k-CoRating: Filling Up Data to Obtain Privacy and Utility

Feng Zhang[1], **Victor E. Lee**[2], and **Ruoming Jin**[3]
[1]School of Computer Science, China University of Geosciences, Wuhan, Hubei, China
{jeff.f.zhang}@gmail.com
[2]Department of Mathematics and Computer Science,
John Carroll University, University Heights, OH, USA
[3]Department of Computer Science , Kent State University, Kent, OH, USA

Abstract

For datasets in Collaborative Filtering (CF) recommendations, even if the identifier is deleted and some trivial perturbation operations are applied to ratings before they are released, there are research results claiming that the adversary could discriminate the individual's identity with a little bit of information. In this paper, we propose k-coRating, a novel privacy-preserving model, to retain data privacy by replacing some null ratings with "well-predicted" scores. They do not only mask the original ratings such that a k-anonymity-like data privacy is preserved, but also enhance the data utility (measured by prediction accuracy in this paper), which shows that the traditional assumption that accuracy and privacy are two goals in conflict is not necessarily correct. We show that the optimal k-coRated mapping is an NP-hard problem and design a naive but efficient algorithm to achieve k-coRating. All claims are verified by experimental results.

Introduction

Collaborative Filtering (CF) recommendation predicts which are the best items to recommend to active users based on the tastes of their like-minded users. Almost all the famous electronic commerce and social network websites have used recommender systems. Typical services appear like the follows: *Customers Who Bought This Item Also Bought .....*; *Today's Recommendations For You .....*; *People You May Know*, etc. CF is believed to be the most successful technique applied in recommender systems.

Typically, the input data for CF is a user-item rating matrix as shown in Table 1.

Table 1: Dataset Format of CF

	i_1	i_2	...	i_j	...	i_{m-1}	i_m
u_1	1	2	...	4	...	3	
u_2		2	...	5	...	5	4
...	...	...	...	...	...	...	...
u_i		5	...	4	...	1	4
...	...	...	...	...	...	...	...
u_n	5	2	...	1	...	2	5

u_i is ID of the user involved in a recommender system, and i_j is ID of the item rated by users. Each row in the rating matrix is a user profile represented by a rating vector, which is a list of the user's rating scores on a set of items.

In addition to traditional sparsity and scalability problems, CF has to address a big privacy challenge: even anonymous rating records could, in fact, be used to detect out personal identities (Polat and Du 2003; Canny 2002a; 2002b; Berkovsky et al. 2007). Arvind Narayanan et al. (Narayanan and Shmatikov 2008) demonstrated that they could easily re-identify anonymous rating records when they linked the publicly available IMDB to the anonymous Netflix prize database [1], and knowing a little bit about a record is enough for the re-identification, which is a serious privacy breach (we will call such attacks as *Narayanan's attacks* later in this paper). For this reason, some subscribers sued the organizers for the privacy breach. The dataset was forced to be retracted, and no follow-up contest has taken place.

To address such kind of privacy concerns, a general idea is to mask the original dataset properly before releasing it. There are many methodologies to do the masking: noises addition, data perturbation (Polat and Du 2003), etc. Differential privacy (McSherry and Mironov 2009; Hay et al. 2010; Mohammed et al. 2011; Lee and Clifton 2012; Kasiviswanathan et al. 2013; Ji and Elkan 2013; Mohammed et al. 2013) is a different method to achieve privacy. Instead of adding noises to the original data, it adds noises to the results. All such researches belong to Privacy-Preserving Collaborative Filtering (PPCF) (Berkovsky et al. 2007), an important component of Privacy-Preserving Data Mining (PPDM) (Vaidya, Clifton, and Zhu. 2005).

The common weakness of these studies is that they might lead to a poor recommendation quality or even untrustworthy recommendations compared with the CF conducted on an original rating matrix. The core challenge in PPDM is to maintain the recommendation performance while preserving data privacy since in general thinking, the two goals are in conflict (Brickell and Brickell 2008; Li and Li 2009).

We propose k-coRating, a novel privacy-preserving model to conduct CF properly and efficiently with the guar-

[1]http://www.netflixprize.com. a user-item ratings matrix which was used to run a $1 million grand-prize competition to select best algorithms to predict user preferences

antee of k-coRated privacy, i.e., ensuring that for each user in the rating matrix, there are at least k-1 other users rating the same set of items. Experimental results show that such model could greatly reduce risks of being subject to *Narayanan's attacks*.

Though it seems that k-coRating is similar to k-anonymity (Samarati 2001), k-coRating is designed with three major differences from k-anonymity. The first is that k-coRating handles with such dataset (typically the user-item ratings) that has a large number of attributes, any subset of which behaves in the sense of *quasi-identifier* arised from k-anonymity. The second is that k-coRated privacy rests in that each record, with at least k-1 ones, has and only has non-null values with respect to the same subset of attributes but they don't necessarily get the identical value under each attribute as k-anonymity requires. And the third is privacy of k-coRating is achieved through filling up necessary NULL cells with significant values, but not the generalization and suppression techniques.

