Knowledge-Based Systems 250 (2022) 109044

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Differentially private recommender system with variational autoencoders

Le Fang, Bingqian Du, Chuan Wu [*]

Department of Computer Science, The University of Hong Kong, Hong Kong

ARTICLE INFO

Article history:
Received 24 September 2021
Received in revised form 7 May 2022
Accepted 10 May 2022
Available online 18 May 2022

Keywords:
Recommender system
Differential privacy
Variational autoencoders

ABSTRACT

To provide precise recommendations, traditional recommender systems (RS) collect personal data, user preference and feedback, which are sensitive to each user if such information is maliciously used for extra analysis. In recent years, differential privacy (DP) has been widely applied in RS to provide privacy protection for sensitive information. Prior studies explored the combination of DP and RS, while neglecting the disparate effect on model accuracy of imbalanced subgroups as large user groups control the trained model, and DP can worsen the disparate effect of degrading the performance of recommender systems significantly. Besides, the number of uploaded contributions can differ among users for training a recommender system, so it is necessary to set the user-level privacy guarantee.

In this paper, we make four contributions. First, we propose an efficient way of constructing datasets for training a recommender system based on prior theories. Second, we compute the user-level priors based on user metadata to optimize the VAE model. Besides, we add noise into the calculation process to protect user metadata. Third, we analyze and propose a tighter theoretical bound on gradient updates for DP Stochastic Gradient Descent (DPSGD). Finally, we exploit these theoretical results and propose a novel DP-VAE based recommender system. Extensive experimental results on multiple datasets show that our system can achieve high recommendation precision while maintaining a reasonable privacy guarantee.

1. Introduction

Recommender systems (RS) have been widely applied in various services [1–6], such as in the recommendation engines of Netflix [7], Amazon [8] and Alibaba [9]. A standard RS extracts each user's information, learns the relationship among users and items and provides the recommended item(s) accordingly. The process is quite privacy-intrusive as the user data needed for making recommendations is usually quite sensitive. For example, users may have to answer some sensitive questions raised by the system periodically to achieve a higher recommendation precision, such as location and age [10]. Differential privacy (DP) has been applied for privacy guarantee in RS [11–17]. McSherry et al. [7] first integrated DP to film recommendation and focused on ensuring the same privacy guarantee for each user, while neglecting that the added noise may affect differently on the recommendation performance among users with different amounts of rating information [18]. In case of different user contributions, user-level differential privacy can be applied in a RS, which provides the privacy guarantee at the privacy level that each user specifies according to its need. Collaborative filtering (CF) based methods [19–21] are widely applied in recommender systems, which predict what items a user prefers by measuring the similarities across users and items. Recently, variational autoencoder (VAE) models [22] have been introduced into RS design [23–25] to involve side information for better recommendation, *e.g.*, to combine ratings with other user metadata. It can address the drawbacks of CF-based methods in terms of the rating information matrix sparsity and the cold start issue [26] (that it is hard to recommend items to new users without any previous rating information), as VAE [22,23] is a non-linear model powered by neural network that can capture more complex patterns in the data. We adopt VAE as the RS model and apply user-level DP for a better privacy guarantee.

Besides, most prior works assumed that each user only contributes one rating score for a specific item, which does not address the real-world scenario: one may contribute multiple sets of rating information with different timestamps for a particular item. For example, a user can rate some items with higher or lower scores due to the taste change, which is common in video/commodity recommendation [9,27]. Here allowing more data contribution implies that we need to add more noise to protect individual users with DP, which increases the variance of user contribution and leads to a significant downgrade of the

* Corresponding author.
E-mail addresses: lefang@connect.hku.hk (L. Fang), bqdu@hku.hk (B. Du), cwu@cs.hku.hk (C. Wu).

https://doi.org/10.1016/j.knosys.2022.109044
0950-7051/© 2022 Elsevier B.V. All rights reserved.

performance of RS [28]. Amin et al. [28] discussed the bias–variance trade-off introduced by bounding user contributions with DP, analyzed user contributions (covered portions of the whole dataset) on the expected error of differentially private empirical risk minimization in the model training process, and proposed an optimal clipping bound. Besides, if there is a high contribution bias among users, the bias can cause a disparate effect on the final accuracy of the underrepresented users when differential privacy [18] is applied. Juba et al. [29] analyzed the precision–recall curve considering the classification task of imbalanced data and stated that a larger number of examples would be necessary and sufficient to address class imbalance. Similar observation has been made by McMahan et al. [30] and they argued that increasing the size of the dataset can improve the accuracy while maintaining privacy protection.

We strategically design a DP-VAE RS in this paper, aiming to provide precise recommendations while maintaining an appropriate privacy guarantee for each user. Our contribution is fourfold:

- We propose a novel DP-VAE recommender system to achieve high recommendation precision while maintaining a reasonable privacy guarantee.
- We propose the user-level priors based on the user metadata for optimizing the VAE-based models and add noise into the calculation process of user-level priors to protect the metadata.
- We propose an efficient way of building training datasets, which handles the problem of the uneven contribution among users and avoids the model performance degradation of the trained model.
- We propose an optimal clipping bound for the Gaussian Mechanism when conducting DPSGD, which avoids a significant decline in recommendation performance caused by excessive privacy guarantee.

2. Preliminaries and related work

2.1. Differential privacy

Differential privacy is a mechanism that ensures privacy protection over a group of participants by adding randomness into the training process, which can protect all participants from predicting the participation of each of them concisely.

Definition 1 (*Differential privacy [31,32].*)**.** Given a dataset of n samples, a randomized mechanism $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{O}$ is (ϵ, δ)-differential private if given any pairs of datasets D and D', which differs exactly in one record, the inequality $Pr(\mathcal{M}(\mathcal{D}) \in \mathcal{O}) \leq exp(\epsilon) \cdot Pr(\mathcal{M}(\mathcal{D}') \in \mathcal{O})$ holds with a probability of $1 - \delta$ under the given protection level ϵ:

$$Pr(\mathcal{M}(\mathcal{D}) \in \mathcal{O}) \leq exp(\epsilon) \cdot Pr(\mathcal{M}(\mathcal{D}') \in \mathcal{O}) + \delta \tag{1}$$

Here ϵ and δ are a pair of parameters for controlling the level of privacy protection. $\epsilon \in (0, +\infty)$ is the privacy budget and $\delta \in [0, 1]$ is the given probability. Note that each smaller value of both indicates a higher privacy guarantee. For user-level differential privacy, we can specify different ϵ_k and δ_k for different user k, which is useful when different users are allowed to specify the privacy guarantee within their interests.

Definition 2 (*User-level adjacent datasets [30].*)**.** Considering two datasets D_k and D'_k both from user k, D_k and D'_k are user-level adjacent datasets if D'_k can be formed from D_k by adding or removing one single data sample from user k.

Definition 3 (*User-Level l_2 Sensitivity [31,32].*)**.** Considering two user-level adjacent datasets D_k and D'_k from user k, given the query function for the specific kth user, $f^k : D_k \rightarrow O_k$, the l_2 sensitivity is defined as follows:

$$\triangle^k_{f2} = \max_{D_k, D'_k} \|f(D_k) - f(D'_k)\|_2. \tag{2}$$

$\triangle^k_{f2}$ defines the maximal change after adding/removing any single data sample from the specific kth user. Specially, the l_2 sensitivity is applied in the Gaussian mechanism [31,32], which preserves the (ϵ, δ)-differential privacy by adding noise sampled from Gaussian distribution with standard deviation $\sqrt{2 \log \frac{1.25}{\delta_k}} \frac{\triangle^k_{f2}}{\epsilon_k}$. For SGD-based method, $\triangle^k_{f2}$ is defined as the maximal l_2-norm change of gradients for adjacent datasets D and D'. Noises are added into each gradient update in the training process when adopting differential privacy (Table 1).As introduced in [33], we clip each user's gradient based on a clipping bound S and then add noise into each gradient before averaging the gradient updates.

2.2. Differentially-private recommender system

Differential privacy was first adopted in a movie recommender system for user privacy protection [7], where the noise was added to the rating information when building an item-to-item covariance matrix. Differentially private matrix factorization (DPMF) [13,34] added noise to the sampled rating information for collaborative filtering. Differentially private recommender system with the auto-encoder (DPAE) [14,15] adopted an auto-encoder as the framework and noise was added to the objective function. Unlike prior works, our design adopts the VAE architecture with user-level differential privacy that can efficiently address some issues from representative CF methods such as cold-start, sparsity, etc. Besides, we analyze and propose an optimal gradient clipping bound in DPSGD.

2.3. Recommender system with VAEs

Variational autoencoders have been applied in recommender systems [23,35,36], which consist of an inference model (encoder) and a generative model (decoder). Here we input the user–item matrix containing the rating information into the encoder to generate a low-dimension latent vector with the variational distribution. Then the decoder generates new rating scores by reconstructing the matrix from the latent vector. Items with high predicted scores can be recommended to users accordingly. An illustration is given in Fig. 1.

2.3.1. Encoder

The encoder reduces the original matrix dimension as most valuable features can be preserved in a low-dimension layer, which can reduce the computational cost. Let x_k be the input of user k to the encoder and z_k denote the generated latent vector for user k. The goal of the encoder is to find a plausible $q_\phi(z_k|x_k)$ to approximate the intractable posterior $q(z_k|x_k)$. We compute the true posterior $q_\phi(z_k|x_k)$ parameterized by ϕ and adopt the data-dependent function $g_\phi = [\mu_\phi(x_k), \sigma_\phi(x_k)] \in \mathbb{R}^{2K}$ [37] to let $q_\phi(z_k|x_k)$ be a variational distribution (Gaussian distribution is used). According to another prior work [38], we can approximate the true posterior with a variational distribution: $q_\phi(z_k|x_k) = \mathcal{N}(\mu_\phi(x_k), diag\{\sigma^2_\phi(x_k)\})$, where $\mu_\phi(x_k) \in \mathbb{R}^K$ and $\sigma^2_\phi(x_k) \in \mathbb{R}^{K^2}$ parameterized by ϕ can be learned by the encoder.

