Interaction-level Membership Inference Attack Against Federated Recommender Systems

Wei Yuan
The University of Queensland
Brisbane, Australia
w.yuan@uq.edu.au

Chaoqun Yang
Griffith University
Gold Coast, Australia
chaoqun.yang@griffith.edu.au

Quoc Viet Hung Nguyen
Griffith University
Gold Coast, Australia
henry.nguyen@griffith.edu.au

Lizhen Cui
Shandong University
Jinan, China
clz@sdu.edu.cn

Tieke He
Nanjing University
Nanjing, China
hetieke@gmail.com

Hongzhi Yin*
The University of Queensland
Brisbane, Australia
h.yin1@uq.edu.au

ABSTRACT

The marriage of federated learning and recommender system (FedRec) has been widely used to address the growing data privacy concerns in personalized recommendation services. In FedRecs, users' attribute information and behavior data (i.e., user-item interaction data) are kept locally on their personal devices, therefore, it is considered a fairly secure approach to protect user privacy. As a result, the privacy issue of FedRecs is rarely explored. Unfortunately, several recent studies reveal that FedRecs are vulnerable to user attribute inference attacks, highlighting the privacy concerns of FedRecs. In this paper, we further investigate the privacy problem of user behavior data (i.e., user-item interactions) in FedRecs. Specifically, we perform the first systematic study on interaction-level membership inference attacks on FedRecs. An interaction-level membership inference attacker is first designed, and then the classical privacy protection mechanism, Local Differential Privacy (LDP), is adopted to defend against the membership inference attack. Unfortunately, the empirical analysis shows that LDP is not effective against such new attacks unless the recommendation performance is largely compromised. To mitigate the interaction-level membership attack threats, we design a simple yet effective defense method to significantly reduce the attacker's inference accuracy without losing recommendation performance. Extensive experiments are conducted with two widely used FedRecs (Fed-NCF and Fed-LightGCN) on three real-world recommendation datasets (MovieLens-100K, Steam-200K, and Amazon Cell Phone), and the experimental results show the effectiveness of our solutions.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Recommender System, Federated Learning, Membership Inference Attack and Defense

ACM Reference Format:
Wei Yuan, Chaoqun Yang, Quoc Viet Hung Nguyen, Lizhen Cui, Tieke He, and Hongzhi Yin. 2018. Interaction-level Membership Inference Attack Against Federated Recommender Systems. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In the age of the information explosion, recommender systems have become an essential means to alleviate information overload [5, 41], and many recommendation techniques have been proposed, including matrix factorization [23], deep learning based methods [10, 11], etc. These traditional recommender systems have already achieved good performance in diverse scenarios [44]. However, most of these traditional recommender systems work in a centralized way, i.e., they require collecting and storing users' historical interaction data to train a powerful recommender model in a central server [15]. As the increasing concerns of user privacy and the relevant privacy protection regulations such as the General Data Protection Regulation (GDPR) [32] in European Union and the California Consumer Privacy Act (CCPA) [6] in the United States, centrally collecting users' personal data is harder and even becomes infeasible in many cases [16].

To address the privacy issue, federated learning (FL) [22] has been recently adopted in recommender systems. In federated recommender systems (FedRecs), users can collaboratively train the recommender model but do not need to share their private data with either central servers or other users (clients). Therefore, FedRecs are considered a natural solution to protect users' sensitive information. Generally, FedRecs can be further divided into FedRecs with explicit feedback [16] and FedRecs with implicit feedback, according to their training datasets and optimization objectives. In this paper, we focus on FedRecs with implicit feedback[1]. Since Ammad et al. [1] proposed the first FedRec with collaborative filtering, many studies followed and extended their basic FedRec framework.

*Corresponding author.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

[1]To make the presentation concise, we directly use FedRec refer to FedRec with implicit feedback by default in the remaining part of this paper.

For example, FedFast [24] aims to accelerate the convergence of FedRec training. Imran et al. [13] and Wang et al. [35] focused on the efficiency of FedRecs.

