An Efficient Distributed Privacy-preserving Recommendation System

Frederik Armknecht Thorsten Strufe
Universitaet Mannheim, Germany
[lastname](at)uni-mannheim.de

***Abstract*—Implementing a recommendation system on the data of mobile social networks exploits knowledge about behavior and preferences of its users and hence raises serious privacy concerns. Leveraging the wealth of aggregated information in these services promises an immense benefit by allowing suggestions for presumably appreciated, yet previously unseen restaurants, sights, and further types of locations. Privacy preserving recommenders based on homomorphic encryption have been proposed, which have a systematic draw-back: while recommender systems often store their information as real values, all homomorphic encryption schemes used today process only data from other algebraic structures, e.g., the ring of integers modulo some integer n. Therefore, we present a novel distributed recommender and a homomorphic encryption scheme, which works directly on real numbers and which possesses some remarkable properties: it is conceptually simple, efficient, and provably secure.**

I. Introduction

Mobile, location-based online social networks, like gowalla, foursquare, or facebook places[1] are enjoying an immense increase in utilization and are prognosed to be a major driving force advancing the further growth of social networking services. The providers offer mobile applications that allow the users to publish that they are currently visiting a certain location, vulgo "checking in" at selected "spots", where the spots are automatically identified by the GPS location of the mobile phone. These services, collecting and immense amount of highly personal information including mobility traces, so far yield high revenues for the providers of the services, yet fail to offer benefit for the users beyond the possibility to hunt for achievements and the odd voucher for a free coffee at one of the participating locations. In previous work [1], Berjani and Strufe have proposed the first recommendation system for spots in location-based online social networks (henceforth termed the *BS system*) towards this end.

While such recommendation systems can be highly beneficial both for the users and the service providers, the price is likewise high: the total loss of privacy. For computing a recommendation, one naturally needs to take into account both the preferences of the user and the properties of the considered item. Thus, either the user is forced to reveal the whole information about his preferences, opening the door for the misuse of personal data, or the system provider has to give the user all necessary information required for computing the recommendation on his own. The latter, however, is not acceptable either, as keeping this information confidential may be essential for commercial reasons[2]. This immediately raises the question: Is it possible to construct a recommendation system such that

1) The user does not need to reveal his preferences to the system provider.
2) The system provider does not to reveal need his internal information about spots and preferences.
3) The user gets a correctly computed recommendation.

It has been discussed to use homomorphic encryption schemes for this purpose in the past (cmp., e.g., [2]). A homomorphic encryption scheme preserves the algebraic structure of the data space such that data can be processed in its encrypted form. Consider, for example, an *additively* homomorphic encryption scheme and let encryptions of two arbitrary messages m and m' messages be given, one can compute an encryption of $m + m'$ without knowing m or m'. This allows to outsource the computation on data without revealing the data itself. In particular, this makes homomorphic encryption schemes appealing building blocks for privacy-preserving recommendation systems. The idea in a nutshell is that the user provides his preferences in encrypted form to the service provider. The latter computes the recommendation using his internal information and the encrypted preferences, to get an encrypted recommendation which is returned to the user. The user, being in possession of the decryption key, can decrypt and consider the recommendation. The service provider, however, gained no knowledge on the preferences.

Using a homomorphic encryption scheme that is able to process real values in an efficient and secure way is mandatory for this approach. Here a problem arises as all existing schemes do either not work on real values or are highly inefficient, hence being unsuitable for the application scenario. In [2], it has been proposed to choose a homomorphic encryption scheme where the plaintexts are integers. We see three different drawbacks with this approach: (i) As we will explain in Section III-C, one has to be careful with this approach to achieve correctness and security at the same time. (ii) The ciphertexts are usually quite big, e.g., 1000 bits and more, while the plaintexts may be much smaller e.g., around 100 bits. This unnecessarily increases the effort for transmitting data, being disadvantageous for mobile devices. (iii) These schemes usually require quite complex computations, e.g., multiplication of huge numbers modulo

[1] http://www.[gowalla|foursquare|facebook].com

[2] See, e.g. http://www.teleread.com/paul-biba/goodreads-revs-up-a-book-recommendation-engine/

978-1-4577-0899-2/11/$26.00 ©2011 IEEE

some integer. Also here, a simple and efficient scheme would be better suited for mobile devices.

