AN IMPROVED PRIVACY-PRESERVING COLLABORATIVE FILTERING RECOMMENDATION ALGORITHM

Jingqi Zhang, School of Information, Central University of Finance and Economics, Beijing, China, oliazhang@126.com
Jianming Zhu, School of Information, Central University of Finance and Economics, Beijing, China, tyzjm65@163.com
Ning Zhang, School of Information, Central University of Finance and Economics, Beijing, China, zhangning75@sina.com

Abstract
Privacy-preserving collaborative filtering is an emerging web-adaptation tool to cope with information overload problem without jeopardizing individuals’ privacy. However, Collaborative filtering with privacy schemes commonly suffers from scalability and sparseness. Moreover, applying privacy measures causes a distortion in collected data, which in turn defects accuracy of such systems. In this work, the concept of privacy-preserving intensity weight, its measurement and an improved method of similarity calculation are introduced to solve the accuracy decreasing problem of the Randomized Perturbation Techniques (RPT) based recommendation algorithm. A new formula of similarity is proposed which considers both users’ rating similarity and the level of perturbation. Experimental results show that the improved algorithm outperforms the initial one in accuracy without affecting the effectiveness of privacy protection.

Keywords: Collaborative filtering, Privacy preserving, Randomized perturbation, privacy-preserving intensity weight.

1 This work was supported by the National Natural Science Foundation of China under Grant 61272398, the National Social Science Foundation of China under Grant 13AXW010, Discipline Construction Foundation of Central University of Finance and Economics.
1 INTRODUCTION

Increasing attention to online facilities introduced by the Internet leads an unnecessary access to a burden of material, which is contemporarily called information overload. Especially in E-commerce, huge amount of products are exposed in front of consumers, which provides consumers with more choices, but on the contrary, also increase their cost of searching products. In fact, consumers’ need is vague, which also lead to an increasing of difficulty to find the product they want. Consequently, online vendors start offering automated product recommendation services to boost sales in online stores. There has been a number of ways to produce automated referrals including content-based, collaborative and knowledge-based techniques (R. Burke, 2002; P. Melville, R.J. Mooney, R. Nagarajan, 2002; W. Ziqiang, F. Boqin, 2004). One of the most successful techniques is collaborative filtering (CF) (G. Chen, F. Wang, C.S. Zhang, 2009; P. Symeonidis, A. Nanopoulos, A.N. Papadopoulos, Y. Manolopoulos, 2008). There are also real-life deployments adopting this technique such as Amazon.com and Last.fm (Symeonidis, P., Nanopoulos, A., Papadopoulos, A. N., & Manolopoulos, Y., 2008).

However, unlike conventional transactions, the development of electronic commerce has changed the relationship between Internet merchants and consumers. Consumers cannot remain anonymous in Internet transactions as their data are revealed to Internet merchants easily and unconsciously, which is a great threat to their privacy and benefit. As risks of online shopping, such as profiling users, unsought marketing, price discrimination, being subject to government surveillance, and so on, are much get to be known, the privacy issues have been attracting more attentions from consumers. As a result, customers hesitate to submit their authentic preference, or even give false data, which makes it difficult to estimate dependable referrals. In order to overcome this challenge, some privacy-preserving schemes to produce predictions without jeopardizing privacy are proposed (H. Polat, W. Du, 2005; J. Canny, 2002; S. Berkovsky, Y. Eytani, T. Kuflik, F. Ricci, 2007; Alper Bilge, Huseyin Polat, 2012).

Providing privacy measures within CF applications can solve this problem to a large degree. However, some problems emerge concerning the use of privacy-preserving collaborative filtering (PPCF) systems (G. Chen, F. Wang, C.S. Zhang, 2009; S. Berkovsky, Y. Eytani, T. Kuflik, F. Ricci, 2007; Alper Bilge, Huseyin Polat, 2012). First, scalability problem, which refers to that as more people get oriented in such applications and online vendors supplement new products, and grows and it gets harder to expand such systems. Second, privacy measures provided by such systems require extra computational and storage costs that contribute to the scalability issues. Third, originating from constantly growing nature of PPCF, user–item matrices are generally highly sparse. Fourth, the accuracy of recommendation is inevitably influenced due to privacy-preserving measures.

