Hindawi
Complexity
Volume 2020, Article ID 6683834, 10 pages
https://doi.org/10.1155/2020/6683834

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 Wang**[2]

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

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 ε-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 ε-LDP, after k iterations, the final algorithm is at most $k\varepsilon$-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_j^i is the gradient submitted in iteration i, let $P_{\text{real}}(y_j)$ be the probability that y_j is a real gradient sequence and let $P_{\text{fake}}(y_j)$ be the probability that y_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.

Table 1: Notations.

Notation	Meaning
m	Number of users
n	Number of items
u	The user profile vector
v	The item profile vector
$L(u, v)$	The loss function
M	The set of the ratings
C	Bound of the norm of the gradient in privacy gradient descent
η	The learning rate
λ	The regularization parameters
β	β-Smooth parpameters for the loss function

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 $M: D \longrightarrow R$ with domain D and range R is (ε, δ)-Differential Privacy, if for two adjacent data $d, d' \in D$ and for a subset S of range R, it holds that

$$P(M(d) \in S) \leq e^{\varepsilon} P(M(d') \in S) + \delta. \tag{1}$$

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

$$\ln \frac{P(M(d) \in S)}{P(M(d') \in S)} \leq \varepsilon. \tag{2}$$

If ε 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 $M: D \longrightarrow R$ is (α, ε) Renyi Differential Privacy if for all neighboring d' and $'d'$ we have

$$D_{\alpha}(M(d'), M(d)) = \frac{1}{\alpha - 1} \ln \int \left(\frac{M(d')(z)}{M(d)(z)} \right)^{\alpha} M(d)(z) \mathrm{d}z \leq \varepsilon. \tag{3}$$

When $\alpha \longrightarrow 1$, D_1 is the Kullback-Leibler distance, and when $\alpha = \infty$, Renyi Differential Privacy is equal to $(\varepsilon, 0)$ 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 (ε, δ)-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 $\Delta_2(f) = \max_{d', d} \| f(d) - f(d) \|$.

Using this definition, we have the following theorem in [29].

Theorem 2. *If f is a map from D to $\mathbb{R}^d$. Then the randomized algorithm $M(D)$: $f(D) + n$ where*

$$P(n) \propto \exp\left(\frac{\varepsilon \|n\|}{\Delta_2(f)}\right), \tag{4}$$

achieves ε-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

$$\begin{aligned} &L(u, v) + \lambda\left(\sum_i \|u_i\|^2 + \sum_j \|v_j\|^2\right) \\ &= \frac{1}{n}\sum_{i,j\in M}\left(r_{ij} - u_i^T v_j\right)^2 + \lambda\left(\sum_i \|u_i\|^2 + \sum_j \|v_j\|^2\right). \end{aligned} \tag{5}$$

We can use gradient descent

$$\begin{aligned} u_i(t+1) &= u_i(t) - \eta\nabla_{u_i} L(u(t), v(t)) + 2\lambda u_i(t), \\ v_j(t+1) &= v_j(t) - \eta\nabla_{v_j} L(u(t), v(t)) + 2\lambda v_j(t). \end{aligned} \tag{6}$$

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

$$\sum_{i,j\in M}\left(r_{ij} - u_i^T v_j\right)^2 = \sum_{i,j} y_{ij}\left(r_{ij} - u_i^T v_j\right)^2, \tag{7}$$

$$\nabla_{v_j} L(u, v) = -\frac{2}{n}\sum_i g_{ij} = -\frac{2}{n}\sum_i y_{ij} u_i\left(r_{ij} - u_i^T v_j\right), \tag{8}$$

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 $\mathbb{M}$. Then the central server sums all these gradients to update the item profile vectors v_j. Using this random perturbation, ε-differential privacy can be achieved by adjusting the distribution of noise.

The whole process is shown in Algorithm 1.

Note that there are two types of private information. One is the ratings of the users and the other one is the items have been rated by the users.