Table 1
Summary of all notations.

k	refers to a specific user	U	the total number of users
i	refers to a specific item	I	the total number of items
X	refers to the user–item matrix and $X \in \mathbb{R}^{U\times I}$	C	the total number of item categories
K	the dimension of the latent representation vector in VAE	c_k	the preference category list for user k
c_i	the category for item i	x_k	uploaded ratings from user k
x'_k	new ratings for user k	z_k	sampled latent vector for user k
β	regulation parameter for ELBO	ω, δ	the accuracy, the probability
d	the dimension of selected categories for dataset construction	p_k	the weight factor determining the actual dataset size for user k
r^t_{ki}	refers to the tth rating score of item i uploaded by user k	R_{ki}	the final rating score of item i for user k
$\mathcal{U}$	the number of sampled users in each batch for DPSGD	$\mathcal{J}$	the number of sampled examples for each user in each batch
S	the optimal clipping bound on gradient updates for DPSGD	η	refers to the learning rate for DPSGD
$\bar{g}_k$	the gradient by adding noise into $\hat{g}_k$	g_k, $\hat{g}_k$	the original gradient and the clipped gradient for user k
r	the ranking position in the recommendation list	$i_k(r)$	the recommended item at rank r for user k
Q_k	items in the recommendation list for user k	ϵ_k, δ_k	privacy parameters for user k
o	refers to the number of metadata categories	ϕ, θ	parameter pairs for encoder and decoder
m_k	represented metadata vector for user k	μ_w, γ_w	mean–variance pairs for each metadata category, $w \in [1, o]$
σ_k	the noise parameter in Gaussian Mechanism	μ_k, γ_k^2	user-level priors for sampling z_k for user k

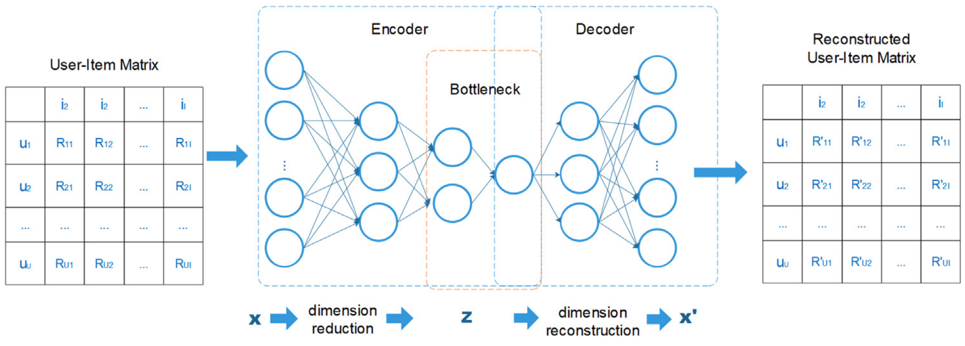


Fig. 1. Overview of the VAE-based RS.

2.3.2. Decoder

The decoder reconstructs the original matrix and sends new item recommendations with high generated rating scores. The decoder starts with sampling a low K-dimensional latent representation vector $z_k \in \mathbb{R}^K$ from the Gaussian prior ($z_k \sim \mathcal{N}(0, \mathbb{I}_K)$) as input. Then a non-linear function $f_\theta \in \mathbb{R}^I$ with parameters θ maps z_k to a probability distribution over I items, which is defined as $\pi^\theta(z_k)$ and $\pi^\theta(z_k) \propto \exp\{f_\theta(z_k)\}$. Besides, we assume that each user–item matrix row x_k (x_{ki} denoting the ith value) is drawn from a multinomial distribution $Mult()$ with probability $\pi^\theta(z_k)$, that is $x_k \sim Mult(N_k, \pi^\theta(z_k))$, where number of item ratings for user k. We can define the log-likelihood for x_k as $\log p_\theta(x_k|z_k) = \sum_{i\in I} \log(\pi_i^\theta(z_k))$.

2.3.3. Learning process

For each user, the objective is to maximize the evidence lower bound (ELBO) as given in Eq. (3), which assures that the distance between $q_\phi(z_k|x_k)$ and $p_\theta(z_k|x_k)$ can be minimized. The first term is the negative reconstruction error that measures how input fits our trained model and qualifies the reconstruction performance; the second term is the Kullback–Leibler (KL) divergence [39] that can be regarded as a regulation term.

$$\mathcal{L}(x_k; \phi, \theta) \equiv \mathbb{E}_{q_\phi(z_k|x_k)}[\log(p_\theta(x_k|z_k))] - KL(q_\phi(z_k|x_k) \parallel p(z_k)) \tag{3}$$

Lemma 1. *The objective function $\mathcal{L}(x_k; \phi, \theta)$ in Eq. (3) for VAE is equivalent to maximizing the log-likelihood of input user–item matrix and minimizing the KL-divergence between posterior of z_k evaluated by encoder and decoder at the same time.*

Proof. We have known that the encoder aims to make $q_\phi(z_k|x_k)$ parameterized by ϕ approximate the intractable posterior $q(z_k|x_k)$

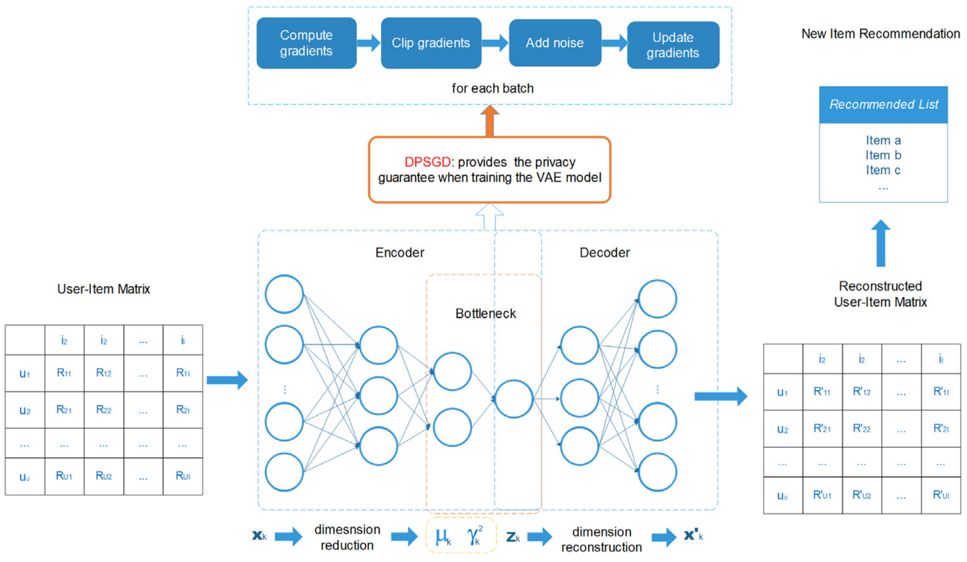


Fig. 2. Overview of the system design.

when considering the input x_k from user k. For the decoder, we can also obtain $p_\theta(z_k|x_k)$ parameterized by θ that should also approximate the true posterior $q(z_k|x_k)$. For both $q_\phi(z_k|x_k)$ and $p_\theta(z_k|x_k)$, we use the Kullback–Leibler (KL) divergence to measure the similarity of them. So we can obtain the basic objective function as follows:

$$\min KL(q_\phi(z_k|x_k) \parallel p_\theta(z_k|x_k)) \tag{4}$$

Besides, for input x_k with $p_\theta(x_k)$, we can define a new function $\mathcal{L}(x_k;\phi,\theta)$ and the log-likelihood of x_k as follows:

$$\log p_\theta(x_k) = KL(q_\phi(z_k|x_k) \parallel p_\theta(z_k|x_k)) + \mathcal{L}(x_k;\phi,\theta) \tag{5}$$

Then we can obtain $\mathcal{L}(x_k;\phi,\theta)$:

$$\begin{aligned}\mathcal{L}(x_k;\phi,\theta) &= \log p_\theta(x_k) - KL(q_\phi(z_k|x_k) \parallel p_\theta(z_k|x_k))\\ &= \mathbb{E}_{q_\phi(z_k|x_k)}[-\log(q_\phi(z_k|x_k)) + \log p_\theta(x_k|z_k)p_\theta(z_k)]\\ &= \mathbb{E}_{q_\phi(z_k|x_k)}[\log(\frac{p_\theta(z_k)}{q_\phi(z_k|x_k)}) + \log p_\theta(x_k|z_k)]\\ &= \mathbb{E}_{q_\phi(z_k|x_k)}[\log(p_\theta(z_k|x_k))] - KL(q_\phi(z_k|x_k) \parallel p_\theta(z_k))\end{aligned} \tag{6}$$

So the new objective function is $\max \mathcal{L}(x_k;\phi,\theta)$, which is proportional to maximizing $\log p_\theta(x_k)$ and minimizing Eq. (4). □

Prior work [23] adopted an additional parameter ($\beta \in [0,1]$) with the ELBO to calibrate the trade-off between how well we can fit the data and how close the approximate posterior $q_\phi(z_k|x_k)$ is to the prior $p(z_k)$. We also adopt the best β in our design to maximize the model precision by gradually increasing β from 0 to 1.

$$\begin{aligned}\mathcal{L}_\beta(x_k;\phi,\theta) \equiv\ & \mathbb{E}_{q_\phi(z_k|x_k)}[\log(p_\theta(x_k|z_k))]\\ & - \beta \cdot KL(q_\phi(z_k|x_k) \parallel p(z_k))\end{aligned} \tag{7}$$

We design our VAE-based model based on the VAE architecture for item recommendation in [23]. We optimize the latent vector z_k by replacing the standard Gaussian distribution $\mathcal{N}(0, \mathbb{I}_K)$ with the user-level priors $\mathcal{N}(\mu_k, \sigma_k^2)$ that can be computed with user metadata such as gender, age, location and hobbies, prior observed item categories. Another work [36] for VAE-based RS proposed an optimized way for generating z_k with user-level priors $\mathcal{N}(\mu_k, \sigma_k^2)$ by training a Latent Dirichlet Allocation (LDA) model [40], while we directly compute these priors based on public pre-trained word embeddings. None of these works studies how to protect user privacy. We study VAE-based RS and apply user-level differential privacy for user privacy protection. Besides, we propose a tight threshold of collecting user training examples for better recommendation precision and identify a trade-off between privacy and accuracy for clipping each gradient in DPSGD and adding noise accordingly before sending gradient updates.

3. Methods

3.1. System overview

As shown in Fig. 2, the input is the user–item matrix $X \in \mathbb{R}^{U\times I}$, where U is the number of users, I is the number of items, and each component is a vector representing the closeness of the relationship between the specific user and the item that the user has interactions with, including two main factors: the ratings of the user to the item and the number of user–item interactions. We decide the number of sampled contributions from each user and ensure better recommendation accuracy. We also propose a technique for building the user–item matrix when a user contributes a set of rating information for a specific item, rather than one. Inspired by prior work [36], we propose an efficient way to generate user-level priors for sampling the latent vector z_k based on the user metadata. We further add user-specified noise into the user-level priors to protect the metadata from adversary analysis. Besides, we use VAE and adopt DPSGD [30] for training the VAE model. Instead of adopting global differential privacy for all users [30], we apply user-level differential privacy, which allows each user to specify its privacy budget. We also propose a gradient clipping bound in DPSGD, which preserves more valuable updates and ensures that the trained model is not controlled by users who contribute more.

3.2. Constructing training datasets

We note that the uneven contribution problem among users can degrade the model performance sharply [28]. To our best knowledge, there is no theoretical study on deciding the size of the training dataset (especially the number of contributions for each user) when training a recommender system. However, prior work has investigated that the size of the training dataset affects the model accuracy when training classifiers [41]. Since each user is different from others considering their personalities, item preferences, *etc.*, we can regard each user as a classifier.

The collected data from a specific user can present the user's preferences, which are distinct from others.