In this study, the concept of privacy-preserving intensity weight, its measurement and an improved method for similarity calculation are introduced to solve the accuracy decreasing problem of the Randomized Perturbation Techniques (RPT) based recommendation algorithm. And by merging users’ ratings into Feature-based profiles (FBPs) (Bilge, A., & Polat, H., 2010; Bilge, A., & Polat, H., 2011), we can reduce typically large and sparse user vectors into compact models, which can relieve the problem of scalability and sparseness of CF.

2 RELATED WORK

Recommendation system is an Information Filtering scheme used to decrease the additional cost of searching information. Gediminas and Alexander provided a formalized definition of Recommendation system: let \( C \) be the set of all users and let \( S \) be the set of all possible items that can be recommended. The space \( S \) of possible items can be very large, ranging in hundreds of thousands or even millions of items in some applications. Let \( u \) be a utility function that measures the usefulness
of item s to user c, i.e., \( u : C \times S \rightarrow R \), where R is a totally ordered set. Then, for each user \( c \in C \), we want to choose such item \( s' \in S \) that maximizes the user’s utility. More formally:
\[
\forall c \in C, \ s' = \arg \max_{s \in S} u(c, s)
\] (1)

In recommender systems, the utility of an item is usually represented by a rating, which indicates how a particular user liked a particular item.

Based on the formalized definition of Recommendation system, we try to provide a more specified formal expression to clarify its nature. Assuming that the recommendation system including \( M \) users and \( N \) items, define the set of users is \( U = \{ u_m \mid n = 1...M \} \), and define the set of items is \( I = \{ i_n \mid n = 1...N \} \). As the core issue, the costumers’ preference of items needs to be clarified.

Define preference matrix is \( P_{M \times N} \), and its elements is \( p_{mn} \), the user \( u_m \)'s preference of item \( i_n \). If the user \( u_m \)'s preference of item \( i_n \) is not recorded, then \( p_{mn} = \text{Null} \).

Assume that the utility function of preference: \( \text{pref} : \text{user} \times \text{item} \times \text{evidence} \rightarrow \text{preference} \), in which \( p_{mn} = \text{pref}(u_m, i_n, \text{evidence}) \). \( \text{evidence} \) means the evidence for the recommendation result.

For some algorithm, it means intermediate result. Therefore, the recommendation result can be defined as follows:
\[
\forall u_x \in U, i_{\text{top}} = \max_{i_{\text{top}}} \left[ \text{pref}(u_x, i_{\text{top}}, \text{evidence}) \right]
\] (2)

The preference function is used to measure preference. And how to define the preference function will specify it as different recommendation algorithms. In order to compare different recommendation algorithms, some advantages and disadvantages of them are listed in Table 1.

<table>
<thead>
<tr>
<th>algorithms</th>
<th>classification</th>
<th>advantages</th>
<th>disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule-based recommendations</td>
<td>Rule-based</td>
<td>• Simple and direct</td>
<td>• Difficult to extract rules, time-wasting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Find new interests</td>
<td>• Difficult to manage the system</td>
</tr>
<tr>
<td>Content-based recommendations</td>
<td>Information filtering-based</td>
<td>• Direct result</td>
<td>• Difficult to distinguish the quality of recourse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Need less knowledge about this field</td>
<td>• Difficult to new interests</td>
</tr>
<tr>
<td>Collaborative recommendations</td>
<td></td>
<td>• Find new interests</td>
<td>• Difficult to depict users’ preference reliably</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Sparsity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Scalability</td>
</tr>
</tbody>
</table>

