Differentially-Private Release of Check-in Data for Venue Recommendation

Daniele Riboni Claudio Bettini
Università degli Studi di Milano, Dept. of Computer Science, EveryWare Lab
via Comelico, 39, I-20135 Milano, Italy
Email: {daniele.riboni,claudio.bettini}@unimi.it

***Abstract*—Recommender systems suggesting venues offer very useful services to people on the move and a great business opportunity for advertisers. These systems suggest venues by matching the current context of the user with the venue features, and consider the popularity of venues, based on the number of visits (*"check-ins"*) that they received. Check-ins may be explicitly communicated by users to geo-social networks, or implicitly derived by analysing location data collected by mobile services. In general, the visibility of explicit check-ins is limited to friends in the social network, while the visibility of implicit check-ins limited to the service provider. Exposing check-ins to unauthorized users is a privacy threat since recurring presence in given locations may reveal political opinions, religious beliefs, or sexual orientation, as well as absence from other locations where the user is supposed to be. Hence, on one side mobile app providers host valuable information that recommender system providers would like to buy and use to improve their systems, and on the other we recognize serious privacy issues in releasing that information. In this paper, we solve this dilemma by providing formal privacy guarantees to users and trusted mobile providers while preserving the utility of check-in information for recommendation purposes. Our technique is based on the use of differential privacy methods integrated with a pre-filtering process, and protects against both an untrusted recommender system and its users, willing to infer the venues and sensitive locations visited by other users. Extensive experiments with a large dataset of real users' check-ins show the effectiveness of our methods.**

I. Introduction

Many mobile and pervasive applications acquire the user's current position and often monitor the user's movement patterns as one of the context parameters used for personalization. As a result, a large amount of location data is stored server-side and can be easily analyzed to identify for each user the recurrent visits to specific venues as well as venues where the user spends significant time. Some of these implicitly revealed venues are sometimes explicitly reported by users when engaged in so called *geo-social networks* like Foursquare and Facebook Places in the form of *check-ins*. In this case venues are usually shops, restaurants, pubs, tourist attractions, or just locations the users want to remember or share with friends. In this paper we use the general term *check-in* referring to both explicitly and implicitly revealed user preferred venues.

Check-ins are extremely valuable information for mobile recommender systems; these systems suggest venues by matching the current context of the mobile user (including e.g., location, time, interests) with the venue features, considering the number of check-ins as an estimate of its popularity. Since these systems have profitable business models, we are experiencing the flourishing of a new market for location data that makes large datasets of user implicit and explicit check-ins a valuable asset.

On the other hand, a check-in or a pattern of check-ins may directly or indirectly unveil private information, and for this reason their visibility can be usually limited to a restricted group of users; for example, this is the case when posting a check-in to a circle of friends in a geo-social network. In some cases they are just kept private to the user and to the mobile service provider; this is the case of many location-based client-server mobile and pervasive applications. Exposing check-ins to unauthorized users is a privacy threat since recurring presence in given locations may reveal political opinions, religious beliefs, sexual orientation, or even health problems. The presence in a venue at a given time also discloses absence from other venues, a privacy threat known as *absence privacy*. A detailed discussion of privacy threats in location sharing applications is reported in [20].

The problem. In this paper, we consider the technical problem of exchanging check-in data without violating the user privacy. To illustrate our solution we focus on a scenario in which a trusted location data collector (geo-social network service or LBS provider) would like to release its users' check-in data to a third party mobile recommender system that has not been authorized by users to see their data. This scenario poses two privacy threats: a) the untrusted recommender system may violate users' privacy by directly inspecting check-in data, and b) its users may be able to craft personalized queries and reconstruct the venues visited by a target user by mining the received recommendations, as it has been shown in the literature [7], [15].

As an example for threat a), suppose Bob regularly uses a check-in service and that the service provider decides to sell its users' check-in counts (i.e., the number of people that checked-in at each venue) collected in the last year to Eve, a third-party that runs a recommender system. Eve knows that Bob is a user of the check-in service and that Bob regularly visits the ill-famed Helltown but would like to know which venues he actually visits. Eve may discover one of these venues, by colluding with the other visitors of the venue and simply subtracting the number of their check-ins from the check-in counts for that place.

A solution to these threats is to appropriately perturb check-in statistics before releasing them. However, in the case of counting, adding/subtracting a fixed number j of check-ins to/from the count of each venue would not defend against the above threat, if the recommender system gets to know j. More

generally, the problem is how to efficiently perturb check-in statistics preserving data utility and providing formal privacy guarantees.

Assumptions, and proposed solution. For the sake of this work, we assume that each user agrees to reveal his location traces and explicit check-ins to the check-in collector as long as they are not revealed to unauthorized third parties. This is equivalent to assume that the collector is trusted and the adversary for a user A is any user not authorized by A to see her check-ins. We assume that the adversary may have external knowledge about one or more target users, and use it to discover check-ins by inspecting check-in data released by the collector, or crafting queries to the recommender system.

Our defense is based on a novel application of differential privacy methods to extract obfuscated check-in statistics. These statistics are shown to be still very useful to generate the recommendations based on context data, while providing formal privacy guarantees. Differential privacy [12] is enforced by a non trivial combination of pruning and noise addition. In more detail, the set of check-ins is pruned based on the venues' location and user location privacy preferences, in order to reduce the magnitude of noise to be added.

Referring to the example, with our defense Bob can still check-in at his preferred venues, and share that information with his close friends and with the trusted check-in collector. The collector will still be able to make a profit by selling to a third party (e.g., the recommender system) useful check-in statistics, but our defense will guarantee that from those statistics it is very unlikely to be able to derive the presence of Bob in any of Helltown's venues.

Contributions. The contributions of this paper can be summarized as follows.