With the remarkable attainment achieved in a short time [39], a few recent studies have started to verify whether FedRecs are "safe" enough. [45] is the first work to analyze the privacy issue of FedRecs. However, it only discussed sensitive attribute information leakage [46] and developed an effective attribute information protection approach. Although [16–19] studied the leakage and protection of user rating information in FedRecs, they all focused on explicit feedback data, which are much different from this work targeting FedRecs with implicit feedback.

Inferring a user's interaction data in FedRecs is one type of membership inference attack (MIA). Although MIA has been widely investigated in federated classification tasks [4, 21, 25, 30, 43, 49], their proposed attack and defense approaches cannot apply to FedRecs due to the following major differences between federated recommendation and federated classification. (1) From the perspective of attack objective, MIA in federated classification aims to infer or predict whether a sample has been used in the federated training process and which client has used it for the local training. However, in FedRecs, the associated item set of each client can be easily inferred by simply checking which items' embeddings are updated by the client. Furthermore, knowing such an item set is meaningless in FedRecs, since it consists of both positive and negative samples/items, and only positive samples (i.e., interacted items) can leak user privacy. Hence, the membership inference attack on FedRecs aims to infer the user's interacted items (i.e., positive samples), and we name such MIA as Interaction-level Membership Inference Attack (IMIA). (2) From the attack implementation perspective, MIA in federated classification needs to acquire extra i.i.d. data, which is however infeasible in FedRecs. In addition, the federated recommender architecture is significantly different from the federated classification model architecture. A client in FedRecs can have its private parameters (i.e., user embedding), while all model parameters in the federated classification models are shared.

In this paper, we first design a novel IMIA attacker to reveal the risk of leaking user interaction data in FedRecs and then propose an efficient and effective defender. The attack is launched by a central server that is honest but curious. The central server aims to identify a user's interacted items (i.e., positive samples) from its associated items (including both positive and negative samples) by analyzing the user's uploaded parameters without breaking the federated learning protocol. To be specific, given a target client, the attacker iteratively identifies its interacted items by repeating the following procedure. The attacker first randomly assigns ratings (0 or 1) to the client's associated items to construct a shadow training set, based on which a shadow recommender model is trained. Then, the attacker compares the relevance between the client's uploaded item embeddings and the item embeddings in the shadow recommender model to find the correctly guessed items. We implement the IMIA attacker on two representative FedRecs (Fed-NCF [1] and Fed-LightGCN [10]), and evaluate its inference accuracy on three real-world recommendation datasets (MovieLens-100K [7], Steam-200K [2], and Amazon Cell Phone [9]). The experimental results show the high inference accuracy of this new IMIA attacker, highlighting the risk of user interaction data leakage in FedRecs.

Recently, to improve the privacy-preserving ability of federated learning, Local Differential Privacy (LDP) has been employed in FedRecs and quickly becomes a gold standard for privacy preservation because of its effectiveness [20, 33, 40]. Therefore, we also evaluate the performance of the IMIA attacker in the above-mentioned FedRecs equipped with LDPs. It is found that LDP is not effective against such new attacks unless the recommendation performance is largely compromised, highlighting the timely demand for a new defense mechanism against the new IMIA.

In light of this, we propose a novel defense mechanism - IMIA defender. As there are both public and private parameters in FedRecs and only the public parameters can leak user privacy information, we impose a regularization term in the loss function of FedRecs to restrict the update and learning ability of the public parameters and enforce the private parameters to learn more useful patterns and account more for the recommendation performance. In this way, less sensitive information is transmitted to the server via the shared parameters. As shown in our experiments, our proposed defender can significantly decrease the inference accuracy of the IMIA attacker to the level of random guess with negligible influence on the recommendation performance.

In conclusion, the main contributions of this paper are summarized as follows:

- To the best of our knowledge, we are the first to perform a comprehensive privacy analysis of federated recommender systems under interaction-level membership inference attack (IMIA). Our study discloses the privacy risk of user interaction data in FedRecs.
- We find that the commonly used privacy-preserving approach, LDP, cannot effectively defend against the new IMIA attack. Then, we propose a simple yet effective defense mechanism to constrain the update of public parameters, which can significantly degenerate the IMIA attacker's performance to the level of random guesses without hurting the recommendation performance.
- Extensive experiments are conducted with two widely used federated recommender systems (Fed-NCF and Fed-LightGCN) on three real-world recommendation datasets, showing the effectiveness of our attack and defense approaches.