In order to protect the items, one way is to use the random response mechanism introduced in Section 4.1 of [17]. In this method, we generate a y'_{ij} such that $y'_{ij} = 1$ with probability p, and if the original $y_{ij} = 0$, we set a fake rating $r_{ij} = 0$ so the fake gradient is $u_i(0 - u_i^T v_j)$ by (8), and Gaussian noise is added to the final gradient sent to the central server to protect the ratings of users.

However, it is shown in the discussion of Section 4.1 of [17] that the error caused by these fake ratings is not small, which will influence the final model accuracy. The main reason is that there are many fake gradients, which lead to a great error in the expectation of the sum of gradients.

One way is to solve this problem is to set the fake gradient F_{ij} to be zero. If $y_{ij} = 0$, the user sends a random variable $\mathbb{M}(0)$ to the central server. This method is used in [17], where $\mathbb{M}(x)$ is a Bernoulli random variable with mean value x. However, the disadvantage of this method is that the distribution of gradients in the $y_{ij} = 0$ case is very different from the distribution of the real gradient. For example, we can collect some data of g_{ij} sent by the user i, and use a statistical test to test if this data obeys the certain distribution of mean 0, then we can know whether $y_{ij} = 0$.

All in all, we need to strike a balance between privacy and accuracy. We need to provide a fake gradient to make sure the accuracy will not be greatly affected and let these two distributions, the fake one and the real one, to be statistically indistinguishable as far as possible.

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 $\mathbb{M}$ we choose is the Gaussian random mechanism, $\mathbb{M}(d) = N(d, \sigma^2)$. 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 (ε, δ)-differentially private for any $\delta > 0$ if we choose*

$$\sigma \geq c_2 C \frac{\sqrt{T\ln(1/\delta)}}{\varepsilon}. \tag{9}$$

Generally, C is chosen to be a prior bound of the gradient norm, so we do not write it in the algorithm description explicitly.

Input: Random mechanism $\mathbb{M}$, learning rate η, and redefined iteration number k
Output: Item profile matrix V
Randomly initialize $u_i(0), v_j(0)$ for all i and j.
for $t = 1, 2, 3 \ldots$ **do**
  Initialize $G_j = 0$ for all j in central server.
  for $i = 1, 2, 3, \ldots, m$ **do**
    On user i: sample j uniformly
    from$\{1, 2, 3, \ldots, n\}$.
    if $y_{ij} = 1$ **then**
      $g_{ij} = u_i(r_{ij} - u_i^T v_j)$
      $g_{ij} = g_{ij}/\max(1, (\|g_{ij}\|^2/C))$
      $g'_{ij} = \mathbb{M}(x_{ij})$
    end
    else
      Generate a fake gradient of F_{ij}.
      set $g_{ij} = F_{ij}$
      $g'_{ij} = \mathbb{M}(x_{ij})$
    end
    $G_j = G_j + x'_{ij}$ for all j.
  end
  For all j:
  $G_j = (G_j/m)$
  $v_j = v_j + \eta G_j$
 for $i = 1, 2, 3 \ldots m$ **do**
    Update u_i on a local device by gradient descent.
  end
end

Algorithm 1: Perturbed Matrix Factorization algorithm.

In the case of the Gaussian random mechanism, it is easy to calculate the K-L distance between distributions. In the following section, we will show that we can find a good choice of the fake gradient.

4.1. Estimating the K-L Distance between Two Distributions. Given a gradient sequence y_j with length k, a probability of y_j can be represented in the following form.

$$P\left(y_j = o_{1:k}\right) = \prod_{i=1}^{k} P\left(y_j^i = o_i | y_j^{i-1} = o_{i-1}\right). \tag{10}$$

Using this form we can calculate K-L distance.