- We propose a novel technique for privacy-aware release of check-in datasets as used in knowledge-based mobile recommender systems. To our knowledge this is the first work that provides a solution for the effective application of differential privacy to this category of systems. Our technique provides strong privacy guarantees based on user preferences, irrespective of the background knowledge available to adversaries.
- The algorithm implementing our proposed defense is computationally efficient, and may be easily extended to support incremental release of check-in statistics.
- We perform an extensive experimental evaluation of our defense technique with a large dataset of real users' check-ins. Results show that our technique provides a good trade-off between privacy and data utility.

The rest of the paper is structured as follows. In Section II we discuss related work. In Section III we present the problem definition and the adversary model. Section IV illustrates our defense and the achieved privacy guarantees. The algorithms to apply our defense are presented in Section V. The experimental evaluation is reported in Section VI. Section VII concludes the paper.

II. Preliminaries and related work

We first briefly review the related research on mobile recommender systems, and then focus on the investigations that have been conducted to address the specific privacy issues arising from the use of these systems. Finally, we report the basic notions of differential privacy, that will be used in the rest of the paper.

A. Mobile recommender systems

Many techniques to provide recommendations of items of interest have been proposed in the literature, and recommender systems are more and more popular in online selling services. A widely adopted classification [6] divides recommender systems in different categories based on the method adopted to provide recommendations. *Collaborative filtering* methods recommend items that "similar" users liked in the past, where the similarity among users is computed based on their ratings for items in common. These methods are less effective in the context of mobile recommendations. For instance, it is very likely that a tourist traveling to a foreign city for the first time has no preferences in common with people who expressed preferences for venues in that city. Our work applies to *Knowledge based* methods that rely on a utility function evaluating how close are the user's current needs with the item features. When applied to venue recommendation, these techniques may be used to recommend the "best" venues (those that received the largest number of preferences) among the ones that match the user's context, including location, time of the day, profile, categories of interest, etc.

Techniques to include context-aware features into recommender systems have been investigated, among others, by Adomavicius and Tuzhilin [1]. However, while those techniques take into account the user's context, they do not consider the items' location; hence, they cannot be seamlessly applied to localized resources like venues. For this reason, several recommender systems specifically targeted to mobile computing have been recently proposed [4], [11], [3], [13]. However, no technique to enforce users' privacy is proposed in those works.

B. Privacy and recommender systems

There have been many studies on preserving context and location privacy in location based systems by obfuscating, suppressing and/or anonymizing data at the time a query is issued [5]. Some of these techniques can also be applied when querying recommender systems. However, in this paper we are not addressing the protection of users making queries to a recommender system, but the protection of check-in data used to provide the recommendations. In principle, a privacy violation may occur by acquiring check-in data from other sources and by providing a set of recommendations revealing the check-in of an individual that is not a user of the recommender system. Our work is focused on how to use check-in data to provide venue recommendations without violating the privacy of the individuals associated with the check-ins.

The anonymization of check-ins through the use of pseudonyms or numeric codes does not provide any formal

privacy guarantees due to the well known possibility of re-identification based on location and/or on other re-identifying data combined with background knowledge. The most straightforward alternative would be to simply *suppress* those check-ins that are considered sensitive by the user. However, suppression would obviously have a negative impact on the quality of produced recommendations. Moreover, in many cases it may be very difficult for an individual to understand whether a check-in may reveal private information. For instance, a check-in may be insensitive when considered in isolation, but it may reveal private information when matched with other check-ins, or when joined with background knowledge about the subject.

Other *micro-data obfuscation* methods, like spatial or spatio-temporal cloaking, may not be applicable to check-ins at the time the check-in is posted by the user without compromising the service. For example, in a friend-finder service the user may want to provide precise and real-time information to a close friend. The methods may be applied by the check-in collector for off-line release of a check-in data set. However, spatially and temporally generalized information would lose most of its value. For example, counting of check-ins in a specific venue at a given time is very valuable for a recommendation system but will not be available if cloaking is applied. In our defense, instead of applying cloaking to single check-ins, we perturb check-in aggregated data such that the issuer of any of the check-in cannot be re-identified.

Other approaches for privacy preservation include the use of encryption and multi-party computation [8], [2]. These techniques are usually computationally expensive and we are not aware of any proposal for their application to the problem we are considering.

A technique for privacy-conscious recommendation of venues has been presented by Sato et al. [19]. Acquired user interests in terms of categories of visited venues are used to provide recommendations. In order not to release the interest of a specific individual, they propose a method to perturb the values in a users/interests matrix. However, the proposed method is not supported by formal privacy guarantees, especially in presence of external knowledge available to an adversary. On the contrary, our technique, being based on differential privacy [12], provides strong privacy guarantees even in presence of external knowledge. In the context of recommender systems based on collaborative filtering, McSherry [17] showed that they can be adapted to enforce differential privacy, without significantly degrading the quality of recommendations. However, as we explained before, collaborative filtering methods are not always well suited to the recommendation of venues. Hence, in this paper we tackle the application of differential privacy to a knowledge-based recommender system.

We presented preliminary ideas on applying differential privacy to this problem in [18], but we did not address location privacy, and we did not have an implementation nor experimental evaluation.

C. Differential privacy

Generally speaking, differential privacy guarantees that the probability distribution of the outcomes of a given computation is essentially the same, irrespective of whether or not a user's data is present in the knowledge base.

Formally, a randomized computation C satisfies ϵ-differential privacy if, for any possible datasets D_1 and D_2 that differ in at most one record, and any subset O of possible outcomes of C:

$$\mathbf{Pr}[(C(D_1) \in O)] \leq \exp(\epsilon) \times \mathbf{Pr}[(C(D_2) \in O)], \quad (1)$$

where **Pr**[] represents the probability distribution of outcomes. As a consequence, irrespective of the background knowledge available to an adversary, the inference about the presence of a single record is bounded by the factor $\exp(\epsilon)$. In the experimental evaluation section, we discuss which values of ϵ are considered safe in the literature.