2 PRELIMINARIES

In this section, we first revisit the fundamental settings of FedRecs, and then formally define interaction-level membership inference attack and defense. Note that the bold lowercase (e.g. $\mathbf{a}$) represents vectors, bold uppercase (e.g. $\mathbf{A}$) denotes matrices, and squiggle uppercase (e.g. $\mathcal{A}$) signifies sets.

2.1 Federated Recommender System

Let $\mathcal{U}$ and $\mathcal{V}$ denote the sets of users (clients) and items, respectively. In FedRec, each user/client u_i has a local training dataset $\mathcal{D}_i$, which consists of user-item interactions (u_i, v_j, r_{ij}). $r_{ij} = 1$ means that user u_i has interacted with item v_j; otherwise, $r_{ij} = 0$, that is v_j is a negative sample. We use $\mathcal{V}_i^+$ and $\mathcal{V}_i^-$ to denote the interacted item set and negative sample set of user u_i. The FedRec is trained to predict $\hat{r}_{ij}$ between u_i and non-interacted items. Finally, FedRec

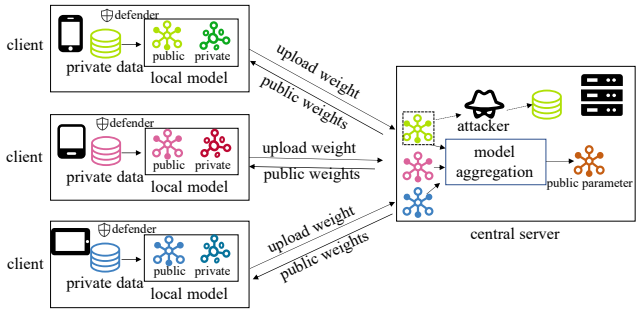


Figure 1: A typical federated recommender system with IMIA attacker and defender.

will recommend top-K ranked items with the highest predicted ratings to each user u_i.

In FedRec, a central server coordinates a large number of clients. The federated training process mainly contains four steps. First, the central server randomly selects a batch of users/clients as participants and dispenses the global parameters to these clients. Second, after receiving global parameters, each client combines these public parameters with their private parameters to form a local recommendation model and optimize this model on their local datasets regarding a certain objective function (e.g., BPRLoss [27]). Third, after local training, each client sends the updated public parameters back to the central server. Finally, the central server aggregates received public parameters with a certain aggregation strategy (e.g., FedAvg [22]). The above steps form a global training epoch in FedRec and will be repeated many times until the model convergence or meet some pre-defined requirement.

2.2 Interaction-level Membership Inference Attack and Defense

Adversary's Goal. In this paper, we assume the central server is honest-but-curious, i.e., the server is curious about user private data, but it will not break FedRec's learning protocol. The goal of the curious server is to infer the set of interacted items on each client u_i based on its uploaded public parameters:

$$\hat{\mathcal{V}}_i^+ \leftarrow IMIA(\mathrm{V}_i^t) \tag{1}$$

where $\hat{\mathcal{V}}_i^+$ is the inferred set of u_i's interacted items, and V_i^t represents public or shared parameters that user u_i sends to the server at epoch t. Without loss of generality, the public parameters mainly refer to item embeddings in this paper. The central curious server aims to accurately infer each client's interacted items, and meanwhile, it does not expect its inference attack to affect FedRec's normal learning process and recommendation performance.

Adversary's Knowledge. To be more realistic, we assume that the server has the following prior knowledge: (1) the target user u_i's uploaded public parameters (or gradients), which is consistent with the FedRec protocol; and (2) a few basic learning hyper-parameters, such as learning rate lr and the ratio of negative sampling η. In FedRecs, these hyper-parameters are pre-defined by the central server and broadcast to each participant client, therefore, this assumption of prior knowledge is reasonable.

Defense. The defense is launched locally by each client to defend against the curious server's inference attack. The client anticipates the defense method can significantly reduce the server's inference accuracy to protect their interaction data without much recommendation performance loss and extra computation footprint.