In this work, we close this gap by presenting a novel encryption scheme that is perfectly suited for privacy preserving recommendation systems, especially if mobile devices are involved. More precisely, the proposed scheme is able to directly process real valued data. Thus, there is no need to adapt the scheme or the data for being suitable in a recommendation system. In particular, the security analysis takes the algebraic structure into account, making a separate security analysis superfluous. Furthermore, the scheme itself is conceptually very simple and requires only few basic operations (addition and multiplication) over the field $\mathbb{R}$ of real numbers. For example, neither exponentiations nor taking the modulus are required.

The remainder of this paper is organized as follows: we introduce the basic foundations recommender systems and present a spot recommender from previous work in Section II. Section III subsequently introduces our recommender based on homomorphic encryption, and Section IV our new homomorphic encryption scheme to implement the recommender. We conclude our paper in Section V and give an outlook on future extensions.

II. Spot Recommendation

The main aim of a spot recommender is, given the personalized preferences of users as input, to predict the user's response to a selected item, or suggest items that are predicted to appeal to the user. Collaborative filtering (CF) approaches are an accepted and promising class for recommender systems. Their general intent is to transform the observed ratings r as given in the sparse matrix Y of n users vs. m items (cmp. Table I), to a dense model of latent factors. The model then can be used to predict the response $\rho_{(i,j)}$ of a user i to a previously unseen item j, following different strategies.

	item_1	$\cdots$	item_j	$\cdots$	item_m
user_1	$r_{(11)}$		$r_{(1j)}$		$r_{(1m)}$
$\vdots$					
user_i	$r_{(i1)}$		$\rho_{(\mathbf{i,j})}$?		$r_{(im)}$
$\vdots$					
user_n	$r_{(1n)}$		$r_{(nj)}$		$r_{(nm)}$

TABLE I
User/Spot matrix of ratings r and predictions ρ

Exploring the potential of different approaches in [1], we have suggested that Regularized Matrix Factorization (RMF) exhibited the most promising results. The BS system considers spots that users visit as items and the checkins as an evidence for preference. Location-based online social networks lack an explicit rating of spots. The recommender hence assumes the logarithm of the number of checkins as a balanced valuation and performs a binning into equal width intervals over the resulting values ($LogEWI$) to achieve an explicit preference definition. Leveraging RMF the model then is learned as dense explicit matrices of chosen dimension d, computing a mapping to factor vectors. $P \in \mathbb{R}^{n\times d}$ represents the rows, or user preferences of the input matrix, and $Q \in \mathbb{R}^{m\times d}$ the columns, or spot rating factors. The challenge is to find a mapping, such that $Q^\top P$ well estimates Y. The recommendation task of predicting the response of user i to a previously unvisited spot j in consequence can be performed by simply multiplying the according factor vectors $< \overrightarrow{p_i}, \overrightarrow{q_j} >$. Here, $\langle \overrightarrow{p_i}, \overrightarrow{q_j} \rangle$ refers to the scalar product of two vectors. That is, if $\overrightarrow{v} = (v_1, \ldots, v_n)$ and $\overrightarrow{w} = (w_1, \ldots, w_n)$ are two vectors in $\mathbb{R}^n$, then $\langle \overrightarrow{v}, \overrightarrow{w} \rangle = \sum_{i=1}^{n} v_i \cdot w_i$. The BS system employs the squared error and the stochastic gradient descent for the learning process. All factor vectors are initialized with arbitrary values. Looping over all ratings of the training set the algorithm calculates the predicted rating ρ and the corresponding prediction error as:

$$e_{(i,j)} = r_{(i,j)} - \rho_{(i,j)}. \tag{1}$$

The factor vectors then are updated opposite to the gradient:

$$\overrightarrow{p}'_i = \overrightarrow{p_i} + \gamma \cdot \left(e_{(i,j)} \cdot \overrightarrow{q_j} - \lambda \cdot \overrightarrow{p_i}\right) \tag{2}$$

$$\overrightarrow{q}'_j = \overrightarrow{q_j} + \gamma \cdot \left(e_{(i,j)} \cdot \overrightarrow{p_i} - \lambda \cdot \overrightarrow{q_j}\right) \tag{3}$$

with the learning rate $\gamma \in \mathbb{R}$ and the regularizer parameter $\lambda \in \mathbb{R}$.

