


Article

DPMF: Decentralized Probabilistic Matrix Factorization for Privacy-Preserving Recommendation

Xu Yang [†] , Yuchuan Luo ^{*,†}, Shaojing Fu ^{*}, Ming Xu and Yingwen Chen

College of Computer, National University of Defense Technology, Changsha 410073, China

* Correspondence: luoyuchuan09@nudt.edu.cn (Y.L.); fushaojing@nudt.edu.cn (S.F.)

† These authors contributed equally to this work.

Abstract: Collaborative filtering is a popular approach for building an efficient and scalable recommender system. However, it has not unleashed its full potential due to the following problems. (1) Serious privacy concerns: collaborative filtering relies on aggregated user data to make personalized predictions, which means that the centralized server can access and compromise user privacy. (2) Expensive resources required: conventional collaborative filtering techniques require a server with powerful computing capacity and large storage space, so that the server can train and maintain the model. (3) Considering only one form of user feedback: most existing works aim to model user preferences based on explicit feedback (e.g., ratings) or implicit feedback (e.g., purchase history, viewing history) due to their heterogeneous representation; however, these two forms of feedback are abundant in most collaborative filtering applications, can both affect the model, and very few works studied the simultaneous use thereof. To solve the above problems, in this study we focus on implementing decentralized probabilistic matrix factorization for privacy-preserving recommendations. First, we explore the existing collaborative filtering algorithms and propose a probabilistic matrix co-factorization model. By integrating explicit and implicit feedback into a shared probabilistic model, the model can cope with the heterogeneity between these two forms of feedback. Further, we devise a decentralized learning method that allows users to keep their private data on the end devices. A novel decomposing strategy is proposed for users to exchange only non-private information, in which stochastic gradient descent is used for updating the models. Complexity analysis proves that our method is highly efficient with linear computation and communication complexity. Experiments conducted on two real-world datasets *FilmTrust* and *Epinions* show that our model gains a guarantee of convergence as the RMSE decreases quickly within 100 rounds of iterations. Compared with the state-of-the-art models, our model achieves lower model loss in rating prediction task and higher precision in item recommendation task.

Keywords: privacy-preserving; probabilistic matrix factorization; recommendation; decentralized learning



Citation: Yang, X.; Luo, Y.; Fu, S.; Xu M.; Chen Y. DPMF: Decentralized Probabilistic Matrix Factorization for Privacy-Preserving Recommendation. *Appl. Sci.* **2022**, *12*, 11118. <https://doi.org/10.3390/app122111118>

Academic Editor: Gianluca Lax

Received: 21 September 2022

Accepted: 31 October 2022

Published: 2 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the explosive growth of information, it becomes more difficult for people to find what they are interested in. To address this problem, Recommender System (RS) [1] is proposed to provide users with personalized recommendations which we call items. Its key idea is to profile users and items and model the relation between them. As one of the most popular techniques for building RSs, Collaborative Filtering (CF) [2] is based on the past behavior of users such as their previous viewing history and rating records.

Challenges. In real-world applications, although CF has proven to be effective and scalable in predicting user preferences, it still suffers from some problems.

(1) Conventional CF models have possible risks to privacy.

Most service providers in CF tend to collect users' historical behaviors to train the recommendation model, which might jeopardize user privacy since the plaintext infor-

mation is exposed to the service provider. Thus, some data privacy regulations, such as General Data Protection Regulation (GDPR) [3], have been published. These regulations attempt to place restrictions on the collection, storage, and use of user data. To tackle this privacy problem, federated learning (FL) [4] is proposed to legally and efficiently make use of users' data. FL distributes the model learning process to users' end devices, making it possible to train a global model from user-specific local models, which ensures that user's private data never leaves their end devices. However, CF under federated learning still has privacy issues as it is susceptible to some attacks. For example, Mothukuri et al. [5] prove the practicality in conducting backdoor attacks in federated learning. Zhang et al. [6] study and evaluate poisoning attack in federated learning system based on generative adversarial nets (GAN). Recently, decentralized learning (DL) [7] has drawn people's attention in many real-world applications. As shown in Figure 1, in decentralized learning the model is learned collaboratively by a group of users without needing a centralized server. However, malicious participants might steal privacy from other users in the communication phase [8]. Thus, we propose a decomposing strategy in our decentralized scheme to ensure that users exchanging only non-private information with each other, which shows practicality in preserving user privacy.

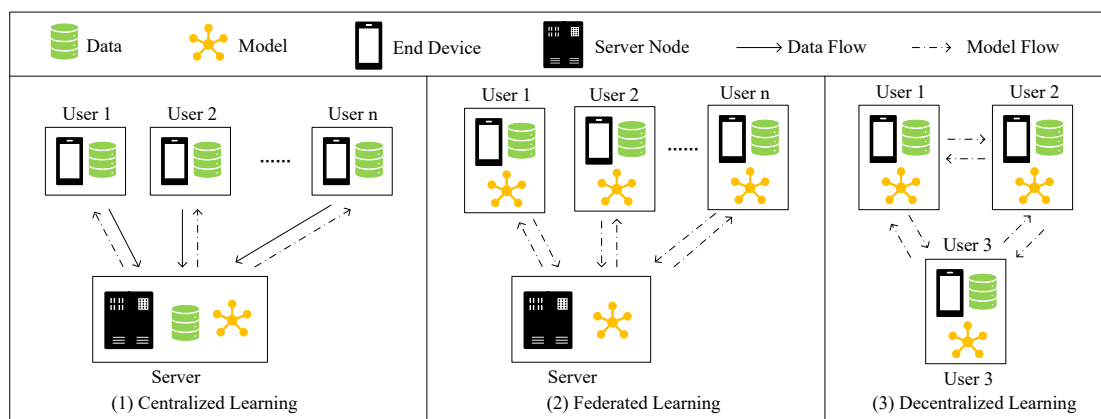


Figure 1. The comparison between centralized learning, federated learning, and decentralized learning.

(2) Conventional CF models might be limited by high resource requirements.

Except for privacy concerns, Kairouz et al. [9] point out that in order to maintain the user data and train the CF model, centralized learning needs a server with high storage and computing capacity. Although the server in federated learning is only responsible for aggregating and distributing the model, it still has to store a large amount of model information and coordinate communication with various users. This might become a bottleneck when the learning scale becomes large and further lead to single point of failure in practice [10]. Thus, in this study, we focus on implementing a decentralized learning scheme for building a CF model. As shown in Figure 1, by performing local training and exchanging some non-private information with neighbors, users collaboratively learn a global model. In this way, the storage and computing load are transferred to users which improves the scalability and stability of the system.

Most CF models cannot learn from both explicit and implicit feedback simultaneously.