Contributions

1. Inspired by k-anonymity, we introduce a novel model of k-coRating to preserve the privacy of rating-style datasets. To the best of our knowledge, it is *the first study to conduct CF with the result of higher recommendation accuracy while ensuring data privacy with a solid privacy model*.

2. We formally prove that finding the optimal k-coRated dataset is an NP-hard problem, and we employ a greedy strategy to design feasible algorithms to convert the ratings to a new dataset satisfying k-coRating.

3. To our knowledge, so far there are no countermeasures against Narayanan's attacks. We show empirically that the k-coRated privacy could greatly reduce the risks of suffering such attacks. We also explain how and why k-coRated privacy model protects against such attacks.

Collaborative Filtering Recommendation

The underlying rationale behind CF is the assumption that users like the items which their similar users like as well. To generate recommendations for an active user, the first step is to find its similar users, which are collectively called the nearest neighbors.

We use the popular measure, pearson correlation similarity, which is formalized as follows:

$$sim(u,v) = \frac{\sum_{i\in\Omega(u,v)}(R_{u,i}-\bar{R}_u)(R_{v,i}-\bar{R}_v)}{\sqrt{\sum_{i\in\Omega(u,v)}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum_{i\in\Omega(u,v)}(R_{v,i}-\bar{R}_v)^2}} \quad (1)$$

where $\Omega(u,v)$ denotes the set of items rated by both user u and user v; $R_{u,i}$ and $R_{v,i}$ represent user u's and user v's rating score respectively for item i; $\bar{R}_u$ and $\bar{R}_v$ are the average of ratings by user u and user v respectively.

Once the nearest neighbors, denoted with NBS_u, to an active user u is obtained, a typical way formalized as Equation (2) (Breese, Heckerman, and Kadie 1998) is used to predict user u's rating score for its unrated item i: $P_{u,i}$.

$$P_{u,i} = \bar{R}_u + \frac{\sum_{v\in NBS_u} sim(u,v)(R_{v,i}-\bar{R}_v)}{\sum_{v\in NBS_u} sim(u,v)} \quad (2)$$

The items who has the highest predicted scores are recommended to the user u.

Methodologies of PPCF

Problem Definitions

Filling data to alleviate the sparsity of a dataset and thus achieve an increased accuracy in CF is not a new idea. However, to the best of our knowledge, *applying filling data to privacy-preserving CF with better accuracy* is novel. Our study is devoted to this. Before articulating the idea in detail, we first formalize the problems in the following definitions.

Definition 2 (coRated Equivalence).
We define

$$u[i] = \begin{cases} 0 & \text{if } r_{u,i} \text{ is NULL} \\ 1 & \text{if } r_{u,i} \text{ is NOT NULL} \end{cases}$$

Two users $u, v \in U$ are of coRated Equivalence if for all $i \in I$, $u[i] = v[i]$.

Definition 3 (k-coRated Privacy).
A rating matrix M satisfies k-coRated privacy if every user $u \in U$ has at least k - 1 coRated Equivalent users, u_{t_1}, $u_{t_2}, \cdots, u_{t_{k-1}} \in U$.

We use $\overline{M}$ to represent a rating matrix satisfying k-coRated privacy.

It is obvious for any rating matrix M, there are many possible matrices satisfying k-coRated privacy. But in them there exist the optimal one(s).

Definition 4 (Optimal k-coRated Mapping). κ is a function which maps a rating matrix M to $\overline{M}$ to make it satisfy k-coRated privacy by filling up some NULL rating cells with specified values. The cost of the filling operation, $\mathcal{C}(\kappa)$, is the total number of filling values in $\overline{M}$. κ is an optimal k-coRated mapping iff $\mathcal{C}(\kappa)$ is minimized.

Theorem 1 (Optimal k-coRated Mapping is an NP-hard Problem, $k > 2$).

Proof Sketch: By reduction from **Edge Partition Into Triangles** (Kann 1994), the hardness of k-coRated Mapping could be proved. For the space limitation, we ignore the details in this version of paper.

Trusts Derivation

One critical concern is what values to be used to fill up the NULL cells to achieve k-coRating. In our experiments, we will evaluate the performances using four kinds of filling methods: trust derivation, pearson correlation similarity, item average ratings and random values.

We focus on deriving trusts to generate significant filling values. In recent years, this method has been used as an important technique to alleviate many critical problems in CF, such as the sparsity problem, the cold start problem, and to generate better recommendations (Golbeck 2005; Massa and Avesani 2007; Jamali and Ester 2009). In this paper, we try to compare it with other filling methods to evaluate the performance of balancing the privacy and utility.

The trust metric signifying how much user u trusts user p is formalized as Equation (3).

$$t_{u\to p} = \frac{1}{|I_u \cap I_p)} \sum_{i\in(I_u\cap I_p)} (1 - \frac{|P^p_{u,i} - R_{u,i}|}{m}) \quad (3)$$

where I_u and I_p refer to the set of rated items of u and p, respectively, and m is the size of the rating score range.