As proven in existing work [41], a trained recommendation model can achieve an accuracy more than $1-\omega$ (error rate ω) with the probability at least $1-\delta$ when setting the lower bound $\Omega(\frac{1}{\omega}(d+\log\frac{1}{\delta}))$ and upper bound $O(\frac{1}{\omega}(d+\log\frac{1}{\delta}))$ with the dimension d of selected categories (here d approximates the number of items I if we set the bound for each user), respectively, for collecting the training examples from each user. Here we unify both bounds into $\Theta(\frac{1}{\omega}(d+\log\frac{1}{\delta}))$ on labeled examples to achieve the accuracy $1-\omega$ with probability $1-\delta$. We decide the number of training examples for user k as $\frac{p_k}{\omega}(d+\log\frac{1}{\delta})$ with the weight factor $p_k \in (0,1]$ representing user contributions: $p_k=0$ indicates that user k does not upload any examples and $p_k=1$ means that user k contributes enough examples satisfying the contribution bound. Furthermore, we can bound $p_k \in [p_{min},1]$ and filter users with $p_k < p_{min}$, where p_{min} determines the minimal user contribution and will be set to the value allowing the maximal model precision, by gradually increasing value of p_{min} from 0 to 1. The total number of examples is then $\frac{\sum_{k=1}^{U} p_k}{\omega}(d+\log\frac{1}{\delta}))$.

Theorem 1. *We set the bound of uploaded examples for user k as $\frac{p_k}{\omega}(d+\log\frac{1}{\delta})$ with the weight factor p_k and the trained model can achieve approximate accuracy $1-\frac{\omega}{\sum_{k=1}^{U} p_k}$ with a probability at least $1-\delta$.*

Proof. We define $V=\{x_1,x_2,\ldots,x_U\}$, where each x_k is a set of samples from user k and independent from other users. We define a process $h(b)$ to predict if the rating score b belongs to x_k and return values -1 (wrong prediction) or 1 (correct prediction) accordingly, where $h \in \{h_1,h_2,\ldots,h_U\}$. So we define the list of all the examples with wrong predictions as $ER(h)=\{b \in V : h(b)=-1\}$ and the error rate of the prediction process as $er(h)$. Then we define a "Majority Function" $Maj(V_j)=2\times \mathbb{1}[\sum_{b\in V_j} h(b)>0]-1$ with $V_j \subseteq V (j\in\mathbb{N})$, where $\mathbb{1}$ is the indicator function. Note that $Maj(V_j)=1$ indicates that most predictions are correct while $Maj(V_j)<-1$ show that there are more wrong predictions.

Algorithm 1: $\mathbb{A}(V,Z)$

```
   Input: two datasets $V$ and $Z$
   Output: a set of subsets $Q$
1  Compute the length of $V$ as $l_s$;
2  if $l_s \le 3$ then
3      $Q \leftarrow V \cup Z$ ;
4      Return $Q$ ;
5  end
6  else
7      for $j \in \{0,1,2,3\}$ do
8          $V_j \leftarrow V_{jl_s/4+1:(j+1)l_s/4}$ ;
9      end
10     $Z_1 \leftarrow V_2 \cup V_3 \cup Z, Z_2 \leftarrow V_1 \cup V_3 \cup Z\ \ Z_3 \leftarrow V_1 \cup V_2 \cup Z$ ;
11     Return $\mathbb{A}(V_0,Z_1) \cup \mathbb{A}(V_0,Z_2) \cup \mathbb{A}(V_0,Z_3)$ ;
12 end
```

To prove the theorem, we need to prove that with the probability at least $1-\delta$ $(\delta \in (0,1))$, we have the error rate $er \le \frac{\omega}{\sum_{k=1}^{U} p_k}$ when the number of uploaded examples is $\frac{\sum_{k=1}^{U} p_k}{\omega}(d+\log\frac{1}{\delta}))$. With the probability $1-\delta$ and $m' \le c\log(\frac{1}{\delta})-1 (m'\in\mathbb{N})$, we set the numerical constant and get the inequality as follows:

$$er(h) \le 1 \le \frac{c}{m'+1}(d+\log\frac{1}{\delta}). \tag{8}$$

Besides, we denote $V_0=V_{1:m/4}, V_j=V_{jm/4+1:(j+1)m/4}$, where $j\in\{1,2,3\}$. Besides, we define $L(\cdot)$ as the empirical risk minimizer. And we can denote $h_j = Major(L(\mathbb{A}(V_0,V/\{V_j,V_0\})))$. Hanneke [41] proposed Algorithm 1 to generate finite subsets, where each subset contains no more than 3 training examples. Then we can define $h_{maj}=Maj(L(\mathbb{A}(V,\emptyset)))$. And Hanneke [41] has proved that if we set $m > c\log(\frac{1}{\delta})-1 > m'$, we can obtain the following:

$$er(h_{maj}) \le \frac{c}{m+1}(d+\log\frac{1}{\delta}). \tag{9}$$

As c is a numerical value, we can set c =1, select $m = \frac{\sum_{k=1}^{U} p_k}{\omega}(d+\log\frac{1}{\delta})$ and obtain the following:

$$\begin{aligned} er(h_{maj}) &\le \frac{1}{m+1}(d+\log\frac{1}{\delta}) \\ &\le \frac{1}{\frac{\sum_{k=1}^{U} p_k}{\omega}(d+\log\frac{1}{\delta})}(d+\log\frac{1}{\delta}) = \frac{\omega}{\sum_{k=1}^{U} p_k}. \end{aligned} \tag{10}$$

This shows that the error rate is no more than $\frac{\omega}{\sum_{k=1}^{U} p_k}$, ending the proof. □

As we allow each user to contribute a set of rating scores for a specific item, we can compute the rating information in the user–item matrix by $R_{ki}=\sum_{t=1}^{m} r_{ki}^{t}(\frac{2t}{m(m+1)})$, where R_{ki} is the final rating score between user k and item i, r_{ki}^{t} is the tth rating score of item i uploaded by user k and t ranges in $[1,m]$. As new rating scores can commonly describe user's preference better, we adopt factor $\frac{2t}{m(m+1)}$ $(\sum_{t=1}^{m}\frac{2t}{m(m+1)}=1)$ in calculating R_{ki}.

3.3. Training the VAE model

3.3.1. Estimating the user-level priors

We focus on the VAE design introduced in [23], but use an optimized ELBO and combine it with user-level differential privacy, which can achieve both good recommendation performance and user-specified privacy guarantee. Inspired by another work [36], we replace the standard Gaussian distribution by adopting the user-level priors $\mu_k \in \mathbb{R}^K$ and $\gamma_k^2 \in \mathbb{R}^{K\times K}$ $(z_k \sim \mathcal{N}(\mu_k,\gamma_k^2))$, which can be computed based on user metadata (as shown in Eq. (11)). We note that it is not compulsory to access user metadata for training the VAE model. Herein we aim to provide the privacy guarantee when using the metadata to augment the recommender systems [36].

To estimate μ_k and γ_k^2, we use the "Global Vectors" (GloVe) embeddings [42], which are pre-trained vector representations that show the similarities among words and can be used to compute the embeddings of user metadata, where μ_k can be regarded as the mean of the user's embeddings, and γ_k^2 is a diagonal covariance matrix where each diagonal value equals the deviation of the user's mean embeddings.

Here we select o kinds of metadata provided by each user, such as gender, occupation, age, etc. We define the metadata matrix $m_k \in \mathbb{R}^{o\times K}$ for user k as $m_k=[m_{k1},m_{k2},m_{k3},\ldots,m_{ko}]^{\top}$ and m_{kw} $(w\in[1,o])$ is a K-dimensional vector where each component m_{kwj} ranges in $[0,1]$ $(j \in [0,K]$ indicates the jth dimension of the latent vector). Then we can define the K-dimensional mean–variance pairs for each metadata as (μ_w,γ_w^2), where $\mu_{wj}=\frac{1}{U}\sum_{k=1}^{U} m_{kwj}$, $\gamma_{wj}^2=\frac{1}{U}\sum_{k=1}^{U}(m_{kwj}-\mu_{wj})^2$. We can compute each value μ_{kj} in μ_k and each diagonal value γ_{kj}^2 in γ_k^2 (as shown in Eq. (11)). Besides, we add noise sampled from $\mathcal{N}(0,\sigma_k^2)$ with $\sigma_k=\sqrt{2\log\frac{1.25}{\delta_k}}\frac{\triangle_{f2}^{k}}{\epsilon_k}$ into the final results based on the user-level privacy for user k, as the metadata can release some exact information of user k. For example, adversaries may access and collect the exact location, age, ID number of user k for illegal use after confirming the user information. Applying DP into the calculation prevents the adversary attacker from inferring whether the priors belong to a target user. Therefore, the attacker cannot be sure if the metadata belongs to the user even when the attacker accesses the metadata.

$$\mu_{kj} = \sum_{w=1}^{o} m_{kwj}\mu_{wj} + \mathcal{N}(0, \sigma_k^2),$$
$$\gamma_{kj}^2 = \sum_{w=1}^{o} (m_{kwj} - \mu_{kj})^2 + \mathcal{N}(0, \sigma_k^2). \tag{11}$$

Theorem 2. *To estimate user-level priors, we add user-level differential privacy into the mean–variance pairs under the Gaussian Mechanism with noise sampled from $\mathcal{N}(0, \sigma_k^2)$ with $\sigma_k = \sqrt{2\log\frac{1.25}{\delta_k}}\,\frac{\triangle_{f2}^k}{\epsilon_k}$ for protecting the metadata from user k; our computation process is (ϵ^k, δ^k)-differentially private.*

Proof. We select o kinds of metadata provided by each user, such as gender, age, etc. Then we compute the mean–variance pairs μ_k and γ_k^2 for user k. For example, we can extract two vector representations for gender words "male" and "female" according to the method proposed by Pennington et al. [42]. Then we can compute the normalized element-wise mean and the diagonal covariance matrix for the "gender" metadata after collecting the "gender" information from U users. Following the same way, we can compute the mean–variance pairs for other metadata.

For two given neighboring metadata matrices m_k and m_k', which only differ in one vector $m_{kw}(w \in [1, o])$ from user k's metadata. We define the function of computing the scalar value μ_{kj} as $f : m_k \to O_k$, where $f = \sum_{w=1}^{o} m_{kwj}\mu_{wj}$ and the noise y_k is sampled from $\mathcal{N}(0, \sigma_k^2)$.[1] According to Eq. (2), we can define the l_2-sensitivity for user k as $\triangle_{f2}^k = \max_{m_k, m_k'} \|f(m_k) - f(m_k')\|_2$.

To prove that our process is differentially private, we have to assure the privacy loss can be bounded by the privacy budget ϵ_k by adding the noise y_k sampled by $N(0, \sigma_k^2)$. So we consider the absolute privacy loss for any plausible output ψ_k with noise as follows:

$$\begin{aligned}\mathcal{L}_{|f(m_k)||f(m_k')|} &= \left|\ln\frac{Pr[f(m_k) + y_k = \psi_k]}{Pr[f(m_k) + y_k + \triangle_{f2}^k = \psi_k]}\right| \\ &= \left|\ln\frac{e^{(-1/2\sigma_k^2)y_k^2}}{e^{(-1/2\sigma_k^2)(y_k+\triangle_{f2}^k)^2}}\right| \\ &= |\ln e^{(-1/2\sigma_k^2)[y_k^2-(y_k+\triangle_{f2}^k)^2]}| \\ &= \left|-\frac{1}{2\sigma_k^2}(2y_k\triangle_{f2}^k + (\triangle_{f2}^k)^2)\right|.\end{aligned} \tag{12}$$

So here we obtain that $\mathcal{L}_{|f(m_k)||f(m_k')|}$ can be bounded by ϵ_k when $|y_k| < \frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}$. To assure the privacy loss is bounded by ϵ_k under the probability $1 - \delta_k$, we require

$$Pr[|y_k| \geq \frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}] < \delta_k. \tag{13}$$

As y_k is sampled from the standard Gaussian distribution, if we only consider the scenario $y_k \geq \frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}$, we need to maintain

$$Pr[y_k \geq \frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}] < \frac{\delta_k}{2}. \tag{14}$$

[1] Note that the process of computing γ_{kj} is as same as the way of calculating μ_{kj} (as shown in Eq. (11)), so we omit the discussion on the privacy guarantee for this one.