The primary method to apply differential privacy to numerical data is to add Laplace noise with mean 0 and scale parameter $\frac{\Delta C}{\epsilon}$ [12]. The scale parameter determines the noise magnitude, and depends on the maximum contribution ΔC that a single record may have on the outcome of the computation: the higher the contribution, the larger the magnitude of noise to be added. That contribution is called *sensitivity*. Formally, the sensitivity of C is:

$$\Delta C = \max_{D_1,\, D_2} ||C(D_1) - C(D_2)||, \quad (2)$$

for all D_1, D_2 that differ in at most one record.

For instance, in the simple case in which each record is a check-in, and C computes the number of check-ins at a specific venue, the sensitivity of C is 1. However, as explained in Section IV, in this work the computation of sensitivity is more complicated, since each record represents the set of check-ins of a single user, and C computes a vector with the number of check-ins for each venue inside a specific region.

III. Problem definition and adversary model

The fact that a user checked-in at a given venue, or at any venue within a sensitive area, may reveal private information. For instance, visits to a gluten-free shop may be a clear indicator of celiac disease; frequent visits to any venue within a hospital area may indicate health issues. On the other hand, it could be argued that check-ins at some categories of venues (e.g., theaters) are not sensitive information, and can be freely released to the public. However, privacy is a subjective matter, and a universally accepted distinction between sensitive and insensitive venues is just not possible. For this reason, in this paper we make the conservative assumption that every check-in can be revealed only to trusted entities.

A. Reference architecture

The general architecture of the system is shown in Figure 1. Users may communicate their location traces and check-ins to a trusted service provider (collector), which stores them locally, and releases check-in statistics to an untrusted third-party. We point out that the architecture complies to a common business model. For instance, consider Foursquare, one of the most popular check-in apps, having more than 45M users worldwide, and collecting millions of check-ins every day[1]. Foursquare provides APIs for which any third-party can get statistics about the check-ins at each venue, such as the total number of check-ins, and the total number of people that

[1] https://foursquare.com/about

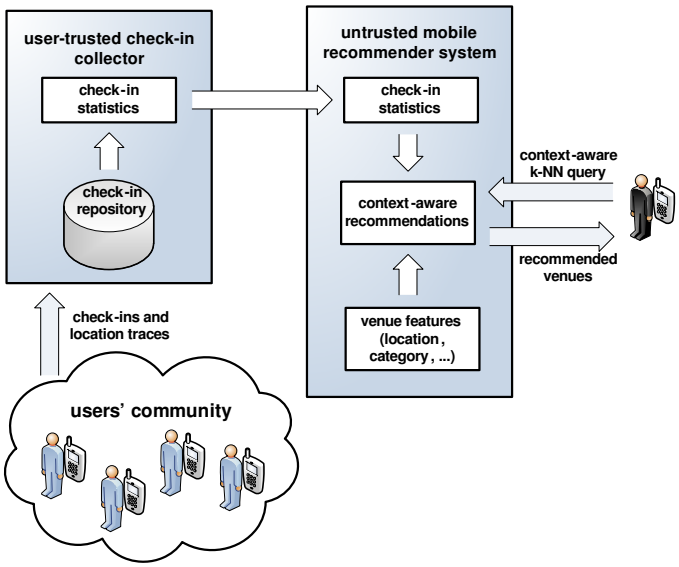


Fig. 1. Reference architecture

checked-in there. The service has different terms depending on the detail and amount of requested information. Without loss of generality, for the sake of this work we assume that the untrusted third-party is a mobile recommender system. However, the same threats and defense are applicable to any other untrusted third-party obtaining check-ins statistics.

In order to obtain venue recommendations, users issue *context-aware top-k queries* from an app on their mobile device. Queries include the number k of venues to be recommended, the current location of the user, a maximum distance radius, and a set of search categories. The query response returns the k venues of the requested categories that received the highest number of check-ins among the ones whose distance from the user is below the maximum search radius. If less than k venues of the requested categories are within the search radius, the user is prompted to either select a larger radius, or choose different categories.

Note that in this paper we do not address the problem of preserving location or context privacy for the user that is submitting a query to the recommender system. Standard techniques can be applied for that purpose [5]. For example, if a spatial cloaking technique is applied, the location of the querying user is generalized from a specific position to a region. This region is totally independent from the location privacy preferences that our check-in service users specify (the parameter L defined in Definition 2). The latter are used by the trusted collector to appropriately modify the check-in statistics before they are released. Our proposed defense works even if fake locations or any other privacy or context protection technique is used by the user when querying the recommender system.

B. Problem definition

Definition 1 (Check-in): We denote by $V = \langle v_1, v_2, \ldots, v_n \rangle$ a vector of venues, and by U a set of users that may visit venues in V. A check-in r is represented by a tuple $\langle u, v, t \rangle$, meaning that user u visited a venue v at time t.

The collector computes statistics S about the dataset D of check-ins, and perturbs them in order to protect privacy. We denote the perturbed statistics by S^*. Statistics S^* are released to an untrusted third party recommender system. Given S^* and the request parameters, the mobile recommender system deterministically returns a list of recommended venues. Hence, a user of the recommender system may be able to reconstruct S^* by submitting a large set of queries with different search parameters. In our model, the adversary may be the untrusted recommender system, or any entity that reconstructs S^* by querying the recommender system. The goal of the defense is to perturb check-ins statistics such that, even if the adversary has arbitrary background knowledge, the perturbed statistics cannot help him to perpetrate a *privacy violation*.

Definition 2 (Privacy violation): We state that a *privacy violation* occurs when an adversary infers with high confidence that a user checked-in at any venue within a square-shaped region of side-length less than a user-defined location privacy threshold L.

Each user providing check-ins and location traces to the trusted collector can choose different location privacy thresholds based on location; for instance, small L when in the center of the city, large L when in the countryside.

Referring to the example given in Section I, Bob may set the value L so that any square-shaped region containing his preferred club in Helltown would cover many other venues (or the whole Helltown). In this case he can still check-in precisely at the club and share that information with his close friends. However, the released check-in counts should be perturbed to enforce protection.

C. Adversary model

For the sake of this work, we make the following assumptions.