CF aims to model user preference based on user feedback, which generally has two categories: explicit feedback and implicit feedback [11]. Explicit feedback is often the form of numeric ratings given by users to express the extent to which they like the items. It could measure user preference in a direct and granular way, but some users are reluctant to have such extra operations [12]. In contrast, implicit feedback is easier to be collected. It includes users' behaviors that indirectly reflect their opinion (e.g., browse history, clicking record). However, compared with explicit feedback, it has lower accuracy and granularity. For example, a woman buys a skirt online and finds out she dislikes it after wearing it. Through the above analysis, it is clear that these two forms of feedback are complementary

to each other since explicit feedback is higher in quality while implicit feedback is higher in quantity. As far as we know, few CF models are based both on explicit and implicit feedback. Thus, in this study we devise a matrix co-factorization model to cope with the heterogeneity between these two forms of feedback.

Our contributions. In this study, we first explore a probabilistic model specifically suitable for handling both explicit and implicit feedback and then devise a decentralized method called DPMF to protect users' sensitive information. To the best of our knowledge, this is the first privacy-preserving framework for recommendation with both explicit and implicit feedback. The main contributions are listed as follows.

- We devise a novel probabilistic matrix factorization method for recommender systems. It uses both explicit and implicit feedback simultaneously to model user preferences and item characteristics, which is practical and interpretable in rating prediction and item recommendation.
- We propose a novel decomposing strategy to decompose the shared information among users into two parts, and only share the non-private part. In this way, the model not only gains a guarantee of convergence by exchanging the public information, but also maintains user privacy as the private information is kept locally by users.
- We propose a secure and efficient method to train our model. By finding neighbors from the trust statement, users exchange public model information with others. The public and personal model gradients are updated through stochastic gradient descent. Extensive experiments on two real-world datasets show that our method outperforms the existing state-of-the-art CF methods with lower RMSE loss in rating prediction task and higher precision in item recommendation task.

The rest of this study is organized as follows. We introduce the background in Section 2 and then discuss the preliminaries and the system model in Section 3. We conduct experiment and discuss the model performance in Section 4. Finally, we conclude this study in Section 5.

2. Background

In this section, we first introduce the probabilistic model that we have to use in Section 3, and then discuss the existing decentralized learning scheme.

2.1. Probabilistic Matrix Factorization

The basic hypothesis for CF is that users with similar past behavior tend to like similar items. Matrix factorization (MF), as one of the most famous techniques in CF, aims to embedding users and items into low-dimensional dense vector space. By computing the inner product of user and item latent factor vector, it can predict user i 's preference on item j :

$$\hat{R}_{ij} = U_i^T V_j, \quad (1)$$

where U_i^T and V_j represent the latent factor vector for user i and item j , respectively. \hat{R}_{ij} is the predicted rating of user i for item j .

Based on this assumption, probabilistic matrix factorization (PMF) model [13] is proposed which defines the conditional distribution over the observed ratings:

$$p(R|U, V, \sigma_R^2) = \prod_{i=1}^m \prod_{j=1}^n [\mathcal{N}(R_{i,j} | g(U_i^T V_j), \sigma_R^2)]^{I_{ij}^R}, \quad (2)$$

where $g(\cdot)$ defines the logistic function, with $\mathcal{N}(x|\mu, \sigma^2)$ representing the Gaussian distribution with its mean value μ and variance σ^2 . The function I_{ij}^R equals 1 if R_{ij} is available in the observed data and 0 otherwise. Some other notations are defined in Table 1. Furthermore, they also place zero-mean spherical Gaussian priors on user and movie feature vectors:

$$p(U|\sigma_U^2) = \prod_{i=1}^m \mathcal{N}(U_i|0, \sigma_U^2 I), \quad (3)$$

$$p(V|\sigma_V^2) = \prod_{j=1}^n \mathcal{N}(V_j|0, \sigma_V^2 I). \quad (4)$$

By introducing Bayesian method, it defines the posterior distribution over matrix U and V :

$$p(U, V|R, \sigma_R^2, \sigma_U^2, \sigma_V^2) \propto p(U|\sigma_U^2) \cdot p(V|\sigma_V^2) p(R|U, V, \sigma_R^2) \quad (5)$$

To obtain optimal parameter U and V , we maximize (5) and then have:

$$E = \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^n I_{ij}^R [R_{ij} - g(U_i^T V_j)]^2 + \frac{\lambda_U}{2} \|U\|_F^2 + \frac{\lambda_V}{2} \|V\|_F^2, \quad (6)$$

where $\|\cdot\|_F^2$ denotes the Frobenius norm of a matrix.

Based on this, Cai et al. [14] further propose constrained probabilistic matrix factorization which is called Constrained-PMF. It introduces an additional method of constraining user-specific feature vectors that has strong effect on infrequent users, which define the conditional distribution over the observed ratings as:

$$p(R|U, V, Q, \sigma_R^2) = \prod_{i=1}^m \prod_{j=1}^n [\mathcal{N}(R_{ij}|g([U_i + \frac{\sum_{k=1}^n I_{ik}^R Q_k}{\sum_{k=1}^n I_{ik}^R}] V_j), \sigma_R^2)]^{I_{ij}^R}, \quad (7)$$

where $U_i + (\sum_{k=1}^n I_{ik}^R Q_k) / \sum_{k=1}^n I_{ik}^R$ denotes the user latent feature vector with $I_{ik}^R Q_k$ capturing captures the effect of a user i having rated item k . Moreover, it also places spherical Gaussian prior on matrix Q as:

$$p(Q|\sigma_Q^2) = \prod_{k=1}^m \mathcal{N}(Q_k|0, \sigma_Q^2 I). \quad (8)$$

Table 1. Notation Table.

Notation	Definition
\mathcal{U}	user set
\mathcal{V}	item set
R	explicit feedback matrix
\hat{R}	predicted rating matrix
M	implicit feedback matrix
W	implicit feedback weighting matrix
U	user latent factor matrix
V	item rating latent factor matrix
Z	item selecting latent factor matrix
Q	user offset latent factor matrix
m	size of the user set
n	size of the item set
$g(\cdot)$	logistic function
I	identity matrix
I_{ij}^R	indicator that equals 1 when user i rates item j and 0 otherwise
\mathcal{L}	loss function
$\mathcal{N}^d(i)$	d closest neighbors of user i

Table 1. *Cont.*

Notation	Definition
$\lambda_U, \lambda_V, \lambda_Z, \lambda_Q, \lambda$	regularization parameters
θ	learning rate
T	numbers of training iterations
d	number of neighbors for a user
K	dimension of the latent factors

