Research Article

A Smart Privacy-Preserving Learning Method by Fake Gradients to Protect Users Items in Recommender Systems

Guixun Luo,1 Zhiyuan Zhang,2 Zhenjiang Zhang,3 Yun Liu,2 and Lifu Wang2

1School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
2School of Electronic and Information Engineering, Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education, Beijing Jiaotong University, Beijing 100044, China
3School of Software Engineering, Beijing Jiaotong University, Beijing 100044, China

Correspondence should be addressed to Zhiyuan Zhang; zhangzhiyuan@bjtu.edu.cn

Received 22 October 2020; Revised 2 November 2020; Accepted 27 November 2020; Published 17 December 2020

Academic Editor: Abd E.I.-Baset Hassanien

Copyright © 2020 Guixun Luo et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this paper, we study the problem of protecting privacy in recommender systems. We focus on protecting the items rated by users and propose a novel privacy-preserving matrix factorization algorithm. In our algorithm, the user will submit a fake gradient to make the central server not able to distinguish which items are selected by the user. We make the Kullback–Leibler distance between the real and fake gradient distributions to be small thus hard to be distinguished. Using theories and experiments, we show that our algorithm can be reduced to a time-delay SGD, which can be proved to have a good convergence so that the accuracy will not decline. Our algorithm achieves a good tradeoff between the privacy and accuracy.

1. Introduction

Recommender systems, which help the electronic commerce websites to give more useful suggestions, are becoming more and more important. However, to provide users with appropriate options, the server will collect users’ data, which includes lots of sensitive information.

Data in electronic commerce, economics, supply chain, financial system [1–10], etc., are generally very sensitive. In the electronic commerce case, it is shown in many studies, such as [11, 12] that user data in recommender systems, shopping records, movies a user has watched, and ratings for the restaurants contain lots of very private information such as political attitudes, sexual orientation, etc. In this paper, we study the privacy-protecting problem in electronic commerce data. Privacy has been an important issue for a long time, not only in the recommender system but also in almost all algorithms in data mining and machine learning.

Differential privacy [13] is a popular method to protect privacy in machine learning algorithms. For recommender systems, there are many works applying differential privacy, such as [14–16]. Differential privacy matrix factorization algorithms are introduced in [17, 18], etc. Traditional differential privacy method is centralized, in other words, relying on a trustworthy data collector. When we want the central server not to be able to get privacy information, local differential privacy (LDP) should be used. Every user will add noise to their private data in their own device before being submitted to the central server. Recommender systems with LDP are studied in [19–21]. LDP has been used in Google's Chrome browser [22] and Apple iOS 10 [23] to collect user data.

In local differential privacy, there are two important things to be protected. The first one is which items this user has rated and the second one is the ratings of the user. In some situations, which items have been rated is much more sensitive than the rating itself. For example, shopping record contains a lot of private information, but the ratings can only represent the quality of goods. The work in [19] can only protect the ratings but not both. Shin et al. [17] proposed a novel LDP matrix factorization algorithm to protect both kinds of privacy information based on the work in [24]. Their method is to let the user submit a noisy gradient, whose value is either $B$ or $-B$. The
algorithm is $\epsilon$-LDP, and in each round of the training process, and since the output is binary, the adversary can not learn about which items are rated in a single iteration process.

However, if the adversary can get noisy gradients in multiple iterations since the noisy gradients obey the Bernoulli distribution with a mean 0, the items which have not been rated can be identified by a statistical test. The intensity of the privacy protection for the ratings and items after multiple iterations can be guaranteed by composition theorems for LDP [25, 26]. If every iteration is $\epsilon$-LDP, after $k$ iterations, the final algorithm is at most $ke$-LDP. But these analyses are not a direct guarantee to protect the items rated by the users. We can turn to a new perspective on this question. After performing $k$ iterations, given a sequence with length $k$ denoted by $y^i$, where $y^i_j$ is the gradient submitted in iteration $i$, let $P_{\text{real}}(y^i)$ be the probability that $y^i_j$ is a real gradient sequence and let $P_{\text{fake}}(y^i)$ be the probability that $y^i_j$ is fake. Using these two probabilities, we can consider testing two hypotheses, the sequence is real and the sequence is fake. So now comes the question, how can we make it difficult to distinguish the two situations?