1) The check-in collector is trusted.
2) Both the recommender system, and the users that query the system to obtain venue recommendations are honest but curious adversaries[2], whose goal is to perpetrate a privacy violation.
3) Adversaries may have arbitrary external knowledge about the users.

IV. Defense method and privacy guarantees

In this section, we describe our defense technique, and the achieved privacy guarantees.

For each venue $v \in V$ in a potentially sensitive region of interest, the statistical measure that the check-in collector needs to extract from the set D of check-ins is the number c_v of check-ins at v by users in U:

$$c_v = \left|\{\langle u, v', t \rangle \in D : v' = v\}\right|.$$

In order to apply differential privacy to the check-in statistics, we need to formally define the computation that generates the statistics, and to compute its sensitivity (2).

[2]Technically, a honest but curious adversary follows the protocol but analyses the data in order to infer some additional information.

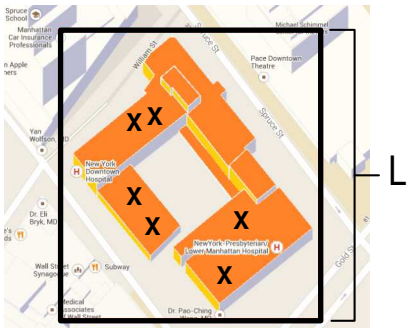


Fig. 2. A set of check-ins (represented by $\times$) of a single user u. That set satisfies $(L, 6)$-density, but it does not satisfy $(L, 5)$-density.

Definition 3 (Computation F_L): Given any square-shaped region R_L of side-length L, and the set V_L of venues in it, we denote by F_L a function:

$$F_L : \mathcal{D} \rightarrow \mathbb{N}_0^{|V_L|},$$

that takes as input a set of check-ins, and returns a vector that stores, for each venue $v \in V_L$, the number of check-ins at v.

As formally defined in Section II, the sensitivity of a computation measures the maximum change of the computation output when a single record is ignored. For the sake of this work, a record contains the whole set of check-ins of a single individual. Clearly, if each individual could contribute an unlimited number of check-ins to the statistics, the sensitivity of F_L would be infinite, since a record may account for infinite check-ins; in that case, differential privacy could not be applied without totally disrupting the utility of statistics [12]. Hence, we need to limit the individual contribution that each user may provide to the statistics. To this aim, we introduce the following property.

Property 1 ((L, j)-density): Given a set D of check-ins by a set U of users, an integer value $j > 0$, and a length L, we state that D satisfies (L, j)-*density* if, for any user $u \in U$, and any square-shaped region R_L of side-length L whose sides are parallel to the axes of the coordinate system, the region R_L contains at most j check-ins of u.

For instance, consider the region illustrated in Figure 2, corresponding to a hospital area in Manhattan, and the set D_u of check-ins (represented by $\times$) provided by a single user u. In that example, D_u satisfies $(L, 6)$-density, but it does not satisfy $(L, 5)$-density, since there exists one square-shaped region of side-length L (the one within the black square) containing more than 5 check-ins by the same user. Note that, while L is a privacy parameter chosen by the user, the value j is a system parameter tuned accordingly to experimental results to optimize utility.

Observe that, if a set D of check-ins satisfies Property 1 ((L, j)-density), then the sensitivity of F_L applied on D is j.

The following definition formalizes the notion of (ϵ, L, j)-private statistics.

Definition 4 ((ϵ, L, j)-private statistics): Given a dataset D of check-ins, a vector V of venues, a length L, an integer value $j > 0$, and a real value $\epsilon > 0$, we denote as (ϵ, L, j)-*private statistics* a vector

$$S^* = \langle c^*_{v_1}, c^*_{v_2}, ..., c^*_{v_n} \rangle$$

such that every $c^*_{v_i}$ is the result of the perturbation of c_{v_i} obtained by the application of ϵ-differential privacy with sensitivity $\Delta F = j$.

Given the above definitions, we can formalize our defense. Our defense consists in two main steps:

- At first, we obtain a dataset D' enforcing (L, j)-density by selecting, from the whole dataset of the check-in collector, only those check-ins that do not violate Property 1 (Algorithm 1 in Section V).
- Then, we extract (ϵ, L, j)-private statistics from D' (Algorithm 2 in Section V) in order to obtain S^*, that can be safely released to untrusted third parties, and used to provide recommendations.

V. Algorithms

In this section we present the algorithms to enforce (L, j)-density and extract (ϵ, L, j)-private statistics. For the sake of presentation, we illustrate the algorithms implementing our defense ignoring the possibility for a user to specify different values of L in different areas. However, the algorithms can be easily extended to deal with this case without loosing their properties.

A. Enforcing (L, j)-density

Enforcing (L, j)-density is not trivial. The simplest method would be to consider, for each user u, all the square-shaped regions of side-length L, and use no more than j check-ins from each of these regions to build the dataset satisfying (L, j)-density. However, in our model, the number of square-shaped regions of side-length L is infinite; hence, that method cannot be applied.

For this reason, we have devised an efficient algorithm, shown in Figure 3. The algorithm takes as input the original set D of check-ins, the set V of venues, the set U of users, values L of (L, j)-density for each user in U, and value j of (L, j)-density; it returns the set $D' \subseteq D$ of check-ins satisfying (L, j)-density.

Since, in our model, there exists an infinite number of square-shaped regions of side-length L, we need to discretize the search space. Suppose that a region R_L exists, which contains j check-ins from the same user (the dashed square in Figure 4(a)). In general, there exist infinite other regions of the same dimension containing the same check-ins, that can be obtained by translating R_L (e.g., the dotted squares in Figure 4(a)). Hence, for each set of check-ins, we represent the infinite set of square-shaped regions of side-length L that contain them by the rightmost and topmost square (called the *representative square*) of side-length L that contains those check-ins (the solid square in Figure 4(a)). That square is identified by the venues of the leftmost and bottommost check-ins in the set (represented in bold in Figure 4(a)).