Generating the factor model is a computationally intensive task based on aggregating a wealth of highly personal information of a selected base of users. The providers of recommendation systems hence have high interest in protecting their model, represented in P and Q. The preference vectors P of users describe the behavior and preference of their respective user at the same time. Concealing their preference vector from the provider consequently is a quite comprehensible objective for users of the recommender systems.

We hence devise a privacy preserving spot-recommendation scheme. It processes behavior and preferences of plain users, which are disclosed to the provider, to derive the model. Premium users, however, are enabled to consume a privacy preserving service of the provider: They are concealing their behavior and preferences from the provider, who still computes recommendations and updates to their preference vectors on encrypted data.

III. Privacy-preserving Recommender Systems based on Homomorphic Encryption

A. General Setting

Although the work is motivated by spot recommendation systems as explained in Sec. II, we expect that the proposed solution is applicable to a broader set of scenarios. Therefore, we give in the following a description of the considered setting in general terms. We explain first the plain setting only, that is the recommendation system without any privacy preserving means. Involved are two parties: a recommendation system operated by a system provider $\mathcal{S}$ and a user $\mathcal{U}$. $\mathcal{S}$ stores information on a number of objects, e.g., spots in the case of a location-based recommender system, on which the user

can ask for recommendations. The system builds on a base of plain users in order to learn the dense factor matrices Q and P. These may be the users of a primary service (like, e.g., a location based online social network as in the cases of gowalla and foursquare, or an online shop). Privacy enabled users consume the premium service of the recommendation system only, and they hence conceal their preferences and behavior, thus protecting their privacy.

More precisely, for each object an information vector $\overrightarrow{q_j} = (q_{(j,1)}, \ldots, q_{(j,d)}) \in \mathbb{R}^d$, and for each plain user a preference vector $\overrightarrow{p_i} = (p_{(i,1)}, \ldots, p_{(i,d)})$ is stored where $\mathbb{R}$ denotes the field of real numbers and d the dimension. The exact meaning and the number of entries depend on the actual use case.

Premium users similarly store their personal preference vector $\overrightarrow{p_i} = (p_{(i,1)}, \ldots, p_{(i,d)}) \in \mathbb{R}^d$. If the user is interested to get a recommendation for an object j, he sends the the preference vector to $\mathcal{S}$. $\mathcal{S}$ uses the preference vector and the information stored on the object to compute a recommendation value $\rho_{(i,j)}$. That is, $\mathcal{S}$ computes

$$\rho_{(i,j)} := f(\overrightarrow{p_i}, \overrightarrow{q_j}) \tag{4}$$

where f is the recommendation function. In this work, we assume that f is linear with respect to the entries in $\overrightarrow{p_i}$. That is, it holds that

$$\rho_{(i,j)} = \sum_{l=1}^{d} f_l(\overrightarrow{q_j}) \cdot p_{(i,l)} \tag{5}$$

where $f_l : \mathbb{R}^d \to \mathbb{R}$ are the partial recommendation functions. This setting covers several concretely proposed recommendation systems, including the BS system [1].

B. Prerequisite: A Homomorphic Encryption Scheme over Real Numbers

Before we explain the privacy preserving recommendation system, we specify the prerequisites. For the system, we require a homomorphic encryption scheme with some particular properties. These are as follows:

1) For each key k, the plaintext space is $\mathbb{R}$. That is, for a value $m \in \mathbb{R}$, the encryption function accepts m as input and generates a ciphertext c.
2) The encryption scheme is additively homomorphic. That is, given encryptions of two values $m, m' \in \mathbb{R}$, one can compute an encryption of $m \cdot m' \in \mathbb{R}$ without knowing m or m'.
3) Let c be an encryption of a value $m \in \mathbb{R}$ and $\lambda \in \mathbb{R}$ be an arbitrary real value. Then, given an encryption of m, one can compute an encryption of $\lambda \cdot m$ even if m is secret.