3 METHOD

In this section, we will first describe the base federated recommenders used in this paper and then present the details of the IMIA attacker and defender. Fig. 1 shows the framework of FedRec with IMIA attack and defense and the whole procedure is also described in Alg. 1.

3.1 Base Federated Recommender

Generally, a federated learning framework can be applied to most deep learning-based recommendation models. Among these recommenders, neural collaborative filtering (NCF) [11] and graph neural network (GNN) [29] are the two most widely used techniques. Hence, we extend an NCF-based centralized model and a LightGCN-based [10] centralized model to Fed-NCF and Fed-LightGCN respectively, which will be then used as our base FedRecs to show the effectiveness of our attacker and defender.

Neural Collaborative Filtering. NCF extends collaborative filtering (CF) by leveraging an L-layer feedforward network (FFN) to capture the complex patterns of user-item interactions as follows:

$$\hat{r}_{ij} = \sigma(\mathrm{h}^\top FFN([\mathrm{u}_i, \mathrm{v}_j])) \tag{2}$$

where u_i and v_j are user u_i's and item v_j's embedding; h denotes a learnable weight vector; $[\cdot]$ is concatenation operation, and $\hat{r}_{ij}$ is the predicted preference score of user u_i on item v_j.

LightGCN. In graph-based recommenders, the user-item interactions can be constructed as a bipartite graph. Then, LightGCN treats all users and items as distinct nodes. After that, user and item embeddings are learned by propagating their neighbor nodes' embeddings:

$$\mathrm{u}_i^l = \sum_{j \in \mathcal{N}_{u_i}} \frac{1}{\sqrt{|\mathcal{N}_{u_i}|}\sqrt{|\mathcal{N}_{v_j}|}} \mathrm{v}_j^{l-1}, \quad \mathrm{v}_j^l = \sum_{i \in \mathcal{N}_{v_j}} \frac{1}{\sqrt{|\mathcal{N}_{v_j}|}\sqrt{|\mathcal{N}_{u_i}|}} \mathrm{u}_i^{l-1} \tag{3}$$

where $\mathcal{N}_{u_i}$ and $\mathcal{N}_{v_j}$ denote the sets of u_i's and v_j's neighbors. l is the propagation layer. Note that under the federated learning setting, each user/client can only access its own data, thus they can only perform the above calculation on their local bipartite graphs.

After L layers propagation, we aggregate all layers' embedding together as the final user and item embeddings:

$$\mathrm{u}_i = \sum_{l=0}^{L} \mathrm{u}_i^l, \quad \mathrm{v}_j = \sum_{l=0}^{L} \mathrm{v}_j^l \tag{4}$$

Then, as done in NCF, E.q. 2 is adopted to compute the predicted preference scores.

FedRec Learning Protocol. In FedRec, the parameters can be divided into private and public parameters. Each client initializes its private parameters, i.e., user embedding u_i, and the public parameters V are initialized by a central server s. At the beginning of a global training epoch t, the server s randomly selects a group of clients as participants $\mathcal{U}_t$ and sends V_t to each participant. The

participant combines V_t with its private parameters to form a local recommender and trains the recommender on its local dataset $\mathcal{D}_i$ with the following loss function:

$$\mathcal{L}^{rec} = -\sum_{(u_i,v_j,r_{ij})\in\mathcal{D}_i} r_{ij}\log\hat{r}_{ij} + (1-r_{ij})\log(1-\hat{r}_{ij}) \quad (5)$$

After the local training, the client u_i locally updates its private user embedding u_i and uploads the updated public parameters V_i^t to the central server s. Then, the server utilizes FedAvg [22] to update the global parameters:

$$\mathrm{V}_{t+1} = \sum_{u_i\in\mathcal{U}_t} \mathrm{V}_i^t \quad (6)$$

The above steps iterate until the system converges or meets certain requirements.

Local Differential Privacy. As one of the most popular ways to protect users' sensitive data, LDP has been integrated into many FedRecs [38]. In this paper, we perform the analysis of IMIA attacks on not only the vanilla FedRecs but also FedRecs with the LDP mechanism. Following [37], before uploading public parameters to server s, the client adds some noises to V_i^t:

$$\mathrm{V}_i^t \leftarrow \mathrm{V}_i^t + \mathcal{N}(0,\lambda^2\mathrm{I}) \quad (7)$$

where $\mathcal{N}$ is the normal distribution and λ controls the scale of noise.

