2 Basic Idea of Our Solution

In order to reduce the privacy leakage ΔC, we obfuscate X to obtain $\hat{X}$ based on a probabilistic obfuscation function $p_{\hat{X}|X}$, which encodes the conditional probability of releasing $\hat{X}$ when observing X. Intuitively, $p_{\hat{X}|X}$ should be designed such that any inference attack on Y should be rendered weak. Meanwhile, it also keeps some utility of $\hat{X}$ by limiting the distortion budget in the obfuscation process, which can be modeled by a constraint ΔX as follows;

$$E_{\hat{X},X}(dist(\hat{X}, X)) \leq \Delta X \quad (4)$$

where $dist(\hat{X}, X)$ is a certain distance metric that measures the difference between $\hat{X}$ and X. ΔX limits the expected distortion w.r.t. the probabilistic obfuscation function $p_{\hat{X}|X}$. The data distortion budget can ensure the utility of the released data. Considering the data utility for enabling personalized ranking-based recommendation, we measure and bound the data distortion budget using ranking distance. In summary, the key idea of our solution is to *learn* $p_{\hat{X}|X}$ *that minimizes* ΔC *under a given distortion budget* ΔX.

5 Historical Data Publishing

To publish historical public data in a privacy-preserving way, the key idea is to probabilistically obfuscate a user's historical public data vector to that of another user, which are similar but have less privacy leakage. In this context, data obfuscation operates on one's whole public data vector, rather than obfuscating her individual activity records one by one (over the user's activity stream). Compared to the streaming scheme, we show that such a historical data obfuscation scheme can achieve the same level of privacy protection with a lower data distortion budget (see Section 7.3 for details).

Figure 3 gives an overview of the historical data publishing process. First, aiming at reducing the problem complexity stemming from learning the optimal obfuscation function, we incorporate a clustering step in our framework to cluster a large number of users into a limited number of groups based on their public data, as similar user activities often cause similar privacy leakage [2]. Second, based on the user clusters, we quantitatively measure the privacy leakage of user-specified private data (e.g. gender) from public data, and then learn the optimal obfuscation function by minimizing the privacy leakage under a given distortion budget which bounds the ranking loss. Finally, based on the learned obfuscation function, we perform probabilistic data obfuscation. Customized privacy protection is achieved in the way that, for the specified private data (e.g., gender), a corresponding obfuscation function is generated.

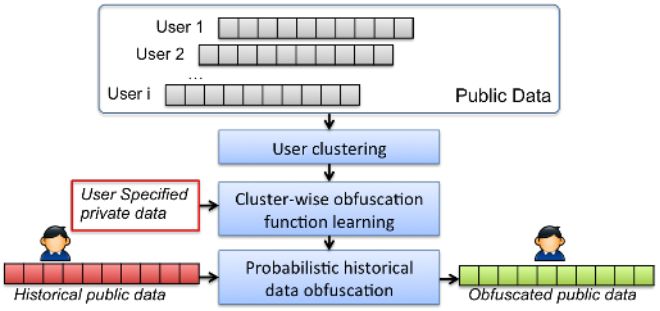


Fig. 3. Historical data publishing

5.1 User Clustering

We try to obfuscate a user's historical public data vector to that of another user. Directly learning the optimal obfuscation function $p_{\hat{X}|X}$ from individual user's public data incurs the complexity growing quadratically with the number of users $|\mathcal{U}|$. To reduce the problem complexity, the user clustering phase clusters users into a limited and fixed number of groups according to their public data vector. Then, the complexity related to learning the optimal obfuscation function between user clusters rather than between individual users is hence reduced and independent with $|\mathcal{U}|$. Specifically, we cluster the set of users $\mathcal{U}$ based on their historical public data vector V^u. We adopt average-linkage hierarchical clustering [17] using Euclidean distance for the sake of simplicity. Based on the clustering results, we obtain the mapping from users $\mathcal{U}$ to clusters $\mathcal{G}$, where each element G ($G \in \mathcal{G}$) is the centroid of the corresponding cluster.

5.2 Cluster-wise Obfuscation Function Learning

The optimal obfuscation function is learned based on user clusters. *Therefore, values for the public data* $\mathcal{X}$ *and for the released public data* $\hat{\mathcal{X}}$ *refer to user clusters* $\mathcal{G}$. Without loss of generality, in the following derivation, we keep using X and Y for public and private data, respectively. In the following, we first formally present our privacy-utility tradeoff, and then the utility guarantee by bounding ranking loss, followed by the obfuscation function learning algorithm.

5.2.1 Balancing Privacy and Utility

The privacy leakage in this paper is measured by ΔC, which represents the information gain of an adversary after observing the released public data $\hat{X}$. When using a log-loss cost function, Calmon et al. [13] proved that ΔC becomes the mutual information between the released public data $\hat{X}$ and the specific private data Y:

$$\Delta C = I(\hat{X}, Y) = \sum_{\hat{x}\in\hat{X}, y\in Y} p(\hat{x}, y) \log \frac{p(\hat{x}, y)}{p(\hat{x})p(y)} \quad (5)$$

As noted above, we use the probabilistic obfuscation function $p_{\hat{X}|X}$ to generate the released public data $\hat{X}$. Therefore, the joint probability of $\hat{X}$ and Y can be computed as:

$$p(\hat{x}, y) = \sum_{x \in X} p_{\hat{X}|X}(\hat{x}|x) p_{X,Y}(x, y) \tag{6}$$

The marginal probability $p_{\hat{X}}(\hat{x})$, $p_X(x)$ and $p_Y(y)$ can be calculated as follows:

$$p_{\hat{X}}(\hat{x}) = \sum_{x \in X, y \in Y} p_{\hat{X}|X}(\hat{x}|x) p_{X,Y}(x, y) \tag{7}$$

$$p_X(x) = \sum_{y \in Y} p_{X,Y}(x, y),\ p_Y(y) = \sum_{x \in X} p_{X,Y}(x, y) \tag{8}$$

Combined with the above Equations, the mutual information between the release public data $\hat{X}$ and the private data Y can be derived as:

$$I(\hat{X}, Y) = \sum_{\hat{x} \in \hat{X}, y \in Y} p(\hat{x}, y) \log \frac{p(\hat{x}, y)}{p(\hat{x})} - \sum_{y \in Y} p(y) \log p(y) \tag{9}$$

where the second term is the entropy of Y, i.e., $-\sum_{y \in Y} p(y) \log\ p(y)$, which is a constant for the specified private data (e.g., gender) in a given dataset. Hence, we ignore this term in the following derivations and obtain:

$$I(\hat{X}, Y) = \sum_{\hat{x} \in \hat{X}, y \in Y} p(\hat{x}, y) \log \frac{p(\hat{x}, y)}{p(\hat{x})} \tag{10}$$

Combined with Equations 6 and 7, the mutual information can then be derived as a function of only two factors, namely the joint probability $p_{X,Y}$ which can be empirically obtained from a given dataset, and the obfuscation function $p_{\hat{X}|X}$:

$$\begin{aligned} I(\hat{X}, Y) = & \sum_{\substack{\hat{x} \in \hat{X} \\ x \in X \\ y \in Y}} p_{\hat{X}|X}(\hat{x}|x) p_{X,Y}(x, y) \\ & \cdot \log \frac{\sum\limits_{x' \in X} p_{\hat{X}|X}(\hat{x}|x') p_{X,Y}(x', y)}{\sum\limits_{\substack{x'' \in X \\ y' \in Y}} p_{\hat{X}|X}(\hat{x}|x'') p_{X,Y}(x'', y')} \end{aligned} \tag{11}$$

The optimal obfuscation function $p_{\hat{X}|X}$ is learned such that $I(\hat{X}, Y)$ is minimized under a given distortion budget ΔX.