In Sec. IV, we introduce a new encryption scheme that fulfills all conditions mentioned above.

C. On the Necessity of an Appropriate Homomorphic Encryption Scheme

Before we proceed with the description of the recommender system in Sec. III-D, we explain why existing homomorphic encryption schemes are not adequate building blocks. The reason is that many homomorphic encryption schemes are homomorphic with respect to an algebraic structure that differs significantly from the field $\mathbb{R}$ of real numbers. The candidates that support structures that are the most similar to $\mathbb{R}$ are schemes where the plaintexts are integers. Examples are RSA [3], ElGamal [4], and Paillier [5]. In [2], Erkin et al. suggest to pick such a scheme[3] and to treat the real values as integer values (i.e., all real values are multiplied with an appropriate expansion factor). However, here one has to pay attention to the details.

All these schemes use a plaintext space which is *not* the ring of integers $\mathbb{Z}$ but the ring $\mathbb{Z}_n$ for some integer value n. Recall that $\mathbb{Z}_n$ refers to the integers in the range of $\{0, \ldots, n-1\}$ and all computations are done modulo n. For example, consider the Paillier encryption scheme, being one of the few additive homomorphic encryption schemes. Let m and m' denote two plaintexts which are taken from the range $\{0, \ldots, n-1\}$ and let c and c' be encryptions of m and m', respectively. The ciphertexts are values in $\mathbb{Z}_{n^2}$ and the product $c \cdot c'$ (being again a value in $\mathbb{Z}_{n^2}$) gives an encryption of $m + m'$ modulo n. Obviously, $c \cdot c'$ is an encryption of $m + m'$ (taken as sum in $\mathbb{Z}$) if and only if $m + m' < n$. That is, one has to take care that the results of *all* possible computations never exceed n. In the worst case, this may require to choose large values for n, which immediately implies that the ciphertexts may get very large. Indeed, for all schemes mentioned above, n needs to be big anyway for security reasons. For example, in the schemes [3], [5], [6], n needs to be at least 1024 bits long for a minimum security level. This may lead to a huge data overhead with respect to the size of the ciphertexts compared to the size of the plaintexts.

Another effect is that the plaintexts are not uniformly distributed in the plaintext space anymore. This may compromise security. Take as an example the RSA encryption scheme. Here, the plaintext space is $\mathbb{Z}_n$ and a value $m \in \mathbb{Z}_n$ is encrypted by $c := m^e \bmod n$. Now assume that m is very small compared to n such that m^e does not produce a carry over, that is $m^e < n$. In this case, one can easily compute m from c by taking the e-th root of c. Observe that this problem is conjectured to be hard in $\mathbb{Z}_n$ (if n is chosen appropriately) but is easy in $\mathbb{Z}$. Summing up, restricting the set of plaintexts to small numbers (compared to n) can induce new security risks. In any case, the security of the previously proposed schemes need to be re-evaluated. Theoretically, one may use a fully homomorphic encryption scheme where the plaintext space is simply one bit cf., e.g., Gentry [7] and follow-up works. However, none of these are yet sufficiently efficient for practical applications. For example, Gentry and Halevi [8] have been able to implement all aspects of the scheme [7]. Interestingly enough, they proposed several challenges [4] with different level of security. It is worth to mention that the

[3]More precisely, they consider a scheme by Damgård et al. [6], being a variant of the Paillier

[4]https://researcher.ibm.com/researcher/view_project.php?id=1548.

corresponding public-key size ranges from 70 MBytes for the smaller setting to 2.3 GBytes for the larger setting and the time effort for one homomorphic operation from 30s (small setting) to 30mins (large setting).