Then we use the tail bound $Pr[y_k > a] \leq \frac{a\sigma_k}{\sqrt{2\pi}}e^{\frac{-a^2}{2\sigma_k^2}}$ in [32] and set $a = \frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}$. So we require

$$\begin{aligned}&\frac{\sigma_k}{a\sqrt{2\pi}}e^{\frac{-a^2}{2\sigma_k^2}} < \frac{\delta_k}{2} \Leftrightarrow \frac{\sigma_k}{a}e^{\frac{-a^2}{2\sigma_k^2}} < \sqrt{2\pi}\frac{\delta_k}{2} \\ &\Leftrightarrow \frac{a}{\sigma_k}e^{\frac{a^2}{2\sigma_k^2}} > \frac{2}{\sqrt{2\pi}\delta_k} \Leftrightarrow \ln(\frac{a}{\sigma_k}) + \frac{a^2}{2\sigma_k^2} > \ln(\frac{2}{\sqrt{2\pi}\delta_k}) \\ &\Leftrightarrow \ln[\frac{1}{\sigma_k}(\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2})] + \frac{(\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2})^2}{2\sigma_k^2} > \ln(\sqrt{\frac{2}{\pi}}\frac{1}{\delta_k}).\end{aligned} \tag{15}$$

Here we make sure the first term $\ln[\frac{1}{\sigma_k}(\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2})] \geq 0$. And we rewrite $\sigma_k = \frac{b_k\triangle_{f2}^k}{\epsilon_k}$, where $b_k \geq 1$. So we can obtain

$$\begin{aligned}\frac{1}{\sigma_k}(\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}) &= \frac{\epsilon_k}{b_k\triangle_{f2}^k}[(\frac{b_k\triangle_{f2}^k}{\epsilon_k})^2 \times \frac{\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2}] \\ &= \frac{\epsilon_k}{b_k\triangle_{f2}^k}(\frac{b_k^2\triangle_{f2}^k}{\epsilon_k} - \frac{\triangle_{f2}^k}{2}) = b_k - \frac{\epsilon_k}{2b_k}.\end{aligned} \tag{16}$$

As $\epsilon_k \leq 1 \leq b_k$, we know that $b_k - \frac{\epsilon_k}{2b_k} \geq b_k - 1/2$. So we have $b_k \geq 3/2$ to support $\ln[\frac{1}{\sigma_k}(\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2})] \geq 0$.

Then we consider the second term:

$$\begin{aligned}\frac{(\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} - \frac{\triangle_{f2}^k}{2})^2}{2\sigma_k^2} &= \frac{\epsilon_k^2}{2b_k^2}[\frac{\sigma_k^2\epsilon_k}{\triangle_{f2}^k} \times \frac{b_k^2(\triangle_{f2}^k)^2}{\epsilon_k^2} - \frac{\triangle_{f2}^k}{2}]^2 \\ &= \frac{1}{2}(\frac{b_k^2}{\epsilon_k} - \frac{1}{2})^2 \times \frac{\epsilon_k^2}{b_k^2} = \frac{1}{2}(b_k^2 - \epsilon_k + \frac{\epsilon_k^2}{4b_k^2}).\end{aligned} \tag{17}$$

As we have $\epsilon_k \leq 1$ and we need $b_k \geq 3/2$, so we have $b_k^2 - \epsilon_k + \frac{\epsilon_k^2}{4b_k^2} \geq b_k^2 - 8/9 > 2\ln(\sqrt{\frac{2}{\pi}}\frac{1}{\delta_k})$. And we obtain that $b_k > \sqrt{2\ln(\frac{1.25}{\delta_k})}$ that can satisfy condition (15).

We set events $Y_1 = \{y_k : |y_k| \leq \frac{b_k^2\triangle_{f2}^k}{\epsilon_k} - \frac{\triangle_{f2}^k}{2}\}$ and $Y_2 = \{y_k : |y_k| > \frac{b_k^2\triangle_{f2}^k}{\epsilon_k} - \frac{\triangle_{f2}^k}{2}\}$, and $O_{k1} = \{f(m_k) + y_k | y_k \in Y_1\}$, $O_{k2} = \{f(m_k) + y_k | y_k \in Y_2\}$, for any plausible output O_k. We have:

$$\begin{aligned}Pr[f(m_k) \in O_k] &= Pr[f(m_k) + y_k \in O_{k1}] \\ &\quad + Pr[f(m_k) + y_k \in O_{k2}] \\ &= Pr[f(m_k) + y_k \in O_{k1}] + Pr[Y_2] \\ &\leq e^{\epsilon_k}Pr[f(m_k') \in O_k] + \delta_k.\end{aligned} \tag{18}$$

This shows our calculation process is (ϵ^k, δ^k)-differentially private for user k, ending the proof. □

Based on the user-level priors, the optimized ELBO (ELBO-O) for user k can be modified as:

$$\begin{aligned}\mathcal{L}_\beta(x_k; \phi, \theta, \mu_k, \gamma_k^2) &\equiv \mathbb{E}_{q_\phi(z_k|x_k)}[\log(p_\theta(x_k|z_k))] \\ &\quad - \beta \cdot KL(q_\phi(z_k|x_k) \parallel p(z_k; \mu_k, \gamma_k^2)).\end{aligned} \tag{19}$$

3.3.2. Computing the optimal clipping bound

To apply DP, we clip each sample's gradient with a clipping bound and then add Gaussian noise into the clipped gradient accordingly. In [33], they adopted the median value of all gradient norms as the threshold for clipping each gradient. Another work [28] proposed a $1 - \frac{1}{U\epsilon}$-quantile method based on the Laplace mechanism and chose the $\{\frac{1}{\epsilon}\}^{th}$ largest gradient norm as the threshold. For each batch, suppose we sample $\mathcal{U}$ users, and each user contributes $\mathcal{J}$ samples in the model training process. We compute the sum of gradients as follows: $\hat{g} = \frac{1}{\mathcal{U}}\sum_{k\in\mathcal{U}}\hat{g}_k$, where $\hat{g}_k$ is the average gradient of user k after clipping and noise addition. Given the original gradient, g_{kj}, which is the gradient of the jth sample for user k, the clipping operation is performed as $\bar{g}_{kj} = g_{kj}\times\min(1, \frac{S}{\|g_{kj}\|_2})$, and then the average gradient for user k is $\tilde{g}_k = \frac{1}{\mathcal{J}}\sum_{j=1}^{\mathcal{J}}\bar{g}_{kj}$. The noise addition operation is $\hat{g}_k = \tilde{g}_k + Z_k$, where Z_k is the l-dimensional random variable with each dimension sampled independently from the distribution $\mathcal{N}(0, \sigma_k^2)$, with $\sigma_k = \sqrt{2\log\frac{1.25}{\delta_k}}\frac{\triangle_{f2}^k}{\epsilon_k}$.

To find the optimal clipping bound S, we define an objective function $H(S)$ in the proof of Theorem 3 to minimize the expected error of actual gradients and estimated ones, and find S from all computed gradients based on a recursive algorithm (Algorithm 2).

Algorithm 2: *Optimal_Clip*(L_g, ϵ, δ)

```
Input: A list of gradients $L_g$, privacy parameters $\epsilon$, $\delta$
Output: the optimal clipping bound $S$
1  Compute the length of $L_g$ as $l$;
2  for $j \in \{\lfloor\frac{l}{2}\rfloor - 1, \lfloor\frac{l}{2}\rfloor, \lfloor\frac{l}{2}\rfloor + 1\}$ do
3      $H_j \leftarrow H(||g_j||_2)$ ;
4  end
5  if $H_{\lfloor\frac{l}{2}\rfloor} \le H_{\lfloor\frac{l}{2}\rfloor-1}, H_{\lfloor\frac{l}{2}\rfloor} \le H_{\lfloor\frac{l}{2}\rfloor+1}$ then
6      Return $||g_{\lfloor\frac{l}{2}\rfloor}||_2$ ;
7  end
8  else
9      if $H_{\lfloor\frac{l}{2}\rfloor+1} \le H_{\lfloor\frac{l}{2}\rfloor} \le H_{\lfloor\frac{l}{2}\rfloor-1}$ then
10         $L_g = \{g_{\lfloor\frac{l}{2}\rfloor+1}, ..., g_l\}$ ;
11     end
12     else
13         $L_g = \{g_1, ..., g_{\lfloor\frac{l}{2}\rfloor-1}\}$ ;
14     end
15     Return Optimal_Clip$(L_g, \epsilon, \delta)$ ;
16 end
```

Theorem 3. *For the user-level DPSGD training process with noise scale of* $\sqrt{2\log\frac{1.25}{\delta_k}}\frac{\triangle_{f2}^k}{\epsilon_k}$ *for user k, the optimal clipping bound S for all users can be decided by a recursive algorithm. The algorithm complexity is* $O(\log(\mathcal{U}\mathcal{J}))$.

Proof. For each user, we can compute the expected error caused by estimated gradient $\hat{g}$ (which describes the distance of the actual gradients and perturbed ones.), as follows:

$$\mathbb{E}|\hat{g} - g| \le \mathbb{E}|\hat{g} - \tilde{g}| + \mathbb{E}|\tilde{g} - g|$$
$$\le \frac{1}{\mathcal{U}}\sum_{k\in\mathcal{U}}[\mathbb{E}|\hat{g}_k - \tilde{g}_k| + \frac{1}{\mathcal{J}}\sum_{j\in\mathcal{J}}(\mathbb{E}|\tilde{g}_k - \bar{g}_{kj}| + \mathbb{E}|\bar{g}_{kj} - g_{kj}|)]$$
$$\le \frac{1}{\mathcal{U}}\sum_{k\in\mathcal{U}}\{l\sqrt{2/\pi}\sqrt{2\log\frac{1.25}{\delta_k}}\frac{S}{\epsilon_k} + \frac{1}{\mathcal{J}}\sum_{j\in\mathcal{J}}[\mathbb{E}|\tilde{g}_k - \bar{g}_{kj}|$$
$$+ \frac{|g_{kj}|}{\|g_{kj}\|_2}\times\max(0, \|g_{kj}\|_2 - S)]\}.$$

Then we can define a function $H(S)$:

$$H(S) = \frac{1}{\mathcal{U}}\sum_{k\in\mathcal{U}}\{l\sqrt{2/\pi}\sqrt{2\log\frac{1.25}{\delta_k}}\frac{S}{\epsilon_k} + \frac{1}{\mathcal{J}}\sum_{j\in\mathcal{J}} [\mathbb{E}|\tilde{g}_k - \bar{g}_{kj}| + \frac{|g_{kj}|}{\|g_{kj}\|_2}\times\max(0, \|g_{kj}\|_2 - S)]\}. \quad (20)$$

So our objective is to minimize the convex function $H(S)$ [28]. As shown in Eq. (20), if we set $S = 0$, the error is $\frac{1}{\mathcal{U}\mathcal{J}}\sum_{k\in\mathcal{U}}\sum_{j\in\mathcal{J}}|g_{kj}|$, where the first term can be quite large if we set a large S. We give a plausible assumption that some gradient norms are no larger than S while other gradient norms are larger than S [28]. We sort all l_2 gradients from the largest to the smallest based on the gradient norms as $L_g = \{g_1, g_2, \dots, g_l\}$. Then we obtain S by recursively using the bisection method. As shown in Algorithm 2, we can obtain the optimal clipping bound S = *Optimal_Clip* $(L_g, \epsilon_k, \delta_k)$, which assures that $H(S)$ is minimized considering all norms of gradients. □

3.3.3. *Designing the complete algorithm*

We next present the complete user-level DP-VAE algorithm in Algorithm 3. The privacy parameter pairs (ϵ_k, δ_k) can be set initially for each user. Besides, for additional metadata (such as gender, location, age and hobbies) provided by the user k, we can compute the user's word embeddings for generating mean–variance pairs (μ_k, γ_k^2) before training the VAE model. For user k, the computed gradients of the jth training example x_{kj} are $\triangledown_\theta\mathcal{L}_\beta(x_{kj})$ and $\triangledown_\phi\mathcal{L}_\beta(x_{kj})$. We compute the clipping bound S = *Optimal_Clip*$(L_g, \epsilon_k, \delta_k)$ based on the ordered gradient list L_g and privacy parameters. Then we clip each gradient based on the clipping bound S, average the gradients for each user and add noise based on the user-level differential privacy. Finally, we compute the average gradients and carry out gradient updates.