In order to improve the ability of protecting privacy, we want the probability error to be large. Note that the average negative log probability of error is well-known deduced from the Chernoff–Stein lemma.

**Theorem 1** (Theorem 11.8.3 in [27]). $X \sim Q$ is a random variable; consider the hypothesis test between two alternatives, $Q = P_1$ and $Q = P_2$, where $D(P_1, P_2)$, the K-L distance, is finite. Then the average negative log probability of error of this hypothesis testing is $D(P_1, P_2)$.

Using this result, although we can not obtain the distribution of the real sequence, in Section 4, we will show that for the Gaussian noise based differential privacy algorithm, we can estimate the mean value of K-L distance and optimize the value of fake gradient to make the two distributions to be difficult to distinguish.

In this paper, we propose a novel algorithm such that if the item has not been rated by the user, the user will submit a fake gradient. Else, the user can submit the real one, but all the submitted data will eventually be noise added. The paper is organized as follows. In Section 2, we introduce differential privacy briefly as preliminaries. In Section 3, we introduce the framework of the general differential privacy matrix factorization algorithm. And in Section 4, we will show that our algorithm can reduce the average K-L distance between the fake and real gradient distributions, such that it can improve the intensity to protect the privacy items. Meanwhile, we can prove that our algorithm has the form of SGD with time delay, which can be proved that the accuracy of the model will not be reduced by our updating rules so that our algorithm achieves a tradeoff between accuracy and privacy. In Section 5, we use experiments to show the effectiveness of our algorithm. The related work is reviewed in Section 6. In the final section, we conclude.

### 2. Preliminaries

In this paper, the notations we used are listed in Table 1.

#### 2.1. Differential Privacy

Differential Privacy is first introduced by Dwork et al. [13], the aim of which is to make it difficult for an attacker to obtain privacy from the output data by adding noise.

**Definition 1.** A randomized algorithm $M: D \rightarrow R$ with domain $D$ and range $R$ is $(\epsilon, \delta)$-Differential Privacy, if for two adjacent data $d, d'$ in $D$ and for a subset $S$ of range $R$, it holds that

$$P(M(d) \in S) \leq e^\epsilon P(M(d') \in S) + \delta.$$  

(1)

Note that this definition is to compare the two probability. If $\delta = 0$, it can be expressed as

$$\ln \frac{P(M(d) \in S)}{P(M(d') \in S)} \leq \epsilon.$$  

(2)

If $\epsilon$ is small, such that it is hard to distinguish whether the output data is come from $d$ or $d'$. As in [28], one can link differential privacy with mutual-information.

Another way to describe Differential Privacy is to use the distance between distributions. We say a randomized algorithm $M: D \rightarrow R$ is $(\alpha, \epsilon)$ Renyi Differential Privacy if for all neighboring $d'$ and $d'$ we have

$$D_\alpha(M(d'), M(d)) \leq \frac{1}{\alpha - 1} \ln \left( \frac{M(d')}{M(d)} \right)^\alpha M(d)dz \leq \epsilon.$$  

(3)

When $\alpha \rightarrow 1$, $D_1$ is the Kullback–Leibler distance, and when $\alpha = \infty$, Renyi Differential Privacy is equal to $(\epsilon, 0)$-Differential Privacy. So we can see Differential Privacy is to make the output distributions with different inputs to be indistinguishable (the distributions have small distances).

One may ask how to achieve $(\epsilon, \delta)$-Differential Privacy in machine learning process. A basic paradigm to achieve "-differential privacy is to examine a query $L_2$-sensitivity in [29].

**Definition 2.** $f$ is a map from the data in the dataset $D$ to a vector. The $L_2$-sensitivity of $f$ is $\Delta_2(f) = \max_{d' \neq d} \|f(d) - f(d')\|$. 