D. Description

In this section, we describe the privacy preserving recommendation system. The setting is as formalized in Section III-A. However, the main difference now is that the user is interested to get a recommendation from $\mathcal{S}$ (correctness) without the need to reveal his preferences (privacy) At this end, $\mathcal{U}$ sends to $\mathcal{S}$ the preference vector only in encrypted form, using an encryption scheme that meets the requirements listed in Sec. III-B. More precisely, let $\overrightarrow{p_i} = (p_{(i,1)}, \ldots, p_{(i,d)})$ the preference vector. The user chooses an encryption key k and encrypts $\overrightarrow{c_l} := \mathsf{Enc}_k(p_{(i,l)})$ for $l = 1, \ldots, d$. Whenever the user is interested into getting a recommendation for object i, he sends $C := (\overrightarrow{c_1}, \ldots, \overrightarrow{c_d})$ to $\mathcal{S}$. If the encryption scheme is secure, $\mathcal{S}$ can derive no information on $\overrightarrow{p_i}$ from C.

Upon reception of C, the service provider $\mathcal{S}$ computes

$$\hat{\rho}_{(i,j)} := \sum_{l=1}^{d} f_l(\overrightarrow{q_j}) \cdot \overrightarrow{c_l} \tag{6}$$

Because of the properties requested in Sec. III-B, it holds

$$\begin{aligned}
\hat{\rho}_{(i,j)} &:= \sum_{l=1}^{d} f_l(\overrightarrow{q_j}) \cdot \overrightarrow{c_l} && (7)\\
&= \sum_{l=1}^{d} f_l(\overrightarrow{q_j}) \cdot \mathsf{Enc}_k(p_{(i,l)}) && (8)\\
&= \sum_{l=1}^{d} \mathsf{Enc}_k(f_l(\overrightarrow{q_j}) \cdot p_{(i,l)}) && (9)\\
&= \mathsf{Enc}_k(\sum_{l=1}^{d} f_l(\overrightarrow{q_j}) \cdot p_{(i,l)}) && (10)\\
&= \mathsf{Enc}_k(\rho_{(i,j)}). && (11)
\end{aligned}$$

That is, the user receives with $\hat{\rho}_{(i,j)}$ an encryption of the recommendation value $\rho_{(i,j)}$. By decrypting $\hat{\rho}_{(i,j)}$, the user can eventually get his recommendation. However, the system provider did not learn anything on $\overrightarrow{p_i}$ as long as the encryption scheme is secure. Furthermore, $\mathcal{S}$ was not obliged to reveal anything on $\overrightarrow{q_j}$. More precisely, the only information the user can learn about $\overrightarrow{q_j}$ is what he can deduce from $\rho_{(i,j)}$ and the knowledge of $\overrightarrow{p_i}$.

It remains to discuss how the user can get his preference vector. Motivated again by the BS system [1], we assume that $\overrightarrow{p_i}$ is first initialized with random values. Using the stochastic gradient descent as learning algorithm, the preference vector $\overrightarrow{p_i}$ can be updated based on the recommendation computed by $\mathcal{S}$. More precisely, the formula is

$$\overrightarrow{p_i}' := \overrightarrow{p_i} + \gamma \cdot \left(e_{(i,j)} \cdot \overrightarrow{q_j} - \lambda \cdot \overrightarrow{p_i}\right) \tag{12}$$

where $e_{(i,j)}$ can be computed by the user from the received recommendation. For example, $e_{(i,j)}$ may be the difference between the computed recommendation and a binary value that indicates whether the user visited spot i more often than the average user or not (see [1] for details). Given an encryption scheme as explained in Sec. III-B, one can likewise use the encryption scheme in such a way that the user can execute the learning algorithm without the need to reveal any information. More precisely, the user $\mathcal{U}$ sends to the service provider $\mathcal{S}$ an encryption of $\overrightarrow{p_i}$ and of $e_{(i,j)}$ (both are known to the user). S in turns computes

$$\begin{aligned}
& \mathsf{Enc}_k(\overrightarrow{p_i}) + \gamma \cdot \left(\mathsf{Enc}_k(e_{(i,j)}) \cdot \overrightarrow{q_j} - \lambda \cdot \mathsf{Enc}_k(\overrightarrow{p_i})\right)\\
= \; & \mathsf{Enc}_k\left(\overrightarrow{p_i} + \gamma \cdot \left(e_{(i,j)} \cdot \overrightarrow{q_j} - \lambda \cdot \overrightarrow{p_i}\right)\right)\\
= \; & \mathsf{Enc}_k\left(\overrightarrow{p_i}'\right)
\end{aligned}$$

and sends the result to the user $\mathcal{U}$. The user can decrypt the ciphertext to learn the updated preference vector while the system provider again did not learn anything on $\mathcal{U}$'s preferences.