Algorithm 3: Training the DP-VAE model with user-specified noise

```
Input: user-item matrix $X \in \mathbb{R}^{U\times I}$, learning rate $\eta$,
       randomly selected $\phi$, $\theta$, the total training rounds $T$,
       privacy parameters $\{(\epsilon_1, \delta_1), ..., (\epsilon_U, \delta_U)\}$, clipping
       bound $S$, computed mean-variance pairs
       $\{(\mu_1, \gamma_1^2), ..., (\mu_U, \gamma_U^2)\}$
Output: $\phi$, $\theta$
1  for $t \in \{1, 2, .., T\}$ do
2      Sample $\mathcal{U}$ users as a batch ;
3      for each sampled user $k$ do
4          Sample $z_k \in \mathbb{R}^k$ via $\mathcal{N}(\mu_k, \gamma_k^2)$ ;
5          Sample $\mathcal{J}$ training examples ;
6          for $j \in \{0, 1, ..., \mathcal{J}\}$ do
7              Compute the gradients: $g_{kj\phi} \leftarrow \triangledown_\phi\mathcal{L}_\beta(x_{kj})$ and
                $g_{kj\theta} \leftarrow \triangledown_\theta\mathcal{L}_\beta(x_{kj})$ ;
8          end
9          Allocate gradients into a list and sort it from
            largest to smallest as $L_g$ ;
10         Update $S$: $S$ = Optimal_Clip$(L_g, \epsilon_k, \delta_k)$ ;
11         for $j \in \{0, 1, ..., \mathcal{J}\}$ do
12             Clip the gradients: $\bar{g}_{kj\phi} \leftarrow g_{kj\phi}\times\min(1, \frac{S}{||g_{kj}||_2})$
                and $\bar{g}_{kj\theta} \leftarrow g_{kj\theta}\times\min(1, \frac{S}{||g_{kj}||_2})$ ;
13         end
14         Average the gradients: $\tilde{g}_{k\phi} \leftarrow \frac{1}{\mathcal{J}}\sum_{j\in\mathcal{J}}\bar{g}_{kj\phi}$ and
            $\tilde{g}_{k\theta} \leftarrow \frac{1}{\mathcal{J}}\sum_{j\in\mathcal{J}}\bar{g}_{kj\theta}$ ;
15         Add noise: $\hat{g}_{k\phi} \leftarrow \tilde{g}_{k\phi} + \mathcal{N}(0, \sigma_k^2)$ and
            $\hat{g}_{k\theta} \leftarrow \tilde{g}_{k\theta} + \mathcal{N}(0, \sigma_k^2)$ ;
16     end
17     Average the gradients: $\hat{g}_\phi \leftarrow \frac{1}{\mathcal{U}}\sum_{k\in\mathcal{U}}\hat{g}_{k\phi}$ and
        $\hat{g}_\theta \leftarrow \frac{1}{\mathcal{U}}\sum_{k\in\mathcal{U}}\hat{g}_{k\theta}$ ;
18     Update $\phi$ and $\theta$: $\phi \leftarrow \phi - \eta\hat{g}_\phi$ and $\theta \leftarrow \theta - \eta\hat{g}_\theta$ ;
19 end
```

Theorem 4. *For Gaussian Mechanism* $\mathcal{M}(g_k) = f(g_k) + Z_k$ *for user k in DPSGD, where* S = *Optimal_Clip*$(L_g, \epsilon_k, \delta_k)$ *and noise*

$Z_k \sim \mathcal{N}(0, \sigma_k^2)$, the Gaussian Mechanism satisfies (ϵ_k, δ_k)-differential privacy.

Proof. In Algorithm 3, we add noises sampled from $Z_k \sim \mathcal{N}(0, \sigma_k^2)$ to $\tilde{g}_{k\phi}$ (which equals to $\hat{g}_{k\phi}$).

Prior work [30] have proven that: there exists constants c_1, c_2, sampling probability q and the number of steps T such that for function f with sensitivity smaller than 1, for any $\epsilon_k < c_1 q^2 T$, the mechanism $\mathcal{M}(g_k) = f(g_k) + Z_k$ satisfies (ϵ_k, δ_k) for any $\delta_k > 0$ if we choose $\sigma_k \geq c_2 \frac{q\sqrt{T \log(1/\delta_k)}}{\epsilon_k}$.

Due to the clipping operation $\bar{g}_{kj} = g_{kj} \times \min(1, \frac{S}{\|g_{kj}\|_2})$, the sensitivity of f is upper bounded by S. Since the variance of our mechanism is σ_k^2 for user k, and we only select $\mathcal{J}$ examples with the probability $\frac{\mathcal{U}}{U}$ for T rounds, it suffices to have $\sigma_k \geq c_2 \frac{\mathcal{U}\sqrt{\mathcal{J} T \log(1/\delta_k)}}{U\epsilon_k}$, which satisfies (ϵ_k, δ_k)-differential privacy for user k. □

4. Evaluation

In this section, we evaluate our proposed methods and the DP-VAE based RS, to answer the following questions:

RQ1 How effective is our method of training dataset construction?
RQ2 How effective are our proposed user-level priors for training the DP-VAE model?
RQ3 How effective is the optimal clipping bound on gradient updates for the DP-VAE model?
RQ4 Can our proposed DP-VAE model outperform other standard schemes?
RQ5 How does the user-level privacy parameter (ϵ_k) affect the performance of the DP-VAE model?
RQ6 Can we apply our proposed DPSGD algorithm with the optimal clipping bound to other models?

4.1. Evaluation setup

4.1.1. Datasets

We evaluate our methods on three standard benchmark datasets for testing recommender systems: Movielens-100k (ML-100K) [43], Movielens-20 m (ML-20M) [44], Netflix Prize (Netflix) [45]. Table 2 gives details of these datasets. "OL" indicates the original version while "PD" presents the pre-processed one. As the original Netflix dataset is very large, we randomly extract around 70 million ratings. We set $\omega = 0.05$ and $\delta = 0.1$ with $p_k \in [0.001, 1]$ for all users. Table 2 summarizes details of the pre-processed datasets, where we omit users and items that do not constitute enough training examples satisfying $\frac{p_k}{\omega}(d + 1 + \log \frac{1}{\delta})$. We split each dataset and set the ratio of training, validation and test examples to 8:1:1.

The sparsity is measured as follows:

$$Sparsity = 1 - R \div (U \times I) \tag{21}$$

4.1.2. Metrics

We adopt *Recall@R* and the truncated normalized discounted cumulative gain (*NDCG@R*) to evaluate the performance of the recommender system. *Recall@R* compares the predicted rank of an item with the true rank by sorting R items. *NDCG@R* is the normalized *DCG@R* which ranges in $[0, 1]$. *DCG@R* adopts a monotonically increasing discount to show that higher ranks affect the performance of RS more than lower ranks. Let $i_k(r)$ be the recommended item i at rank r to user k. Q_k indicates all recommended items. *Recall@R*, *DCG@R* can be calculated as follows:

$$Recall@R(k) = \frac{\sum_i^R \mathbb{1}[i_k(r) \in Q_k]}{\min(Q_k, R)},$$
$$DCG@R(k) = \sum_i^R \frac{2^{\mathbb{1}[i_k(r)\in Q_k]} - 1}{\log_2(i+1)}. \tag{22}$$

For both metrics, the higher values imply the better performance of the recommender system. For example, considering user k with 3 recommended items, the best $DCG@3(k)$ is $DCG@3(k)_{Best} = \frac{1}{\log_2 2} + \frac{1}{\log_2 3} + \frac{1}{\log_2 4} = 2.13$. And the true $DCG@3(k)$ is $DCG@3(k) = \frac{1}{\log_2 2} + \frac{0}{\log_2 3} + \frac{1}{\log_2 4} = 1.5$ if $i_k(2)$ does not belong to Q_k. So the $NDCG@3(k) = \frac{1.5}{2.13} = 0.70$. In this case, $Recall@3(k) = \frac{2}{3} = 0.67$. In our final experimental results, we follow the VAE model in [23] and mainly select *NDCG@*100, *Recall@*20 and *Recall@*50 for model comparisons.

4.1.3. Default settings

For the user-level DP-VAE model, we set the privacy budget ϵ_k in $(0, 1]$, which simulates that each user can randomly choose the privacy guarantee level within their interests. Herein another privacy parameter δ_k is set to the inverse of the training data size as suggested by Dwork et al. [32]. For other DP models, we set the privacy budget to 1 if not specified. Table 3 summarizes these parameters for training the VAE model. Besides, we adopt the best β optimizing the DP-VAE model, with which both *NDCG@R* and *Recall@R* reach the peak.

4.1.4. Baselines

We compare our DP-VAE recommendation model with the following schemes:

(i) k-nearest neighbor (KNN) [19] recommendation: ; (ii) singular value decomposition (SVD) based recommendation, which is a standard method widely applied in RS design [20]; (iii) a hybrid matrix factorization model called LightFM [46], which combines both content-based and collaborative models; (iv) the non-DP VAE model in [23]; two state-of-the-art DP models (v) DPMF, a differentially private model using matrix factorization [13,34] and (vi) DPAE, a differentially private autoencoder model [14,15]. Note that we do not compare our model with the Bayesian-based approach [47], as the similar work [23] has verified that its performance cannot be on par with VAE-based methods. We further summarize all details of the model structures and parameters in Table 4.

4.1.5. Implementation

We use TensorFlow 2.0 [49] to train our models, on a server equipped with one Nvidia GTX 1080 GPU, an Intel Xeon E5-1620 CPU with 4 cores, and 32 GB memory. In our evaluation part, we repeat each experiment five times with different random seeds to verify the effectiveness of the proposed methods and then calculate the average Recall and NDCG values as the final results.