$P^p_{u,i}$ signifies user u's predicted rating score for item i by another user p.

$$P^p_{u,i} = \bar{R}_u + (R_{p,i} - \bar{R}_p) \quad (4)$$

We adopt the technique, *trust propagation* (Hwang and Chen 2007; Andersen et al. 2008) by inferring the indirect relationships, i.e., propagated relationships, between users. The inferred local trust score of u with respect to t through m is computed by the weighted average of the two direct relationships of $s \to m$ and $m \to t$.

$$\begin{aligned} t_{u\overset{m}{\to}t} &= t_{u\to m} \oplus t_{m\to t} \\ &= \frac{n(I_u \cap I_m)t_{u\to m} + n(I_m \cap I_t)t_{m\to t}}{n(I_u \cap I_m) + n(I_m \cap I_t)} \end{aligned} \quad (5)$$

Once the trust web is derived, it can be applied to rating score prediction. So Equation (2) can be generalized to be Equation (6). In the generalization, $sim(u, v)$ is replaced with $w(u, v)$. $w(u, v)$ may be $sim(u, v)$ or $t_{u\to v}$ if there are co-rated items between user u and v; otherwise, it is $t_{u\overset{m}{\to}v}$ if there are no co-rated items between user u and v.

$$P_{u,i} = \bar{R}_u + \frac{\sum_{v\in NBS_u} w(u, v)(R_{v,i} - \bar{R}_v)}{\sum_{v\in NBS_u} w(u, v)} \quad (6)$$

The Algorithms

Our algorithm is named as GeCom (**Ge**nerate k-**co**Rated **M**atrix), which is shown in Algorithm 1 and Algorithm 2. Algorithm 2 shows the whole picture of using greedy strategy to generate a k-coRated rating matrix. It divides the matrix into two parts: the already k-coRated one, M_1, and the non-k-coRated but sorted one, M_2. Then Algorithm 1 converts M_2 to a k-coRated matrix $\overline{M_2}$. It utilizes a heuristic that these rating vectors having most similar number of ratings and rating the most prior common items should be put together.

When the matrix is very large such as the Netflix prize dataset, we have to rely on parallel computing to get results. So a parallel version of GeCom, namely PaGeCom, is also implemented.

Complexity Analysis

M is a rating matrix with n vectors (users) and m items, and M is corresponding to the set of users, U. Suppose M_1 is corresponding to a set of users, U_1; and M_2 is corresponding to another set of users, U_2. $U_1 \bigcup U_2 = U$ and $U_1 \bigcap U_2 = \varnothing$. Suppose $|M| = n$, $|M_1| = n_1$ and $|M_2| = n_2$. ω is size of union of all users' rated items, i.e., $\omega = |\bigcup_{u\in U} I_u|$.

Algorithm 1: sub-GeCom: k-coRating an Already Sorted Matrix M_2

input : a positive integer k; a non-k-coRated matrix M_2, treated as a set of user-item rating vectors.
output: $\overline{M_2}$, a rating matrix satisfying k-coRated privacy.

Locate first vector V in M_2
while M_2 *is NOT NULL* **do**
  Set temporary set $T \leftarrow \phi$;
  while $|T| < k$ **do**
    Append V to T;
    Delete V from M_2;
    Locate first vector V in M_2;
  end
  while M_2 *is NOT NULL and* V *is coRated equivalent to the last element in* T **do**
    Append V to T;
    Delete V from M_2;
    Locate first vector V in M_2;
  end
  if $|M_2| < k$ **then**
    Append all the remaining vectors in M_2 to T;
    Delete all the remaining vectors in M_2;
  end
  Make set I by forming the union of all the items with non-null rating values in T;
  foreach $V \in T$ **do**
    foreach $i \in I$ *and* $r_{u,i}$ *is NULL in* V **do**
      $r_{u,i} \leftarrow$ the predicted value based on different filling methods);
    end
  end
  Append T to $\overline{M_2}$;
end

Algorithm 2: GeCom: Generate k-coRated Matrix $\overline{M}$

input : rating matrix M, treated as a set of user-item rating vectors; a positive integer k; a trust matrix, tM.
output: $\overline{M}$, a rating matrix satisfying k-coRated privacy.

Sort the rating vectors in M first by ascending order of number of ratings (the smaller, the further in front), and then by lexicographic order of items;
Divide M into two parts: M_1 satisfying k-coRating, and M_2 not satisfying k-coRating;
Set $\overline{M} \leftarrow M_1$;
Invoke Algorithm 1 to k-coRate the matrix, M_2, getting a k-coRated matrix $\overline{M_2}$;
Append Matrix $\overline{M_2}$ to $\overline{M}$, forming a new rating matrix $\overline{M}$ and printing it out to a file;

Algorithm 3: PaGeCom: A Parallel Algorithm to Generate k-coRated Matrix $\overline{M}$

input : rating matrix M, treated as a set of user-item rating vectors; a positive integer k; a trust matrix, tM; x, the number of processors.
output: $\overline{M}$, a rating matrix satisfying k-coRated privacy.