Table 1. Comparison of different recommendation algorithms

Due to privacy risks of online shopping, PPCF is becoming more popular. First approaches to build privacy measures on CF applications are distributed solutions by Canny, relying on formation of an aggregate data using cryptographic techniques to hide confidential data in distributed environments. Central server-based applications are more popular, where individuals submit their preferences after perturbing up to a level for concealing their actual ratings and rated products. Data obfuscation methods, such as randomized perturbation techniques (RPTs), randomized response techniques (RRTs), and data substitution, are employed to perform filtering processes without violating privacy. However, recovering accuracy losses due to privacy preservation process is not extensively studied. In this study, we focus on overcoming sparsity challenge as well as producing qualified predictions with comparable accuracy to non-private schemes.
3 PRELIMINARIES

3.1 Collaborative Filtering Recommendation Algorithm

The basic idea of Collaborative Filtering Recommendation Algorithm is to recommend items according to other customers’ ranking of items. Specific for, it is based on two assumptions: 1) customers’ rankings of similar items are similar and 2) customers with similar preferences have similar rankings of items. CF prediction estimation can be thought as a two-step: 1) locating neighbours by computing similarities between $x$ and all other users $y$ in the system and 2) estimating a weighted prediction based on preferences of neighbours. There are various methods to calculate such similarity. Most typically employed measure, Pearson’s correlation coefficient (PCC) is given as:

$$sim(x, y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x)(r_{ys} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}'} (r_{xs} - \bar{r}_x)^2 \sum_{s \in S_{xy}'} (r_{ys} - \bar{r}_y)^2}}$$

(3)

Where $r_{sx}$ is the rating for item $s$ by user $u$, $\bar{r}_c$ is the average rating of user $c$ and $S \subseteq \{s \in S | c \neq \phi\}$, and $S$ is the set of items.

Assuming that $v_{ij}$ is the score of item $j$ from user $i$, $\bar{v}_i$ and $\sigma_i$ are the mean and variance of user $i$’s rating vector, then CF algorithm is based on two calculations: 1) define $z$ to normalize the score in user’s rating vector, where

$$z_{ij} = \frac{(v_{ij} - \bar{v}_i)}{\sigma_i}$$

(4)

and 2) the recommendation ($p_{aq}$) is given as

$$p_{aq} = \bar{v}_a + \sigma_a \frac{\sum_{i=1}^n \omega_{ai} \cdot z_{iq}}{\sum_{i=1}^n |\omega_{ai}|}$$

(5)

$$\omega_{ai} = \sum_k z_{ak} \cdot z_{ik}$$

(6)

where $p_{aq}$ is the predicated score for user $a$ of item $q$; $\omega_{ai}$ is the similarity between user $a$ and the recommended users $i$ and $z_{iq}$ is the normalized score of item $q$ from user $i$.

3.2 Feature-based profiles (FBPs)

The key step for CF algorithm is to calculate the similarity between any two users, which relies on their commonly rated items as seen from Eq (3). However, the portion of co-rated items seems become smaller relatively due to the constantly growing nature of products. Therefore, the sparsity problem comes, which will lead to unreliable similarities. To avoid this problem, we propose to employ merging the rating/preference into FBPs to reduce the large and sparse user vectors into compact models (Bilge, A., & Polat, H., 2011). Instead rate items directly, in FBPs users rate the features of products. Although products are independent, they correlate among themselves as they have common features, where number of such features is much less than that of products. If such features are
determined, then the histograms of absolute frequencies can be produced by giving a weight to corresponding features of a rated item. Those weights can be given according to users’ preference or simply equally to each other (Bilge, A., & Polat, H., 2011). If user $c$ has rated item $i$, then the value of features of item $i$ in FPBs vectors will be added with the scores correspondingly, as described in Algorithm 1.