2.2. Probabilistic Model for Implicit Feedback

Conventional MF methods are tailored to the need for explicit feedback and cannot be directly used for implicit feedback as they have no numeric ratings to build the loss function. To solve this problem, Chen et al. [15] provide a PMF model using implicit feedback to implement item recommendation task. Specifically, M_{ik} denotes two types of implicit feedback including positive and negative feedback. Chen et al. assume M_{ik} is from a Bernoulli distribution that equals 1 when user i and item k have implicit interactions and equals 0 otherwise. The uniform weighting strategy [12] is adopted to sample negative implicit feedback. By calculating the product between the user latent factor vector and the item latent factor vector, the probability distribution of M_{ik} is obtained as:

$$\begin{aligned} p(M_{ik} = 1|U_i, Z_k) &= g(U_i^T Z_k), \\ p(M_{ik} = 0|U_i, Z_k) &= 1 - W_{ik}g(U_i^T Z_k), \end{aligned} \quad (9)$$

where W_{ik} denotes the weight of the negative samples. Similar to Equation (2), it defines the conditional distribution of M as:

$$p(M|U, Z) = \prod_{i=1}^m \prod_{k=1}^n p(M_{ik}|U_i, Z_k) = \prod_{(i,k) \in P} g(U_i^T Z_k) \cdot \prod_{(i,k) \in N} (1 - W_{ik}g(U_i^T Z_k)) \quad (10)$$

where $(i, k) \in P$ and $(i, k) \in N$ means that the interaction between user i and item j are positive and negative implicit feedback, respectively. As we discuss in Section 2.1, it places zero-mean spherical Gaussian priors on user latent matrix U and the item latent matrix Z :

$$p(U|\sigma_U^2) = \prod_{i=1}^m \mathcal{N}(U_i|0, \sigma_U^2 I) \quad (11)$$

$$p(Z|\sigma_Z^2) = \prod_{k=1}^n \mathcal{N}(Z_k|0, \sigma_Z^2 I) \quad (12)$$

2.3. Decentralized Learning

In the current environment of information overload, in order to dig out valuable content from the massive amount of information for user, a large number of research has promoted the rapid development of personalized recommendation system. However, user data are normally distributed in various end devices. Due to privacy constraint, users and institutions are reluctant to share their personal data for centralized training. Decentralized learning [7] is proposed to solve this problem. Recently, it has been applied in many real-world applications, such as navigation [16] and localization [17]. Chen et al. [18] propose a decentralized matrix factorization based on FunkSVD for point-of-interest recommendation. Duriakova et al. [3] propose a general algorithmic framework for decentralized matrix factorization. However, it has a low efficiency problem. To the best of our knowledge, there is no existing work that achieves great performance in decentralizing a matrix factorization model using explicit and implicit feedback simultaneously.

In this study, we propose a decentralized learning scheme where models are shared and learned without needing a server node, as shown in Figure 1. Specifically, we im-

plement a decentralized CF model where a decomposing strategy is applied for users to exchange only non-private information, and thus collaboratively train a model.

3. The Proposed Model

In this section, we first provide overview regarding probabilistic matrix co-factorization, and then we devise a decentralized strategy to secure user privacy while training the model. The frequently used notations in the remainder of the paper are summarized in Table 1.

3.1. Overview

The probabilistic model discussed in Sections 2.1 and 2.2 can be well applied with explicit feedback or implicit feedback. However, it cannot handle problems with explicit and implicit feedback simultaneously. In real-world CF applications, a user chooses an item based on their preference but cannot tell whether the item really suits him/her. Thus, in this study, we relate items with two types of latent factor vectors to distinguish between their behavior and true taste, namely the item rating latent factor vector and the item selecting latent factor vector. A user will select a certain item if the item selecting latent factor vector matches the user latent factor vector and will give a high rating on the item if the item rating latent factor vector matches the user latent factor vector.

According to [11], real-world datasets contain both explicit and implicit feedback in the observed data and both of them are useful for predicting user preferences and modeling item characteristics. However, there is little research that uses both of them simultaneously in one recommendation model, which shows that the recommender system still has room for improvement. Thus, in this section, we propose a recommendation model based on probabilistic matrix factorization with both explicit and implicit feedback. As shown in Figure 2, we derive an implicit feedback matrix M and an explicit feedback matrix R from the original observed data. Then M is decomposed into the product of an item selecting latent factor matrix Z and a user latent factor matrix U . Similarly, the explicit latent factor matrix R is decomposed into the product of an item rating latent factor matrix V and a user latent factor matrix U . As mentioned in Equation (7), we also add an offset matrix Q to the user latent factor matrix to constrain user-specific feature vectors that has a strong effect on infrequent users.

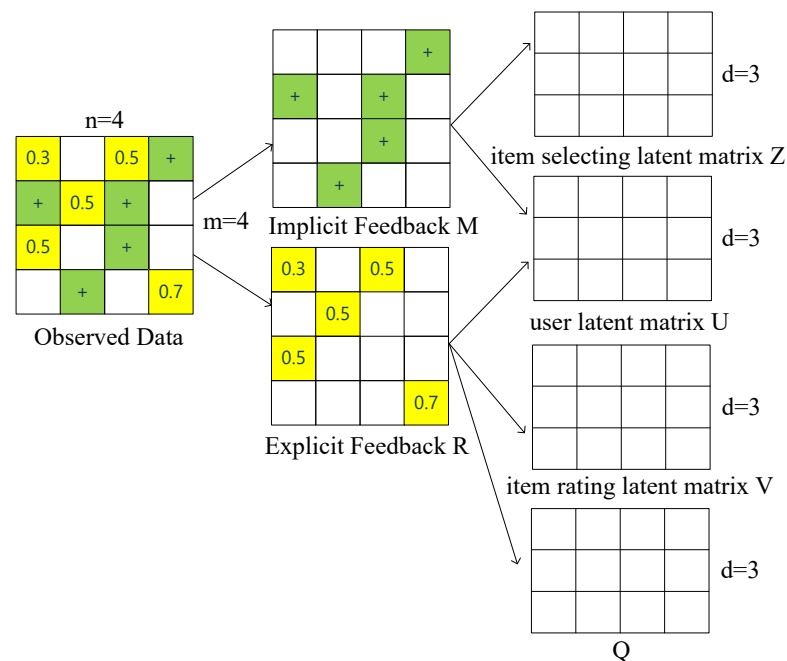


Figure 2. Observed data is divided into implicit and explicit feedback which are used for obtaining user latent matrix and different item latent matrix. '+' denotes the positive implicit feedback.

The main goal of our scheme is to separate rating prediction and item recommendation into two different tasks: once the model is obtained, we can predict the user ratings for the unknown items by calculating the product of U and V ; Further, we can provide a rank list for item recommendation by calculating the product of U and Z .