After having initialized the set D' (line 1), the algorithm considers one user $u \in U$ at a time. At first (lines 3 and 4), the

Algorithm 1: ***Enforcing*** (L, j)***-density***

Input: Set D of check-ins; set V of venues; set U of users; set of values L_u of (L, j)-density for each user in $u \in U$; value j of (L, j)-density.

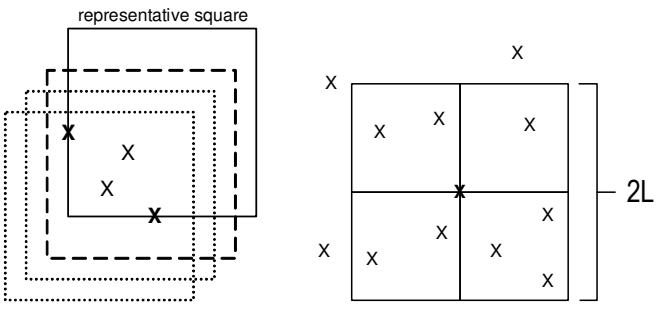
Output: Set $D' \subseteq D$ of check-ins satisfying (L, j)-density.

```
1:  D' := ∅
2:  for all user u ∈ U do
3:      D_u := check-ins of u in D
4:      L_u := value L of (L, j)-density for user u
5:      B-tree T_u := empty
6:      for all check-in c = ⟨u, v, t⟩ ∈ D_u do
7:          R̄ := Get2LRegion(v, L_u)
8:          V̄ := GetVenues(R̄)
9:          densityCondition := true
10:         for all venues ⟨v_x, v_y⟩ ∈ (V̄ × V̄) do
11:             if IsFeasible(⟨v_x, v_y⟩, R̄) then
12:                 n := NumberOfCheckIns(T_u, ⟨v_x, v_y⟩)
13:                 if n == j then
14:                     densityCondition := false
15:                     break
16:                 end if
17:             end if
18:         end for
19:         if densityCondition then
20:             D' := D' ∪ {c}
21:             Update(T_u)
22:         end if
23:     end for
24: end for
25: return(D')
```

Fig. 3. Algorithm 1: Enforcing (L, j)-density

algorithm selects the set D_u of u's check-ins from D, and the value L_u of (L, j)-density for u. Then (line 5), it initializes a B-tree [10] T_u that stores statistics about the number of u's check-ins in representative squares.

The algorithm considers one check-in at D_u at a time (lines 6 to 24). For each check-in $c = \langle u, v, t \rangle \in D_u$, the algorithm selects only those venues $\overline{V} \subseteq V$ that belong to any L-sided square containing the venue v of c; indeed, check-ins at other venues cannot belong to an L-sided square containing v, and can be ignored. $\overline{V}$ is the set of venues within the $2L$-sided square $\overline{R}$ with center in v (see Figure 4(b)). Then, the algorithm considers all the *feasible* representative squares contained in $\overline{R}$, in order to check if all of them contain less



(a) Discretization of the search space

(b) The $2L$-sided square $\overline{R}$ with center in v (v is represented by the bold $\times$)

Fig. 4. Representative squares and venue selection in Algorithm 1

Algorithm 2: ***Extracting*** (ϵ, L, j)***-private statistics***

Input: Check-ins dataset D' satisfying (L, j)-density; set of venues V; parameter j of (L, j)-density; parameter ϵ of differential privacy.

Output: (ϵ, L, j)-private statistics S^*.

```
1: S* = ⟨⟩
2: ⟨c_{v_1}, c_{v_2}, ..., c_{v_n}⟩ := F_L(D')
3: for all venue v_i ∈ V do
4:     c*_{v_i} := c_{v_i} + LaplaceNoise(0, j/ε)
5:     S*.add(c*_{v_i})
6: end for
7: return(S*)
```

Fig. 5. Algorithm 2: Extracting (ϵ, L, j)-private statistics

than j check-ins by u: in that case, c can be added to D'. Note that feasible representative squares are a subset of those regions whose representative square belongs to $\overline{V} \times \overline{V}$. Indeed, some pairs $\langle v_x, v_y \rangle$ ($v_x, v_y \in \overline{V}$) are not feasible. For instance, if v_y is on the left hand side of v_x, the pair $\langle v_x, v_y \rangle$ is unfeasible: indeed, by definition, v_x is assumed to be the leftmost venue of the representative square. Moreover, if v_x is on the right hand side of v, or v_y is below v, any representative square $\langle v_x, v_y \rangle$ must not be checked, since c is at a venue outside $\langle v_x, v_y \rangle$. The algorithm selects the feasible representative squares by iterating venues in $\overline{V}$ based on their x and y axes (lines 10 to 18). For each feasible representative square, the algorithm queries T to obtain the number of check-ins in D' in that square (line 12). If all feasible representative squares contain less than j check-ins, c is added to D', and T is updated (lines 20 and 21).

It is easy to verify by construction that Algorithm 1 returns a set of check-ins satisfying (L, j)-density. The average and worst case time complexity of Algorithm 1 (Figure 3) is $O(n \cdot j^2 \log |V|^2)$, where n is the cardinality of D.

B. Extracting (ϵ, L, j)-private statistics

The pseudo-code of the algorithm for extracting (ϵ, L, j)-private statistics is reported in Figure 5. The algorithm takes as input a check-ins dataset D' satisfying (L, j)-density, the set V of venues, and the parameter ϵ of differential privacy. It returns the (ϵ, L, j)-private statistics S^*.

At first, the vector S^* is initialized (line 1). Then, for each venue v, the algorithm calculates the exact number c_v of check-ins at v (line 2). Differential privacy is applied by adding noise to c_v extracted from a Laplace distribution with mean 0 and scale parameter $\frac{j}{\epsilon}$ (line 4). The resulting value c_v^* is added to S^* (line 5). Finally, the obtained (ϵ, L, j)-private statistics are returned.