1 **foreach** *processor* P_i **do**
2 **Sort** the rating vectors in M first by ascending order of number of ratings (the smaller, the further in front), and then by lexicographic order of items;
3 **Divide** M into two parts: M_1 satisfying k-coRating, and M_2 not satisfying k-coRating;
4 **Accept** an average workload aroused by M_2. Use m_i to denote the workload assigned to processor P_i;
5 **Invoke** Algorithm 1 to k-coRate m_i, getting a k-coRated matrix $\overline{M_i}$, a sub rating matrix satisfying k-coRated privacy;
6 **end**
7 **foreach** P_i, $i > 0$ **do**
8 **Send** its $\overline{M_i}$ to P_0;
9 **end**
10 In P_0, **Combine** the received matrices and M_1 to form a rating matrix $\overline{M}$, printing it out to a file;

$\varpi = \sum_{u \in U} |I_u|$ is the number of non-NULL ratings in M. Similarly, $\varpi_1 = \sum_{u \in U_1} |I_u|$ is the number of non-NULL ratings in M_1, and $\varpi_2 = \sum_{u \in U_2} |I_u|$ is the number of non-NULL ratings in M_2.

Theorem 2 (Complexity of Algorithm 2). The upper bound of the complexity of Algorithm 2 is $O(\varpi log(n) + \varpi log(\omega) + 4\varpi + n\omega)$, which is independent of k.

Proof Sketch:

Ignored.

Lemma 1. The response time of step 18 in Algorithm 1 can be indicated with $f(k) = \sum_{i=1}^{\frac{n_2}{k}} (\varpi_i^k log(\omega_i^k))$. It is not a strictly monotone increasing function of k values, but if $k' > k$ and k' is a multiple of k, i.e., $k' = \rho k$ (ρ is an integer that is bigger than 1), then $\sum_{i=1}^{\frac{n_2}{k'}} (\varpi_i^{k'} log(\omega_i^{k'})) \geq \sum_{i=1}^{\frac{n_2}{k}} (\varpi_i^k log(\omega_i^k))$ must hold. $f(k)$ has a upper bound of $\varpi log(\omega)$.

Proof Sketch: Ignored.

Lemma 2. The response time of steps 19 to 23 in Algorithm 1 can be indicated with $g(k) = \sum_{i=1}^{\frac{n_2}{k}} (\varpi_i^k + k\omega_i^k)$. It is not a strictly monotone increasing function of k values, but if $k' > k$ and k' is a multiple of k, i.e., $k' = \rho k$ (ρ is an integer that is bigger than 1), then $\sum_{i=1}^{\frac{n_2}{k'}} (\varpi_i^{k'} + k'\omega_i^{k'}) \geq \sum_{i=1}^{\frac{n_2}{k}} (\varpi_i^k + k\omega_i^k)$ must hold. $g(k)$ has a upper bound of $\varpi + n\omega$.

Proof Sketch: Ignored.

Analysis of Privacy Preservation

The privacy is preserved because what the malicious users observe is not the original data but the hybrid one (the original data plus the predictions), they have no way to recover original data from the hybrid one with certainty.

Observation 1 Even if the underlying trust relationship is learned or it is published information, identities attached to ratings cannot be discriminated with certainty.

Proof Sketch: This is essentially to solve a multivariate system of equations. But the number of variates is more than the number of equations, and the relationship between variates are not linear, so it is impossible for an adversary to recover with certainty the original rating scores even it knows the trust relationship.

Observation 2 Even if adversaries who have uniquely linked one set of ratings to a specific individual, it is still impossible for them to make sure that they have made a successful attack.

Proof Sketch: Two cases to make this observation sound. The first is that the ratings used to link public data may be generated by the filling process, which results in a false positive detection; The second is, even if it is a true positive detection, since the user has been k-coRated, what the adversary sees is not the original ratings, but the hybrid one, so the privacy leakage about the subscriber is not so serious as Narayanan et al. claim (Narayanan and Shmatikov 2008)

Experiments

Experimental Design

Experiments have been done using four popular benchmark datasets: two Movielens datasets[2], one Epinions dataset[3] and one Netflix prize dataset[4]. General characteristics of these datasets are described in Table 2.

All the algorithms were implemented in C/C++. The implementation was conducted on a laptop of an Intel Core i7-2640M CPU 2.80GHz with 8GB RAM running on an Ubuntu 12.04 virtual machine with a host of Windows 8 64-bit operating system.

For the Netflix prize dataset, the laptop's computing resources were insufficient, so we implemented and ran a parallel version of the algorithm GeCom (Algorithm 3, PaGeCom) on the Ohio Supercomputer Center [5].

For the MovieLens 100K dataset, we used the prepared 80%/20% splits of the dataset, i.e., u1.base and u1.test through u5.base and u5.test, to do the 5-fold cross-validation experiments; and for other datasets, we used the 10-fold cross-validation method to evaluate the prediction accuracy.

For the trust derivation, we computed the propagation at most two times.