3.2. Matrix Co-Factorization

The key idea of matrix co-factorization is to associate implicit and explicit feedback together with the PMF model. Specifically, the model of decomposing explicit feedback matrix R and implicit feedback matrix M are linked by the user latent factor matrix U . Based on Bayesian law [19], we define the posterior distribution over user latent factor matrix U , item selecting latent factor matrix Z and item rating latent factor matrix V as:

$$\begin{aligned} & \ln p(U, V, Z, Q | R, M; \sigma_U^2, \sigma_V^2, \sigma_Z^2, \sigma_R^2, \sigma_Q^2, W) \\ & \propto p(R | U, V, Q; \sigma_R^2) p(M | U, Z; W) p(U | \sigma_U^2) p(V | \sigma_V^2) p(Z | \sigma_Z^2) p(Q | \sigma_Q^2) \\ & = \prod_{i=1}^m \prod_{j=1}^n [\mathcal{N}(R_{ij} | g([U_i + \frac{\sum_{k=1}^n I_{ik}^R Q_k}{\sum_{k=1}^n I_{ik}^R} V_j], \sigma_R^2)]^{I_{ij}^R} \prod_{(i,k) \in P} g(U_i^T Z_k) \prod_{(i,k) \in N} (1 - W_{i,k} g(U_i^T Z_k)) \quad (13) \\ & \times \prod_{i=1}^m \mathcal{N}(U_i | 0, \sigma_U^2) \prod_{j=1}^n \mathcal{N}(V_j | 0, \sigma_V^2) \prod_{j=1}^n \mathcal{N}(Z_j | 0, \sigma_Z^2) \prod_{k=1}^n \mathcal{N}(Q_k | 0, \sigma_Q^2) \end{aligned}$$

Similar as we obtain the loss function in Equation (6), by maximizing the probability of the posterior distribution in Equation (13), we have the following loss:

$$\begin{aligned} \min_{U, V, Z, Q} \mathcal{L} &= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^n I_{ij}^R (g([U_i + \frac{\sum_{k=1}^n I_{ik}^R Q_k}{\sum_{k=1}^n I_{ik}^R} V_j] - R_{ij}))^2 \\ & - \lambda (\sum_{(i,k) \in P} \ln g(U_i^T Z_k) + \sum_{(i,k) \in N} \ln(1 - W_{i,k} g(U_i^T Z_k))) \quad (14) \\ & + \frac{\lambda_U}{2} \|U\|_F^2 + \frac{\lambda_V}{2} \|V\|_F^2 + \frac{\lambda_Z}{2} \|Z\|_F^2 + \frac{\lambda_Q}{2} \|Q\|_F^2 \end{aligned}$$

where $\lambda = \sigma_R^2$, with $\lambda_U, \lambda_V, \lambda_Z, \lambda_Q$ defines similar regularization parameters. The main idea for our method is training over explicit and implicit feedback using stochastic gradient descent (SGD) [20], so first we need to obtain the gradients of \mathcal{L} with respect to U_i, V_j, Z_j and Q_k :

$$\frac{\partial \mathcal{L}}{\partial U_i} = \begin{cases} -\lambda \frac{g'(U_i^T Z_j)}{g(U_i^T Z_j)} Z_j + \lambda_U U_i, & \text{if } (i, j) \in P, \\ \lambda \frac{W_{i,j} g'(U_i^T Z_j)}{1 - W_{i,j} g(U_i^T Z_j)} + \lambda_U U_i, & \text{if } (i, j) \in N, \\ g'(\hat{U}_i^T V_j) (g(\hat{U}_i^T V_j) - R_{i,j}) V_j + \lambda_U U_i, & \text{if } I_{ij}^R = 1, \end{cases} \quad (15)$$

$$\frac{\partial \mathcal{L}}{\partial V_j} = g'(\hat{U}_i^T V_j) (g(\hat{U}_i^T V_j) - R_{i,j}) \hat{U}_i + \lambda_V V_j, \text{ if } I_{ij}^R = 1, \quad (16)$$

$$\frac{\partial \mathcal{L}}{\partial Z_j} = \begin{cases} -\lambda \frac{g'(U_i^T Z_j)}{g(U_i^T Z_j)} U_i + \lambda_Z Z_j, & \text{if } (i, j) \in P, \\ \lambda \frac{W_{i,j} g'(U_i^T Z_j)}{1 - W_{i,j} g(U_i^T Z_j)} U_i + \lambda_Z Z_j, & \text{if } (i, j) \in N, \end{cases} \quad (17)$$

If $I_{ij}^R = 1$, then for all k that satisfies $I_{ik}^R = 1$ we have:

$$\frac{\partial \mathcal{L}}{\partial Q_k} = g'(\hat{U}_i^T V_j)(g(\hat{U}_i^T V_j) - R_{i,j})V_j \frac{I_{ik}^R}{\sum_{k=1}^n I_{ik}^R} + \lambda_Q Q_k, \quad (18)$$

where \hat{U}_i denotes $U_i + (\sum_{k=1}^n I_{ik}^R Q_k) / \sum_{k=1}^n I_{ik}^R$ to simplify the equations.

After T rounds of training, the model reaches its convergence, as in Algorithm 1. As shown in (19), by mapping the product of U_i and V_j in the range of $[0, 1]$, we can predict user i 's rating for the unknown items, which corresponds to the rating prediction task. Similarly, the logistic function value of the product $U_i^T Z_j$ represents the predicted probability of user i selecting item j , which corresponds to the item recommendation task.

$$\begin{aligned} \hat{R}_{i,j} &= g(U_i^T V_j) \\ \hat{M}_{i,j} &= g(U_i^T Z_j) \end{aligned} \quad (19)$$

Algorithm 1: Stochastic gradient descent learning for probabilistic matrix co-factorization

Input: Observed data matrix, regularization parameter $\lambda_U, \lambda_V, \lambda_Z, \lambda_Q, \lambda$, numbers of iterations T , and learning rate θ

Output: latent factor matrixes U, V, Z , and Q

- 1 Derive the implicit feedback matrix M and the explicit feedback matrix R from the observed data;
 - 2 Initialize elements of U, V, Z, Q by zero-mean Gaussian distribution;
 - 3 **for** $i = 1$ to T **do**
 - 4 Shuffle training data;
 - 5 **foreach** user-item pair (i, j) in R, P , and N **do**
 - 6 Calculate gradients based on Equations (15)–(18);
 - 7 update latent factor matrix as follows:
 - 8 $U_i \leftarrow U_i - \theta \frac{\partial \mathcal{L}}{\partial U_i}$;
 - 9 $V_j \leftarrow V_j - \theta \frac{\partial \mathcal{L}}{\partial V_j}$;
 - 10 $Z_j \leftarrow Z_j - \theta \frac{\partial \mathcal{L}}{\partial Z_j}$;
 - 11 $Q_k \leftarrow Q_k - \theta \frac{\partial \mathcal{L}}{\partial Q_k}$;
 - 12 **end**
 - 13 **end**
 - 14 **return** U, V, Z , and Q
-