3.2 Interaction-level Membership Inference Attacker

In this work, the curious-but-honest central server is the IMIA attacker, who attempts to infer target user u_i's interacted item set $\mathcal{V}_i^+$. Basically, if the server has more prior information, such a membership attack is easier to implement with high accuracy. For example, if the server s can access u_i's private user embedding or a part of u_i's interaction data, it can simply train a shadow recommender to infer its other interacted items. However, these strong prior knowledge assumptions are unrealistic in real-world FedRecs. Therefore, we assume that the malicious server can only access the public parameters V_i^t uploaded by each client and some training hyper-parameters including the learning rate lr and the negative sampling ratio η.

Based on the public parameters V_i^t updated by u_i, the server can easily infer which items are involved during the local training according to their embedding updates. That is, for item v_j, if its embedding is updated by the client u_i, v_j participates in u_i's local training. But such simple inference is not useful since v_j can also be a negative sample. The malicious server would like to further infer whether v_j is positive or not for user u_i (i.e., the value of r_{ij}). Once the r_{ij} is accurately predicted, u_i's private interaction dataset $\mathcal{D}_i$ is exposed to the server. Thus, the membership inference attack problem transforms to predict r_{ij} for item v_j in $\mathcal{V}_i$.

Our attacker design is inspired by the following interesting empirical observation. Assume there is a local model M_i trained on its local dataset $\mathcal{D}_i$. $\mathrm{M}_i^{'}$ is also trained on $\mathcal{D}_i$ but its private parameters (i.e., user embedding) have different initial values. $\mathcal{D}_i^j$ represents a dataset in which v_j's rating r_{ij} is reversed, and all the other ratings are the same as in $\mathcal{D}_i$. For example, if $r_{ij}=1$ in $\mathcal{D}_i$, r_{ij} will be reversed to 0 in $\mathcal{D}_i^j$. $\mathrm{M}_i^{''}$ is trained on $\mathcal{D}_i^j$ with a different private parameter initial point. Before training, these three models' public parameters are the same. After training, we obtain the following interesting observation: $dist(\mathrm{v}_j,\mathrm{v}_j^{'}) < dist(\mathrm{v}_j,\mathrm{v}_j^{''})$. $dist(\cdot)$ denotes a distance function and the Euclidean metric is adopted in our paper. v_j, $\mathrm{v}_j^{'}$, and $\mathrm{v}_j^{''}$ are v_j's embeddings from model M_i, $\mathrm{M}_i^{'}$, and $\mathrm{M}_i^{''}$, respectively. It is worth noting that u_i's embeddings in M_i, $\mathrm{M}_i^{'}$, and $\mathrm{M}_i^{''}$ have different initial values. Table 1 provides a proof-of-concept. For each user, we randomly select one item from its local dataset and reverse the item's rating to construct the dataset $\mathcal{D}_i^j$. Once M_i, $\mathrm{M}_i^{'}$, $\mathrm{M}_i^{''}$ are trained, we can infer the rating r_{ij} in $\mathcal{D}_i$ only based on the item's rating in $\mathcal{D}_i^j$ and the distance of the item's embeddings in these three models. As shown in Table 1, the inference accuracy is higher than 90% in most cases, showing the effectiveness of this inference attack method. Based on this observation, if all other item ratings in $\mathcal{D}_i$ are known, we can infer v_j's rating r_{ij} by training $\mathrm{M}_i^{'}$ and $\mathrm{M}_i^{''}$ and then comparing their v_j's item embedding distance with the uploaded parameters V_i^t.

Table 1: Accuracy of inferring randomly select items' ratings for all users based on comparing Euclidean distances $dist(\mathrm{v}_j,\mathrm{v}_j^{'})$ and $dist(\mathrm{v}_j,\mathrm{v}_j^{''})$.