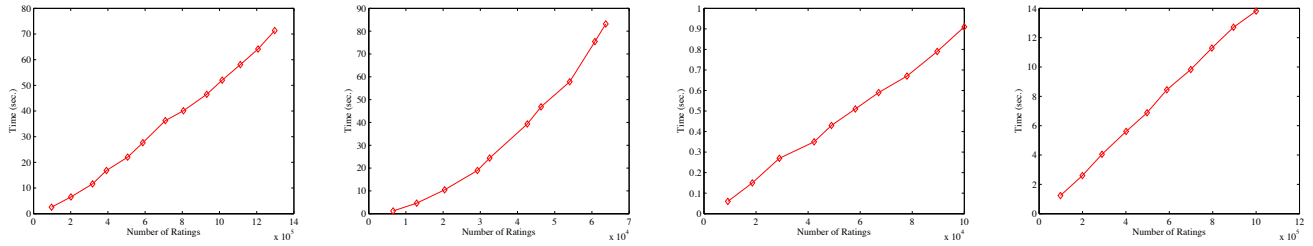
Evaluation Metrics

Root Mean Square Error (RMSE), was used to measure the prediction error, i.e., to measure the deviation between the

[2]http://www.grouplens.org/node/73
[3]http://www.epinions.com/
[4]http://www.netflixprize.com
[5]https://osc.edu/

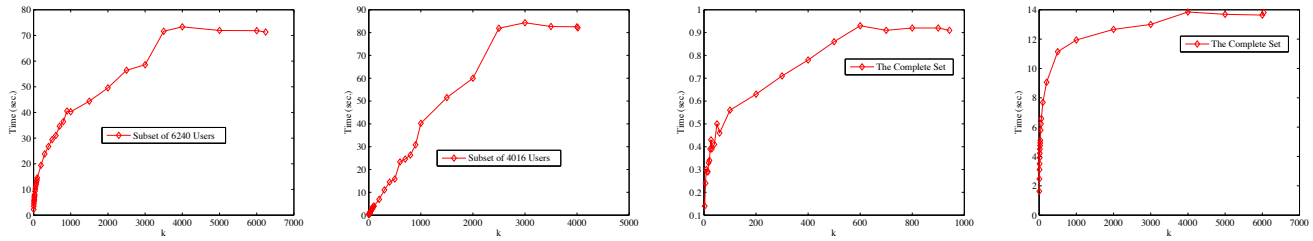
Table 3: RMSE VS k

	Netflix (0.86696)				Epinions (1.22974)				Movielens I (0.98417)				Movielens II (0.92648)			
	trusted	sim	average	random	trusted	sim	average	random	trusted	sim	average	random	trusted	sim	average	random
k=3	0.86598	0.86676	0.88490	0.89122	1.12785	1.13255	1.13771	1.30770	0.97838	0.97853	0.99564	1.02753	0.92374	0.92256	0.94218	0.95899
k=6	0.86498	0.86615	0.88983	0.91761	1.11453	1.11728	1.13563	1.28405	0.97788	0.97733	1.00389	1.04584	0.92359	0.92162	0.95251	0.97398
k=9	0.86476	0.86587	0.89344	0.91873	1.11212	1.11356	1.13770	1.26942	0.97750	0.97650	1.00792	1.05555	0.92298	0.92123	0.95560	0.97560
k=12	0.86465	0.86465	0.89879	0.92461	1.10512	1.11444	1.14032	1.26514	0.97735	0.97699	1.01078	1.05528	0.92233	0.92110	0.95839	0.97627
k=15	0.86449	0.86567	0.90231	0.93331	1.10872	1.11623	1.14370	1.26218	0.97780	0.97646	1.01207	1.05876	0.92192	0.92108	0.95967	0.97674
k=18	0.86421	0.86592	0.90982	0.93234	1.10987	1.11791	1.14645	1.26107	0.97818	0.97679	1.01365	1.06094	0.92200	0.92132	0.96044	0.97723
k=21	0.86478	0.86601	0.91234	0.93612	1.10914	1.11844	1.14647	1.26067	0.97851	0.97763	1.01538	1.05961	0.92179	0.92191	0.96121	0.97714
k=30	0.86629	0.86621	0.91873	0.94123	1.10998	1.12083	1.14990	1.25996	0.97874	0.97789	1.01807	1.06055	0.92241	0.92273	0.96275	0.97804
k=100	0.86986	0.87019	0.92343	0.95123	1.12506	1.13043	1.15493	1.26384	0.98251	0.98344	1.03448	1.05407	0.92792	0.93123	0.97147	0.98505



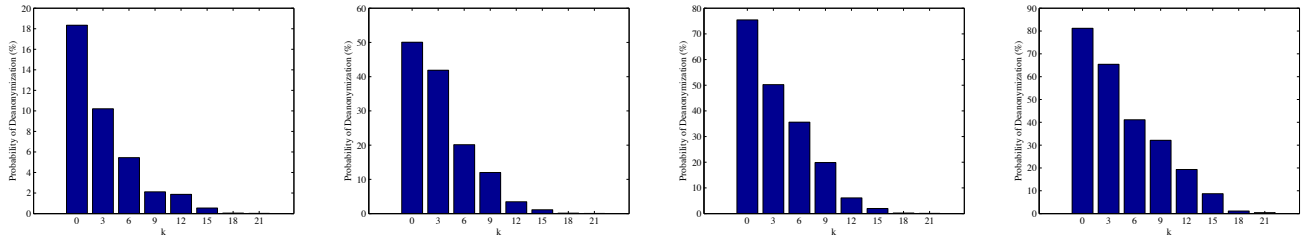
(a) Netflix (b) Epinions (c) Movielens I (d) Movielens II

Figure 1: GeCom: Time VS Number of Ratings



(a) Netflix (b) Epinions (c) Movielens I (d) Movielens II

Figure 2: GeCom: Time VS k



(a) 2-of-8 (b) 4-of-8 (c) 6-of-8 (d) 8-of-8

Figure 3: De-anonymization: Effect of k-coRating

items' predicted rating scores and actual rating scores.