Table 2
Attributes of benchmark datasets.

Dataset		ML-100K	ML-20M	Netflix
# of users	OL	943	138,493	311,266
	PD	943	96,772	279,128
# of items	OL	1,682	26,744	17523
	PD	1,682	11,210	17,509
# of ratings	OL	100,000	20,000,263	70,613,395
	PD	100,000	18,723,768	69,511,359
Sparsity	OL	93.70%	99.46%	98.71%
	PD	93.70%	98.27%	98.58%

Table 3
Experimental settings for training the VAE model.

Dataset	Batch Size	Epoch	β	Latent Dimension	# of parameters
ML-100K	10	30	0.5	20	275,802
ML-20M	20	150	0.2	30	5,384,804
Netflix	20	250	0.1	50	3,587,070

Table 4
Parameters and network structures for compared models.

Model	Model Structure	Optimizer	Parameters Settings[a]
KNN [19]	The K-nearest neighbor (KNN) algorithm, which classifies each input based on its closest Top-K neighbors	Not applicable	Compute the Euclidean distance between input tuples and select the optimal K in [1, 40] by retraining the KNN model
SVD [20]	The Singular value decomposition (SVD) algorithm, which is a MF technique by producing low-rank approximations	SGD	Select the lower-dimensional latent factor (rank) to 100, as a higher latent typically brings in better prediction results
LightFM [46]	The hybrid content-collaborative model, which is close to the factorization machine (FM) algorithm	SGD	Implement the Weighted Approximate-Rank Pairwise (WARP) loss that outperforms the Bayesian Personalized Ranking (BPR) loss [48]
Non-DP VAE [23]	The variational autoencoder (VAE) model as shown in Fig. 1	SGD	Adopt the same parameters settings as the DPVAE model except for all the privacy parameters
DPMF [13,34]	The matrix factorization (MF) technique, which can decompose the item-based matrix and the user profile matrix	SGD	Apply DP into the decomposed user profile matrix, privacy parameters $\epsilon = 1$, same δ as the DPVAE
DPAE [14,15]	The autoencoder and the latent dimension is set to 20, 30, 50 separately for ML-100K, ML-20M and Netflix	DPSGD	Truncate, aggregate and update gradients based on the DPSGD [33], privacy parameters $\epsilon = 1$, same δ as the DPVAE

[a] Note that the we set the learning rate to 0.005 and regulation term to 0.02 if not specified.

Table 5
Recommendation model performance comparison on original/pre-processed datasets: "OL" indicates the original dataset, "PD" indicates the pre-processed dataset, "PP" indicates the improvement percentage point.

Metrics	NDCG@100			Recall@20			Recall@50		
ML-20M	OL	PD	PP(%)	OL	PD	PP (%)	OL	PD	PP(%)
DP-VAE	0.24	**0.28**	**16.67**	0.45	**0.47**	4.44	0.42	**0.43**	2.38
KNN	0.21	**0.26**	**23.81**	0.36	**0.38**	5.56	0.39	**0.40**	2.56
SVD	0.22	**0.27**	**22.73**	0.40	0.40	–	0.41	**0.42**	2.44
LightFM	0.15	**0.16**	6.67	0.13	**0.14**	7.69	0.22	**0.24**	**9.09**
VAE	0.33	0.33	–	0.51	**0.53**	3.92	0.49	**0.50**	2.04
DPMF	0.19	**0.21**	**10.53**	0.31	**0.32**	3.23	0.32	**0.34**	6.25
DPAE	0.22	0.22	–	0.32	**0.35**	**9.38**	0.33	**0.36**	**9.09**
Netflix	OL	PD	PP(%)	OL	PD	PP(%)	OL	PD	PP(%)
DP-VAE	0.50	**0.52**	4	0.62	**0.64**	3.23	0.58	**0.60**	3.45
KNN	0.48	**0.49**	2.08	0.14	**0.16**	**14.29**	0.16	**0.17**	6.25
SVD	0.50	0.50	–	0.14	**0.17**	**21.43**	0.18	**0.21**	**16.67**
LightFM	0.16	**0.17**	6.25	0.12	**0.16**	**33.33**	0.17	**0.19**	**11.76**
VAE	0.52	**0.54**	3.85	0.65	**0.66**	1.54	0.60	**0.61**	1.67
DPMF	0.35	0.35	–	0.13	**0.15**	**15.38**	0.16	**0.18**	**12.5**
DPAE	0.46	**0.47**	2.17	0.35	**0.37**	5.71	0.39	**0.42**	7.69

4.2. Evaluation results

4.2.1. Effectiveness of training dataset construction (RQ1)

We first compare the performance of the recommendation model trained using the original dataset and pre-processed dataset, respectively. As the computed bound on the number of examples contributed by each user on ML-100k is [3, 1772], all samples are preserved after pre-processing. We hence only use the other two datasets. We use a constant clipping bound $S = 2$ based on prior studies [18]. In Table 5, "OL" indicates the original dataset, and "PD" refers to the processed dataset, "PP" indicates the improvement percentage point. The results in Table 5 show that our method of constructing training datasets can improve the RS performance significantly for all recommendation models.

4.2.2. Effectiveness of the user-level priors (RQ2)

We further conduct experiments on three processed datasets to test the effectiveness of the user-level priors. Similar to prior settings [18], we adopt a constant clipping bound $S = 1.5$ for ML-100k and $S = 2$ for ML-20M and Netflix, in our DP-VAE model. Besides, we set a randomly chosen privacy budget ϵ_k within (0, 1] for user k. In Table 6, "ULP" denotes the VAE-based model with user-level priors, and "NUP" represents the original VAE-based

Table 6
Effectiveness of using the user-level priors: 'ULP" denotes the VAE-based model with user-level priors, and "NUP" represents the original VAE-based model, "PP" indicates the improvement percentage point.

Metrics	NDCG@100			Recall@20			Recall@50		
DP-VAE	NUP	ULP	PP(%)	NUP	ULP	PP(%)	NUP	ULP	PP(%)
ML-100K	0.40	**0.42**	5	0.52	**0.53**	1.92	0.50	**0.52**	4
ML-20M	0.28	0.28	–	0.47	**0.48**	2.12	0.43	**0.44**	2.32
Netflix	0.52	**0.53**	1.92	0.64	**0.65**	1.56	0.60	**0.62**	3.3
VAE	NUP	ULP	PP(%)	NUP	ULP	PP(%)	NUP	ULP	PP(%)
ML-100K	0.42	**0.44**	4.76	0.61	**0.62**	1.64	0.60	0.60	–
ML-20M	0.33	**0.34**	3.03	0.53	**0.55**	3.77	0.50	**0.52**	4
Netflix	0.54	0.54	–	0.66	**0.68**	3.03	0.61	**0.62**	1.64

Table 7
Effectiveness of using the optimal clipping bound: "CCB" indicates using a constant clipping bound and "OCB" refers to using the optimal clipping bound, "PP" indicates the improvement percentage point.

Metrics	NDCG@100			Recall@20			Recall@50		
DP-VAE	CCB	OCB	PP(%)	CCB	OCB	PP(%)	CCB	OCB	PP(%)
ML-100K	0.40	**0.41**	2.44	0.52	**0.53**	1.92	0.50	**0.52**	4
ML-20M	0.28	**0.29**	3.57	0.47	0.47	–	0.43	**0.45**	4.65
Netflix	0.52	**0.53**	1.92	0.64	**0.66**	3.13	0.60	**0.61**	1.67

Table 8
Comparison Results on ML-100K.

MODEL	DP-VAE	KNN	SVD	LightFM	VAE	DPMF	DPAE
NDCG@100	**0.41**	0.26	0.26	0.25	0.44	0.21	0.26
Recall@20	**0.53**	0.35	0.34	0.18	0.62	0.17	0.36
Recall@50	**0.52**	0.36	0.35	0.28	0.60	0.21	0.37

Table 9
Performance of the optimized DP-VAE model: "NV" indicates the original DP-VAE model and "OV" refers to the optimized version that apply both the optimal clipping bound and user-level priors, "PP" indicates the improvement percentage point.

Metrics	NDCG@100			Recall@20			Recall@50		
DP-VAE	NV	OV	PP(%)	NV	OV	PP(%)	NV	OV	PP(%)
ML-20M	0.28	**0.30**	7.14	0.47	**0.48**	2.13	0.43	**0.45**	4.65
Netflix	0.52	**0.54**	3.85	0.64	**0.68**	6.25	0.60	**0.62**	3.33

model. The experimental results show that adopting the user-level priors can improve the recommendation performance of VAE-based models as we calculate the user-level priors from the user metadata that can indicate user preferences. Especially, for the DP-VAE model on ML-100K, we achieve a 5% improvement for *NDCG*@100 and a 4% improvement for *Recall*@50, indicating that using the metadata information can generate a larger latent VAE space that improves the recommendation performance.

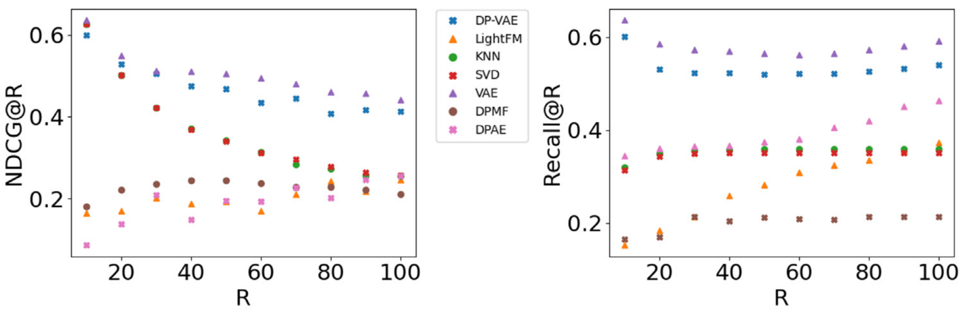
4.2.3. *Effectiveness of the optimal clipping bound (RQ3)*

We next evaluate the effect of using the optimal clipping bound in the DPSGD training process of our DP-VAE model. In Table 7, "CCB" indicates using a constant clipping bound and "OCB" refers to using the optimal clipping bound. Based on prior work [18], we set a constant clipping bound $S = 1.5$ for ML-100k and $S = 2$ for both ML-20M and Netflix. Note that we do not apply the user-level priors here so that each latent vector z_k is sampled from the standard Gaussian distribution. The results in Table 7 show that with the optimal clipping bound, the performance of the DP-VAE model is improved on all datasets, where an optimal clipping bound assures that the most effective updates are preserved, and large gradient updates cannot control the trained model.