Since the extraction of random samples from a Laplace probability distribution incurs negligible overhead, the execution of Algorithm 2 has no impact on the overall computational complexity of our defense. Hence, the complexity of our defense corresponds to the one of Algorithm 1.

VI. Experimental evaluation

In this section we experimentally evaluate our defense.

A. Experimental setup

We have applied our defense to a large dataset of real users' check-ins retrieved from location sharing services [9], in order to evaluate its impact on the data utility. The dataset has been collected by mining geo-tagged Twitter feeds automatically generated by different location sharing services (including Foursquare, Gowalla, and Facebook Places). More than 22 million check-ins were collected from September 2010 to January 2011, provided by more than $225,000$ users. Among other data, each check-in includes the user ID, the venue ID, the time of the check-in, and a textual comment. The highest density of check-ins is in North America, Western Europe, South and Pacific Asia. For carrying out our experiments, we have chosen the area of Manhattan, New York, because of the large number of venues it contains: this is the most challenging case when location privacy must be protected, since the density of check-ins is high. For venue recommendation purposes, we were interested in counting the number of different people that checked-in at each venue; hence, for our experiments we have removed repeated check-ins performed by the same user at the same venue. However, we obtained similar results considering also repeated check-ins. In order to associate the correct category to each venue, we have mined Foursquare to retrieve the micro-category (e.g., "Italian restaurant") and macro-category (e.g., "Food") corresponding to each venue ID. We kept only check-ins from Foursquare, distributed in 378 micro- and 8 macro-categories (Education, Home/Work/Other, Entertainment, Nightlife, Food, Shop, Travel, Outdoors). The resulting dataset consists of more than $231,000$ check-ins at about $30,000$ Manhattan venues by more than $15,000$ users.

We performed our experiments with varying values of ϵ, L and j. For simplicity, we assumed the value L selected by a user not to change in different areas of Manhattan. Parameters ϵ and L have an impact on both the achieved privacy guarantees, and data utility (in general, the more privacy, the less utility). The larger L, the more location privacy is provided. The smaller ϵ, the less an adversary can infer about the presence of a user's data by observing the output of the system. Generally, values of $\epsilon = 0.1$ or less are considered strong, while values of $\epsilon = 10$ or more are considered weak [16]. On the contrary, parameter j does not impact privacy, but only data utility. Indeed, the smaller j, the more check-ins must be pruned, but the less noise must be added to statistics S^*, since j corresponds to the sensitivity of the function generating S^*.

B. Rate of pruned check-ins

We executed Algorithm 1 (Enforcing (L, j)-density) to evaluate the rate of check-ins in D that were not added to D' (named *pruned* check-ins in the following) according to different values of L and j. Results are shown in Figure 6. With small values of j and large values of L, a high rate of check-ins is pruned. With medium levels of location privacy ($L = 500$m), the rate of pruned check-ins is much lower. The relatively high rate of pruning is in part due to the fact that we executed the experiments in the Manhattan area, in which venues are very dense. In less dense areas, we expect a lower rate of pruning. However, pruning a relatively high number of check-ins does not necessarily determine poor data utility of statistics; indeed, the loss of information due to pruning is counterbalanced by the need to inject less noise due to the reduced sensitivity. We evaluate this aspect in the following experiments.

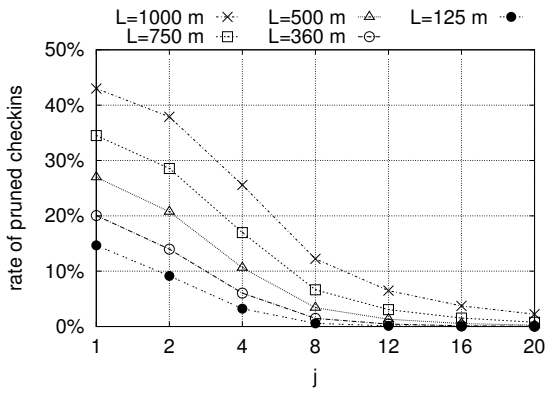


Fig. 6. Rate of pruned check-ins based on L and j

C. Data utility

We measure the data utility based on the quality of recommendation obtained after applying our defense. We consider the following *context-aware top-k query* issued by a user:

SELECT TOP-k venues FROM RECOMMENDER_DB
WHERE DISTANCE(u_coord, venue_coord) $< d$
AND CATEGORY = C
AND SEARCH_HOURS = $[h_b, h_e]$

In order to answer the above query, at first the recommender system selects those venues belonging to category C and having maximum distance d from the user's current location. Then, it sorts them according to the number of check-ins during search hours $[h_b, h_e]$ (e.g., late night $[00, 05]$, morning $[06, 11]$), and returns the top-k venues. We call T the set of venues returned based on the non-sanitized data (the "true" top-k venues), and T_ϵ the set of those returned based on the application of differential privacy with parameter ϵ. We measure the error E introduced by the application of our defense as:

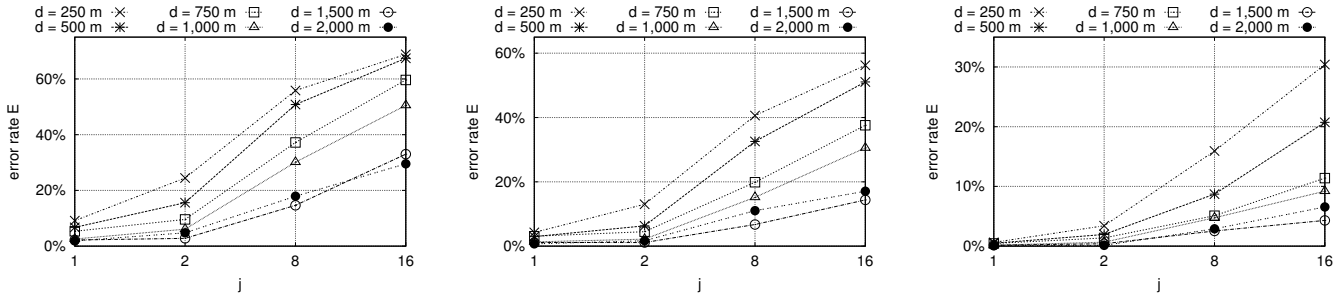
$$E = 1 - \frac{|T_\epsilon \cap T|}{k}$$

i.e., E is the percentage of venues suggested by the privacy-preserving system that were not the true top-k venues. In all the data utility experiments we calculate the average error by issuing the query from 10 different random locations in Manhattan. We claim that an average error below 10% provides reasonably good data quality in our considered scenario.