- **RMSE**

Given the N actual/predicted pairs $(R_{u,i}, P_{u,i})$, the RMSE of the N pairs is computed as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(P_{u,i} - R_{u,i})^2}{N}} \quad (7)$$

where $P_{u,i}$ and $R_{u,i}$ hold the same meanings with those in Equation (2). It is obvious that the lower the RMSE values,

Table 2: General Dataset Description

dataset	# of ratings	# of users	# of items	sparsity
Movielens 1	100,000	943	1,682	99.9403%
Movielens 2	1,000,209	6,040	3,706	95.5316%
Epinions	664,824	40,163	139,738	99.9882%
Netflix	100,480,507	480,189	17,770	98.8224%

Table 4: Filling Methods Details

filling methods	details on the filling values
trusted k-coRating	the trusts between users[6]
sim k-coRating	the pearson correlation similarity[7]
average k-coRating	the average ratings for items
random k-coRating	random values in $\{1,2,3,4,5\}$

the more accurate the predictions, signifying a better performance of a recommender system.

Experimental Results

The experiments could be divided into three parts.

1. Experiments to show that k-coRating can lead to a better accuracy. We varied the values of k and compared the performances between the trust-based filling k-coRating with three other filling methods: correlation-based similarity rating values, average rating values and random rating values.

2. Experiments to show the response time of the algorithm GeCom. We showed how the response time varied with the number of ratings and the values of k increased correspondingly.

3. Experiments to show how much the k-coRated model can reduce the risks of identity breach.

We have done experiments to evaluate performance of the parallel PaGeCom algorithm, which scales well. For the space limitation, we do not report the results here.

The results of the *first part* are shown in Table 3. The experiments were run on the datasets, Netflix, Epinions, Movielens I and Movielens II, respectively. Four techniques were used as the filling methods. They are trust-based prediction (trusted k-coRating), pearson correlation similarity prediction (sim k-coRating), average rating values (average k-coRating) and random rating values (random k-coRating), which correspond to the four columns in each multi-column of Table 3: *trusted*, *sim*, *average* and *random*, respectively.

A detailed explanation about how the filling values are derived are shown in Table 4.

The results in Table 3 show that not all data filling methods guarantee better accuracy: trusted k-coRating and sim k-coRating perform best, better than the baseline for most k values; random k-coRating achieves the worst accuracy, worse than the baseline; while average k-coRating achieves the *average* accuracy, better than the random one but worse than its trusted and sim counterparts. This is the reason why generating "well-predicted" ratings is important. So rather than introducing random factors, we compute well-predicted ratings to strengthen recommendations. In addition, one noteworthy trend of these experiments is when the values of k are big enough, all the k-coRated approaches may perform worse than the original CF does. It seems to disclose that for the k-coRated model, when k grows, more data ought to fill in the matrix, and there may be a turning point where the filling data impose more negative impacts (act more like noises) rather than positive impacts (act more to decrease the sparsity) upon the accuracy.

Fig. 1 and Fig. 2 illustrate the results of the *second part* experiments. They show the performance (measured by response time) of GeCom. Fig. 1 shows how the response time varies as the number of ratings increases while the values of k are set to numbers of the users in the current datasets. And Fig. 2 shows how the response time varies as k increases at a fixed size of dataset. These results reveal the properties of Theorem 2, Lemma 1 and Lemma 2. The response time is not a strictly monotone increasing function of k values (in some cases we do see a bigger k value results in a smaller response time), but it definitely has the upper bound of $O(\varpi log(n) + \varpi log(\omega) + 4\varpi + n\omega)$.

To get Fig. 1a and Fig. 2a, we ran the experiments on a small subset of Netflix since the full dataset is too large for the memory and CPU power of the laptop. But the results illustrated in the two figures show us that they nicely follow up the same rules as their three counterparts do.

One of the most interesting results of k-coRating is that it can greatly reduce the risks of Narayanan's attacks demonstrated in (Narayanan and Shmatikov 2008). In Narayanan's attacks, an adversary knowing a little bit about an individual subscriber can easily identify it in the dataset. They used an algorithm called Scoreboard-RH (Algorithm 4) to achieve the attack.

We implemented the algorithm and simulated the experiments in (Narayanan and Shmatikov 2008) but ignored the dates. Then the results of the *fourth part* experiments are illustrated in Fig. 3. Similar to (Narayanan and Shmatikov 2008), ***n of m*** means an adversary knows n out of m ($n \leq m$) ratings of a subscriber, i.e., he gets m rating values of the subscriber, but only n values are correct and the other $m-n$ ratings are wrong.

All the four sub-figures in Fig. 3 show similar results: as k increases, the risks of identity being detected out are greatly reduced. When k increases from 0 to above 21, the probability of becoming a victim decreases dramatically from over 80% to below 0.1%. Such results have been encouraging since to the best of our knowledge, no other studies have been found to challenge the Narayanan's attacks.