Observe that in the computation above, $\mathsf{Enc}_k(\overrightarrow{p_i})$ refers to the componentwise encryption of $\overrightarrow{p_i}$. That is if $\overrightarrow{p_i} = (p_{(i,1)}, \ldots, p_{(i,d)})$, then

$$\mathsf{Enc}_k(\overrightarrow{p_i}) = (\mathsf{Enc}_k(p_{(i,1)}), \ldots, \mathsf{Enc}_k(p_{(i,d)})). \tag{13}$$

Similarly, if $\overrightarrow{q_j} = (q_{(j,1)}, \ldots, q_{(j,d)})$, then $\mathsf{Enc}_k(e_{(i,j)}) \cdot \overrightarrow{q_j}$ refers to

$$\mathsf{Enc}_k(e_{(i,j)}) \cdot \overrightarrow{q_j} = (\mathsf{Enc}_k(e_{(i,j)}) \cdot q_{(j,1)}, \ldots, \mathsf{Enc}_k(e_{(i,j)}) \cdot q_{(j,d)}). \tag{14}$$

IV. Encryption Scheme

A. Description

In this section, we present a novel encryption that meets the requirements mentioned in Sec. III-B. Recall that an encryption scheme is composed of three different algorithms:

Setup: The Gen algorithm gets as input some parameters (that depend on the actual scheme) and outputs a secret key k.

Encryption: The encryption algorithm Enc gets as input a secret key k, a message m, and outputs a ciphertext c.

Decryption: The decryption algorithm Dec gets as input a secret key k, a ciphertext c, and outputs a plaintext m.

In addition, the following conditions need to be met:

1) All algorithms are efficient.
2) For all keys k and all messages m, the decryption of an encryption of m (under the same key) yields m again. Formally:

$$\mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m. \tag{15}$$

The proposed encryption scheme is defined as follows:

1) Setup: The algorithm Gen takes as input a value n and samples randomly a vector $\overrightarrow{k} \in \mathbb{R}^n \setminus \{\overrightarrow{0}\}$. The vector $\overrightarrow{k}$ represent the secret key. The choice of n depends on the concrete application scenario, the range of possible values, and the aimed security level.

2) Encryption: The encryption algorithm Enc transforms values $m \in \mathbb{R}$ into ciphertexts $\vec{c} \in \mathbb{R}^n$. More precisely, the encryption algorithm takes as input a secret key $\vec{k} \in \mathbb{R}^n$ and a plaintext $m \in \mathbb{R}$ and samples a vector $\vec{c} \in \mathbb{R}^n$ such that

$$\langle \vec{c}, \vec{k} \rangle = m. \tag{16}$$

We explain in Sec. IV-D how an appropriate vector $\vec{c}$ can be efficiently sampled.

3) Decryption: The decryption operation is directly given by the condition formalized in eq. (16). Given a ciphertext $\vec{c} \in \mathbb{R}^n$ and a key $\vec{k} \in \mathbb{R}^n$, the decryption operation is to compute is $m = \langle \vec{c}, \vec{k} \rangle$.

B. Correctness

Before we discuss the properties of the encryption scheme, we shortly explain its correctness. Observe that an encryption $\vec{c}$ of a plaintext m is defined by a vector that meets the condition formalized in eq. (16). Furthermore, this equation describes exactly the decryption operation. Thus, a ciphertext of m is defined as a vector that decrypts to m. This shows the correctness.

It remains to discuss the existence of ciphertexts. In other words, given a plaintext m, we need to show that a vector $\vec{c}$ exists which fulfills eq. (16). As $\vec{k} \neq \vec{0}$ by definition, there exists a vector $\vec{v} \in \mathbb{R}^n$ such that $\langle \vec{v}, \vec{k} \rangle = \lambda \neq 0$. As $\lambda \neq 0$ and as $\mathbb{R}$ is a field, there exists $\lambda^{-1} \in \mathbb{R}$. Obviously, it holds for $\vec{c} = m \cdot \lambda^{-1} \cdot \vec{v}$ that eq. (16) is fulfilled. In particular, $\vec{c}$ is an encryption of m.