5.2.2 Bounding Ranking Loss for Utility

To provide optimal utility guarantees for personalized ranking-based recommendation, we consider bounding the data distortion $dist(\hat{X}, X)$ based on ranking loss. There are typically three types of ranking loss functions [16], namely pointwise, pairwise, and listwise, which are defined on the basis of single items, pairs of items, and all ranking items, respectively. As a pointwise loss function measures the loss of (ranking) score for individual items, it is analogous to non-ranking-based distance metrics. A theoretical study on these three types of ranking loss functions [16] shows that pairwise and listwise losses are indeed upper bounds of two quantities 1-MAP and 1-NDCG, respectively, where Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) are two popular metrics for evaluating ranking-based information retrieval algorithms [18], [19]. In other words, for ranking-based recommendation algorithms, minimizing pairwise/listwise loss is equivalent to maximizing the predicted ranking quality measured by MAP or NDCG. Following this idea, we also want to bound the data distortion incurred from data obfuscation by limiting the pairwise ranking loss when obfuscating X into $\hat{X}$.

We choose to measure the pairwise ranking loss using a widely known metric, i.e., the Kendall-τ rank distance [8]. It measures the number of pairwise disagreements between two ranking lists. For two users a and b, we denote their public data vectors as V^a and V^b, respectively. The Kendall-τ rank distance $K(V^a, V^b)$ is then computed as:

$$K(V^a, V^b) = \sum_{V_i^a > V_j^a} \mathbb{1}_{V_i^b < V_j^b} \tag{12}$$

where V_i^a is the ranking score of item i in list V^a, and so on. $\mathbb{1}_{cond}$ is an indicator function which is equal to 1 when $cond$ is true and 0 otherwise. As Eq. 12 counts the absolute number of pairwise disagreements, we normalize it by dividing by $|\mathcal{I}|(|\mathcal{I}| - 1)/2$, so that the normalized Kendall-τ distance lies in the interval [0,1]:

$$K(V^a, V^b) = \frac{1}{|\mathcal{I}|(|\mathcal{I}| - 1)/2} \sum_{V_i^a > V_j^a} \mathbb{1}_{V_i^b < V_j^b} \tag{13}$$

A value of 1 indicates maximum disagreement while 0 indicates that the two lists express the same ranking. For the sake of simplicity, all terms of the Kendall-τ distance refer to the normalized Kendall-τ distance (Eq. 13) in the following.

In practice, a large number of items yields a high cost when computing the Kendall-τ distance. Since the computation of the Kendall-τ distance requires a total of $|\mathcal{I}|(|\mathcal{I}| - 1)/2$ pairwise comparisons, the resulting computation complexity is $O(n^2)$, where n is the number of items $|\mathcal{I}|$. To efficiently compute the Kendall-τ distance for large item sets, we propose to use a bootstrap sampling process [20] to approximate the Kendall-τ distance. Specifically, instead of computing all $|\mathcal{I}|(|\mathcal{I}| - 1)/2$ comparisons, we randomly sample S pairs of items for comparison. After counting the absolute number of disagreements in S sampled pairs, we then normalize it by dividing by $|S|$:

$$K(V^a, V^b) \approx \frac{1}{|S|} \sum_{V_i^a > V_j^a, (i,j) \in S} \mathbb{1}_{V_i^b < V_j^b} \tag{14}$$

5.2.3 Optimal Obfuscation Function Learning

Considering the above ranking loss as a constraint to ensure high data utility, we now present our algorithm that learns the optimal cluster-wise obfuscation function $p_{\hat{G}|G}$. For a given dataset, we can empirically determine $p_{G,Y}$ according to private data Y (e.g., gender). Thus, the obfuscation function $p_{\hat{G}|G}$ can be learned by Algorithm 1, which contains a convex optimization problem with three constraints (can be solved by many solvers such as CVX [21]). The first constraint is for the distortion budget that bounds the expected Kendall-τ distance w.r.t. the probabilistic obfuscation function $p_{\hat{G}|G}$. Note that it is easy to compute the Kendall-τ distance for $\hat{G}$ and G. The last two constraints are probability constraints of $p_{G\hat{|}G}$. To stress the protected private data Y, we denote the corresponding optimal obfuscation function

Algorithm 1 Cluster-wise obfuscation function learning

Require: Joint probability $p_{G,Y}$, and distortion budget ΔX

1: Solve the optimization problem for $p_{\hat{G}|G}$

$$\min_{p_{\hat{G}|G}} \; I(\hat{G}, Y)$$
$$s.t., \; E_{\hat{G},G}(K(\hat{G}, G)) \leq \Delta X$$
$$p_{\hat{G}|G}(\hat{g}|g) \in [0,1], \forall g, \hat{g} \in \mathcal{G}$$
$$\sum_{\hat{g}} p_{\hat{G}|G}(\hat{g}|g) = 1, \forall g \in \mathcal{G}$$

2: **return** $p_{\hat{G}|G,Y}$

Algorithm 2 Probabilistic historical data obfuscation

Require: Obfuscation functions $p_{\hat{G}|G}$ for all possible Y

1: **for** $u \in \mathcal{U}$ **do**
2: Get u-specified private data Y
3: Get obfuscation function $p_{\hat{G}|G,Y}$ for Y
4: Get u's cluster G, where $u \in G$
5: Obfuscate the user's cluster G to $\hat{G}$ based on $p_{\hat{G}|G,Y}(\hat{G}|G)$
6: Randomly select a user $\hat{u}$ in cluster $\hat{G}$
7: Obfuscate $V^{\hat{u}}$ to V^u
8: **end for**

as $p_{\hat{G}|G,Y}$. Note that we do not assume any inference attack methods in our framework, and that any inference attacks on Y should be rendered weak.

As we try to find the optimal obfuscation probability between each pair of user clusters, the problem complexity of learning optimal $p_{\hat{G}|G}$ in Algorithm 1 is $O(n^2)$, where n is the number of user clusters $|\mathcal{G}|$ rather than the number of users $|\mathcal{U}|$. The later evaluation shows that a small number of $\mathcal{G}$ can indeed provide an efficient privacy protection (see Section 7.7 for details).

5.3 Probabilistic Historical Data Obfuscation

Since the learned obfuscation function is based on user clusters, we still need to bridge the gap between clusters and users to obfuscate individual public data vectors. Algorithm 2 describes the probabilistic data obfuscation process. Specifically, for a user u, we first obtain the corresponding obfuscation function $p_{\hat{G}|G,Y}$ to protect her private data Y (Line 2-3). We then obfuscate her cluster G to another $\hat{G}$ based on the obfuscation function $p_{\hat{G}|G,Y}(\hat{G}|G)$ (Line 4-5). Finally, since all users in cluster $\hat{G}$ share the similar public data vectors, we randomly select one user $\hat{u}$ in the cluster $\hat{G}$, and leverage her public data vector $V^{\hat{u}}$ to obfuscate (replace) V^u (Line 6-7).