Given two probability measures P_1 and P_2 in length k sequence space, we have

$$\begin{aligned} D(P_1, P_2) &= \int P_1(o_{1:k}) \sum_{i=1}^{k} \log \frac{P_1(o_i|o_{i-1})}{P_2(o_i|o_{i-1})} \mathrm{d}o \\ &= \sum_{i=1}^{k} \int P_1(o_i|o_{i-1}) P_1(o_{i-1}) \log \frac{P_1(o_i|o_{i-1})}{P_2(o_i|o_{i-1})} \mathrm{d}o_i \mathrm{d}o_{i-1} \\ &= \sum_{i=1}^{k} \int P_1(o_{i-1}) D(P_1(o_i|o_{i-1}), P_2(o_i|o_{i-1})) \mathrm{d}o_{i-1}. \end{aligned} \tag{11}$$

In each iteration, the user will sent a perturbed gradient g'_{ij} to the central server, which has the following forms:

$$g'_{ij} = \begin{cases} u_i\left(r_{ij} - u_i^T v_j\right) + N\left(0, \sigma^2\right), & \text{if } y_{ij} = 1, \\ F_{ij} + N\left(0, \sigma^2\right), & \text{if } y_{ij} = 0. \end{cases} \tag{12}$$

The we have $P_{\text{real}}(o_t|o_{t-1}) = P(x_t + \eta_{ij} = o_t)$, where x_t is the gradient calculated from o_{t-1}, so $D(P_{\text{real}}(o_t|o_{i-1}), P_{\text{fake}}(o_t|o_{i-1})) = D(N(x_t^{ij}, \sigma^2), N(G_{ij}, \sigma^2)$.

This is the K-L distance between two Gaussian distributions with the same σ. We can show that $D(N(x_t^{ij}, \sigma^2), N(G_{ij}, \sigma^2)) = \text{const} + (1/2\sigma^2)\|x_t^{ij} - G_{ij}\|^2$.

From equation (11), if we want to optimize the K-L distance, we need to consider

$$\sum_t \int P_{\text{real}}\left(x_t^{ij}\right) \frac{1}{2\sigma^2} \left\|x_t^{ij} - F_{ij}\right\|^2 \mathrm{d}x_t^{ij}. \tag{13}$$

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 $y_{ij} = 1$.

$$\int P_{\text{real}}\left(x_t^{ij}\right) \frac{1}{2\sigma^2} \left\|x_t^{ij} - F_{ij}\right\|^2 \mathrm{d}x_t^{ij} \sim \sum_{i \in S} \frac{1}{2\sigma^2} \left\|x_t^{ij} - F_{ij}\right\|^2. \tag{14}$$

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.

$$\sum_{i \in S} \frac{1}{2\sigma^2} \left\|x_t^{ij} - F_j\right\|^2. \tag{15}$$

In order to minimize this K-L distance, we should set $F_j = \sum_{i\in S}(x^{ij}/\#S)$. $G_j(t) \approx (1/m)\sum_{i\in S} u_i(t)(r_{ij} - u_i^T(t)v_j(t)) = \nabla_{v_j} L(v_j(t))$. However, at time t, the user i can not get the current gradient $\nabla_{v_j} L(v_j(t))$. However, in the following section, we will show that in our algorithm we can estimate it from the previous gradient $\sum_i g_{ij}(t-1)$.

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:

$$G_j = \sum_{i \text{ with } y_{ij}=1} u_i\left(r_{ij} - u_i^T v_j\right) + N\left(0, \sigma^2\right) + \sum_{i \text{ with } y_{ij}=0} F_j + N\left(0, \sigma^2\right). \tag{16}$$

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_j 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.

$$\begin{aligned} G_j = &\sum_{i \text{ with } y_{ij}=1} u_i\left(r_{ij} - u_i^T v_j\right) + N\left(0, \sigma^2\right) \\ &+ \sum_{i \text{ with } y_{ij}=0} F_j + N\left(0, \sigma^2\right) - N_j F_j. \end{aligned} \tag{17}$$

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 $(\theta(1-\theta)/n) + (p(1-p)/n(2p-1)^2)$, where θ 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, $\widehat{\theta} \approx \theta$.

The whole process is shown in Algorithm 2.