C. Homomorphic Properties

In the following, let $\vec{c} \in \mathbb{R}^n$ be an encryption of $m \in \mathbb{R}$, $\vec{c}' \in \mathbb{R}^n$ be an encryption of $m' \in \mathbb{R}$, and $\lambda, \lambda' \in \mathbb{R}$ be arbitrary real numbers. Then it holds for $\vec{c}'' := \lambda \cdot \vec{c} + \lambda' \cdot \vec{c}'$ that

$$\begin{aligned} \langle \vec{c}'', \vec{k} \rangle &= \langle \lambda \cdot \vec{c} + \lambda' \cdot \vec{c}', \vec{k} \rangle && (17) \\ &= \lambda \cdot \langle \vec{c}, \vec{k} \rangle + \langle' \vec{c}', \vec{k} \rangle && (18) \\ &= \lambda \cdot m + \lambda' \cdot m'. && (19) \end{aligned}$$

This shows that the scheme provides the homomorphic properties as described in Sec. III-B

D. Effort

In this section, we analyze the effort of the encryption scheme. The setup procedure is to sample a key $\vec{k} \in \mathbb{R}^n \setminus \{\vec{0}\}$. This can be done by sampling all n entries independently. For practical reasons, one will sample these from a limited range, e.g., $\{-5, -0.49, \ldots, 0.49, 0-5\}$. If s marks the size of the range, i.e., the number of values, then the probability to get a vector $\neq \vec{0}$ is $1 - s^{-n}$. This should be close enough to 1 in most practical use cases. In the case of need, one can repeat this procedure a few times until a correct key has been sampled.

The decryption operation is to compute the value $\langle \vec{c}, \vec{k} \rangle$ which is equivalent to compute n products and sum them up. More precisely, we need n multiplications and n summations.

It remains to investigate the encryption operation. The task is to find a vector $\vec{c}$ such that $\langle \vec{c}, \vec{k} \rangle = m$. To this end, we propose a precomputation step. Within this step, a basis for the kernel of $\langle \vec{k} \rangle$ is computed. As $\langle \vec{k} \rangle$ has the dimension 1, its kernel has the dimension $n-1$ and a basis $\vec{b}_1, \ldots, \vec{b}_{n-1}$ exists which can be computed with an effort in $\mathcal{O}(n^3)$ using Gaussian elimination. Recall that this implies

$$\langle \vec{k}, \vec{b}_i \rangle = 0 \tag{20}$$

for all $i = 1, \ldots, n-1$. Furthermore, we set

$$\vec{e} := \frac{1}{\langle \vec{k}, \vec{k} \rangle} \cdot \vec{k}. \tag{21}$$

The effort is linear in n. Observe that $\langle \vec{k}, \vec{k} \rangle \neq 0$ as $\vec{k} \neq \vec{0}$. Furthermore, it holds that $\langle \vec{e}, \vec{k} \rangle = 1$.

Given this, simple arguments from basic linear algebra show that an encryption of a message m can be computed by choosing random values $\lambda_1, \ldots, \lambda_{n-1} \in \mathbb{R}$ and to set

$$\vec{c} := m \cdot \vec{e} + \sum_{i=1}^{n-1} \lambda_i \cdot \vec{b}_i. \tag{22}$$

The effort is to sample $n-1$ elements in $\mathbb{R}$ and to do $\mathcal{O}(n^2)$ basic operations in $\mathbb{R}$.

E. Security

In this section, we analyze the security of the proposed encryption scheme. The probably most obvious observation is that each known pair of plaintext/ciphertext yields a linear equation in the unknown entries of $\vec{k}$. Thus, an attacker that has sufficiently many such pairs at his disposal could easily compute $\vec{k}$. The consequence is that the scheme becomes insecure as soon as too many fresh encryptions are available to the attacker. In classical cryptographic scenarios, this would immediately rule out the scheme. However, this attack does not apply to the use scenario considered here for the following two reasons.