**Input:** user rating matrix $RP[m]$, feature matrix of products $Feature[m][n]$

**Output:** FBPs

1: function FBP($RP[m]$, $Feature[m][n]$)

Initialize:
2: $FBP[n] \leftarrow 0$

Calculate FBP:
3: for $i=1:m$
4: if $RP[i] \neq \emptyset$ then
5: for $j=1:n$
6: if $Feature[i][j] \neq \emptyset$ then
7: $FBP[j] += RP[i]$
8: else $j++$
9: end if
10: end for
11: else $i++$
12: end if
13: end for

**Algorithm 1. FBP**

To understand the FBP produce procedure, a small user-item matrix is given in Table 2, including ratings for songs in a 5-star scale, where $\times$ indicates an unrated item. Also, suppose that the genre features of songs are provided as having at least one from the feature set {rock, jazz, lyric, electronic, rap, blues} in Table 3. FBPs of users’ are given in Figure 1.

<table>
<thead>
<tr>
<th>User</th>
<th>Item</th>
<th>s1</th>
<th>s2</th>
<th>s3</th>
<th>s4</th>
<th>s5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td></td>
<td>5</td>
<td>$\times$</td>
<td>3</td>
<td>$\times$</td>
<td>1</td>
</tr>
<tr>
<td>Bob</td>
<td></td>
<td>2</td>
<td>1</td>
<td>$\times$</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 2. User-item matrix**

<table>
<thead>
<tr>
<th>Item</th>
<th>rock</th>
<th>jazz</th>
<th>lyric</th>
<th>electronic</th>
<th>rap</th>
<th>blues</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>s2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>s5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 3. Item-feature matrix**
Figure 1. FPs of Alice and Bob

According to Algorithm 1, Alice’s FBP is [9, 5, 3, 0, 5, 0] and Bob’s FBP is [3, 3, 3, 4, 2, 3]. Moreover, in the real-world data sets, every user rates different number of items and similarly number of features might vary from different items. Therefore, the obtained FBPs should be normalized because they might fluctuate. We assign each feature the same weight and the FBPs are updated as [9/22, 5/22, 3/22, 0, 5/22, 0] for Alice and [3/18, 3/18, 3/18, 4/18, 218, 3/18] for Bob.

3.3 Privacy protection by randomization

The accuracy and privacy is two important issues of recommendation system. However, there is inherent conflict between them. High quality predictions can only be produced upon authentic data. While privacy-preserving schemes generally require a level of distortion in user profiles, which inevitability leads to accuracy loss. RPT is such a technique that the privacy parameters can be well-tuned as not allowing the server to extract valuable information from user profiles and yet still be able to produce precise predictions. It protects privacy by applying a preferred level of distortion on data to hide individual preference and conceal list of rated items. Specifically, it masks personal data by randomly perturbing each vote in the profile and randomly fills some fraction of the empty cells (Polat, H., & Du, W. 2005).

In terms of PPCF, RPTs offer to disguise a vote entry $v$ by replacing it with $r + v$, where $r$ is a random number drawn from either a uniform or Gaussian distribution with mean ($\mu$) being zero and a standard deviation ($\sigma$).

The value of $\sigma$ can control the range of produced random numbers and thus control the level of distortion. Also, users insert additional random numbers to $\beta$ % of the empty cells, which are uniformly randomly chosen as fake ratings. And the number of fake rating cells is determined according to the density of the user-item matrix. The algorithm of PPCF based on RPT is given in Algorithm 2.

```
Input: rating matrix A, target user a, item q
Output: target user a’s predicted score, $v_{aq}$, of item q

1: begin
2:   $Z \leftarrow z_{scores}(A)$
3:   for i=1 to m do

```
Algorithm 2. RPTCF

4 AN IMPROVED PRIVACY-PRESERVING COLLABORATIVE FILTERING ALGORITHM

4.1 Privacy-preserving intensity

In PPCF, the recommendation system uses distorted data and fake rating as real rating data to predict. However, the more the perturbation intensity is, the less the accuracy will be. Therefore, it is inconsiderate to calculate similarities using perturbed data simply. By adjusting similarity equation and building calculation model accordingly, privacy-preserving intensity can make it balance between privacy and accuracy.