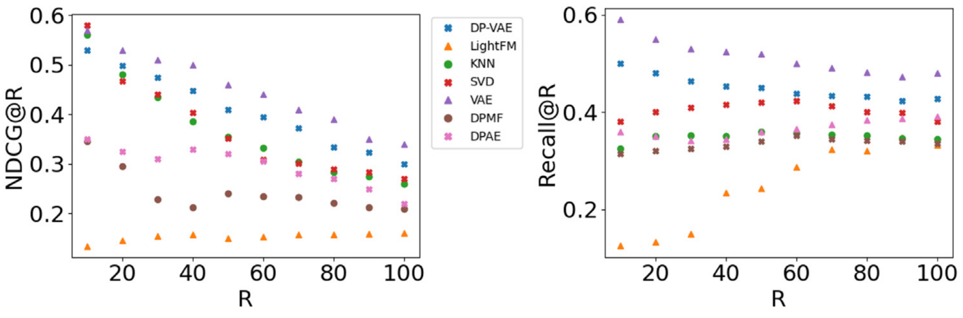
4.2.4. *Performance comparison with baselines (RQ4)*

Here we apply both user-level priors and optimal clipping bound for our DP-VAE model. We train the recommendation models on ML-100K, and the results are given in Table 8. Comparing the performance of DP-VAE and VAE, we observe that there is only about 6.8% loss of *NDCG*@100. Though the Recall values are around 16% lower, our proposed DP-VAE model can still perform better than other standard models. We note that the performance loss is unavoidable as the clipping operation and noise addition in DPSGD may discard some valuable features. As shown in Fig. 3(a), we also vary R values and obtain the experimental results on ML-100k, which show that DP-VAE still outperforms other baselines compared to *Recall@R*. For *NDCG@R*, although KNN and SVD perform slightly better than DP-VAE for *NDCG*@10 (which are still worse than VAE), their performance degrades sharply when increasing R.

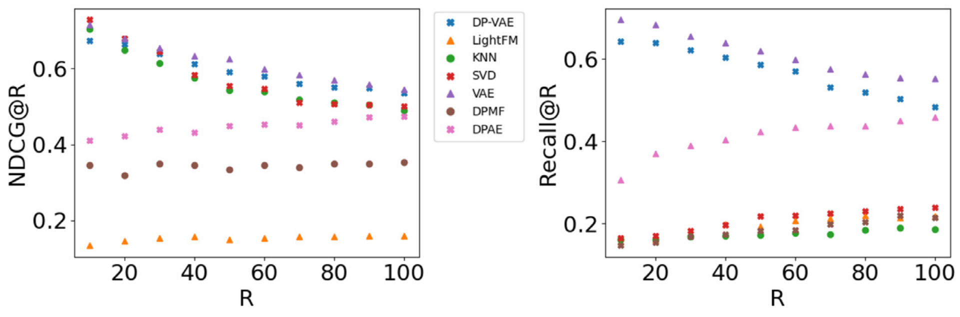
We further test our DP-VAE model on the pre-processed ML-20M and Netflix datasets. The evaluation results are given in Table 9. Note that "NV" indicates the original DP-VAE model and "OV" refers to the optimized version that applies the optimal clipping bound and user-level priors. Compared with other baseline models (except for the VAE-based model) in Table 5, our proposed DP-VAE model achieves better performance in terms of *NDCG*@100. For both *Recall*@20 and *Recall*@50, our proposed DP-VAE model also outperforms other standard baseline models. We further evaluate the performances of all models on ML-20M and Netflix by varying the R values. As shown in Fig. 3(b) and Fig. 3(c), we observe that DP-VAE and VAE based methods still outperform other baselines compared to *Recall@R*. Besides, except for the VAE-based model, our proposed DP-VAE model can outperform other baselines in terms of *NDCG@R* with a bigger $R \in [20, 100]$.



(a) NDCG and Recall vs. R on ML-100k



(b) NDCG and Recall vs. R on ML-20M



(c) NDCG and Recall vs. R on Netflix

Fig. 3. NDCG and Recall vs. R.

Table 10
Experimental results by varying the privacy budgets.

Model	VAE	DP-VAE (0.3)	DP-VAE (0.5)	DP-VAE (1)	DP-VAE (1.5)
NDCG@100	0.44	0.28	0.38	0.41	0.42
Recall@20	0.62	0.20	0.49	0.53	0.55
Recall@50	0.60	0.22	0.48	0.52	0.54

4.2.5. Trade-off between the privacy guarantee and model performance (RQ5)

We further set the highest privacy budget to 0.3, 0.5, 1.0, and 1.5 separately. For user-level differential privacy, each user can randomly choose the privacy budget no bigger than 0.3, 0.5, 1.0, and 1.5 accordingly. We test the trade-off between privacy guarantee and the model performance on ML-100k and obtain the results shown in Table 10. The experimental results verify that our solutions can achieve comparable performance under a higher privacy guarantee ($\epsilon_k \leq 0.5$). When we reduce the highest privacy budget to 0.5, the DP-VAE model is still much better than the baselines except for the VAE model, comparing with the performances of baselines in Table 8. However, if we set the highest privacy budget to 0.3 with a strict privacy guarantee, the performance of the DP-VAE model degrades shapely considering Recall@20 and Recall@50. So we recommend setting the privacy budget ϵ_k to 0.5 when training the DP-VAE model, which achieves the comparable performance with other baselines and provides an appropriate privacy guarantee.

4.2.6. Generalization of the DPSGD algorithm with the optimal clipping bound (RQ6)

To test the generalization ability of the DPSGD algorithm with the optimal clipping bound, we select three models widely applied to recommendations with the SGD optimizer: SVD, MF and the autoencoder model (AE). We set all model parameters and structures the same as described in Table 4. To verify the generalization of our DP strategy, we compare our method with the general DPSGD algorithm [33] on these models and set the privacy budget to 1 for a fair comparison. We evaluate all models on ML-100K and summarize all results in Table 11: “OL” refers to the original model, “DP” indicates the model with the general DPSGD algorithm [33], “DP-O” represents the model with our

Table 11
Generalization performance of our DPSGD algorithm with the optimal clipping bound: "OL" refers to the original model without differential privacy, "DP" indicates the model with the DPSGD algorithm [33], "DP-O" represents the model with our DP strategy.

Models	SVD			MF			AE		
Versions	OL	DP	DP-O	OL	DP	DP-O	OL	DP	DP-O
NDCG@100	0.26	0.22	**0.24**	0.24	0.20	**0.22**	0.33	0.26	**0.28**
Recall@20	0.34	0.29	**0.32**	0.21	0.18	**0.19**	0.42	0.36	**0.38**
Recall@50	0.35	0.31	**0.33**	0.25	0.21	**0.23**	0.45	0.37	**0.41**

DP strategy. Considering *NDCG*@100, *Recall*@20, and *Recall*@50, the "DP-O" version outperforms the "DP" one for each selective model, which verifies that our DP strategy can be effective in other models as well.

5. Discussions

5.1. Differential privacy

In this paper, we investigate the application of differential privacy to variational autoencoders for recommendation tasks to provide privacy guarantees from the user-level perspectives. A recommender system analyzes taste similarities of users, but prior work [7] pointed out that it can be overly sensitive to the input of individual users. Narayanan et al. [50] analyzed the privacy leakage issue and demonstrated that analyzing users' historical data can disclose a large amount of sensitive information such as users' hobbies and health recordings, etc. For example, movie recommendations from a service provider (such as Netflix) are based on users' general information and viewing histories. Then an attacker may resemble a target user by creating a fictitious profile and obtaining the movie recommendations within the user's interests. So the attacker could infer that a new recommendation might be included in the target user's recommendation list. So it is significant and pivotal to provide privacy protection mechanisms for a recommender system.

Differential privacy conquers the privacy leakage issue as mentioned above and provides strong privacy protection, even when an attacker may access a substantial amount of personal information about the target user [32]. Herein, we point out that differential privacy is a property that results from the computation process (by adding noise) but not from the output itself. In other words, the computation process is differentially private when the model with similar entries produces the same output. Like the above instance for movie recommendations, a differentially private recommender system assures that the attacker cannot correctly infer whether the recommended movie is in the recommendation list of the target user. However, differential privacy provides this property by adding noise (scaled by the privacy parameters) into the computation process, which degrades the model performance. To be specific, there is a trade-off between the model performance and the privacy protection level based on the privacy budget ϵ_k: higher ϵ_k typically indicates lower privacy protection (versus poorer model performance), while a lower ϵ_k usually presents higher privacy protection (versus better model performance).

5.2. Novelties of our work and applications in other models

In this part, we further clarify the novelties of our work and discuss potential applications of our proposed methods in other algorithms/models from three perspectives.

First, our proposed dataset construction method can be an intuitive guideline for other works when training their RS models. Especially, our proposed benchmark bound $\frac{p_k}{\omega}(d+1+\log\frac{1}{\delta})$, which is adopted to handle the uneven contribution problem [28] among users, has been proven to improve the model performance based on both theoretical analysis and experimental results.

Second, we adopt differential privacy when metadata is adopted to provide better model performance [36], where we inject Gaussian noise into the intermediate latent vectors. As word embeddings [42] are widely employed in existing recommender systems [36,51], our proposed method can be applied to the calculation process of generating the word representations, which can provide privacy protection for those models.

Finally, we study the existing DPSGD [33] and convert the process of computing the optimal clipping bound to a simple convex problem. Our proposed clipping bound can achieve a better trade-off between the model performance and privacy guarantee compared with the original clipping bound. As numerous RS models adopt SGD or DPSGD as the optimizer, e.g., machine learning models [13,19,20], deep learning-based methods [52], the emerging graph neural network models [53–55], we believe our proposed DPSGD algorithm with the optimal clipping bound can be effective to avoid severe privacy leakage and maintain good model performance.

5.3. Limitations

Although prior works [56,57] analyzed the contribution size in their experimental parts, we investigate and propose the contribution size of each user from the theoretical perspective when building the datasets for an RS model. However, a tighter bound should be considered from the trade-off between the model performance and dataset size based on the current research findings [29,41]. Although the evaluation performance verifies the effectiveness of the proposed benchmark threshold, we still need more profound theoretical research and more experimental results. Especially for the uneven contribution problem among users, we mainly handle the disparate effect that causes severe model degradation. However, investigating the "fairness" problem [58] could be more critical in designing a fair RS, where the RS model tends to provide better performance for users with higher contributions. Besides, we mainly apply our methods to the VAE model and test the DP strategy on some baseline models. Although we discuss the potential applications on other RS algorithms or models, more extensive designs on SOTA models should be conducted to verify the generalization of proposed methods, as our future research directions.

6. Conclusion and future work

This paper presents a novel DP-VAE model for recommender systems. We first present an efficient way of deciding the number of training examples from each user to ensure that the dataset is not controlled by users with large contributions. We then compute the user-level priors with differential privacy to optimize the VAE-based model and protect user metadata. We next present an optimal clipping bound for DPSGD with user-level differential privacy. We conduct experiments on various benchmark datasets. The results verify that our proposed solutions can achieve high recommendation performance while maintaining an appropriate privacy level. As future work, we plan to explore differential privacy in emerging neural network methods, e. g. , in GNN-based RS [53–55,59,60], aiming to provide higher privacy protection and better recommendation precision.

CRediT authorship contribution statement

Le Fang: Methodology, Writing – original draft, Validation, Investigation. **Bingqian Du:** Writing – review & editing. **Chuan Wu:** Conceptualization, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by grants from Hong Kong RGC under the contracts 17204619, 17208920, 17207621, and C5026-18G (CRF). All Authors agreed the version of manuscript to be published.