First, recall that the encryption scheme is taken to encrypt the entries of the preference vector. Therefore, one knows in advance an upper bound on the number of ciphertexts. If the length of the key exceeds this bound, there will never be enough ciphertexts that allow for this attack. In case that a user wants to update his preference vector, we suggest to encrypt the result with another key.

Second, the whole purpose of the proposed scheme is to conceal the preference from the service provider. In case that an attacker would figure them out completely, there is nothing left that needs to be protected.

Therefore, we can restrict to an attacker that only knows

1) the encryptions of the d entries in p_i but not more ciphertexts and
2) some but not all value p_i in clear.

Given this, one sees easily that the knowledge of the ciphertexts gives only an underdefined system of linear equations. More precisely, assume that the attacker knows ℓ entries from $\vec{p_i}$. This gives ℓ linear equations in the secret entries of $\vec{k}$,

reducing the set of possible values for $\overrightarrow{k}$. Observe that these form a vector space of dimension $n - \ell$ if the entries of p_i and likewise the entries of $\overrightarrow{k}$ are all uniformly from $\mathbb{R}$. Thus, without any additional information, the attacker just has not enough information for determining the secret key $\overrightarrow{k}$.

Of course, in practice the entries are only taken from a certain range and the attacker may have some assumptions on $\overrightarrow{p_i}$, e.g., if one spot has been visited quite often, this probably holds for a nearby spot as well. These additional information could be cleverly combined to reduce the set of possible keys further. However, this effect can be thwarted by either choosing the range and/or the value n big enough. For example, if the size n of the key $\overrightarrow{k}$ is significantly bigger than the dimension d of $\overrightarrow{p_i}$, it will be hard in general to derive sufficient information $\overrightarrow{k}$ from ℓ known plaintext/ciphertext pairs. The concrete choice of the parameters highly depends on the use case and the aimed level of security. We leave the determination of appropriate parameters for concrete cases for future work.

V. Conclusion

This paper deals with recommendation systems for mobile social networks and their implications on the privacy of their users.

Location-based online social networks currently are a highly accepted service on the Internet, even though their benefit for users is rather limited. Devising a personalized recommender that suggests chosen places to the expected liking of the users could decisively change this situation. Existing recommendation schemes, however, generally come at the loss of privacy of their users. Even though privacy preserving schemes have been proposed, they are based on homomorphic encryption on integer values. Recommender systems calculating on real values, these schemes have to be adapted, and either their security or their correctness may be at risk.

This paper attempts first steps to designing a distributed recommendation scheme towards this end. It divides the user base into two groups of users. Plain users are using the recommendation scheme (free of charge, or by simply being active in the location-based online social network) at loss of their privacy, thus allowing the provider to establish a prediction model. Premium users, however, are enabled to learn a local, personal factor vector and receive recommendations from the system without disclosing any information to the provider, which in turn keeps his recommender model, which may represent a significant commercial asset, hidden from any third party. The system utilizes a new homomorphic encryption scheme, which is conceptually and computationally simple, efficient, and provably secure.

Acknowledgment

The authors would like to thank Betim Berjani and Markus Weimer for discussions about and suggestions for the recommendation scheme.

References

[1] B. Berjani and T. Strufe, "A Recommendation System for Spots in Location-Based Online Social Networks," in *Proceedings of the EuroSys Workshop on Social Network Systems*, 2011.

[2] Z. Erkin, M. Beye, T. Veugen, and R. Lagendijk, "Privacy enhanced recommender system," in *Thirty-first Symposium on Information Theory in the Benelux*, Rotterdam, 2010, pp. 35–42.

[3] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 26, no. 1, pp. 96–99, 1983.

[4] T. E. Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.

[5] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT*, 1999, pp. 223–238.

[6] I. Damgård, M. Geisler, and M. Krøigaard, "Efficient and secure comparison for on-line auctions," in *Proceedings of the 12th Australasian conference on Information security and privacy*, ser. ACISP'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 416–430. [Online]. Available: http://portal.acm.org/citation.cfm?id=1770231.1770269

[7] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC*, 2009, pp. 169–178.

[8] C. Gentry and S. Halevi, "Implementing Gentry's fully-homomorphic encryption scheme," Accepted to EUROCRYPT'11, 2011.