3.3. Decentralized PMF

Inspired by [18], in this study, we devise an efficient and privacy-preserving method to train the model. Specifically, the user latent factor matrix U and the offset matrix Q are kept on users' end devices and are updated locally using each user's private data. The item rating latent factor matrix V and the item selecting latent factor matrix Z are learned collaboratively by users. Since V and Z are stored on each user's hand, we use V_j^i and Z_j^i to denote the local version of V_j and Z_j stored on user i 's end device. To prevent malicious inference, V_j^i, Z_j^i are decomposed into personal parts vq_j^i, zq_j^i that are stored at user i 's end and public parts vp_j, zp_j that users collaboratively learn:

$$V_j^i = vp_j + vq_j^i \quad (20)$$

$$Z_j^i = zp_j + zq_j^i \quad (21)$$

As shown in Figure 3, in real-world scenarios, vp_j and zp_j are the public latent factor vector that all users collaboratively learn, thus at every moment there is a copy in each user's hand. That is to say, what user i actually holds are vp_j^i and zp_j^i :

$$V_j^i = vp_j^i + vq_j^i \quad (22)$$

$$Z_j^i = zp_j^i + zq_j^i \quad (23)$$

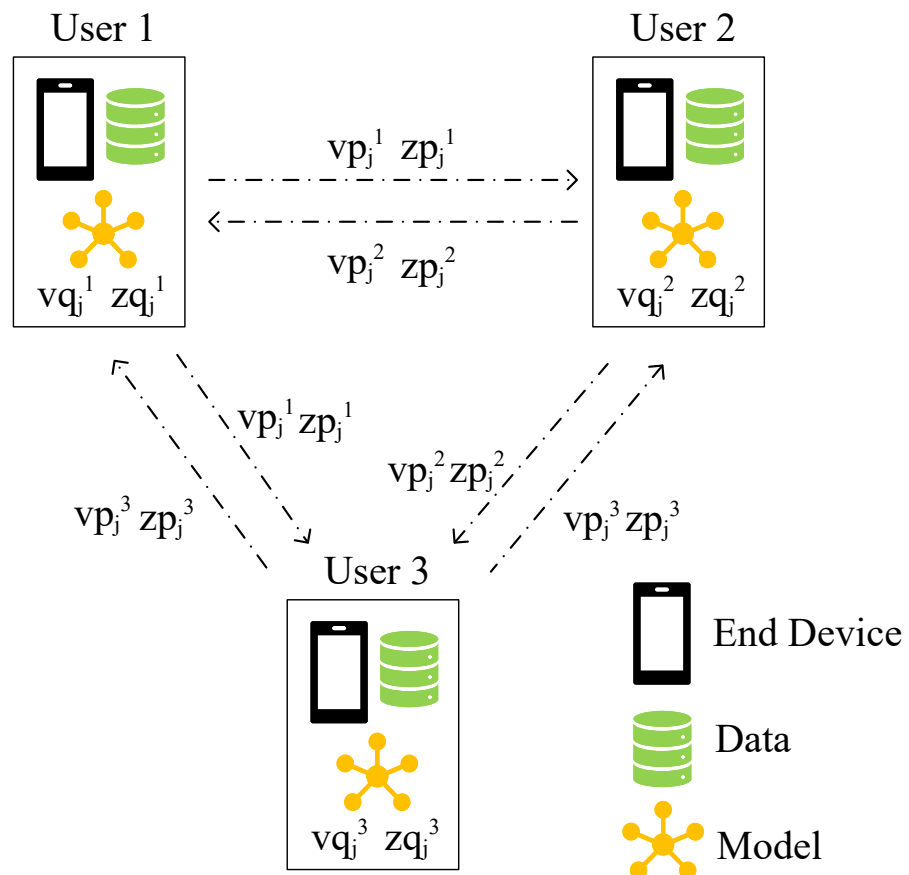


Figure 3. Users share public gradients with its neighbors while keeping personal gradients at their end.

Another question for implementing training is determining how users decide which neighbors they should communicate with. Normally, there is a large number of users in a decentralized learning model, so it is impractical to let all users communicate with each other for two reasons: (1) the communication cost is heavy, (2) fewer users are online simultaneously. Thus, we let users only communicate with a small group of other users, which we call neighbors in this study. According to [3], different neighbors have a different impact on learning a global model with user i , therefore we need to add a weight for each of user i 's neighbor j . To solve the above problems, we define a user adjacency matrix E with each of its element E_{ij} denotes the relationship between user i and user j . Normally user i and user j having a close connection leads to a large E_{ij} . Furthermore, we obtain d closest users for user i ($\mathcal{N}^d(i)$) by sorting all $E_{ij}, j \in 1, 2, \dots, n$ and select the largest d ones. For the sake of simplicity, we also call $\mathcal{N}^d(i)$ as user i 's neighbors in this study. Once user i has finished one round of its local training, it will send its public gradients to its neighbors to help them learn a global model. The overall process is summarized in Algorithm 2.

Algorithm 2: Decentralized PMF Algorithm

Input: implicit feedback (R), explicit feedback (M), weight matrix for implicit feedback (W), learning rate (θ), user adjacency matrix (E), neighbors of user i ($\mathcal{N}^d(i)$), regularization parameter ($\lambda_U, \lambda_V, \lambda_Z, \lambda_Q, \lambda$), number of iterations (T)

Output: latent factor vectors $U_i, v p_j^i, v q_j^i, z p_k^i, z q_k^i$

```

1 for  $i = 1$  to  $m$ ,  $j = 1$  to  $n$ , and  $k = 1$  to  $n$  do
2   | Initialize  $U_i, v p_j^i, v q_j^i, z p_k^i, z q_k^i, Q_k^i$ 
3 end
4 for  $t = 1$  to  $T$  do
5   | Derive the implicit feedback matrix  $M$  and the explicit feedback matrix  $R$  from
   | the observed data;
6   | Shuffle training data;
7   foreach user-item pair  $(i, j)$  in  $R, P$ , and  $N$  do
8     | Compute gradients according to (15)–(18) Update  $U_i$  by  $U_i \leftarrow U_i - \theta \frac{\partial \mathcal{L}}{\partial U_i}$ 
9     | if  $I_{ij}^R == 1$  then
10      | for  $k$  that satisfies  $I_{ik}^R = 1$  do
11      |   | Update  $Q_k$  by  $Q_k \leftarrow Q_k - \theta \frac{\partial \mathcal{L}}{\partial Q_k}$ 
12      |   end
13      | Update  $v p_j^i$  by  $v p_j^i \leftarrow v p_j^i - \theta \frac{\partial \mathcal{L}}{\partial v p_j^i} \frac{\partial V_j^i}{\partial v p_j^i}$ 
14      | Update  $v q_j^i$  by  $v q_j^i \leftarrow v q_j^i - \theta \frac{\partial \mathcal{L}}{\partial v q_j^i} \frac{\partial V_j^i}{\partial v q_j^i}$ 
15      | end
16      | else
17      |   | Update  $z p_j^i$  by  $z p_j^i \leftarrow z p_j^i - \theta \frac{\partial \mathcal{L}}{\partial z p_j^i} \frac{\partial Z_j^i}{\partial z p_j^i}$ 
18      |   | Update  $z q_j^i$  by  $z q_j^i \leftarrow z q_j^i - \theta \frac{\partial \mathcal{L}}{\partial z q_j^i} \frac{\partial Z_j^i}{\partial z q_j^i}$ 
19      |   end
20      | for user  $i'$  in  $\mathcal{N}^d(i)$  do
21      |   | if Receive  $\frac{\partial \mathcal{L}}{\partial v p_j^i}$  from user  $i$  then
22      |   |   |  $v p_j^{i'} \leftarrow v p_j^{i'} - \theta E_{ii'} \frac{\partial \mathcal{L}}{\partial v p_j^i}$ 
23      |   |   end
24      |   | if Receive  $\frac{\partial \mathcal{L}}{\partial z p_j^i}$  from user  $i$  then
25      |   |   |  $z p_j^{i'} \leftarrow z p_j^{i'} - \theta E_{ii'} \frac{\partial \mathcal{L}}{\partial z p_j^i}$ 
26      |   |   end
27      |   end
28      | end
29 end
30 return  $U_i, v p_j^i, v q_j^i, z p_k^i, z q_k^i, Q_k^i$ 