Models	MovieLens-100K	Steam-200K	Amazon
Fed-NCF	93.9%	97.6%	99.9%
Fed-LightGCN	79.7%	90.5%	91.15%

However, the IMIA attacker does not know any item rating in $\mathcal{D}_i$, so the above method cannot be directly used as the attack approach for FedRecs. To implement IMIA attacks, we relax the requirement and generalize the observation: if most samples are the same on two datasets $\mathcal{D}$ and $\mathcal{D}^{'}$, and we train two models M and $\mathrm{M}^{'}$ on them respectively, the embeddings of counterpart items will be close if their ratings are the same. Based on this assumption, when the server is curious about user u_i's interaction data at epoch t, the server first randomly assigns ratings (i.e., 0 and 1) for each item in $\mathcal{V}_i$ according to the negative sampling ratio η. For example, if η is $1:4$, the server will randomly choose 25% items as positive items and the remaining items as negative ones, thus constructing a fake dataset $\mathcal{D}_i^{fake}$. Since the negative samples are empirically several times more than the positive items, $\mathcal{D}_i^{fake}$ and $\mathcal{D}_i$ still have a portion of common ratings. Still taking $\eta = 1:4$ as an example, although in the worst case all positive items are wrongly assigned with rating 0, $\mathcal{D}_i^{fake}$ and $\mathcal{D}_i$ still have 50% the same item ratings. Then, the server trains a shadow model M_i^{fake} based on $\mathcal{D}_i^{fake}$. After that, the malicious server calculates the distance between item embeddings from M_i^{fake} and the uploaded item embeddings V_i^t, and it chooses $\gamma * |\mathcal{V}_i|$ items with the smallest distance as "correct guess". The ratings of "correct guess" items will be fixed in the next iteration. Repeat the above steps several times until the server finishes inferring the positive item set $\hat{\mathcal{V}}_i^+$ for user u_i. Since the whole inference attack process happens on the malicious server side by using uploaded public parameters, the client is unaware of the IMIA attack. In addition, the malicious server can also store the target user's uploaded parameters and asynchronously execute the inference attack process without interrupting the normal training of

FedRecs. Lines 23-32 in Alg. 1 describe the process of the proposed IMIA attack with pseudo-code.

Algorithm 1 FedRec with IMIA attacker and defender.

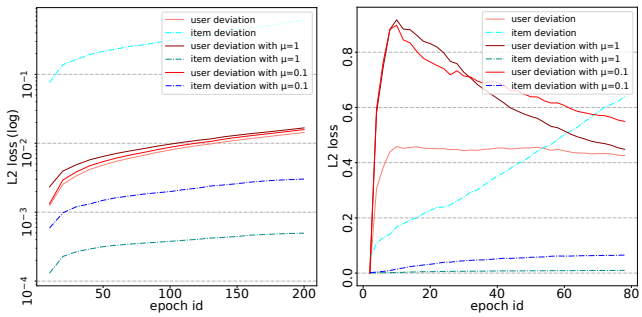
Input: global epoch T; local epoch L; learning rate lr, negative sampling rate η, ...;
Output: global parameter V, local client embedding $\mathrm{u}_i|_{i\in\mathcal{U}}$;

Initializing global parameter V_0;
for each round t = 0, 1, ..., T **do**
  sampling a fraction of clients $\mathcal{U}_t$;
  for $u_i \in \mathcal{U}_t$ **do**
    $\mathrm{V}_i^t \leftarrow$ ClientTrain(u_i, V_t, L);
    if curious about u_i's data **then**
      $\hat{\mathcal{V}}_i^+ \leftarrow$Attacker($\mathrm{V}_i^t$,$\eta$, γ);
    end if
  end for
  $\mathrm{V}_{t+1} = \sum_{u_i\in\mathcal{U}_t} \mathrm{V}_i^t$;
end for
function ClientTrain(u_i, V_t, L)
  downloading V_t from the server;
  sampling negative items $\mathcal{V}_i^{neg}$;
  if use IMIA defender **then**
    $\mathrm{u}_i^{t+1}, \mathrm{V}_i^t \leftarrow$ training L epochs with E.q. 8;
  else
    $\mathrm{u}_i^{t+1}, \mathrm{V}_i^t \leftarrow$ training L epochs with E.q. 5;
  end if
  if use LDP, add noise with E.q. 7;
  return V_i^t
end function
function Attacker($\mathrm{V}_i^t, \eta, \gamma$)
  $\hat{\mathcal{V}}_i^+ = \{\}$
  $\mathcal{V}_i \leftarrow$ select updated items according to V_i^t and V_t;
  while $\left|\hat{\mathcal{V}}_i^+\right| < \eta\,|\mathcal{V}_i|$ **do**
    randomly assign ratings to $v_j \in \mathcal{V}_i \setminus \hat{\mathcal{V}}_i^+$;
    train fake model M_i^{fake} on constructed dataset;
    $\hat{\mathcal{V}}_i^+, \hat{\mathcal{V}}_i^- \leftarrow$ select $\gamma * |\mathcal{V}_i|$ items using $dist(\mathrm{V}_i^t, \mathrm{V}_i^{fake})$;
  end while
  return $\hat{\mathcal{V}}_i^+$
end function