6 Online Data Stream Publishing

After a user subscribed to third-party services, the service providers have access to the user's future activity streams. Therefore, we protect her private data by obfuscating her activity stream on-the-fly. Different from historical data publishing, the streaming nature of user activity imposes the following constraint on online data obfuscation: Due to time and space efficiency requirements of real-time data publishing (i.e., single-pass processing with limited memory) [22], *online data obfuscation can only be performed based on the incoming activity data instance itself* (e.g., a new

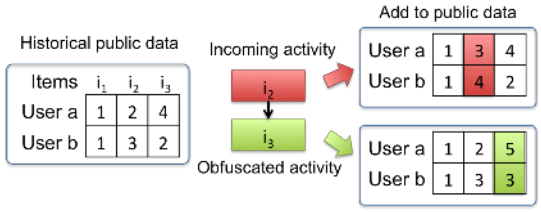


Fig. 4. A toy example of online data obfuscation causing different ranking losses. The public data vector encodes the count of check-ins at POIs (i.e., the cumulative count of a user's interactions with items), and incrementally incorporates incoming check-ins from user activity feed. Suppose the incoming activity is a check-in at POI i_2, and the obfuscation maps it to i_3. By adding this activity/obfuscated activity to two different users' public data, we observe different ranking losses. While this obfuscation causes no ranking loss for user a (i.e., we always find $i_3 > i_2 > i_1$ before and after obfuscation), it causes some ranking loss for user b (i.e., the ranking of $i_2 > i_3$ no longer holds after obfuscation).

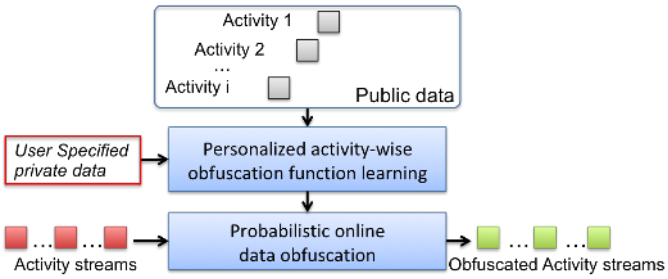


Fig. 5. Online data publishing

rating/tagging/checking-in activity on an item), without accessing the user's historical data. In other words, we want to obfuscate each activity to another with less privacy leakage. However, a rating/tagging/checking-in activity of a user will probably lead to a certain modification of the user's public data vector that encodes a certain type of user preference, such as the user's ratings (on a 1-5 scale), her tags/thumbs-ups (in a binary format), or her cumulative number of interactions (e.g., the number of check-ins on POIs). Therefore, obfuscating an activity to another often causes different ranking loss for different public data vectors (i.e., different users). Figure 4 shows a toy example where the same obfuscation causes different ranking losses for two different users' public data. Therefore, the probabilistic obfuscation function here should be personalized, i.e., ranking loss measured based on the user's own public data.

Figure 5 shows the online data publishing process. First, by measuring the privacy leakage of the user-specified private data from each activity, we learn the personalized optimal obfuscation function such that the privacy leakage is minimized under a given distortion budget. Second, for each incoming data instance of a user from the user activity data stream, we perform the probabilistic obfuscation according to the learned obfuscation function of that user.

6.1 Personalized Activity-wise Obfuscation Function Learning

The activity-wise obfuscation function is learned based on individual activities, where an activity refers to a rating/tagging/checking-in activity on an item. The idea here is to obfuscate an activity on one item using another

Algorithm 3 Personalized activity-wise obfuscation function learning

Require: Joint probability $p_{i,Y}$, distortion budget ΔX, user public data vector V^u

1: Solve the optimization problem for $p_{\hat{i}|i}$

$$\min_{p_{\hat{i}|i}} \ I(\hat{i}|i,Y)$$
$$s.t., \ E_{\hat{i},i}(K(V^u+i,V^u+\hat{i})) \leq \Delta X$$
$$p_{\hat{i}|i}(\hat{i}|i) \in [0,1], \forall i,\hat{i} \in \mathcal{I}$$
$$\sum_{\hat{i}} p_{\hat{i}|i}(\hat{i}|i) = 1, \forall i \in \mathcal{I}$$

2: **return** $p^u_{\hat{i}|i,Y}$

Algorithm 4 Probabilistic online activity obfuscation

Require: Obfuscation functions $p^u_{\hat{i}|i}$ for all possible Y, an incoming activity of user u on item i

1: Get u-specified private data Y
2: Get personalized obfuscation function $p^u_{\hat{i}|i,Y}$ for u and Y
3: Obfuscate i to $\hat{i}$ based on $p_{\hat{i}|i,Y}(\hat{i}|i)$
4: **return** obfuscated activity of user u on item $\hat{i}$

item with less privacy leakage. *Therefore, values for the public data $\mathcal{X}$ and for the released public data $\hat{\mathcal{X}}$ refer to the item set $\mathcal{I}$.* To minimize the privacy leakage of private data Y from public data i ($i \in \mathcal{I}$), we follow the same privacy-utility trade-off framework for cluster-wise obfuscation function learning (Section 5.2.1), and try to minimize $I(\hat{i},Y)$.

The personalized obfuscation function is learned by considering individual users' public data in order to better bound the ranking loss. Let $V^u + i$ denote the public data resulting from adding an incoming activity i to the public data V^u of user u. For an obfuscation from item i to item $\hat{i}$, we measure the ranking loss caused by this obfuscation as $K(V^u + i, V^u + \hat{i})$. Note that we can easily fast compute $K(V^u+i, V^u+\hat{i})$ instead of using bootstrap sampling based approximation, as the potential pairwise ranking differences can only be related to ranking pairs involving item i or item $\hat{i}$. Subsequently, the computation complexity becomes $O(n)$, where n is the number of items $|\mathcal{I}|$.

In summary, we first empirically calculate $p_{i,Y}$, and learn the optimal obfuscation function $p^u_{\hat{i}|i,Y}$ for each user u using Algorithm 3 which is also a convex optimization problem. The complexity of solving the optimization problem in Algorithm 3 depends only on $|\mathcal{I}|$.

6.2 Probabilistic Online Activity Obfuscation

Based on the learned obfuscation functions, we obfuscate each incoming activity from a user's activity stream using Algorithm 4. For each incoming activity on item i, we first obtain the corresponding obfuscation function $p^u_{\hat{i}|i,Y}$ to protect u-specified private data Y (Line 1-2). We then obfuscate the activity on item i based on $p^u_{\hat{i}|i,Y}(\hat{i}|i)$, and map the activity onto item $\hat{i}$ as obfuscated data (Line 3-4).

7 Experimental Evaluation

We empirically evaluate the effectiveness and efficiency of our framework. Specifically, based on real-world datasets, we first investigate the trade-off between privacy protection and personalization performance for ranking-based recommendation. Second, we study the continuous privacy protection performance by evaluating the privacy leakage over time. Third, we evaluate the customization performance of privacy protection by comparing the privacy leakage of user-specified private data and that of other data. Fourth, we further explore the utility guarantee for ranking-based recommendation under different loss metrics. Fifth, based on synthetic datasets, we study the impact of private data settings. Finally, we evaluate the runtime performance of our framework. We start by introducing our experimental setup below before reporting on the evaluation results.

7.1 Experimental Setup

7.1.1 Dataset

Although there are many public social media datasets available for benchmarking recommendation systems, very few of them provide the corresponding private data for privacy studies. Therefore, we collected our own dataset from a location based social network Foursquare for evaluation. Specifically, users can share their real-time presence on Foursquare by checking-in at Point of Interests (POIs), e.g., a bar or a supermarket. Such spatiotemporal user activity data are widely used for enabling various personalized recommendations [23], [24], [25], as check-in data can be regarded as "foot rating", where a higher visiting frequency of a POI implies a more positive preference. Using the method described in [26], [27], we crawled Foursquare check-in data via Twitter Public Streams[1] for about 18 months (from Apr. 2012 to Sep. 2013) in two big cities (i.e., New York City and Tokyo), and consider them as *public data*. Table 1 shows the statistics of the resulting datasets we collected.