```

4. Evaluation

In this section, we compare our work with the state-of-the-art works. First, we provide a complexity analyze to prove the efficiency of our method. Then, by implementing the experiments on a multicore server using parallel toolbox, we explore the performance on real-world datasets.

4.1. Setting

4.1.1. Datasets

As shown in Table 2, we implement our experiment on two real-world datasets, namely *FilmTrust* [21] and *Epinions* [22,23]. The *FilmTrust* dataset contains 35,497 records with 1508 users and 2071 items. Moreover, 1853 trust statements are given as the relationship between users. Another dataset used in our experiment is *Epinions* collected from online website Epinions.com, which is also a famous benchmark for evaluating recommendation systems. However, we only sample 30,000 records randomly from the whole dataset due to its large scale. It consists of 1336 users and 6584 items with 7465 trust statements. For both datasets, we scale the ratings into the range of $[0, 1]$.

Table 2. Dataset Analysis.

Dataset	#Rating	#User	#Item	#Trust
<i>FilmTrust</i>	35,497	1508	2071	1853
<i>Epinions</i>	30,000	1336	6584	7465

4.1.2. Implicit Data

Since DPMD is a model investigating both explicit and implicit feedback, and both *FilmTrust* and *Epinions* contain only explicit ratings, we need to make modifications to the original dataset. Specifically, we randomly sample user i and its rated item j from the observed data at a certain percentage α to obtain the positive implicit feedback M_{ij} , and the rest remains for representing the explicit feedback R_{ij} . For the sake of simplicity, we set α as 50% in the following experiment. Moreover, 70% of the dataset is used for training and the rest is for testing.

4.1.3. Neighbor Adjacent Matrix

For dataset *TrustFilm* and *Epinions*, they both have a trust statement issued by users that are denoted as $\{(i, j, trust) | i \in \mathcal{U}, j \in \mathcal{V}\}$. We set $E_{ij} = \frac{1}{d}$ if $(i, j, trust)$ is in the trust statement. At each training iteration, only d users from $\{j | E_{ij} = \frac{1}{d}\}$ can receive user i 's public gradients, and those users are denoted as $\mathcal{N}^d(i)$.

4.1.4. Metric

For rating prediction task, we adopt root mean square error (RMSE) to measure the distance between the predicted result and the test data. With Y denotes all the user-item pairs in the test set of rating prediction task, we define RMSE as follows:

$$RMSE = \sqrt{\frac{1}{Y} \sum_{(i,j) \in Y} (\hat{R}_{ij} - R_{ij})^2} \quad (24)$$

Moreover, we follow the work of [24] to use $P@k$ and $R@k$ as two metrics to evaluate the item recommendation task. For user i , they are defined as $P@k = \frac{|S_i^T \cap S_i^R|}{k}$, $R@k = \frac{|S_i^T \cup S_i^R|}{S_i^T}$, where S_i^T denotes the items that user i chooses in the test set and S_i^R denotes the recommended items of user i which contains k results.

4.1.5. Baseline Methods

We compare SecSVD++ with three existing works as follows.

- DMF [18] is a decentralized model based on MF. It is mainly designed for point-of-interest recommendation, therefore we simplify the setting and make it practical for handling the same rating prediction problem as DPMD.
- SVD++ [25] is an improved version for MF version that takes users' historical interactions into consideration.

- WRMF [12] is a typical centralized matrix factorization method for implicit feedback.
- PMF is the probabilistic model that we introduced in Section 2.2.

4.2. Complexity Analysis

As concluded in Algorithm 2, the model is trained collaboratively by users. Here, we define K as the dimension of the latent factor vectors, and we use $|R|$ and $|M|$ to denote the size of the explicit data and the size of the implicit data, respectively.

The computation that our method carries out is mainly two parts: (1) users locally compute gradients of \mathcal{L} with respect to U_i and other latent factor vectors as shown in line 8 of Algorithm 2, and (2) users update the latent factors as shown from line 9 to line 27 of Algorithm 2. The computation complexity is $(|R| + |M|) \cdot K$ for (1) and $(|R| + |M|) \cdot K \cdot d$ for (2), so the total computation complexity is $(|R| + |M|) \cdot K \cdot d$ in one iteration. Normally, we can manually set d and K as small values so the computation overhead is linear with the size of the training set. As for the communication overhead, line 21 and line 24 in Algorithm 2 show that at a certain iteration, user i' receives public gradients $\frac{\partial \mathcal{L}}{\partial v_{p_j^i}}$ or $\frac{\partial \mathcal{L}}{\partial z_{p_j^i}}$ from user i . Thus, the communication cost is $(|R| + |M|) \cdot K \cdot d$, which is also linear with the size of the training set. The complexity comparison with other state-of-the-art methods is listed in Table 3, we can see that we have a comparable efficiency with latest DMF model.

Table 3. Complexity Comparison.

Methods	SVD++	DMF	WRMF	PMF	DPMF
Computation	$ R \cdot K$	$ R \cdot K \cdot d$	$m \cdot n \cdot K$	$ M \cdot K$	$(R + M) \cdot K \cdot d$
Communication	-	$ R \cdot K \cdot d$	-	-	$(R + M) \cdot K \cdot d$

4.3. Rating Prediction

First, we uniformly sample the same number of negative implicit feedback as the positive implicit feedback. Then we set $\lambda = 0.02$, $W_{ij} = 0.5$, $K = 3$, $d = 3$ and validate the convergence of our model on both datasets. As the number of iterations increases, the RMSE decreases for models using both datasets, and they both achieve convergence after around 100 rounds of iterations, as shown in Figure 4.

Furthermore, there are two important parameters in the loss function. One is λ , which adjusts the proportion of each part in the loss function. The other is W , which represents the weight of the negative implicit data. By varying the values of these two parameters and placing the RMSE results of the experiment on the contour plot as Figure 5, we can observe that when the model is training on *FilmTrust*, the RMSE reaches its minimum when $W = 0.5$ and $\lambda = 0.02$, and for *Epinions* the minimum RMSE is obtained when $W = 1$ and $\lambda = 0.06$.