In the following we report the result of different data quality experiments with a fixed value of $L = 500$ m and the other parameters values varying as described:

1) Search radius d: Figure 7 shows the average error considering values from 1 to 16 for the j parameter, from 0.5 to 3 for ϵ, and from 250m to $2,000$m for the search radius d. We set the parameter k to 10, category to "any" and search hours $[00, 24]$.

As expected, the average error is higher with lower values of ϵ. Indeed, the lower ϵ, the higher the privacy, the higher



(a) $\epsilon = 0.5$ (b) $\epsilon = 1.0$ (c) $\epsilon = 3.0$

Fig. 7. Average error based on parameters j, ϵ, and d. $k = 10$; search hours = $[00, 24]$; category = any.

the amount of noise that is added to the data used to provide recommendations. On the other hand, the average error increases with smaller values of d. This is due to the fact that in the dataset there are a few venues having a large number of check-ins, while the majority of venues has very few check-ins. Hence, in a typical query, after adding random noise, it is relatively easy to correctly select the top 3 to 5 venues (the most popular, which have a large number of check-ins), but it is more difficult to recognize less popular venues, since the added random noise is close to their actual number of check-ins, thus reducing the utility of perturbed data. The larger the radius, the more probable to find very popular venues. However, with values of $j = 1$ and $j = 2$, the average error is small, even using a small search radius. This indicates that our pruning method has very positive effects on the utility of sanitized data, since by pruning part of check-ins we are able to inject much less noise to the private check-in statistics. In the following experiments, we set $\epsilon = 1$.

2) Number k of requested venues: We evaluated the average error based on different values of the number k of requested venues. We set the search radius to $1,000$ m, category to "any" and search hours to $[00, 24]$. Figure 8(a) shows the average error based on values of k from 3 to 20 and j from 1 to 16. The average error tends to be higher, the larger the number of requested venues. As in the previous experiments, the reason is that it is more difficult to recognize less popular venues than very popular ones; when the number of requested venues is large, it is probable that the query captures many non-popular venues. However, even with large values of k, the error is very low when small values of j are used. As in the previous experiment, in most cases the best results are obtained with $j = 1$ or $j = 2$.

3) Search hours: Even if existing mobile recommenders generally do not take into account temporal features, we believe that time is an important aspect to consider for this kind of services. Hence, in the next experiments we measured the average error based on different search hours.

We set the distance radius to $1,000$ m and search category to "any". Figure 8(b) shows the average error based on six different values of search hours. The average error is lowest if we consider the largest search hours $[00, 24]$. The error grows considering more specific hours (e.g., $[06, 11]$ and $[12, 15]$), during which the number of submitted check-ins is relatively small (see Table I). Indeed, in those cases, it is more difficult to recognize the top-k venues, since the introduced noise tends to be of the same order of magnitude of the true value (number of check-ins submitted during those hours), thus reducing the utility of the data. However, when considering time-specific queries, the average error is well below 10% using values of $j \leq 2$. Moreover, we expect to obtain better results using larger datasets, as those owned by today's geo-social networks and location sharing services.

TABLE I. NUMBER OF CHECK-INS SUBMITTED DURING DIFFERENT SEARCH HOURS

Search hours	Number of check-ins
[00, 05]	70,556
[06, 11]	12,521
[12, 15]	26,127
[16, 19]	56,468
[20, 24]	65,557
[00, 24]	231,229

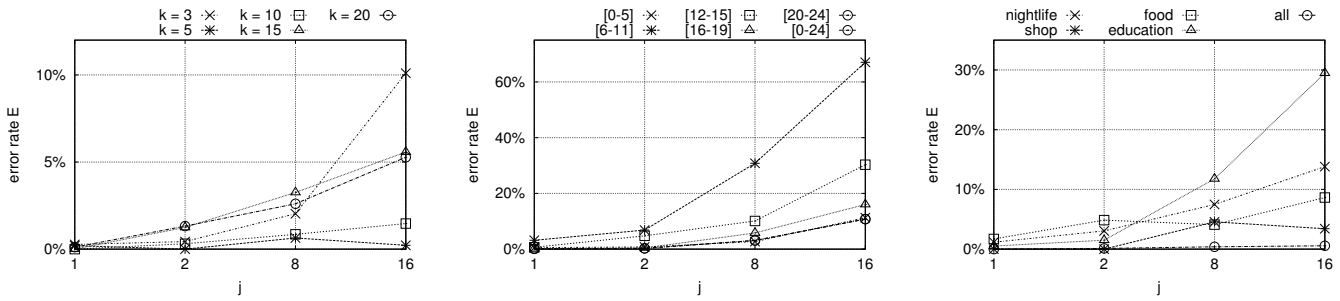
TABLE II. NUMBER OF CHECK-INS ACCORDING TO PART OF THE VENUE CATEGORIES

Search category	Number of check-ins
Education	11,983
Nightlife	56,082
Food	82,883
Shop	29,255

4) Search category: Finally, we measured the average error based on different search categories c. Figure 8(c) shows the average error considering part of the macro-categories of Foursquare. Similarly to temporal queries, the average error tends to be higher if we consider categories whose venues have received a small number of check-ins (e.g., *education*). Table II reports the number of check-ins submitted for the venues categories considered in our experiments. As explained above, this problem can be alleviated by using larger datasets. However, even with our dataset, the application of our pruning method with low values of j is sufficient to keep the error below 10% for any category.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a technique and architecture to address the challenging research issue of providing formal privacy guarantees for a context-aware recommender system of venues. We have proposed a defense based on differential privacy. Extensive experiments with a large dataset of real users' check-ins showed that our defense provides a good trade-off between privacy and data utility.