### Table 1: Notations.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>Number of users</td>
</tr>
<tr>
<td>$n$</td>
<td>Number of items</td>
</tr>
<tr>
<td>$u$</td>
<td>The user profile vector</td>
</tr>
<tr>
<td>$v$</td>
<td>The item profile vector</td>
</tr>
<tr>
<td>$L(u,v)$</td>
<td>The loss function</td>
</tr>
<tr>
<td>$M$</td>
<td>The set of the ratings</td>
</tr>
<tr>
<td>$C$</td>
<td>Bound of the norm of the gradient in privacy gradient descent</td>
</tr>
<tr>
<td>$\eta$</td>
<td>The learning rate</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>The regularization parameters</td>
</tr>
<tr>
<td>$\beta$</td>
<td>$\beta$-Smooth parameters for the loss function</td>
</tr>
</tbody>
</table>

### Complexity

- **M**: Number of users
- **n**: Number of items
- **u**: The user profile vector
- **v**: The item profile vector
- **L**: The loss function
- **M**: The set of the ratings
- **C**: Bound of the norm of the gradient in privacy gradient descent
- **\eta**: The learning rate
- **\lambda**: The regularization parameters
- **\beta**: $\beta$-Smooth parameters for the loss function
Theorem 2. If $f$ is a map from $D$ to $\mathbb{R}^d$, then the randomized algorithm $M(D): f(D) + n$ where
\[ P(n) \propto \exp \left( \frac{\epsilon \|n\|}{\Delta_2(f)} \right), \tag{4} \]
achieves $\epsilon$-Differential Privacy.

This theorem provides a basic method to achieve Differential-Privacy-Machine-Learning.

3. The Framwork of Perturbed Matrix Factorization Algorithm

The program of Matrix Factorization algorithm with privacy protection has been studied by many authors, such as [17, 19]. When minimizing the cost function
\[ L(u, v) + \lambda \left( \sum_i \|u_i\|^2 + \sum_j \|v_j\|^2 \right) \]
\[ = \frac{1}{n} \sum_{i,j \in M} (r_{ij} - u_i^T v_j)^2 + \lambda \left( \sum_i \|u_i\|^2 + \sum_j \|v_j\|^2 \right), \tag{5} \]
we can use gradient descent
\[ u_i(t + 1) = u_i(t) - \eta \nabla_{u_i} L(u(t), v(t)) + 2\lambda u_i(t), \]
\[ v_j(t + 1) = v_j(t) - \eta \nabla_{v_j} L(u(t), v(t)) + 2\lambda v_j(t). \tag{6} \]
The vector $u_i$ is the user profile vector for user $i$, and $v_j$ is the item profile vector for item $j$.

Note that we have
\[ \sum_{i,j \in M} (r_{ij} - u_i^T v_j)^2 \geq \sum_{i,j} y_{ij}(r_{ij} - u_i^T v_j)^2, \tag{7} \]
\[ \nabla_{v_j} L(u, v) = -\frac{2}{n} \sum_i g_{ij} = -\frac{2}{n} \sum_i y_{ij} u_i(r_{ij} - u_i^T v_j), \tag{8} \]
where $y_{ij} = 1$ if $i, j \in M$ else $y_{ij} = 0$.

In this type of program, the user profile vectors $u_i$ are saved and updated on the users’ own devices. As for the item profile vectors, all the users will send the gradient to the central server, and individual users should perturb their gradient $g_{ij}$ using a random mechanism $\mathcal{M}$. Then the central server sums all these gradients to update the item profile vectors $v_j$. Using this random perturbation, $\epsilon$-differential privacy can be achieved by adjusting the distribution of noise.

The whole process is shown in Algorithm 1.

4. The Main Results

In this paper, since we are concerned about the items of users, we will focus on considering the statistical distance of $y_{ij} = 0$ and $y_{ij} = 1$ distributions. We propose a novel algorithm to protect items of the users. In our algorithm, the user will submit a noise-added fake gradient in the $y_{ij} = 0$ case. The K-L distance between the real and fake distributions will be small so that they are hard to be distinguished. On the other hand, we will study how will the fake gradients influence the model accuracy. We will show that in our algorithm, the updating rules can be reduced to a time-delay SGD, which will not influence the accuracy.