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

$$v_j(t+1) = v_j(t) - \eta\left(\nabla_{v_j} L\left(v_j(t), z\right) + \left(\frac{\text{num}_j - \theta_j n_j}{n}\right)\Delta V_j\right), \tag{18}$$

where $\nabla_{v_j} L(v_j(t), z)$ is the sampling stochastic gradient and num_j is the number of $y_{ij} = 0$ terms in sampling.

As for ΔV_j, we know that

$$\begin{aligned} \Delta V_j(t) &= \frac{\sum_{i\in S} g_{ij}(t-1) + \left(\#\{i \notin S\} - \theta_j n_j\right)\Delta V_j(t-1)}{\left(1-\theta_j\right)n_j} \\ &\sim \frac{\sum_{i\in S} g_{ij}(t-1)}{\#S}. \end{aligned} \tag{19}$$

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

$$\left\|\nabla_{v_j} L\left(v_j(t-1)\right) - \nabla_{v_j} L\left(v_j(t)\right)\right\|_2 \leq \beta\left\|v_j(t-1) - v_j(t)\right\|_2. \tag{20}$$

Since $(1/\eta) \gg \beta$, $\nabla_{v_j} L(v_j(t-1))$ is a good approximation of $\nabla_{v_j} L(v_j(t))$.

So we have the folowing:

$$v_j(t+1) = v_j(t) - \eta\left(\nabla_{v_j} L\left(v_j(t), z\right) + \mu\nabla_{v_j} L\left(v_j(t-1), z\right)\right) + \zeta. \tag{21}$$

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 μ a 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, η is the learning rate, B is the batch size, and T is the time delay. If

$$\beta B\eta + 2\beta^2 B^2 T\eta \sum_k^T \eta \leq 1, \tag{22}$$

after K iterations, we have with high probability,

$$\min_k \left\|\nabla f\left(x_k\right)\right\|^2\Big) \leq 4\sqrt{\frac{\left(f\left(x_1\right) - f(x*)\right)\beta}{BK}}\sigma, \tag{23}$$

Where $f(x*)$ is the global minimum of f and σ is the standard deviation of stochastic gradients.

Proof of Theorem 5. In this case, the stochastic gradients $G_{m,t}$ sent by the node m at time t can be written as $G_{m,t} = \nabla f(x_{t-\tau_{m,t}}) + \zeta_{t,m}$, where $\tau_{m,t}$ is the time delay of the gradient and $\zeta_{t,m}$ is the noise (including noise from the stochastic gradients and the Gaussian noise we added). In our case, $\zeta_{t,m}$ is a sub-Gaussian random variable. To simplify the description, we assume $\zeta_{t,m}$ is σ-sub-Gaussian.