Definition 1 (Privacy-preserving intensity, PPI) In the RPT applied in PPCF, the random numbers are generated with zero mean and $\sigma$ in Gaussian distribution. And random numbers are generated over the range $[-\alpha, +\alpha]$, where $\alpha$ is a constant and $\sqrt{3}\sigma$ is in uniform distribution, in which $\sigma$ is defined as perturbation intensity. Besides, the value of $\beta$ also influences the level of perturbation. Therefore, the privacy-preserving intensity is in direct proportion to two factors, $\sigma$ and $\beta$. Then the privacy-preserving intensity is given as

$$PPI_i = \frac{\beta_i}{100} e^{\sigma_i}, \beta_i \in [0,100], \sigma_i \in (0, +\infty]$$

(7)

Definition 2 (Privacy-preserving intensity weight, PPIW) Privacy-preserving intensity weight is decided by the privacy-preserving intensity, given as

$$PPIW_i = \frac{1}{PPI_i} = \frac{100}{\beta_i} e^{-\sigma_i}, \beta_i \in [0,100], \sigma_i \in (0, +\infty]$$

(8)

where $PPIW_i$ is the privacy-preserving intensity weight between $[0,1]$, $\sigma_i$ is the standard deviation of random numbers, and $\beta_i$ is the percentage of inserted empty cells. Therefore, privacy-preserving intensity weight is inversely proportional to privacy-preserving intensity.

Definition 3 (Adjusted similarity) Adjusted similarity considers both similarity and privacy-preserving intensity weight, given as
\[ sim_{ai} = \omega_{ai} \times PPIW_i, \quad PPIW_i \in [0,1] \]  

where \( \omega_{ai} \) is the original similarity and \( W_i \) is the user \( i \)'s and privacy-preserving intensity weight. Then the predicted-rating equation, Eq.(5), is adjusted accordingly as

\[
p_{aq} = \bar{v}_a + \sigma_a \cdot \sum_{i=1}^{n} \frac{sim_{ai} \cdot z_{iq}}{|sim_{ai}|}
\]

### 4.2 Improved PPCF algorithm with privacy-preserving intensity weight

The algorithm is deployed on client-server structure and users communicate with each other directly with central server directly. Specific steps are as follows:

- **Step 1** Target-user provides request for recommendation

  Target-user decides his/her rating items to be calculated and privacy-preserving intensity weight (\( \sigma, \beta \)) according to his/her privacy preference. Then send the perturbed rating items and privacy-preserving intensity weight to server.

- **Step 2** Server provides recommendation

  After the server receives the data from user, it calculates the adjusted-similarity with the concealed-data and sends the recommendation result back to user.

- **Step 3** Recommendation feedback

  Target user gets recommendation result from the server.

The algorithm of improved PPCF is given in Algorithm 3.

---

**Input:** target-user a, predicting items q  
**Output:** target-user a’s predicting score, \( p_{aq} \), of items q

1: begin  
2: Itema[{item1, item2,..., itemn}] ← getItem(a)  
3: \( Z_a \leftarrow z_{scores} \)(Itema)  
4: ItemaP[{item1, item2,..., itemN}] ← getItemP(Za)  
5: \( R_a(r1, r2,...rN) \leftarrow normrnd(0, \sigma_a, N) \)  
6: or \( R_a(r1, r2,...rN) \leftarrow Unifrnd(N, -\sqrt{\sigma_a}, -\sqrt{\sigma_a}) \)  
7: \( Z_a^{'} \leftarrow \{\text{Itema, ItemaP, } R_a\} \)  
8: for i=1 to m do  
9: if (server responds) then  
10: UserM ← getUser(i)  
11: end if  
12: end for  
13: for every user in UserM do  
14: repeat line2~8  
15: \( \omega_{ai} \leftarrow \text{col}(Z_a^{'} \cdot Z_i) \)  
16: end for  
17: end for