In our algorithm, the random mechanism $\mathcal{M}$ we choose is the Gaussian random mechanism, $\mathcal{M}(d) = N(d, \sigma^2)$. One of the advantages is that there is a very good composition theorem [26] which gives a much tighter estimate on the multi-iteration privacy loss for Gaussian mechanism-based differential privacy gradient descent algorithm.

Theorem 3 (Theorem 1 in [26]). Let $C$ be the gradient bound in privacy gradient descent, there exist two constants $c_1$ and $c_2$ such that the after $k$ iterations, the Gaussian noisy privacy gradient descent algorithm is $(\epsilon, \delta)$-differentially private for any $\delta > 0$ if we choose
\[ \sigma \geq c_2 C \frac{\sqrt{T \ln (1/\delta)}}{\epsilon}. \tag{9} \]

Generally, $C$ is chosen to be a prior bound of the gradient norm, so we do not write it in the algorithm description explicitly.
In each iteration, the user will sent a perturbed gradient $g'_{ij}$ to the central server, which has the following forms:

$$g'_{ij} = \begin{cases} u_i(r_{ij} - u^T v_j) + N(0, \sigma^2), & \text{if } y_{ij} = 1, \\ F_{ij} + N(0, \sigma^2), & \text{if } y_{ij} = 0. \end{cases}$$

Although we do not know the distribution of real gradients, this means value can be estimated by sampling. Let $S$ be the set of user $i$ such that $y_{ij} = 1$.

$$\int P_{\text{real}}(x_{ij}^j) \frac{1}{2\sigma^2} \| x_{ij}^j - F_{ij} \|^2 dx_{ij}^j \sim \sum_{i \in S} \frac{1}{2\sigma^2} \| x_{ij}^j - F_{ij} \|^2.$$

And in our algorithm, for a given item $j$, all the users will use the same $F$—in other words, we $F_{ij}$ is independent of $i$. Then the above equation is a function of the quadratic form.

$$\sum_{i \in S} \frac{1}{2\sigma^2} \| x_{ij}^j - F_j \|^2.$$
In order to minimize this K-L distance, we should set 
\( F_j = \sum_{i \in S} (x_i^j - \bar{x}) S \). \( G_j(t) = \frac{1}{(m)} \sum_{i \in S} g_i(t) (r_{ij} - u_{ij}(t)) v_j(t) = V_{ij} L(v_j(t)). \) However, at time \( t \), the user \( i \) can not get the current gradient \( V_{ij} L(v_j(t)) \). However, in the following section, we will show that in our algorithm we can estimate it from the previous gradient \( \sum g_i(t) \). 

4.2. Algorithmic Description. In Algorithm 1 with Gaussian random mechanism, we can see that the central server will receive the gradients submitted from the users, whose summation is as follows:

\[
G_j = \sum_{i \text{ with } y_{ij} = 1} u_i (r_{ij} - u_i^j v_j) + N(0, \sigma^2) + \sum_{i \text{ with } y_{ij} = 0} F_j + N(0, \sigma^2).
\]

Suppose \( F_j = 0 \), \( G_j \) is just a Langevin stochastic gradient [30] whose expected value is the total gradient. When \( F_j \neq 0 \), using \( G_i \) to update the parameters will generally influence the accuracy of the model. One way to solve this problem is to subtract a value in the central server.

\[
G_j = \sum_{i \text{ with } y_{ij} = 1} u_i (r_{ij} - u_i^j v_j) + N(0, \sigma^2)
+ \sum_{i \text{ with } y_{ij} = 0} F_j + N(0, \sigma^2) - N_j F_j.
\]

In order to determine the value of \( N_j \) to make the \( F_j \) part small, we can use the Random Response mechanism.