$$
\begin{aligned}
f(x_{\tau+1}) - f(x_0) &= \sum_{k=0}^{\tau} f(x_{k+1}) - f(x_k) \\
&\le \sum_{k=0}^{\tau} -\frac{B\eta}{2}\left\|\nabla f(x_k)\right\|^2 + \left(\frac{3\eta^2 L}{4} - \frac{\eta}{2B}\right)\left\|\sum_{m=1}^{B}\nabla f\left(x_{k-\tau_{k,m}}\right)\right\|^2 \\
&\quad + \beta^2 TB\eta \sum_{j=k-T}^{k-1}\eta^2\left\|\sum_{m=1}^{B}\nabla f\left(x_{j-\tau_{j,m}}\right)\right\|^2 - \eta\langle\nabla f(x_k), \sum_{m=1}^{B}\zeta_{k,m}\rangle + \frac{3\eta^2\beta}{2}\left\|\sum_{m=1}^{B}\zeta_{k,m}\right\|^2 \\
&\quad + \beta^2 B\eta^3\left\|\sum_{j=k-\tau_k^{\max}}^{k-1}\sum_{m=1}^{B}\zeta_{j,m}\right\|^2 = \sum_{k=0}^{\tau} -\frac{B\eta}{2}\left\|\nabla f(x_k)\right\|^2 \\
&\quad + \sum_{k=0}^{\tau}\left(\frac{3\eta^2\beta}{4} - \frac{\eta}{2B}\right)\left\|\sum_{m=1}^{B}\nabla f\left(x_{k-\tau_{k,m}}\right)\right\|^2 + \beta^2 TB\eta\sum_{j=k-T}^{k-1}\eta^2\left\|\sum_{m=1}^{B}\nabla f\left(x_{j-\tau_{j,m}}\right)\right\|^2 \\
&\quad - \sum_{k=0}^{\tau}\eta\langle\nabla f(x_k), \sum_{m=1}^{B}\zeta_{k,m}\rangle + \frac{3\eta^2\beta}{2}\left\|\sum_{m=1}^{B}\zeta_{k,m}\right\|^2 + \beta^2 B\eta^3\left\|\sum_{j=k-\tau_k^{\max}}^{k-1}\sum_{m=1}^{B}\zeta_{j,m}\right\|^2 \\
&\le \sum_{k=0}^{\tau} -\frac{B\eta}{2}\left\|\nabla f(x_k)\right\|^2 + \sum_{k=0}^{\tau}\left(\eta^2\left(\frac{3\beta}{4} + \beta^2 BT^2\eta\right) - \frac{\eta}{2B}\right)\left\|\sum_{m=1}^{B}\nabla f\left(x_{k-\tau_{k,m}}\right)\right\|^2 \\
&\quad \underbrace{-\sum_{k=0}^{\tau}\eta\langle\nabla f(x_k), \sum_{m=1}^{B}\zeta_{k,m}\rangle + \sum_{k=0}^{\tau}\frac{3\eta^2\beta}{2}\left\|\sum_{m=1}^{B}\zeta_{k,m}\right\|^2}_{T_{2,a}} + \underbrace{\sum_{k=0}^{\tau}\beta^2 B\eta^3\left\|\sum_{j=k-\tau_k^{\max}}^{k-1}\sum_{m=1}^{B}\zeta_{j,m}\right\|^2}_{T_{2,b}}.
\end{aligned}
\tag{24}
$$

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

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

$$
-\sum_{k=0}^{\tau}\eta\langle B\nabla f(x_k), \zeta_k\rangle \le \frac{\eta B}{8}\sum_{k=0}^{\tau}\left\|\nabla f(x_k)\right\|^2 + c\eta\sigma^2\iota. \tag{25}
$$

This is from Lemma 30 in [33].

With high probability,

$$
\sum_{k=0}^{\tau}\frac{3\eta^2\beta}{2}\left\|\sum_{m=1}^{B}\zeta_{k,m}\right\|^2 \le \frac{3\eta^2\beta}{2}Bc\sigma^2(\tau + 1 + \iota). \tag{26}
$$

And with high probability,

$$
\begin{aligned}
&\sum_{k=0}^{\tau}\beta^2 B\eta^3\left\|\sum_{j=k-\tau_k^{\max}}^{k-1}\sum_{m=1}^{B}\zeta_{j,m}\right\|^2 \\
&\le \beta^2 TB\eta^3 Bc\sigma^2\left(\frac{\tau}{TB\eta + 1 + T + \iota}\right) \\
&\le \frac{\eta^2 L}{2}Bc\sigma^2(\tau + 1 + \iota).
\end{aligned}
\tag{27}
$$

We have the following:

$$
T_1 \le \frac{\eta B}{8}\sum_{k=0}^{\tau}\left\|\nabla f(x_k)\right\|^2 + c\eta\sigma^2\iota + 2\eta^2\beta Bc\sigma^2(\tau + 1 + \iota), \tag{28}
$$

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

$$
\begin{aligned}
f(x_{\tau+1}) - f(x_0) &\le \sum_{k=0}^{\tau} -\frac{3B\eta}{8}\left\|\nabla f(x_k)\right\|^2 + c\eta\sigma^2\iota \\
&\quad + 2\eta^2\beta Bc\sigma^2(\tau + 1 + \iota).
\end{aligned}
\tag{29}
$$

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.