In addition, we also collected the corresponding user profile data as *private data*. Due to Foursquare's privacy policy, only limited profile data (i.e., name and gender) is included in the check-ins. Fortunately, as the dataset is collected via Twitter, we also have access to the corresponding Twitter profiles, which typically include additional information such as number of followers and "followings", etc. In this paper, due to the limited availability of user profile data in the collected dataset, we define two attributes as private, i.e, gender (male/female) and social status [28] (a yardstick to measure the popularity of a user in social network). For a user u, social status is computed as the ratio of the number of u's followers to the number of users u follows (i.e., "followings"): $social(u) = \frac{\#followers(u)}{\#followings(u)}$. We also discretize u' social status as popular ($social(u) > 1$) and non-popular ($social(u) \leq 1$). We note that our framework is not limited to these two types of private data, and it can incorporate any categorical attributes as private data.

7.1.2 Evaluation Use Cases and Metrics

Privacy evaluation is traditionally based on simulations, and tries to show that the defined privacy is satisfied with a reasonable computation overhead [29]. In this paper, we take a step forward to quantitatively evaluate both our

1. https://dev.twitter.com/streaming/public

TABLE 1
Characteristics of the experimental dataset

Dataset	New York City	Tokyo
User number	3,669	6,870
POI number	1,861	2,811
Check-in number	893,722	1,290,445

privacy protection and data utility. Specifically, we implement two inference attack methods to directly assess the performance of our privacy protection and use two real-world ranking-based recommendation use cases to evaluate the resulting utility of the obfuscated data.

Privacy. Inference attacks [3] on private data try to infer a user's private information Y (e.g., gender) from her released public data $\hat{X}$, which can be regarded as a classification problem for discrete data. Therefore, we adopt here two common classification algorithms as inference attack methods, namely Support Vector Machine (SVM) and Naive Bayes (NB). We assume that adversaries have trained their classifiers based on the original public data X and private data Y from some non privacy-conscious users [13], who do not care about their privacy and publish all their data. We randomly sample 50% of all users as such non privacy-conscious users for training the classifiers, and then perform inference attacks on the private data Y of the rest of the users based on their obfuscated activity data $\hat{X}$. We use the *Area Under the Curve (AUC)*, which is a widely used metric for classification problem [30], to evaluate the performance of the inference attacks. We report the value (1-AUC) as a privacy protection metric in the experiments. *Higher value of (1-AUC) implies better privacy protection*. The ideal privacy protection is achieved when $AUC = 0.5$ (i.e., $1 - AUC = 0.5$), which implies that any inference attack method performs no better than a random guess.

Utility. In this work, utility refers to the *ranking-based recommendation performance*. We select two typical use cases, i.e., POI recommendation [23] and context-aware activity recommendation [24], as our target scenarios.

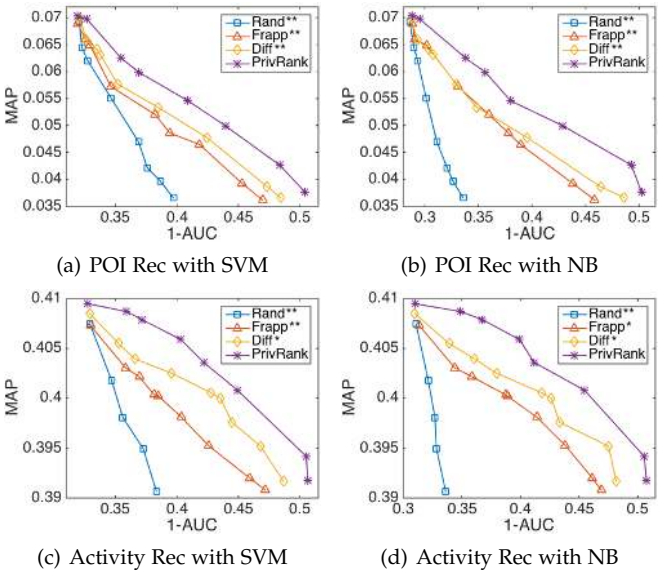
- ***POI Rec.*** POI recommendation [23] tries to recommend to a user a list of POIs that she would be interested in. To implement this use case, we first consider the cumulative check-in number of a user on a POI as the rating (i.e., the cumulative number of interactions as preference score) to build a user-POI matrix, and then leverage a Bayesian personalized ranking algorithm [31] to predict the ranked list. Note that the POI Rec is a common use case for user-item recommendation.
- ***Activity Rec.*** Context-aware activity recommendation [24] tries to come up with a list of activities (represented by POI categories, e.g., restaurant or bar) that a given user may be interested in based on her current context (i.e., location and time). We first discretize the context (i.e., time slots and location grid cells) of check-in data to build a user-context-activity tensor using the 0/1-based scheme (i.e., binary format of preference score), and then leverage a ranking tensor factorization algorithm [32] for ranking prediction.

For both use cases, we first randomly split the original public data X into a training dataset X_{train} (80%) and a test set X_{test} (20%), and then use our framework to obfuscate X_{train} into $\hat{X}_{train}$. Subsequently, we apply the recommendation algorithms on the obfuscated data $\hat{X}_{train}$, and then make predictions for the test dataset X_{test}, which represents the users' true preference. *Our goal is to verify that the obfuscated data $\hat{X}_{train}$ can still be used to accurately predict the users' true preference in X_{test}.* To evaluate the quality of the resulting recommendations, we use *Mean Average Precision (MAP)* [18], which is a widely used metric in information retrieval to assess the quality of rankings. *Higher value of MAP implies better performance.* Each reported result is the mean value of ten repeated trials.

7.1.3 Baseline Approaches

In order to demonstrate the effectiveness of our framework, we compare it with the following baselines:

- Random obfuscation (*Rand*). For historical data obfuscation, it randomly obfuscates each user public data vector V^u to another $V^{u'}$ with a given probability p_{rand}. For online activity obfuscation, it randomly obfuscates each user activity on i to another item i' with probability p_{rand}. Here, p_{rand} controls the distortion budget in both cases.
- *Frapp* [33]. It is a generalized matrix-theoretic framework of data perturbation for privacy-preserving mining. Its key idea is to obfuscate one's activity data to itself with higher probability than to others. For historical data obfuscation, it obfuscates a user u's public data vector V^u to $V^{u'}$ with probability $p_{frapp} = \gamma e$ if $u = u'$, otherwise $p_{frapp} = e$. Here e is used for probability normalization, i.e., $e = \frac{1}{\gamma + |\mathcal{U}| - 1}$. For online data obfuscation, it obfuscates each activity on item i to another item i' with probability $p_{frapp} = \gamma e$ if $i = i'$, otherwise $p_{frapp} = e$ (here $e = \frac{1}{\gamma + |\mathcal{I}| - 1}$). The distortion budget is controlled by γ in both cases.
- Differential privacy (*Diff*) [12] is a state-of-the-art method to protect privacy regardless of the adversary's prior knowledge. It can be implemented for different types of data, such as numeric data [34], categorical data [35], set-valued data [36] or location data [37]. Here we adopt exponential mechanism [35], [38], [39] in our experiments as it fits our use case of categorical data. More importantly, it is straightforward to be implemented for online data obfuscation, which we explicitly consider in PrivRank. We exclude other sophisticated differential privacy methods for categorical data (such as [40]), as they do not handle online data obfuscation. We implement exponential mechanism as follows. For historical data obfuscation, it obfuscates V^u to $V^{u'}$ with a probability that decreases exponentially with the distance $d(V^u, V^{u'})$, i.e., $p_{diff}(V^{u'}|V^u) \propto \exp(-\beta d(V^u, V^{u'}))$, where $\beta \geq 0$. β actually controls the distortion budget. The exponential mechanism satisfies $2\beta d_{max}$-differential privacy, where $d_{max} = \max_{u,u' \in \mathcal{U}} d(V^u, V^{u'})$. For online activity obfuscation, this method obfuscates each activity of user u on item i with $p_{diff}(i'|i) \propto \exp(-\beta d(V^u + i', V^u + i))$. Here we also use the Kendall-τ distance for $d(\cdot, \cdot)$. The distortion budget is controlled by β in both cases.