The Random Response mechanism [31] is a well-known method to obtain statistical information on sensitive issues, e.g., the proportion of AIDS. In our algorithm, we will use the Random Response mechanism to count the number of \( y_{ij} = 0 \) items, which is used for the central server to correct the sum of the gradients.

The procedure of the Random Response mechanism is that the responder will give the true answer with probability \( p > 0.5 \), and with probability \( 1-p \), the answer will give an opposite answer.

**Theorem 4** (Warner, 1965, in [31]). Suppose the number of the answer of \( y = 0 \) is \( n_1 \), and the total number of the responders is \( n \). If \( p \neq (1/2), (p - 1/2p - 1) + (n_1/(2p - 1))n \) is an unbiased estimate of the \( y = 0 \) ratio with variance \( \frac{(1 - \theta)}{n} + (1 - p)/(2p - 1)^2 \), where \( \theta \) is the real ratio of \( y = 0 \) items.

The variance is \( O(1/n) \), so if the total number of the users is large enough, with a high probability, \( \hat{\theta} \approx \theta \).

The whole process is Algorithm 2.

It is easy to see that, in the central server, the update process has the following forms:

\[
v_j(t + 1) = v_j(t) - \eta \bigg( \nabla v_j L(v_j(t), z) + \frac{\text{num}_j - \theta \mu_j}{n} \Delta V_j \bigg),
\]

where \( \nabla v_j L(v_j(t), z) \) is the sampling stochastic gradient and \( \text{num}_j \) is the number of \( y_{ij} = 0 \) terms in sampling.

As for \( \Delta V_j \), we know that

\[
\Delta V_j(t) = \sum_{i \in S} g_{ij}(t) (1 - \theta_j) n_j (1 - \theta_j) n_j \Delta V_j(t - 1)
\]

\[
= \sum_{i \in S} g_{ij}(t) (1 - \theta_j) n_j
\]

\[
\# S.
\]

Note that since the regularization term bound the norm of matrix \( U \) and \( V \), there exits a small constant \( \beta \) to make the loss function \( L(u, v) \) to be \( \beta \)-smooth, that is to say,

\[
\| \nabla v_j L(v_j(t) - v_j(t)) \|_\infty \leq \beta \| v_j(t) - v_j(t) \|_\infty.
\]

Since \( 1/n \gg \beta \), \( \nabla v_j L(v_j(t) - v_j(t)) \) is a good approximation of \( \nabla v_j L(v_j(t)) \).

So we have the following:

\[
v_j(t + 1) = v_j(t) - \eta \bigg( \nabla v_j L(v_j(t), z) + (\mu_j \nabla v_j L(v_j(t), z)) \bigg) + \xi.
\]

One can easily prove that the variances of all these estimations are \( O(1/n) \).

4.3. The Influence of Model Accuracy. We can see the form of updating rule (21) is a stochastic gradient descent with time delay. It can be shown that even if \( \mu \) is not small, time delay SGD will still have good convergence.

The convergence of SGD with time delay is proved in [32]. In this paper, Lian proved the convergence of asynchronous stochastic gradient descent which has the same form as equation (21).

**Theorem 5** (Theorem 1 in [32]). Assume the loss function is \( \beta \)-smooth, \( \eta \) is the learning rate, \( B \) is the batch size, and \( T \) is the time delay. If

\[
\beta B n \eta + 2B^2 T \eta \sum_k \eta \leq 1,
\]

after \( K \) iterations, we have with high probability,

\[
\min_k \| \nabla f(x_k) \|^2 \leq 4 \sqrt{(f(x_k) - f(x^*))} B \eta \sigma, \]

Where \( f(x^*) \) is the global minimum of \( f \) and \( \sigma \) is the standard deviation of stochastic gradients.