Input: Redefined iteration number k, learning rate η, probability p for Random Response and Standard deviation of Gaussian distribution σ.
Output: Item profile matrix V
For all items j, use the probability p Random Response method to estimate the ratio of the users with $y_{ij} = 0$ as θ_j.
Randomly initialize $u_i(0), v_j(0)$ for all i and j.
for $t = 1, 2, 3 \ldots$ **do**
  Initialize $G_j = 0$, $n_j = 0$ for all $j = 1, 2, \ldots,$ n in central server.
  for $i = 1, 2, 3, \ldots, m$ **do**
    On user i: sample B items $S = \{S_1, S_2, \ldots, S_B\}$ uniformly from$\{1, 2, 3, \ldots, n\}$
    for $j \, in \, S$ **do**
      $n_j = n_j + 1$
      if $y_{ij} = 1$ **then**
        $g_{ij} = u_i(r_{ij} - u_i^T v_j)$
        Draw $g'_{ij} \sim N(x_{ij}, \sigma^2)$
      end
      else
        if $t \neq 1$ **then**
          $F_{ij} = \Delta V_j$
        end
        else
          $F_{ij} = u_i(0 - u_i^T v_j)$
        end
        $g_{ij} = F_{ij}$
        Draw $g'_{ij} \sim N(x_{ij}, \sigma^2)$
      end
    end
    $G_j = G_j + x'_{ij}$.
  end
  for $j = 1, 2, \ldots, n$ **do**
    if $n_j = 0$ **then**
    $G_j = 0$
    $\Delta V_j = 0$
    end
    else
      $G_j = G_j - \theta_j n_j \times \Delta V_j(t-1)$
      $\Delta V_j(t) = (G_j / ((1 - \theta_j) n_j))$
      $G_j = (G_j / m)$
      $v_j = v_j + \eta G_j$
    end
  end
  for $i = 1, 2, 3, \ldots, m$ **do**
    Update u_i on the local device by gradient descent.
  end
end

Algorithm 2: Noisy matrix factorization with fake gradient.

It is easy to prove that the Random response mechanism is $\ln(p/1-p)$-Differential Privacy. We know from Theorem 3 that $\varepsilon \sim O(\sqrt{k \ln(1/\delta)})$ 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 then ε.

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 ζ 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.

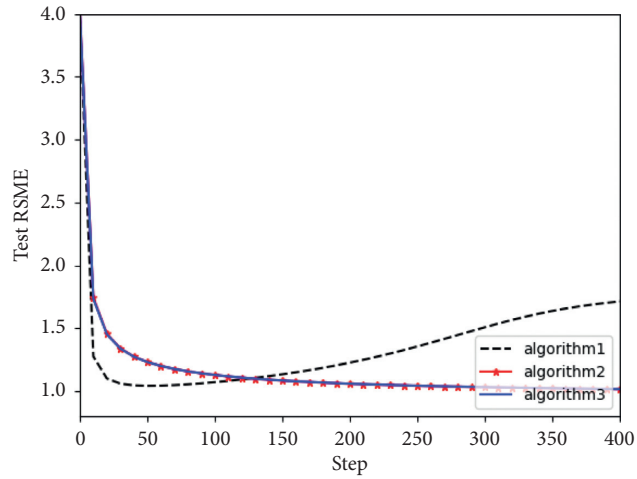

Figure 1: RMSE with 50% density.