(a) POI Rec with SVM (b) POI Rec with NB

(c) Activity Rec with SVM (d) Activity Rec with NB

Fig. 6. Privacy-Utility trade-off with different methods on the NYC dataset (All baselines are significantly outperformed by PrivRank at the **0.01 or *0.05 level (p-value) with paired t-test.)

7.2 Privacy & Utility Trade-off

In this experiment, we vary the parameters that control the distortion budget for our method and baselines, and observe the resulting trade-off between privacy (AUC) and utility (MAP). As our framework and baselines use different parameters to control the distortion budget, they are not directly comparable. Therefore, we tune the obfuscation budgets in different methods to directly show the privacy-utility trade-off, which is more informative for comparison. We consider gender as private data in all experiments except explicitly mentioned otherwise. For the sake of runtime efficiency, we empirically select the number of user clusters $|\mathcal{G}|$ and the number of pair samples $|S|$ when approximating the kendall-τ distance for each use case: $|\mathcal{G}| = 200$, $S = 10^4$ for POI Rec and $|\mathcal{G}| = 200$, $S = 10^3$ for Activity Rec. The selection of these parameters is discussed below in the experiment for runtime performance in Section 7.7.

Figure 6 shows the privacy-utility trade-off results for different privacy-preserving data obfuscation methods on the NYC dataset. First, we observe clearly the trade-off between privacy protection and utility of enabling ranking-based recommendation for all methods. On one hand, a better privacy protection can be achieved with a higher distortion budget, as highly distorted public data makes it harder for adversaries to infer user private data. On the other hand, higher distortion budgets incur a higher loss of data utility, as highly distorted public data also prevent recommendation algorithms from accurately predicting user preferences. Second, we observe that compared to other methods, PrivRank consistently achieves better privacy protection and higher utility at the same time (i.e., the resulting data points are closer to the upper-right corner of the plot) in all cases. We also conduct a paired t-test between each baseline method and PrivRank, and find that PrivRank significantly outperform all baselines at either 0.01 or 0.05 level (p-value). Due to the space limitation, we only show results obtained for the NYC dataset in all the experiments (results for the Tokyo dataset are similar).

7.3 Privacy Protection over User Activity Streams

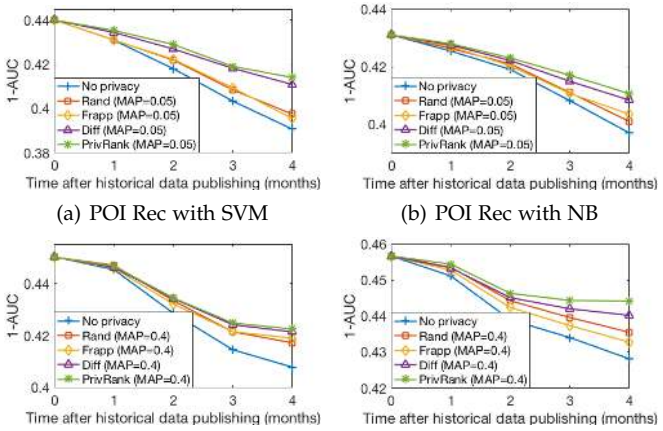
To study the privacy protection performance over time, we first obfuscate the historical data using our method, and then compare different online obfuscation methods for the future activity streams. Specifically, we select the first 14 month data as historical data, and the last 4 month data as activity streams. Such a setting corresponds to the case that a user subscribes to a third-party service at the end of the 14th month, where the service can access her activity stream from that time. To obfuscate the historical data, we set the distortion budget of our method to 0.2, leading to a privacy protection performance of 1-AUC=0.44 (with SVM) and a utility of MAP=0.05 for POI Rec, and 1-AUC=0.45 (with SVM) and MAP=0.4 for Activity Rec. Based on the obfuscated historical data, we now focus on how different online data obfuscation methods perform over time.

First, as our method and baselines use different parameters to control the distortion budget for online data obfuscation, we tune these parameters to *maintain the same level of data utility* (i.e., MAP=0.05 for POI Rec and MAP=0.4 for Activity Rec), and shows the privacy protection performance over time. Figure 7 shows the results on the NYC dataset. We observe that although historical data obfuscation can effectively protect user private data (at time 0), the privacy protection performance rapidly decreases over time if attackers observe the actual user activity streams (shown as *No privacy* in Figure 7). To provide continuous privacy protection, online data obfuscation can actually alleviate this problem to some extent, as we observe a slower decrease of the privacy protection performance over time for all data obfuscation methods. More importantly, our method outperforms all baselines by achieving the highest values of 1-AUC over time in all cases. However, all the methods still show decreasing privacy protection performance over time.

Second, we *maintain the same level of privacy protection for our online data obfuscation method* by increasing the distortion budget, and obtain PrivRank (MAP=0.043) for POI Rec and PrivRank (MAP=0.36) for Activity Rec (as shown in Figure 8). Such an observation shows that for the same level of privacy protection, online data obfuscation needs to sacrifice more utility (caused by higher data distortion budget) than historical data obfuscation. This is due to the real-time data publishing constraint, i.e., online obfuscation can only be done based on the incoming activity data instance itself without an overview of users' whole public data history. Meanwhile, PrivRank still outperforms all baseline methods by achieving the best privacy protection performance over time under the same data utility guarantee.

7.4 Performance of Customized Privacy Protection

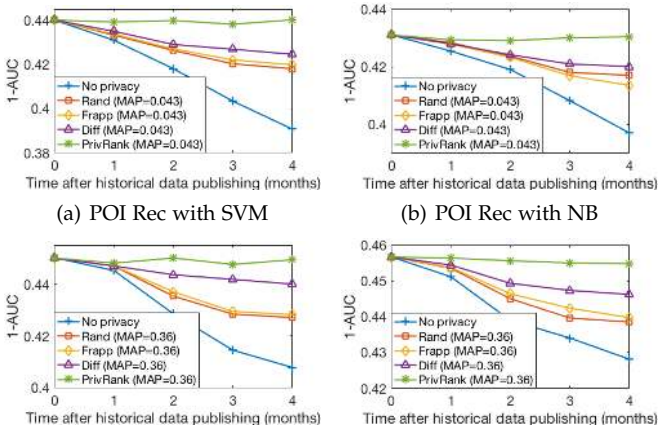
Since users often have different privacy requirements, our framework is designed to protect user-specific private data. In this experiment, we consider aforementioned two types of private data, i.e., gender and social status. Specifically, users have three options according to the types of private data to be protected: 1) protecting gender only (PrivRank-Gender); 2) protecting social status only (PrivRank-Social); 3) protecting both gender and social status (PrivRank-Both).