In a summary of k-coRating, Table 3 shows that, with well-predicted filling values, it may provide a better recommendation accuracy while varying the values of k. Fig. 3 shows us it is able to greatly reduce the risks of suffering

[7] trusts generated based on equation 3, 5 and 6

[8] similarities generated based on equation 1 and 2

[9] $supp(i)$ is the number of subscribers who have rated item i; ρ_i is the rating of item i in the auxiliary information, aux, i.e., the information that the adversary knows and ρ'_i is the rating of item i in the candidate record $r', r' \in D$.

[9] ϕ is a fixed parameter called the eccentricity.

Algorithm 4: Scoreboard-RH (Narayanan and Shmatikov 2008)

input : the auxiliary information aux; the rating data D; parameters ρ_0, σ, and ϕ.
output: the matching record, or an empty set.

Compute Score: $Score(aux, r') = \sum_{i \in supp(aux)} wt(i) Sim(aux_i, r'_i)$ where $wt(i) = \frac{1}{log|supp(i)|}$ and $Sim(aux_i, r'_i) = e^{\frac{-|\rho_i - \rho'_i|}{\rho_0}}$[8];

Compute $max = max(S)$, $max_2 = max_2(S)$ and $\sigma = \sigma(S)$, i.e., the highest and second-highest scores and the standard deviation of the scores, where $S = \{Score(aux, r') : r' \in D\}$;

if $\frac{max - max_2}{\sigma} < \phi$[9] **then**

  Return $\varnothing$, denoting there is no match;

else

  Return the matching record with the highest score;

end

the Narayanan's attacks. Fig. 1 and Fig. 2 illustrate running performance of the algorithms GeCom, which are also very encouraging. From the experimental results and analysis, we can conclude that our method has the advantage of preserving privacy with a better recommendation accuracy.

In contrast, to the best of our knowledge, the previous privacy-preserving data mining studies employing perturbation method cannot avoid the weakness which inevitably hurts the recommendation accuracy (data utility). (Polat and Du 2003) and (Berkovsky et al. 2007) are two examples. In (Polat and Du 2003), one experimental result shows that the values of MAE (a metric whose value increases/decreases along with RMSE) increases from 0.04 to 0.12, and another one shows it increases from 0.07 to 0.22 when a parameter, γ, partially denoting the amount of privacy increases from 50% to 95%. In (Berkovsky et al. 2007), experimental results show that the impact of specific obfuscation policies are similar: MAE of the prediction increases linearly with the obfuscation rate, which refers to how much privacy is preserved.

Related Work

As stated in the Section of Introduction, data perturbation and differential privacy are two major privacy-preserving methods in data mining. The two methods are applied to CF as well (Polat and Du 2003; Berkovsky et al. 2007; McSherry and Mironov 2009). Another closely related technique is the k-anonymity (Samarati 2001) (and the subsequent ℓ-diversity (Machanavajjhala et al. 2007) and t-closeness (Li, Li, and Venkatasubramanian 2007)) etc. Compared with k-anonymity, k-coRating is applied to a different type of data and thus has a different application scenario; moreover, k-coRating is achieved with a different approach.

Encryption/decryption is another important technique used in PPCF as well as in PPDM (Vaidya, Clifton, and Zhu. 2005; Zhan et al. 2010). With this kind of technique, original data must be encrypted before being utilized. Although such technique can ensure recommendation accuracy, encryption/decryption itself is very time-consuming and is a technique applied in distributed computing scenario. Moreover, since many data mining computations cannot be achieved by encryption/decryption and it is unpractical to encrypt/decrypt large datasets, so its application scopes are restricted.

There are many research results on privacy-preserving CF. John Canny is widely credited with pioneering the problem of privacy-preserving CF (Canny 2002a; 2002b). In (Canny 2002a), CF recommendations are conducted based on Singular Value Decomposition (SVD) and Maximum Likelihood techniques; Canny reduced the CF process to the repeated summary of rating data vectors, so as to preserve the privacy of data by Homomorphic Cryptography. The privacy preserving techniques used in (Canny 2002b) are similar to the ones used in (Canny 2002a), except that (Canny 2002b) generates recommendations based on Expectation Maximization (EM) factor analysis. SVD, as well as factor analysis, might cause losses of information, which will result in poor recommendations. Similar to the above two studies, other efforts on privacy-preserving CF, like estimated concordance measures (Lathia, Hailes, and Capra 2007) and random perturbation slope one (Basu, Vaidya, and Kikuchi 2012), are conducted on the distributed scenario or on the cloud, which are different from our centralized CF model.

Conclusion

The major contribution of this paper is showing that the traditional assumption (Brickell and Brickell 2008; Li and Li 2009) that accuracy and privacy are two goals in conflict is not necessarily correct. k-coRating is presented as a way to achieve both higher utility and privacy. Both goals are achieved by the filling data. The idea is simple but effective.

Interestingly and importantly, k-coRating is empirically proved to be an elegant method to protect subscribers from being victims of the well known Narayanan's attacks. The rationale is straightforward: the filling values not only act as noise to protect subscribers from being identified to be victims but also help hide the actual ratings in the rare circumstance that subscribers become victims.