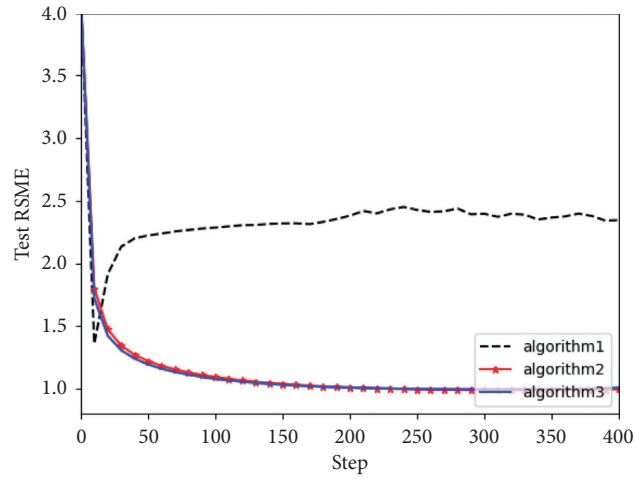

Figure 2: RMSE with 75% density.

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

Table 2: RMSE in experiments.

Density (%)	Method RMSE		
	$F_{ij} = u_i(0 - u_i^T v_j)$	$F_{ij} = 0$	Our algorithm
50	1.7141140	1.0161815	1.0167035
75	2.3660726	0.9966793	1.0073330

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

The Movielens-100K, http://files.grouplens.org/datasets/movielens/.

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

[1] Z. Zhang, Y. Liu, Z. Zhang, and B. Shen, "Fused matrix factorization with multi-tag, social and geographical influences for poi recommendation," *World Wide Web*, vol. 2222 pages, 2018.

[2] Z. Zhang, Y. Liu, G. Xu, and H. Chen, "A weighted adaptation method on learning user preference profile," *Knowledge-Based Systems*, vol. 112, no. 15, pp. 114–126, 2016.

[3] T. G. Alexandru and C. Pupăză, "Machine learning generalization of lumped parameter models for the optimal cooling of embedded systems," *Studies in Informatics and Control*, vol. 29, no. 2, pp. 169–177, 2020.

[4] I. Stoica, "Solving system problems with machine learning," *Studies in Informatics and Control*, vol. 28, no. 2, pp. 119–132, 2019.

[5] K.-S. Moon and H. Kim, "Performance of deep learning in prediction of stock market volatility," *Economic Computation and Economic Cybernetics Studies and Research*, vol. 53, pp. 77–92, 2019.

[6] V. Oona, "Using data mining methods to solve classification problems in financial-banking institutions," *Economic Computation and Economic Cybernetics Studies and Research/Academy of Economic Studies*, vol. 54, no. 1, pp. 159–176, 2020.

[7] K. J. Ban, "Implementation of artificial intelligence system and traditional system: a comparative study," *Journal of System and Management Sciences*, vol. 66, 2019.

[8] S. Y. Y. Ji and L. J. Ku, "A blockchain and internet of things based architecture design for energy transaction," *Journal of System and Management Sciences*, vol. 63, 2020.

[9] S. H. L. Z. X. Li and J. Pan, "A machine learning based method for customer behavior prediction," *Tehnicki vjesnik-Technical Gazette*, vol. 72, 2019.

[10] N. Z. D. Qin and L. L. Yu, "Applying the convolutional neural network deep learning technology to behavioural recognition in intelligent video," *Tehnicki vjesnik-Technical Gazette*, vol. 58, no. 3, 2018.

[11] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, "You might also like: privacy risks of collaborative filtering," *Security & Privacy*, vol. 39, no. 5, 2012.

[12] F. Dan and J. Riedl, "Do you trust your recommendations? an exploration of security and privacy issues in recommender systems," in *Proceedings of the International Conference On Emerging Trends In Information & Communication Security*, Freiburg, Germany, June 2006.

[13] C. Dwork, K. Kenthapadi, F. Mcsherry, I. Mironov, and M. Naor, "Our data, ourselves: privacy via distributed noise generation," in *Proceedings of the International Conference On Advances In Cryptology-Eurocrypt*, Saint Petersburg, Russia, June 2006.

[14] Z. Liu, Y. X. Wang, and A. Smola, "Fast differentially private matrix factorization," *Machine Learning*, vol. 9, 2015.

[15] T. Zhu, L. Gang, Y. Ren, W. Zhou, and X. Ping, "Differential privacy for neighborhood-based collaborative filtering," in *Proceedings of the IEEE/ACM International Conference On Advances In Social Networks Analysis & Mining*, Ontario Canada, August 2013.