---
16: \( W_i \leftarrow \text{Weight}(\sigma_i, \beta_i) \)

17: \( \text{sim}_{ai} \leftarrow \text{Sim}(\omega_{ai}, W_i) \)

18: \( \text{sim\_list} \leftarrow \text{sim}_{ai} \)

19: \( \text{end for} \)

20: \( \text{Neu} \leftarrow \text{Sort(sim\_list, k)} \)

21: \( \text{use Eq.(9) to predict the rating scores of item q} \)

22: \( \text{end} \)

Algorithm 3. Improved PPCF

5 EXPERIMENT AND ANALYSIS

5.1 Experiment dataset and parameters

In order to eliminate the contingency and to test the practicability of this algorithm, we conducted experiments on two dataset, Jester and MovieLens, two accepted dataset in the field of recommendation. These two dataset are different from each other in many ways, like the numbers of users, the interval of score and the sparsity, which are showed in table 4.

<table>
<thead>
<tr>
<th>dataset</th>
<th>user</th>
<th>Number of product</th>
<th>Ranked items</th>
<th>Score interval</th>
<th>sparsity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jester</td>
<td>73421</td>
<td>100</td>
<td>(4.1 \times 10^6)</td>
<td>([-10, 10])</td>
<td>About 55.8%</td>
</tr>
<tr>
<td>MovieLens</td>
<td>943</td>
<td>1682</td>
<td>(\text{About } \times 10^4)</td>
<td>([0, 5])</td>
<td>About 4.73%</td>
</tr>
</tbody>
</table>

Table 4. Experiment Dataset

The parameters are set according to table 5. The value of \(\sigma\) is drawn from a uniform distribution, and the random data is drawn from a Gaussian distribution and its standard deviation is \(\sigma\). Besides, \(\beta\) is a control variable in the experiment and we set 11 values, \([0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]\) to it.

<table>
<thead>
<tr>
<th>parameter</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\beta)</td>
<td>(0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100)</td>
</tr>
<tr>
<td>(\sigma)</td>
<td>(\text{U}(0, 5))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\text{N}(0, \sigma))</td>
</tr>
</tbody>
</table>

Table 5. Value of parameters

5.2 Evaluation criterion

We use mean absolute error (MAE) and standard deviation (SD), the most commonly used criterions to evaluate accuracy. Supposing that \(P = \{p_1, p_2, p_3, \ldots, p_d\}\) is the rating vector of items from users, \(P' = \{p'_1, p'_2, p'_3, \ldots, p'_d\}\) is the prediction rating vector.
Therefore, \( E' = \{\xi_1, \xi_2, \xi_3, \ldots, \xi_d\} = \{p'_1 - p_1, p'_2 - p_2, \ldots, p'_d - p_d\} \) is the deviation, MAE and SD are given as:

\[
MAE = \frac{1}{d} \sum_{i=1}^{d} |\xi_i| \tag{11}
\]

\[
SD = \sqrt{\frac{1}{d-1} \sum_{i=1}^{d} (E - \bar{E})^2} \tag{12}
\]

Obviously, the less the value of MAE is the less the deviation will be, and then the recommendation is more accurate.

**5.3 Result and discussion**

As seen in Figure 2~Figure 5, the value of MAE and SD of improved PPCF algorithm applied on both two different dataset are much lower than those of the original ones. Moreover, with the increase of perturbation ratio, the advantage of the improved algorithm is becoming more obvious. Therefore, this improved PPCF algorithm is practical to a large degree.
6 CONCLUSIONS

We proposed an improved PPCF based on RPT by introducing privacy-preserving intensity weight into similarity calculation, which can improve the accuracy of the recommendation and protect the privacy data at the same time. The adjusted similarity in the improved algorithm is not simply concerns about similarity between users, but it also concerns the level of perturbation of the data. If the data are distorted too much, the influence of these data is decreased by the privacy-preserving intensity weight. Thus the recommendation can be balanced between privacy and accuracy.

References