(a) POI Rec with SVM (b) POI Rec with NB

(c) Activity Rec with SVM (d) Activity Rec with NB

Fig. 7. Privacy protection performance over time on NYC dataset (maintaining the same level of data utility)



(a) POI Rec with SVM (b) POI Rec with NB

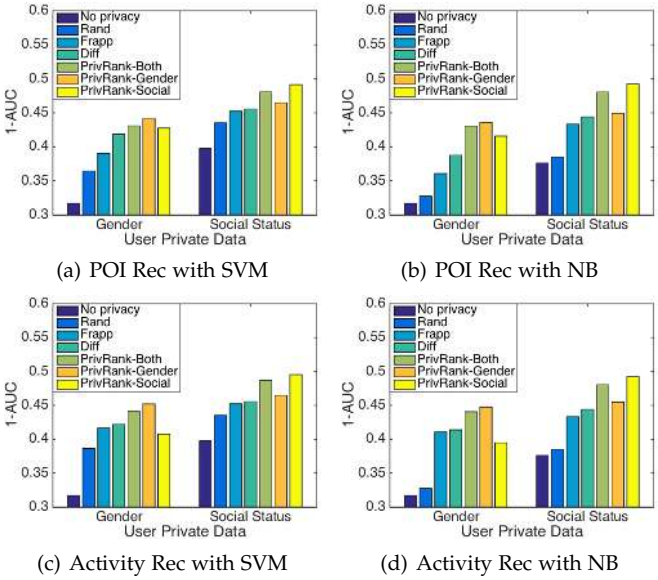
(c) Activity Rec with SVM (d) Activity Rec with NB

Fig. 8. Privacy protection performance over time on NYC dataset (maintaining the same level of privacy protection for PrivRank)

We configure our framework with those three settings, and report on the customized privacy protection performance. We tune the distortion budget for all the data obfuscation methods to keep the same data utility, i.e., MAP=0.05 for POI Rec and MAP=0.4 for Activity Rec.

Figure 9 shows the privacy protection results for both gender and social status on the NYC dataset. We observe that PrivRank-Gender (or PrivRank-Social) outperforms all other methods when protecting the targeted gender data (or social status), by achieving the highest values of 1-AUC. Particularly, compared to PrivRank-Both, which treats both data as private, PrivRank-Gender (or PrivRank-Social) can provide better privacy protection on gender (or social status). In other words, better privacy protection can be achieved under the same data utility guarantee when less private data has to be protected.

In addition, we observe that different types of private data suffer from different levels of privacy leakage. For example, Figure 9 shows that a user's gender can be inferred more accurately than that of her social status. In practice, this observation can be used to help users decide which private data should be protected, by providing them with a quantitative metric based on AUC to indicate the privacy



(a) POI Rec with SVM (b) POI Rec with NB

(c) Activity Rec with SVM (d) Activity Rec with NB

Fig. 9. Customization performance of privacy protection

leakage of all potential private data.

7.5 Utility with Different Loss Metrics

In this experiment, we study the privacy-utility trade-off using our method with different loss metrics, including Euclidean distance, squared L^2 distance, cosine distance, Jensen-Shannon distance (JSD), and two ranking based metrics, i.e., Spearman correlation and Kendall-τ distance. We keep other parameters the same as in previous experiments.

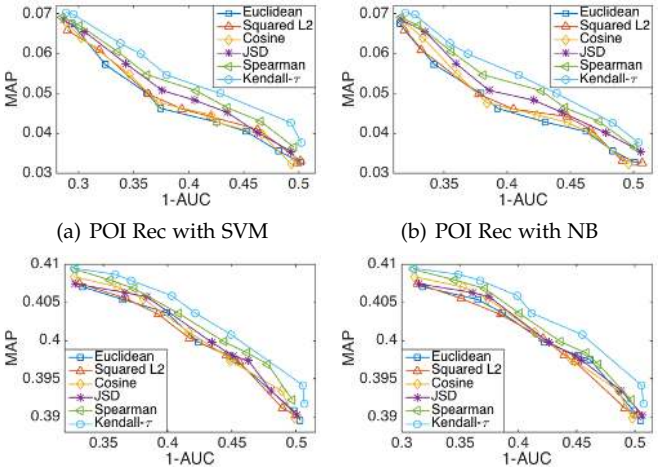
Figure 10 shows the privacy-utility trade-off with different loss metrics. First, we observe that ranking-based loss metrics outperform non-ranking based metrics by achieving better privacy protection and utility at the same time. In other words, bounding the ranking loss incurred from data obfuscation can better preserve the ranking relations in the public data, which leads to smaller utility loss in learning-to-rank algorithms. Moreover, our proposed method using Kendall-τ distance achieves the best results in all cases, as most of the learning-to-rank algorithms actually rely on pairwise/listwise ranking relations in the training data [16], which are optimally preserved by our method.

Compared to our previous work [2] that uses Jensen-Shannon distance, PrivRank can effectively improve the utility of ranking-based recommendation under the same level of privacy protection. For example, for a given privacy protection 1-AUC=0.4 for the use case of POI Rec, PrivRank shows an improvement of 9% and 12% in MAP under the attack methods SVM and NB, respectively.

7.6 Impact of Private Data Setting

In this experiment, we study the impact of different settings for private data, i.e., the number of private attributes and the size of their domain. As our real-world Foursquare dataset has a very limited number of private attributes (only two), here we use synthetic datasets generated by the IBM synthetic data generator for itemsets[2], which is originally designed for frequent itemset mining. We generate

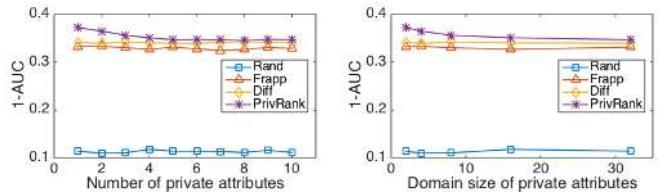
2. https://github.com/zakimjz/IBMGenerator



(a) POI Rec with SVM (b) POI Rec with NB

(c) Activity Rec with SVM (d) Activity Rec with NB

Fig. 10. Privacy-Utility trade-off with different loss metrics for data obfuscation (NYC dataset)



(a) Number of private attributes (b) Private attribute domain size

Fig. 11. Impact of different settings for private data

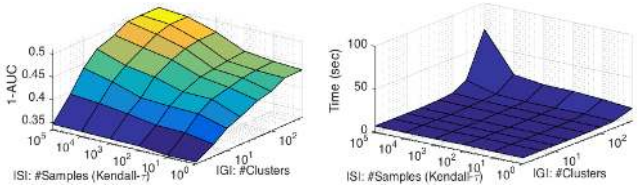
50K transactions with 100 items per transaction on average (1K different items in total), and keep other parameters as default. We regard each transaction as a user, and randomly sample m items as private attributes while regarding the rest as public data. The utility is computed using the POI Rec task, where we now want to recommend items to users. We tune the parameters for each method to let it has the same utility MAP=0.16.

Figure 11(a) shows the average (1-AUC) of all private attributes w.r.t. the number of private attributes. Here we set the domain size of private attributes to two. We observe that PrivRank outperforms all baselines, while its performance slightly decreases with the increasing number of private attributes. With a small number of private attributes, PrivRank is able to put more focus on protecting the specified data by minimizing its privacy leakage from public data, and thus provides better customized privacy protection.