Though preliminary results verifying our points have been obtained, we believe k-coRating is an elegant privacy model that is worthy of more efforts. k-coRating is an effective way to fight back Narayanan's attacks, but it still cannot ensure 100% privacy so far. For the subsequent work, maybe the ℓ-diversity and t-closeness model could give some inspiration to further improve the privacy.

Acknowledgment

This work was supported in part by the following grants: NSF CAREER grant IIS-0953950, NIH/NIGMS grant R01GM103309, OSC (Ohio Supercomputer Center) grant PGS0218, CSC scholarship 2011842045. The authors would like to express their appreciations to Prof. Gansen Zhao for

his insightful suggestions. The authors also want to thank the anonymous reviewers for their invaluable comments.

References

Andersen, R.; Borgs, C.; Chayes, J.; Feige, U.; Flaxman, A.; Kalai, A.; Mirrokni, V.; and Tennenholtz, M. 2008. Trust-based recommendation systems: an axiomatic approach. In *Proceedings of the 17th international conference on World Wide Web*, 199–208.

Basu, A.; Vaidya, J.; and Kikuchi, H. 2012. Perturbation based privacy preserving slope one predictors for collaborative filtering. In *Proceedings of the 6th IFIP WG 11.11 International Conference on Trust Management*, 17–35.

Berkovsky, S.; Eytani, Y.; Kuflik, T.; and Ricc, F. 2007. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *Proceedings of the 2007 ACM conference on Recommender systems*, 9–16.

Breese, J. S.; Heckerman, D.; and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 43–52.

Brickell, J., and Brickell, J. 2008. The cost of privacy: Destruction of data-mining utility in anonymized data publishing. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 70–78.

Canny, J. 2002a. Collaborative filtering with privacy. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, 45–57.

Canny, J. 2002b. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 238–245.

Golbeck, J. A. 2005. *Computing and Applying Trust in Web-based Social Networks*. Ph.D. Dissertation, University of Maryland College Park.

Hay, M.; Rastogi, V.; Miklau, G.; and Suciu, D. 2010. Boosting the accuracy of differentially private histograms through consistency. In *Proceedings of the International Conference on Very Large Data Bases*, 1021–1032.

Hwang, C.-S., and Chen, Y.-P. 2007. Using trust in collaborative filtering recommendation. In *Proceedings of the 20th International Conf. on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 1052–1060.

Jamali, M., and Ester, M. 2009. Trustwalker: A random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 397–405.

Ji, Z., and Elkan, C. 2013. Differential privacy based on importance weighting. *Machine Learning* 93(1):163–183.

Kann, V. 1994. Maximum bounded h-matching is max snp-complete. *Information Processing Letters* 49(6):309–318.

Kasiviswanathan, S. P.; Nissim, K.; Raskhodnikova, S.; and Smith, A. 2013. Analyzing graphs with node differential privacy. In *Proceedings of the 10th theory of cryptography conference on Theory of Cryptography*, 457–476.

Lathia, N.; Hailes, S.; and Capra, L. 2007. Private distributed collaborative filtering using estimated concordance measures. In *Proceedings of the 2007 ACM conference on Recommender Systems*, 1–8.

Lee, J., and Clifton, C. 2012. Differential identifiability. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1041–1049.

Li, T., and Li, N. 2009. On the tradeoff between privacy and utility in data publishing. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 517–526.

Li, N.; Li, T.; and Venkatasubramanian, S. 2007. t-closeness: Privacy beyond k-anonymity and ℓ-diversity. In *Proceedings of the 23rd International Conf. on Data Engineering*, 106–115.

Machanavajjhala, A.; Kifer, D.; Kifer, D.; and Venkitasubramaniam, M. 2007. ℓ-diversity: Privacy beyond k-anonymity. *ACM Transaction on Knowledge Discovery from Data* 1(1):1–52.

Massa, P., and Avesani, P. 2007. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, 17–24.

McSherry, F., and Mironov, I. 2009. Differentially private recommender systems: Building privacy into the net. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 627–636.

Mohammed, N.; Chen, R.; Fung, B. C.; and Yu, P. S. 2011. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 493–501.

Mohammed, N.; Alhadidi, D.; Fung, B. C. M.; and Debbabi, M. 2013. Secure two-party differentially private data release for vertically-partitioned data. *IEEE Transactions on Dependable and Secure Computing* PP(99):In Press.

Narayanan, A., and Shmatikov, V. 2008. Robust de-anonymization of large sparse datasets. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, 111–125.

Polat, H., and Du, W. 2003. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, 625–628.

Samarati, P. 2001. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering* 13(6):1010–1027.

Vaidya, J.; Clifton, C. W.; and Zhu., Y. M. 2005. *Privacy Preserving Data Mining (Advances in Information Security)*. Springer.

Zhan, J.; Zhan, J.; Wang, I.-C.; Hsu, T.-S.; Liau, C.-J.; and Wang, D.-W. 2010. Privacy-preserving collaborative recommender systems. *IEEE Transaction on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 40(4):472–476.