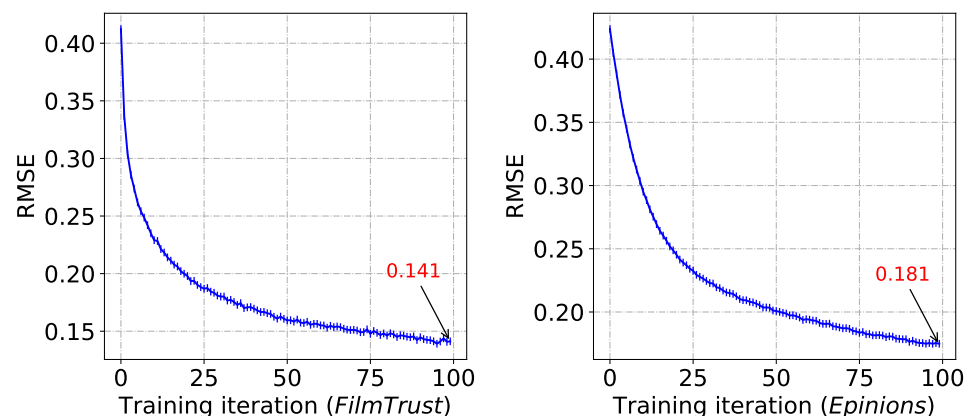


Figure 4. The RMSE of *FilmTrust* and *Epinions* decreases as the number of model iterations increases.

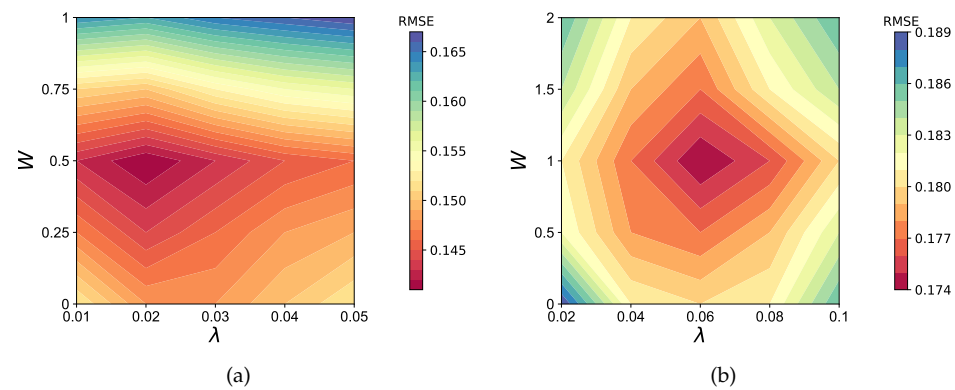


Figure 5. The contour map of RMSE as λ and W changes when training DPMF model on two datasets. (a) *FilmTrust*, (b) *Epinions*.

Based on this result, we further explore how our method compares with other state-of-the-art methods. Specifically, we set $\lambda = 0.02$, $W_{ij} = 0.5$, $d = 3$ for conducting experiment on *FilmTrust* and $\lambda = 0.06$, $W_{ij} = 1$, $d = 3$ for conducting experiment on *Epinions*. As shown in Figure 6, for the experiments on both datasets, the RMSE for all three models decreases as the size of the latent factor vector K increases and our model consistently outperforms the DMF and the SVD++ model. This is mainly because our model fully uses the implicit data which helps to train a matrix co-factorization model.

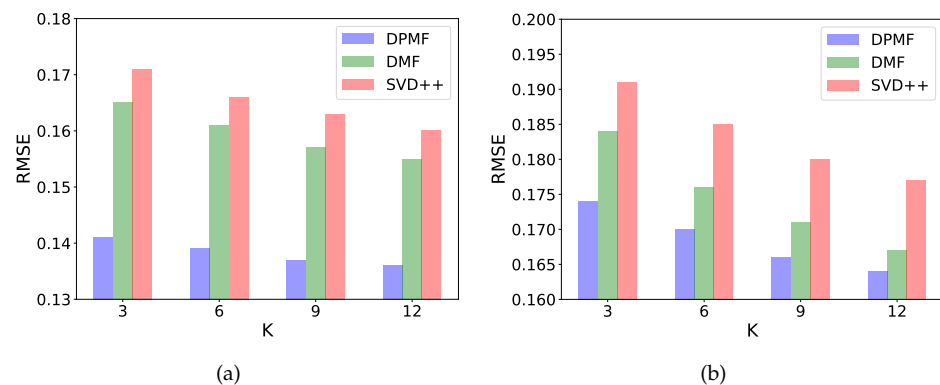


Figure 6. The histogram of RMSE as K changes when training different RS model on two datasets. (a) *FilmTrust*, (b) *Epinions*.

4.4. Item Recommendation

In this section, we compare our method with two famous models for the item recommendation task. One of them is the WRMF model, that assigns a weight and confidence to all user-item pairs, which helps to put the implicit data into the loss function for calculation. The other one is PMF, which we described in Section 2.2. Different from the traditional PMF model, it uses a logistic function to map the results to the range of $[0, 1]$, and then establishes probability distribution function through Bernoulli distribution. These two models are tailored to solve the item recommendation problems using implicit feedback.

As in Section 4.3, first we set $\lambda = 0.02$, $W_{ij} = 0.5$, $d = 3$ for conducting experiment on *FilmTrust* and $\lambda = 0.06$, $W_{ij} = 1$, $d = 3$ for conducting an experiment on *Epinions*. Then we change the value of K to see how precision and recall changes. In Table 4, we can see that (1) as the increase of K , the $P@5$ and $R@5$ of the three models all increase. (2) when K is small ($K = 3$), our model and the other two models have similar performance. (3) when K is large ($K \geq 6$), our model has clear advantages over the other two models. This is mainly because DPMF uses more data to construct the model, so as the size of the latent factors K becomes larger, we are more able to explore the potential information hiding in the data.

Table 4. Performance comparison for item recommendation task.

Metrics	<i>FilmTrust</i>		<i>Epinions</i>	
	<i>P@5</i>	<i>R@5</i>	<i>P@5</i>	<i>R@5</i>
Dimension	<i>K</i> = 3			
PMF	0.0332	0.0711	0.0715	0.1529
WRMF	0.0352	0.0751	0.0717	0.1497
DPMF	0.0342	0.0722	0.0716	0.1524
Dimension	<i>K</i> = 6			
PMF	0.0380	0.0771	0.0761	0.1571
WRMF	0.0379	0.0763	0.0760	0.1563
DPMF	0.0396	0.0796	0.0781	0.1599
Dimension	<i>K</i> = 12			
PMF	0.0410	0.0831	0.0781	0.1611
WRMF	0.0404	0.0823	0.0789	0.1603
DPMF	0.0435	0.0889	0.0821	0.1653

5. Conclusions

In this study, we propose a privacy-preserving recommendation framework based on decentralized probabilistic matrix factorization called DPMF. Specifically, we devise a novel model combining explicit and implicit feedback into a probabilistic matrix co-factorization model by decomposing observed data into explicit and implicit data matrixes and mapping users and items to a shared subspace of low dimensionality. Besides, we propose a novel decomposing strategy under decentralized settings to keep users' private information at their end while users' public information is shared and helps to learn the model collaboratively. The experiments on two real-world datasets demonstrate that compared with classic models, the proposed model improves its performance in lower loss in rating prediction task and higher precision in item recommendation task. Furthermore, the complexity analysis shows that our method is practical with linear computation and communication complexity.