3.3 Interaction-level Membership Inference Defender

In Section 4.5 and 4.6, the experimental results demonstrate that both vanilla FedRecs and FedRecs with LDP are vulnerable to the new attack IMIA, highlighting the need for a new defense mechanism. The experimental results in Table 3 and 4 show that Fed-LightGCN is more resistant to IMIA. This may be because the private user embeddings in Fed-LightGCN learn more useful information and patterns than in Fed-NCF. Since the private user embeddings in Fed-LightGCN capture more user-item interaction patterns, it is harder for the curious server to infer interactions only from public parameters.

To further validate our hypothesis, we compare the deviation of user/item embeddings in the training process from their initial values using L2 loss (i.e., $dist^2(v_i^t - v_i^0)$). Fig. 2 illustrates the trend of the average deviation over training time. In Fig. 2, the deviation of item embeddings is much larger than user embeddings' deviation. In other words, on average, user embeddings do not change as much as item embeddings during the whole training process, therefore user embeddings learn less information and patterns. Further, by comparing Fig. 2a and Fig. 2b, we can see that user embeddings in Fed-NCF vary much less than in Fed-LightGCN, which supports our hypothesis. Note that for the sake of visualization, we log the L2 loss value in Fig. 2a because of the large difference between user and item embedding deviation.



(a) Deviation trend in Fed-NCF. (b) Deviation trend in Fed-LightGCN.

Figure 2: Trend of embedding deviation over time until convergence in Fed-NCF and Fed-LightGCN on MovieLens-100K.

Motivated by the above observation, we propose a novel IMIA defender. The basic idea of LDP is to add noise to the shared parameters to distort the sensitive information behind the shared parameters, leading to catastrophic performance dropping. Unlike LPD, the key idea of our defender is to restrict the learning ability of public parameters so that they will convey less information to the curious central server. To implement that, we add a constraint term in the original FedRec loss function E.q. 5, as follows:

$$\mathcal{L} = \mathcal{L}^{rec} + \mu \left\| \mathrm{V}_i^t - \mathrm{V}_t \right\| \tag{8}$$

The constraint term limits the update of the public parameters V_t on each local client/device. Consequently, to optimize $\mathcal{L}^{rec}$, the recommender model would enforce the private embeddings to learn more information and patterns. Fig. 2 shows the embedding deviation trend after applying our defender to Fed-NCF and Fed-LightGCN. User embedding deviation becomes larger than vanilla FedRecs, while item embedding deviation significantly drops. More details of embedding deviation are in Appendix A.

4 EXPERIMENTS

4.1 Datasets

We use three real-world datasets (MovieLens-100K [7], Steam-200K [2], and Amazon Cell Phone [9]) from various domains (movie recommendation, game recommendation, and cell phone recommendation) to evaluate the performance of our IMIA attacker and

defender. The statistics of these datasets are shown in Table 2. MovieLens-100K contains $100,000$ interactions between 943 users and $1,682$ items. There are $3,753$ users, $5,134$ items, and $114,713$ interactions in Steam-200K. Amazon Cell Phone consists of $13,174$ users, $5,970$ cell phone related items, and $103,593$ interactions. Note that the densities of these three datasets are different. MovieLens-100K is the densest dataset, while Amazon Cell Phone is the most sparse one. Following [47], we binarize the user feedback, where all ratings are transformed to $r_{ij} = 1$ and negative instances are sampled with $1:4$ ratio. Besides, we utilize the leave-one-out method to split the training, validation, and test sets.