**Proof of Theorem 5.** In this case, the stochastic gradients \( g_{mt} \) sent by the node \( m \) at time \( t \) can be written as \( g_{mt} = \nabla f(x_{m - \tau_m}) + \xi_{m, \tau_m} \), where \( \tau_m \) is the time delay of the gradient and \( \xi_{m, \tau_m} \) is the noise (including noise from the stochastic gradients and the Gaussian noise we added). In our case, \( \xi_{m, \tau_m} \) is a sub-Gaussian random variable. To simplify the description, we assume \( \xi_{m, \tau_m} \) is \( \sigma \)-sub-Gaussian.
where $f(x_{t+1}) - f(x_{t}) = \sum_{k=0}^{t} f(x_{k+1}) - f(x_{k})$

$$\leq \sum_{k=0}^{t} \frac{Bn}{2} \left\| \nabla f(x_{k}) \right\|^2 + \left( \frac{3\eta^2L}{2} - \frac{\eta}{2B} \right) \left\| \sum_{m=1}^{B} \nabla f(x_{k-m}) \right\|^2$$

$$+ \beta^2TBn \left( \sum_{j=k-T}^{k-1} \eta^2 B \left\| \nabla f(x_{j-m}) \right\|^2 - \eta \left\langle \nabla f(x_{k}), \frac{B}{m} \sum_{m=1}^{B} \epsilon_{k,m} \right\rangle + \frac{3\eta^2\beta}{2} \left\| \sum_{m=1}^{B} \epsilon_{k,m} \right\|^2$$

$$+ \beta^2Bn^3 \left\| \sum_{j=k-T}^{k-1} \sum_{m=1}^{B} \epsilon_{j,m} \right\|^2 = \sum_{k=0}^{t} \frac{Bn}{2} \left\| \nabla f(x_{k}) \right\|^2$$

$$(24)$$

In order to estimate $T_2 = T_{2,a} + T_{2,b}$, we can use lemmas in [33].

Let $\epsilon_{k} = (1/B) \sum_{m=1}^{B} \epsilon_{k,m}$. With probability $1 - e^{-i}$, we have the following:

$$- \sum_{k=0}^{t} \eta \left\langle B \nabla f(x_{k}), \epsilon_{k} \right\rangle \leq \frac{\eta B}{2} \sum_{k=0}^{t} \left\| \nabla f(x_{k}) \right\|^2 + c\eta^2 \sigma^2.$$

This is from Lemma 30 in [33].

With high probability,

$$\sum_{k=0}^{t} \eta^2 B \sum_{m=1}^{B} \epsilon_{k,m} \leq \frac{3\eta^2B}{2} Bc\sigma^2 (\tau + 1 + i).$$

And with high probability,

$$\sum_{k=0}^{t} \beta^2 Bn^3 \left\| \sum_{j=k-T}^{k-1} \sum_{m=1}^{B} \epsilon_{j,m} \right\|^2$$

$$\leq \beta^2TB n^3 Bc\sigma^2 \left( \frac{\tau}{TBn + 1 + T + i} \right)$$

$$(27)$$

We have the following:

$$T_1 \leq \frac{\eta B}{8} \sum_{k=0}^{t} \left\| \nabla f(x_{k}) \right\|^2 + c\eta^2 \sigma^2 + 2\eta^2Bc\sigma^2 (\tau + 1 + i),$$

$$(28)$$

$\eta^2((3\beta/4 - \beta^2MT^2)\eta) - (\eta/2B) < 0$. With probability at least $1 - 3e^{-i}$,

$$f(x_{t+1}) - f(x_{t}) \leq \sum_{k=0}^{t} \frac{3Bn}{8} \left\| \nabla f(x_{k}) \right\|^2 + c\eta^2 \sigma^2$$

$$(29)$$

The theorem follows.

This theorem has the same form as the convergence theorem of general and SGD, and in our case, we have $T = 1$. So we can show this time delay will not influence the convergence.

4.4. Privacy Loss in the Random Response Mechanism. At the start of our algorithm, we need to use the Random response mechanism to estimate the ratio of $y_{ij}$, which will cause a privacy loss. However, we can show that since we need a large number of iterations in the machine learning algorithm, the initial privacy loss is insignificant.
It is easy to prove that the Random response mechanism is \(\ln(p/1-p)\)-Differential Privacy. We know from Theorem 3 that \(\epsilon \sim O(\sqrt{k\ln(1/\delta)})\) after \(k\) iterations. If \(n\) is large enough, we can choose a \(p\) near 0.5, and when \(k\) is large, \(\ln(p/1-p)\) will be much less than \(\epsilon\).