Figure 11(b) shows the impact of the domain size for one private attribute. We observe that PrivRank still outperforms all baselines. Similar to the impact of the number of private attributes, the performance of PrivRank also slightly decreases with increasing domain sizes, because a larger domain size implies less obfuscation budget on each value for the private attribute.

7.7 Runtime Performance

As our framework includes both a historical and an online data publishing modules, we separately discuss their runtime performance. The prototype of our framework is implemented on a commodity PC (Intel Core i7-4770HQ@2.20GHz, 16GB RAM, OS X), running MATLAB and CVX with MOSEK [21].



(a) Privacy protection (b) Total learning time

Fig. 12. Runtime and privacy performance for POI Rec with SVM

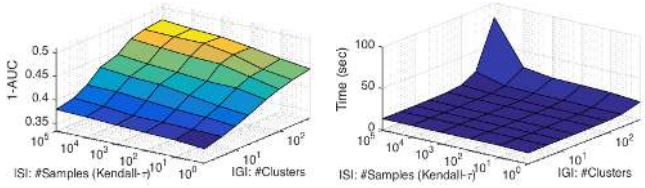
7.7.1 Historical Data Publishing

For learning obfuscation function, we adopt a user clustering step to reduce the problem complexity and a bootstrap sampling process for fast computation of Kendall-τ ranking loss. Subsequently, we study the total learning time of the obfuscation function w.r.t. the number of user clusters $|\mathcal{G}|$, the bootstrap sampling size $|S|$ and the number of users $|\mathcal{U}|$.

First, by varying $|\mathcal{G}|$ and $|S|$, we report privacy protection results (1-AUC) and the total learning time (including user clustering, Kendall-τ distance computation and obfuscation function learning), for the NYC dataset with POI Rec, the NYC dataset with activity Rec and the synthetic dataset, in Figure 12, 13 and 14, respectively. For the NYC dataset, we fix the utility of MAP=0.05 for POI Rec and MAP=0.4 for Activity Rec, and consider gender as private data. For synthetic dataset, we fix the utility of MAP=0.16, and randomly sample one item as private data. On one hand, we observe that the privacy protection performance improves when increasing $|\mathcal{G}|$ and $|S|$. Specifically, larger $|\mathcal{G}|$ can capture finer-grained user groups, which allow our method to find the optimal function that achieves a better privacy protection under the same distortion budget. Meanwhile, a larger bootstrap sampling size $|S|$ can approximate the actual Kendall-τ distance more accurately, which can better measure the ranking loss incurred by the optimal obfuscation function. In addition, we also find that 1-AUC converges after a certain point. On the other hand, the total learning time continuously increases when increasing $|\mathcal{G}|$ and $|S|$. Therefore, the two parameters $|\mathcal{G}|$ and $|S|$ can be selected at the convergence point for 1-AUC.

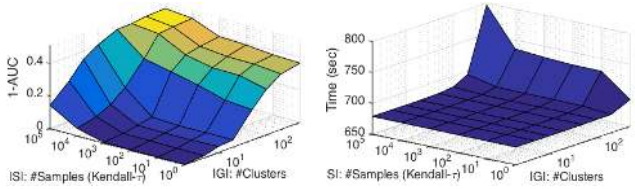
Second, by fixing $|\mathcal{G}|$ and $|S|$ to the convergence point for 1-AUC, we vary the number of users $|\mathcal{U}|$ in synthetic datasets and report the learning time in Figure 15(a). We observe that only user clustering time linearly increases with $|\mathcal{U}|$, while the Kendall-τ distance computation and obfuscation function learning time are independent from $|\mathcal{U}|$. PrivRank can easily scale up to 100K users (taking 3,378 seconds on our test PC). Note that learning the optimal obfuscation function is an offline step, and only depends on the joint probability $p_{G,Y}$. In practice, we can regularly update the obfuscation function to sustain its effectiveness.

Based on the learned obfuscation function, the probabilistic data obfuscation in Algorithm 2 can be performed very efficiently (i.e., 3.3ms per user in all cases).



(a) Privacy protection (b) Total learning time

Fig. 13. Runtime and privacy performance for Activity Rec with SVM



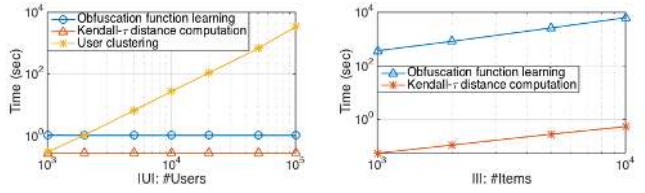
(a) Privacy protection (b) Total learning time

Fig. 14. Runtime and privacy performance on synthetic dataset

7.7.2 Online Data Publishing

The complexity of learning obfuscation function for online data publishing (Algorithm 3) depends only on the number of items $|\mathcal{I}|$. For the synthetic dataset, we vary the number of items and show the its impact on the runtime performance in Figure 15(b). We observe that both Kendall-τ distance computation time and obfuscation function learning time increases with the number of items. PrivRank can easily scale up to a large dataset with 10K items (taking 6,291 seconds on our test PC). For the Foursquare datasets, we keep the same parameters as in previous experiments and report the runtime performance for both NYC and TKY datasets in Table 2. Our test PC is able to learn the optimal obfuscation function in a reasonable time in all cases. In addition, although learning the personalized obfuscation function needs to be performed for each individual for online data publishing, this offline step can be easily parallelized w.r.t. the number of users, as one user's obfuscation function learning process is independent from the others.

Due to the streaming nature of user activity data, the efficiency of probabilistic online data obfuscation is particularly important. Our method (Algorithm 4) is able to perform the obfuscation process with a high speed of 2,200 activity instance per second on all datasets, which can easily accommodate user activity streams from most social media platform. For example, the Foursquare check-in stream has the peak-day record showing 8 million check-ins/day



(a) Impact of $|\mathcal{U}|$ on historical data publishing (b) Impact of $|\mathcal{I}|$ on online data publishing

Fig. 15. Impact of $|\mathcal{U}|$ and $|\mathcal{I}|$ on the scalability

TABLE 2
Runtime performance for online data publishing

Dataset (Utility)	NYC (POI Rec)	NYC (Activity Rec)	TKY (POI Rec)	TKY (Activity Rec)
Obfuscation function learning	662 sec	120 sec	1,438 sec	276 sec

(about 92 check-ins/sec on average) in 2016[3].

8 Conclusions and Future Work

This paper introduced PrivRank, a customizable and continuous privacy-preserving social media data publishing framework. It continuously protects user-specified data against inference attacks by releasing obfuscated user activity data, while still ensuring the utility of the released data to power personalized ranking-based recommendations. To provide customized protection, the optimal data obfuscation is learned such that the privacy leakage of user-specified private data is minimized; to provide continuous privacy protection, we consider both the historical and online activity data publishing; to ensure the data utility for enabling ranking-based recommendation, we bound the ranking loss incurred from the data obfuscation process using the Kendall-τ rank distance. We showed through extensive experiments that PrivRank can provide an efficient and effective protection of private data, while still preserving the utility of the published data for different ranking-based recommendation use cases.

In the future, we plan to extend our framework by considering the data types with continuous values rather than discretized values, and explore further data utility beyond personalized recommendation.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement 683253/GraphInt).

References

[1] S. Salamatian, A. Zhang, F. du Pin Calmon, S. Bhamidipati, N. Fawaz, B. Kveton, P. Oliveira, and N. Taft, "How to hide the elephant-or the donkey-in the room: Practical privacy against statistical inference for large data," in *Proc. of GlobalSIP*. IEEE, 2013.