[16] A. Machanavajjhala, A. Korolova, and A. D. Sarma, "Personalized social recommendations: accurate or private," *Proceedings of the Vldb Endowment*, vol. 4, no. 7, 2011.

[17] H. Shin, S. Kim, J. Shin, and X. Xiao, "Privacy enhanced matrix factorization for recommendation with local differential privacy," *IEEE Transactions on Knowledge & Data Engineering*, vol. 99, p. 1, 2018.

[18] A. Berlioz, A. Friedman, M. A. Kaafar, R. Boreli, and S. Berkovsky, "Applying differential privacy to matrix factorization," in *Proceedings of the Acm Conference On Recommender Systems*, Vienna, Austria, September 2015.

[19] J. Hua, X. Chang, and Z. Sheng, "Differentially private matrix factorization," in *Proceedings of the International Conference on Artificial Intelligence*, Deigo, CL, USA, June 2015.

[20] Y. Shen and H. Jin, "Privacy-preserving personalized recommendation: an instance-based approach via differential privacy," in *Proceedings of the IEEE International*

Conference on Data Mining, Shenzhen, China, December 2014.

[21] Y. Shen and H. Jin, "Epicrec: Towards practical differentially private framework for personalized recommendation," in *Proceedings of the Acm Sigsac Conference on Computer & Communications Security*, London, UK, November 2016.

[22] U. Erlingsson, V. Pihur, and A. Korolova, "Rappor:randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the Acm Sigsac Conference on Computer & Communications Security*, Scottsdale, Ariz, USA, November 2014.

[23] Apple's Differential Privacy Collecting Data," 2016, https://www.wired.com/2016/06/apples-differential-privacy-collecting-data/.

[24] T. T. Nguyen, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," *Computer Science Databases*, vol. 16, 2016.

[25] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," *Foundations of Computer Science Annual Symposium on*, vol. 26, no. 2, pp. 51–60, 2010.

[26] M. Abadi, A. Chu, I. Goodfellow et al., "Deep learning with differential privacy," in *Proceedings of the Acm Sigsac Conference On Computer & Communications Security*, Vienna, Austria, October 2016.

[27] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2003.

[28] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," *Computer and Communications Security*, vol. 28, pp. 43–54, 2016.

[29] C. Dwork, "Calibrating noise to sensitivity in private data analysis," *Lecture Notes in Computer Ence*, vol. 3876, no. 8, pp. 265–284, 2012.

[30] Y. Zhang, P. Liang, and M. Charikar, "A hitting time analysis of stochastic gradient langevin dynamics," *Proceedings of Machine Learning Research*, vol. 65, pp. 1–43, 2017.

[31] S. L. Warner, "Randomized response: a survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.

[32] X. Lian, Y. Huang, Y. Li, and L. Ji, "Asynchronous parallel stochastic gradient for nonconvex optimization," in *Proceedings of the International Conference On Neural Information Processing Systems*, Montreal, Canada, December 2015.

[33] C. Jin, P. Netrapalli, R. Ge, S. M. Kakade, and M. I. Jordan, "Stochastic gradient descent escapes saddle points efficiently," *arXiv: Learning*, vol. 31, 2019.

[34] F. M. Harper and J. A. Konstan, "The movielens datasets: history and context," *Ksii Transactions on Internet and Information Systems*, vol. 5, no. 4, p. 19, 2016.

[35] R. C. Geyer, T. Klein, and M. Nabi, *Differentially private federated learning: a client level perspective*, 2017.

[36] H. Kikuchi and A. Mochizuki, "Privacy-preserving collaborative filtering using randomized response," *Journal of Information Processing*, vol. 21, no. 4, pp. 671–676, 2012.

[37] Y. Xin and T. Jaakkola, "Controlling privacy in recommender systems," in *Proceedings of the International Conference On Neural Information Processing Systems*, Long Beach, CA, USA, December 2014.