Noting that the K-L distance for a length \(k\) sequence is \(O(k)\), the discussion on the K-L distance is the same.

5. Experiments

We now show the performance of our algorithm. We evaluate three types of privacy gradient descent algorithms:

(i) Algorithm 1, the noisy gradient descent with \(F_{ij} = u_i(0 - u_i^T v_j)\). The users will submit a gradient \(F_{ij} = u_i(0 - u_i^T v_j) + \zeta\) if \(y_{ij} = 0\), where \(\zeta\) is a \(N(0, \sigma^2)\) Gaussian random variable.

(ii) Algorithm 2, noisy gradient descent with \(F_{ij} = \zeta\).

(iii) Algorithm 3, our algorithm in this paper.

In the \(F_{ij} = 0\) case, the only noise in the total gradient is caused by Gaussian noise added to the users’ device. This algorithm will be accurate but has no ability to protect the item’s privacy. We will show that the performance of our algorithm is very close to the case \(F_{ij} = 0\) and much better than the algorithm using fake ratings.
We test on MovieLens 100k dataset [34]. This version contains 100k ratings of 1682 movies submitted by 982 users. This dataset is very sparse. In order to test the performance in different situations of sparsity, for every user, we choose a set $F$ of items to be selected to provide fake gradients. We consider different cases that $#F = #S$ (50% fake gradient density), $#F = 3#S$ (75% fake gradient density) to test the algorithm. We set the profile vector dimension $d = 15$, regularization parameters $\lambda = 0.001$, learning rate $\eta = 0.1$, $\sigma^2 = 1$ and use AdaDelta to optimize. The test RMSE is shown in Figures 1 and 2.

After 400 iterations, the test RMSE is listed in Table 2. We see that when the density of fake rating increases, the test RSME of fake rating algorithm is growing rapidly, and the performance algorithm is very close to the zero mean fake gradient algorithm.

6. Related Work

Differential privacy introduced by Dwork [13] is a very strong guarantee to protect privacy. The original version of differential privacy consider a trusted server to provide data to queriers, and the aim is to prevent access to user privacy from queries.

Local differential privacy algorithm, such as RAPPORT [22], is to make sure the central server can not access the privacy of the users. The main technology is to add some noise before submitting the data to the server. In the Chrome browser, Google uses a randomized response mechanism to collect the data of the users’ clicks. Also, there are many works to use local differential privacy to perform machine learning algorithms. For example, Google uses local differential privacy Federated Learning [35] to learn a language model in order to improve the performance of the inputting method.

One of the difficulties in differential privacy machine learning is that when training a model using many iterations, the privacy guarantees will decline rapidly. Differential privacy for multi-iterations is studied in [25, 26] and a much tighter composition theorem is given.

Private recommender system is studied by many authors such as [17–20, 36, 37]. References [17, 18] are based on a matrix factorization recommender system. The algorithm is to adding some noise in users’ devices locally to protect
privacy. The algorithm in [17] can protect both the ratings and the items of the user. Their work is based on the work in [24], where they propose a new randomization mechanism and show that their mechanism is better when the dimension of data is large.

7. Conclusion

In this paper, we propose a novel privacy matrix factorization algorithm. In our algorithm, we use the Random Response method to estimate the selection ratios of the items, and then we use the average value of the gradients in the previous time as the fake gradient to be sent to the central server. Using our method, we can improve the indistinguishability of the real gradient and fake distributions so that improve the ability to protect user private items. Meanwhile, we show that our algorithms will not cut down the accuracy of the model since the updating rule can be reduced to SGD with time delay, which can be proved to convergence to gradient zero points.

Data Availability


Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been supported by the Fundamental Research Funds for the Central Universities (grant number: 2020JBM002) and the National Key Research and Development Program of China (grant no. 2018YFC0831703).

References