[2] D. Yang, D. Zhang, Q. Bingqing, and P. Cudre-Mauroux, "Privcheck: Privacy-preserving check-in data publishing for personalized location based services," in *Proc. of UbiComp'16*. ACM, 2016.

[3] C. Li, H. Shirani-Mehr, and X. Yang, "Protecting individual information against inference attacks in data publishing," in *Advances in Databases: Concepts, Systems and Applications*. Springer, 2007, pp. 422–433.

[4] B. Fung, K. Wang, R. Chen, and P. S. Yu, "Privacy-preserving data publishing: A survey of recent developments," *ACM Computer Survey*, vol. 42, no. 4, p. 14, 2010.

3. http://blog.foursquare.com/post/142900756695/since-foursquare-launched-in-2009-there-have-been

[5] I. A. Junglas, N. A. Johnson, and C. Spitzmüller, "Personality traits and concern for privacy: an empirical study in the context of location-based services," *European Journal of Information Systems*, vol. 17, no. 4, pp. 387–402, 2008.

[6] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proc. of RecSys'10*. ACM, 2010, pp. 39–46.

[7] N. Li, R. Jin, and Z.-H. Zhou, "Top rank optimization in linear time," in *Advances in neural information processing systems*, 2014, pp. 1502–1510.

[8] M. G. Kendall, "Rank correlation methods." 1948.

[9] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[10] L. Sankar, S. R. Rajagopalan, and H. V. Poor, "Utility-privacy tradeoffs in databases: An information-theoretic approach," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 6, pp. 838–852, 2013.

[11] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, p. 3, 2007.

[12] C. Dwork, "Differential privacy," in *Automata, languages and programming*. Springer, 2006, pp. 1–12.

[13] F. du Pin Calmon and N. Fawaz, "Privacy against statistical inference," in *Proc. of Allerton'12*. IEEE, 2012, pp. 1401–1408.

[14] A. Zhang, S. Bhamidipati, N. Fawaz, and B. Kveton, "Priview: Media consumption and recommendation meet privacy against inference attacks," *IEEE Web*, vol. 2, 2014.

[15] S. Salamatian, A. Zhang, F. du Pin Calmon, S. Bhamidipati, N. Fawaz, B. Kveton, P. Oliveira, and N. Taft, "Managing your private and public data: Bringing down inference attacks against your privacy," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 7, pp. 1240–1255, 2015.

[16] W. Chen, T.-Y. Liu, Y. Lan, Z.-M. Ma, and H. Li, "Ranking measures and loss functions in learning to rank," in *Proc. of NIPS*, 2009, pp. 315–323.

[17] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *PNAS*, vol. 95, no. 25, pp. 14 863–14 868, 1998.

[18] R. Baeza-Yates, B. Ribeiro-Neto *et al.*, *Modern information retrieval*. ACM press New York, 1999, vol. 463.

[19] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.

[20] B. Efron and R. J. Tibshirani, *An introduction to the bootstrap*. CRC press, 1994.

[21] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*. Springer, 2008, pp. 95–110.

[22] G. S. Manku, S. Rajagopalan, and B. G. Lindsay, "Approximate medians and other quantiles in one pass and with limited memory," in *ACM SIGMOD Record*, vol. 27, no. 2. ACM, 1998, pp. 426–435.

[23] D. Yang, D. Zhang, Z. Yu, and Z. Wang, "A sentiment-enhanced personalized location recommendation system," in *Proc. of HT'13*. ACM, 2013, pp. 119–128.

[24] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns," *IEEE Transactions on System, Man, Cybernetics: System*, vol. 45, no. 1, pp. 129–142, 2015.

[25] Z. Yu, H. Xu, Z. Yang, and B. Guo, "Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 151–158, 2016.

[26] D. Yang, D. Zhang, L. Chen, and B. Qu, "Nationtelescope: Monitoring and visualizing large-scale collective behavior in lbsns," *Journal of Network and Computer Applications*, vol. 55, pp. 170–180, 2015.

[27] D. Yang, D. Zhang, and B. Qu, "Participatory cultural mapping based on collective behavior data in location-based social networks," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 7, no. 3, p. 30, 2016.

[28] Z. Cheng, J. Caverlee, K. Lee, and D. Z. Sui, "Exploring millions of footprints in location sharing services." *Proc. of ICWSM'11*, vol. 2011, pp. 81–88, 2011.

[29] X. Zhao, L. Li, and G. Xue, "Checking in without worries: Location privacy in location based social networks," in *Proc. of INFOCOM'13*. IEEE, 2013, pp. 3003–3011.

[30] C. X. Ling, J. Huang, and H. Zhang, "Auc: a better measure than accuracy in comparing learning algorithms," in *Advances in Artificial Intelligence*. Springer, 2003, pp. 329–341.

[31] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proc. of UAI'09*. AUAI Press, 2009, pp. 452–461.

[32] D. Yang, D. Zhang, Z. Yu, and Z. Yu, "Fine-grained preference-aware location search leveraging crowdsourced digital footprints from lbsns," in *Proc. of UbiComp'13*. ACM, 2013, pp. 479–488.

[33] S. Agrawal and J. R. Haritsa, "A framework for high-accuracy privacy-preserving mining," in *Proc. of the ICDE'05*. IEEE, 2005, pp. 193–204.

[34] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*. Springer, 2006, pp. 265–284.

[35] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proc. of FOCS'07*. IEEE, 2007, pp. 94–103.

[36] R. Chen, N. Mohammed, B. C. Fung, B. C. Desai, and L. Xiong, "Publishing set-valued data via differential privacy," *PVLDB*, vol. 4, no. 11, pp. 1087–1098, 2011.

[37] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," in *Proceedings of the 26th International Conference on World Wide Web*. ACM, 2017, pp. 627–636.

[38] C. Dwork, "Differential privacy: A survey of results," in *Proc. of TAMC*. Springer, 2008, pp. 1–19.

[39] Z. Huang and S. Kannan, "The exponential mechanism for social welfare: Private, truthful, and nearly optimal," in *Proc. of FOCS'12*, 2012.

[40] Y. Shen and H. Jin, "Privacy-preserving personalized recommendation: An instance-based approach via differential privacy," in *Proc. of ICDM*. IEEE, 2014, pp. 540–549.

Dingqi Yang is a senior researcher in the Department of Computer Science, University of Fribourg, Switzerland. He received his Ph.D. in Computer Science from Pierre and Marie Curie University (Paris VI) and Institut Mines-TELECOM/TELECOM SudParis in 2015, where he won both the Doctorate Award and the Institut Mines-TELECOM Press Mention. His research interests lie in big social media data analytics, ubiquitous computing and smart city applications.

Bingqing Qu is a post-doc researcher in the Department of Computer Science, University of Fribourg, Switzerland. She received her Ph.D. in Computer Science in University of Rennes 1 in 2016. Her research interests include historical document analysis, multimedia content analysis, social media data mining and computer vision.

Philippe Cudre-Mauroux is a Full Professor and the director of the eXascale Infolab at the University of Fribourg in Switzerland. He received his Ph.D. from the Swiss Federal Institute of Technology EPFL, where he won both the Doctorate Award and the EPFL Press Mention. Before joining the University of Fribourg he worked on information management infrastructures for IBM Watson Research, Microsoft Research Asia, and MIT. His research interests are in next-generation, Big Data management infrastructures for non-relational data. Webpage: http://exascale.info/phil