References

[1] M. del Carmen Rodríguez-Hernández, S. Ilarri, AI-Based mobile context-aware recommender systems from an information management perspective: Progress and directions, Knowl.-Based Syst. 215 (2021) 106740.
[2] Z. Erkin, T. Veugen, T. Toft, R.L. Lagendijk, Generating private recommendations efficiently using homomorphic encryption and data packing, IEEE Trans. Inf. Forensics Secur. 7 (3) (2012) 1053–1066.
[3] T. Yu, Y. Shen, H. Jin, A visual dialog augmented interactive recommender system, in: Proceedings of ACM SIGKDD, Association for Computing Machinery, New York, NY, USA, 2019, pp. 157–165.
[4] Y. Wang, L. Wang, L. Zhao, X. Ran, S. Deng, Privacy recommendation based on Bhattacharyya coefficient, Procedia Comput. Sci. 188 (2021) 61–68.
[5] C. Wang, M. Zhang, W. Ma, Y. Liu, S. Ma, Make it a chorus: Knowledge- and time-aware item modeling for sequential recommendation, in: Proceedings of ACM SIGIR, Association for Computing Machinery, New York, NY, USA, 2020, pp. 109–118.
[6] V. Venkatesh, H. Hoehle, J.A. Aloysius, H.R. Nikkhah, Being at the cutting edge of online shopping: Role of recommendations and discounts on privacy perceptions, Comput. Hum. Behav. 121 (2021) 106785.
[7] F. McSherry, I. Mironov, Differentially private recommender systems: Building privacy into the netflix prize contenders, in: Proceedings of ACM SIGKDD, Association for Computing Machinery, New York, NY, USA, 2009, pp. 627–636.
[8] B. Smith, G. Linden, Two decades of recommender systems at amazon.com, IEEE Internet Comput. 21 (3) (2017) 12–18.
[9] H. Yang, BayesIan heteroscedastic matrix factorization for conversion rate prediction, in: Proceedings of ACM CIKM, Association for Computing Machinery, New York, NY, USA, 2017, pp. 2407–2410.
[10] Y. Sun, Y. Zhang, Conversational recommender system, in: Proceedings of ACM SIGIR, Association for Computing Machinery, New York, NY, USA, 2018, pp. 235–244.
[11] E. Aimeur, G. Brassard, J. Fernandez, F. Onana, ALAMBIC: a privacy-preserving recommender system for electronic commerce, Int. J. Inf. Secur. 7 (2008) 307–334.
[12] A. Friedman, B.P. Knijnenburg, K. Vanhecke, L. Martens, S. Berkovsky, Privacy aspects of recommender systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer, Boston, MA, 2015, pp. 649–688.
[13] J. Hua, C. Xia, S. Zhong, Differentially private matrix factorization, in: Proceedings of IJCAI, Morgan Kaufmann, Burlington, MA, USA, 2015, pp. 1763–1770.
[14] X. Liu, Q. Li, Z. Ni, J. Hou, Differentially private recommender system with autoencoders, in: Proceedings of IEEE IThings, IEEE GreenCom, IEEE CPSCom and IEEE SmartData, IEEE, New York, NY, USA, 2019, pp. 450–457.
[15] J. Ren, X. Xu, Z. Yao, H. Yu, Recommender systems based on autoencoder and differential privacy, in: Proceedings of IEEE COMPSAC, IEEE, New York, NY, USA, 2019, pp. 358–363.
[16] Y. Huo, B. Chen, J. Tang, Y. Zeng, Privacy-preserving point-of-interest recommendation based on geographical and social influence, Inform. Sci. 543 (2021) 202–218.
[17] S. Beg, A. Anjum, M. Ahmad, S. Hussain, G. Ahmad, S. Khan, K.-K.R. Choo, A privacy-preserving protocol for continuous and dynamic data collection in IoT enabled mobile app recommendation system (MARS), J. Netw. Comput. Appl. 174 (2021) 102874.
[18] E. Bagdasaryan, V. Shmatikov, Differential privacy has disparate impact on model accuracy, in: Proceedings of NeurIPS, MIT Press, Cambridge, MA, USA, 2019.
[19] D.A. Adeniyi, Z. Wei, Y. Yang, Automated web usage data mining and recommendation system using K-nearest neighbor (KNN) classification method, Appl. Comput. Inf. 12 (1) (2016) 90–108.
[20] N. Hug, Surprise: A Python library for recommender systems, J. Open Source Softw. 5 (52) (2020) 2174.
[21] T. Chen, H. Yin, G. Ye, Z. Huang, Y. Wang, M. Wang, Try this instead: Personalized and interpretable substitute recommendation, in: Proceedings of ACM SIGIR, Association for Computing Machinery, New York, NY, USA, 2020, pp. 891–900.
[22] D. Kingma, M. Welling, Auto-encoding variational Bayes, in: Proceedings of ICLR, WASET, Turkey, 2014.
[23] D. Liang, R. Krishnan, M. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of WWW, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2018, pp. 689–698.
[24] J. Ma, C. Zhou, P. Cui, H. Yang, W. Zhu, Learning disentangled representations for recommendation, in: Proceedings of NeurIPS, MIT Press, Cambridge, MA, USA, 2019.
[25] K. Luo, H. Yang, G. Wu, S. Sanner, Deep critiquing for VAE-based recommender systems, in: Proceedings of ACM SIGIR, Association for Computing Machinery, New York, NY, USA, 2020, pp. 1269–1278.
[26] J. Feng, Z. Xia, X. Feng, J. Peng, RBPR: A hybrid model for the new user cold start problem in recommender systems, Knowl.-Based Syst. 214 (2021) 106732.
[27] C.A. Gomez-Uribe, N. Hunt, The netflix recommender system: Algorithms, business value, and innovation, ACM Trans. Manag. Inf. Syst. 6 (4) (2016) 1–19.
[28] K. Amin, A. Kulesza, A.M. Medina, S. Vassilvitskii, Bounding user contributions: A bias-variance trade-off in differential privacy, in: Proceedings of ICML, PMLR, USA, 2019, pp. 263–271.
[29] B. Juba, H. Le, Precision-recall versus accuracy and the role of large data sets, in: Proceedings of AAAI, AAAI Press, California, USA, 2019, pp. 4039–4048.
[30] B. McMahan, D. Ramage, K. Talwar, L. Zhang, Learning differentially private recurrent language models, in: Proceedings of ICLR, WASET, Turkey, 2018.
[31] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: Proceedings of the Theory of Cryptography Conference, Springer, Heidelberg, Germany, 2006, pp. 265–284.
[32] C. Dwork, A. Roth, The algorithmic foundations of differential privacy, Found. Trends Theor. Comput. Sci. 9 (2014) 211–407.
[33] M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in: Proceedings of ACM CCS, 2016.
[34] Z. Liu, Y. Wang, A. Smola, Fast differentially private matrix factorization, in: Proceedings of ACM RecSys, Association for Computing Machinery, New York, NY, USA, 2015, pp. 171–178.
[35] X. Li, J. She, Collaborative variational autoencoder for recommender systems, in: Proceedings of ACM SIGKDD, Association for Computing Machinery, New York, NY, USA, 2017, pp. 305–314.
[36] G. Karamanolakis, K.R. Cherian, A.R. Narayan, J. Yuan, D. Tang, T. Jebara, Item recommendation with variational autoencoders and heterogeneous priors, in: Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems, Association for Computing Machinery, New York, NY, USA, 2018, pp. 10–14.
[37] M. Jordan, Z. Ghahramani, T. Jaakkola, L. Saul, An introduction to variational methods for graphical models, Mach. Learn. 37 (1999) 183–233.
[38] S. Gershman, N. Goodman, Amortized inference in probabilistic reasoning, in: Proceedings of the Annual Meeting of the Cognitive Science Society, Cognitive Science Society, Austin, USA, 2014, pp. 517–522.
[39] T. van Erven, P. Harremos, Rényi divergence and Kullback-Leibler divergence, IEEE Trans. Inform. Theory 60 (7) (2014) 3797–3820.
[40] D. Blei, A. Ng, M. Jordan, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.
[41] S. Hanneke, The optimal sample complexity of PAC learning, J. Mach. Learn. Res. 17 (1) (2016) 1319–1333.
[42] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of EMNLP, Association for Computational Linguistics, Stroudsburg PA, USA, 2014, pp. 1532–1543.
[43] GroupLens Research, MovieLens 100k dataset, 1998, https://grouplens.org/datasets/movielens/100k/.
[44] GroupLens Research, MovieLens 20 m dataset, 2016, https://grouplens.org/datasets/movielens/20m/.
[45] Netflix, Netflix prize, 2009, https://www.netflixprize.com/.
[46] M. Kula, Metadata embeddings for user and item cold-start recommendations, in: Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems, Association for Computing Machinery, New York, NY, USA, 2015, pp. 89–96.

[47] T. Kulkarni, J. Jälkö, A. Koskela, S. Kaski, A. Honkela, Differentially private Bayesian inference for generalized linear models, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 139, PMLR, 2021, pp. 5838–5849.
[48] J. Weston, S. Bengio, N. Usunier, WSABIE: Scaling up to large vocabulary image annotation, in: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three, in: IJCAI'11, AAAI Press, 2011.
[49] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, TensorFlow: A system for large-scale machine learning, in: Proceedings of USENIX OSDI, USENIX Association, Savannah, GA, 2016, pp. 265–283.
[50] A. Narayanan, V. Shmatikov, Robust de-anonymization of large sparse datasets, in: 2008 IEEE Symposium on Security and Privacy (Sp 2008), 2008, pp. 111–125.
[51] C. Cui, M. Hu, J.D. Weir, T. Wu, A recommendation system for meta-modeling: A meta-learning based approach, Expert Syst. Appl. 46 (2016) 33–44.
[52] S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM Comput. Surv. 52 (1–38)) (2019).
[53] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, D. Yin, Graph neural networks for social recommendation, in: Proceedings of WWW, Association for Computing Machinery, New York, NY, USA, 2019, pp. 417–426.
[54] R. Qiu, H. Yin, Z. Huang, T. Chen, GAG: Global attributed graph neural network for streaming session-based recommendation, in: Proceedings of ACM SIGIR, Association for Computing Machinery, New York, NY, USA, 2020, pp. 669–678.
[55] S. Zhang, H. Yin, T. Chen, Q.V.N. Hung, Z. Huang, L. Cui, GCN-based user representation learning for unifying robust recommendation and fraudster detection, in: Proceedings of ACM SIGIR, Association for Computing Machinery, New York, NY, USA, 2020, pp. 689–698.
[56] A. Majumdar, A. Jain, Cold-start, warm-start and everything in between: An autoencoder based approach to recommendation, in: Proceedings of the International Joint Conference on Neural Networks, IJCNN, 2017.
[57] L. Galke, F. Mai, I. Vagliano, A. Scherp, Multi-modal adversarial autoencoders for recommendations of citations and subject labels, in: Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization, 2018.
[58] J. Leonhardt, A. Anand, M. Khosla, User fairness in recommender systems, in: Companion Proceedings of the the Web Conference 2018, in: WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2018.
[59] B. Chen, W. Guo, R. Tang, X. Xin, Y. Ding, X. He, D. Wang, Tgcn: Tag graph convolutional network for tag-aware recommendation, in: Proceedings of ACM CIKM, Association for Computing Machinery, New York, NY, USA, 2020, pp. 155–164.
[60] Z. Pan, F. Cai, W. Chen, H. Chen, M. de Rijke, Star graph neural networks for session-based recommendation, in: Proceedings of ACM CIKM, Association for Computing Machinery, New York, NY, USA, 2020, pp. 1195–1204.