(a) Impact of k (number of requested venues) (b) Impact of search hours (c) Impact of search categories

Fig. 8. Average error based on parameters j, k, search hours, and search categories. $\epsilon = 1.0$; $L = 500$ m.

Several research issues remain open, and will be investigated in future work. In particular, in this paper we aimed at protecting the sensitive association between a single user and a group of geographically-close venues. However, even frequent check-ins at a semantically-close group of venues (e.g., gluten-free restaurants) may determine a privacy breach. An adversary could also try to discover whether any member of a group of people has a preference for a venue: for instance, "is the Hollywood Disco a preferred venue of A.C. Milan football players?". Such inferences about groups of people/venues increase the sensitivity of queries and, consequently, the amount of noise that must be used to enforce differential privacy, possibly disrupting data utility. Hence, it is necessary to devise data-quality preserving methods to protect against those coarse-grained inferences. Extensions of our technique to support incremental update of statistics will also be investigated.

Another challenging issue regards how to incentivize the user in sharing his/her check-in data. In our defense, each user can fine-tune the desired level of location privacy by setting the value L of (L, j)-density. Of course, the larger L, the more location privacy for the user, and the more of the user's check-ins are pruned. Hence, there are conflicting goals between the user and the check-in collector: the former would like to choose large values of L to have strong location privacy, while the latter would prefer small values to retain more check-in data. In order to address this issue, reward mechanisms should be adopted to incentivize personal data sharing, like proposed in recent works (e.g., [14]).

ACKNOWLEDGMENTS

This work has been partially supported by the TENACE PRIN Project (n. 20103P34XC) funded by the Italian Ministry of Education, University and Research. The authors would like to thank Zhiyuan Cheng, James Caverlee, Kyumin Lee and Daniel Z. Sui for kindly providing the dataset used in our experiments, and Daniela Ramundo and Chiara Maestroni for their work in software development.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 217–253.

[2] F. Armknecht and T. Strufe, "An efficient distributed privacy-preserving recommendation system," in *Proc. of the 10th IFIP Med-Hoc-Net Workshop*. IEEE Comp. Soc., 2011, pp. 65–70.

[3] L. Baltrunas, B. Ludwig, S. Peer, and F. Ricci, "Context-aware places of interest recommendations for mobile users," in *Proc. of HCI*, ser. LNCS, vol. 6769. Springer, 2011, pp. 531–540.

[4] M. J. Barranco, J. M. Noguera, J. Castro, and L. Martínez, "A context-aware mobile recommender system based on location and trajectory," in *Management Intelligent Systems*, ser. Advances in Intelligent Systems and Computing. Springer, 2012, vol. 171, pp. 153–162.

[5] C. Bettini, S. Jajodia, P. Samarati, and X. S. Wang, Eds., *Privacy in Location-Based Applications: Research Issues and Emerging Trends*, ser. LNCS. Springer, 2009, vol. 5599.

[6] R. D. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User-Adapt. Interact.*, vol. 12, no. 4, pp. 331–370, 2002.

[7] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, "You might also like: Privacy risks of collaborative filtering," in *IEEE Symposium on Security and Privacy*. IEEE Comp. Soc., 2011, pp. 231–246.

[8] J. F. Canny, "Collaborative filtering with privacy," in *IEEE Symposium on Security and Privacy*. IEEE Comp. Soc., 2002, pp. 45–57.

[9] Z. Cheng, J. Caverlee, K. Lee, and D. Z. Sui, "Exploring millions of footprints in location sharing services," in *Proc. of ICWSM*. AAAI Press, 2011.

[10] D. Comer, "The ubiquitous b-tree," *ACM Comput. Surv.*, vol. 11, no. 2, pp. 121–137, 1979.

[11] H. Costa, B. Furtado, D. Pires, L. Macedo, and F. A. Cardoso, "Context and intention-awareness in pois recommender systems," in *Proc. of CARS Workshop*, ser. CEUR Workshop Proc., vol. 889. CEUR-WS.org, 2012.

[12] C. Dwork, "Differential privacy," in *Proc. of ICALP*, ser. LNCS, vol. 4052. Springer, 2006, pp. 1–12.

[13] D. Gavalas and M. Kenteris, "A web-based pervasive recommendation system for mobile tourist guides," *Pers. Ubiquit. Comput.*, vol. 15, no. 7, pp. 759–770, 2011.

[14] J.-S. Lee and B. Hoh, "Sell your experiences: a market mechanism based incentive for participatory sensing," in *Proc. of PerCom*. IEEE Comp. Soc., 2010, pp. 60–68.

[15] A. Machanavajjhala, A. Korolova, and A. D. Sarma, "Personalized social recommendations - accurate or private?" *Proc. of the VLDB Endowment*, vol. 4, no. 7, pp. 440–450, 2011.

[16] F. McSherry and R. Mahajan, "Differentially-private network trace analysis," in *Proc. of SIGCOMM*. ACM, 2010, pp. 123–134.

[17] F. McSherry and I. Mironov, "Differentially private recommender systems: Building privacy into the netflix prize contenders," in *Proc. of SIGKDD*. ACM, 2009, pp. 627–636.

[18] D. Riboni and C. Bettini, "Private context-aware recommendation of points of interest: An initial investigation," in *Proc. of PerCom Workshops*. IEEE Comp. Soc., 2012, pp. 584–589.

[19] H. Sato, T. Inoue, H. Iwamoto, and N. Takahashi, "Virtual scent: Finding locations of interest in ambient intelligence environments," in *Proc. of PDCAT*. IEEE Comp. Soc., 2009, pp. 384–389.

[20] M. P. Scipioni and M. Langheinrich, "I'm here! privacy challenges in mobile location sharing," in *Proc. of IWSSI/SPMU Workshop*, 2010.