In the future, we will focus on model compression. The recommendation model has made significant progress in using users' data to predict user preferences and model item characteristics. However, the scale of recommendation model is becoming larger since there are increasing parameters, thus the storage overhead is becoming higher. How to reduce the storage of the recommendation model will be our next stage of work.

Author Contributions: Conceptualization, X.Y., Y.L.; methodology, X.Y., Y.L. and S.F.; software, X.Y.; validation, X.Y.; formal analysis, X.Y. and Y.L.; investigation, X.Y.; resources, X.Y. and M.X.; data curation, X.Y. and M.X.; writing—original draft preparation, X.Y.; writing—review and editing, X.Y., S.F. and Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Nature Science Foundation of China (No.62102429, 62102422, 62072466, 61872372), the Natural Science Foundation of Hunan Province (No.2022JJ30667), and the NUDT Grants (No.ZK19-38).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The principal datasets used in this research can be downloaded from <https://guoguibing.github.io/librec/datasets.html> and <http://www.trustlet.org/epinions.html>. Accessed 25 August 2022.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alhijawi, B.; Awajan, A.; Fraihat, S. Survey on the Objectives of Recommender System: Measures, Solutions, Evaluation Methodology, and New Perspectives. *ACM Comput. Surv.* **2022**. [CrossRef]
2. Cui, Z.; Xu, X.; Fei, X.; Cai, X.; Cao, Y.; Zhang, W.; Chen, J. Personalized recommendation system based on collaborative filtering for IoT scenarios. *IEEE Trans. Serv. Comput.* **2020**, *13*, 685–695. [CrossRef]
3. Duriakova, E.; Huang, W.; Tragou, E.; Lawlor, A.; Smyth, B.; Geraci, J.; Hurley, N. An algorithmic framework for decentralised matrix factorisation. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 307–323.
4. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [CrossRef]
5. Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A survey on security and privacy of federated learning. *Future Gener. Comput. Syst.* **2021**, *115*, 619–640. [CrossRef]
6. Zhang, J.; Chen, J.; Wu, D.; Chen, B.; Yu, S. Poisoning attack in federated learning using generative adversarial nets. In *Proceedings of the 2019 18th IEEE International Conference on Trust, Security Furthermore, Privacy in Computing Furthermore, Communications/13th IEEE International Conference on Big Data Science Furthermore, Engineering (TrustCom/BigDataSE)*, Rotorua, New Zealand, 5–8 August 2019; pp. 374–380.
7. Hegedűs, I.; Danner, G.; Jelasity, M. Decentralized learning works: An empirical comparison of gossip learning and federated learning. *J. Parallel Distrib. Comput.* **2021**, *148*, 109–124. [CrossRef]
8. Song, M.; Wang, Z.; Zhang, Z.; Song, Y.; Wang, Q.; Ren, J.; Qi, H. Analyzing user-level privacy attack against federated learning. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 2430–2444. [CrossRef]
9. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and open problems in federated learning. *Found. Trends® Mach. Learn.* **2021**, *14*, 1–210. [CrossRef]
10. Wittkopp, T.; Acker, A. Decentralized federated learning preserves model and data privacy. In *Proceedings of the International Conference on Service-Oriented Computing*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 176–187.
11. Saito, Y.; Yaginuma, S.; Nishino, Y.; Sakata, H.; Nakata, K. Unbiased recommender learning from missing-not-at-random implicit feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*; Machinery: New York, NY, USA, 2020; pp. 501–509.
12. Li, H.; Diao, X.; Cao, J.; Zheng, Q. Collaborative filtering recommendation based on all-weighted matrix factorization and fast optimization. *IEEE Access* **2018**, *6*, 25248–25260. [CrossRef]
13. Xu, S.; Zhuang, H.; Sun, F.; Wang, S.; Wu, T.; Dong, J. Recommendation algorithm of probabilistic matrix factorization based on directed trust. *Comput. Electr. Eng.* **2021**, *93*, 107206. [CrossRef]
14. Cai, G.; Chen, N. Constrained probabilistic matrix factorization with neural network for recommendation system. In *Proceedings of the International Conference on Intelligent Information Processing*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 236–246.
15. Chen, S.; Peng, Y. Matrix Factorization for Recommendation with Explicit and Implicit Feedback. *Knowl.-Based Syst.* **2018**, *158*, 109–117. [CrossRef]
16. Huang, H.; Savkin, A.V.; Huang, C. Decentralized autonomous navigation of a UAV network for road traffic monitoring. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *57*, 2558–2564. [CrossRef]
17. Alshamaa, D.; Mourad-Chehade, F.; Honeine, P. Decentralized kernel-based localization in wireless sensor networks using belief functions. *IEEE Sens. J.* **2019**, *19*, 4149–4159. [CrossRef]
18. Chen, C.; Liu, Z.; Zhao, P.; Zhou, J.; Li, X. Privacy preserving point-of-interest recommendation using decentralized matrix factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, 4–6 February 2018; Volume 32.
19. Wu, W.; Fu, S.; Luo, Y. Practical Privacy Protection Scheme In WiFi Fingerprint-based Localization. In *Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, Sydney, Australia, 6–9 October 2020; pp. 699–708.
20. Li, Q.; Xiong, D.; Shang, M. Adjusted stochastic gradient descent for latent factor analysis. *Inf. Sci.* **2022**, *588*, 196–213. [CrossRef]
21. Guo, G.; Zhang, J.; Yorke-Smith, N. A novel Bayesian similarity measure for recommender systems. *IJCAI* **2013**, *13*, 2619–2625.
22. Massa, P.; Souren, K.; Salvetti, M.; Tomasoni, D. Trustlet, open research on trust metrics. *Scalable Comput. Pract. Exp.* **2008**, *9*. Available online: <https://personal.ntu.edu.sg/zhangj/paper/ijcai13-guibing.pdf> (accessed on 25 August 2022).
23. Massa, P.; Avesani, P. Trust-aware recommender systems. In *Proceedings of the 2007 ACM Conference on Recommender Systems*, Minneapolis, MN, USA, 19–20 October 2007; pp. 17–24.
24. Castells, P.; Moffat, A. Offline recommender system evaluation: Challenges and new directions. *AI Mag.* **2022**, *43*, 225–238. [CrossRef]
25. Koren, Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, NV, USA, 24–27 August 2008; pp. 426–434.