Table 2: Statistics of recommendation datasets

Dataset	#users	#items	#interactions	Avg.	Density
MovieLens-100K	943	1,682	100,000	106	6.30%
Steam-200K	3,753	5,134	114,713	31	0.59%
Amazon	13,174	5,970	103,593	8	0.13%

4.2 Evaluation Metrics

To measure the effectiveness of IMIA attackers, we employ the widely used classification metric F1 score to evaluate inference performance. To evaluate the recommendation performance, we adopt the widely used hit ratio at rank 10 (Hit@10), which measures the ratio of ground truth items that appear in the top-10 recommendation list.

4.3 Baselines

Since none of the prior works conducts interaction-level membership attacks on FedRecs, we design two baselines.

Random Attack. For each client u_i, the server randomly selects a group of items from $\mathcal{V}_i$ as the positive items based on the negative sampling ratio η. Comparing with Random Attack can reveal whether a privacy issue of user interaction data exists.

K-means Attack. Since we do not have any labels of user-item interaction samples, IMIA can naturally be treated as a clustering problem. We adopt K-means [8] algorithm to divide items into two clusters based on the client's uploaded public parameters V_i^t. Positive items are chosen from the cluster with lower SSE (the sum of squared errors). The intuition of K-means Attack is that for a user, the positive items are more similar to each other than diverse negative items due to the coherence principle of personal interests, therefore, their embeddings will also be more coherent.

4.4 Parameter Settings

For both Fed-NCF and Fed-LightGCN, the dimension of user and item embeddings is 64, and 3 neural layers with dimensions $128, 64, 32$ are used to process the concatenated user and item embedding. The negative sampling ratio η is set to $1:4$, as this ratio can well balance the training effectiveness and efficiency for most pair-wise loss functions and has been widely used. The local training batch size and local epoch size are 64 and 20, respectively. Adam [14] optimizer with 0.001 learning rate is employed to optimize local models. To ensure the model convergence, the maximum global epoch is set to 200. γ is set to 20%. We also perform the sensitivity analysis of key hyper-parameters in the experiment.

4.5 Performance of IMIA Attackers

Table 3 presents three attackers' performances on two FedRecs and three datasets. The results are average F1 scores that reflect the inference effectiveness of the IMIA attacker. The results in Table 3 highlight that **vanilla FedRecs have a high risk of user interaction data leakage**, since the performance of our IMIA attacker is much better than Random Attack. Besides, comparing K-means and our attacker, we can see that the naive clustering method cannot effectively infer user interaction information. Furthermore, by comparing our IMIA attacker's performances crossing datasets, we can find that FedRecs trained on Steam-200K and Amazon Cell Phone are more vulnerable to IMIA than the ones trained on MovieLens-100K. With the statistics of datasets in Table 2, we believe that this phenomenon is related to the number of user interactions because the average number of user interactions on MovieLens-100K is much higher than that on the other two datasets. To further investigate this phenomenon on MovieLens-100K, we cluster users into 20 groups according to their interaction numbers and report their average F1 score in Fig. 3. The results show that **users with fewer interactions have a higher risk of interaction data leakage**. Appendix B analyzes this phenomenon on all datasets.

Table 3: The performance (F1 scores) of attackers on vanilla FedRecs. ML-100K is short for MovieLens-100K, Amazon is short for Amazon Cell Phone.

Model	Attack	ML-100K	Steam-200K	Amazon
	Random	0.2079	0.2019	0.1998
Fed-NCF	**K-means**	0.3183	0.2477	0.2458
	Ours	**0.5928**	**0.6707**	**0.6516**
Fed-LightGCN	**K-means**	0.1460	0.2573	0.2697
	Ours	**0.3900**	**0.6007**	**0.4328**

Finally, the comparison of IMIA attackers' performances on Fed-NCF and Fed-LightGCN shows that **Fed-LightGCN is more resistant to IMIA than Fed-NCF**. This may be because that **private parameters (i.e., user embeddings) in Fed-LightGCN learn more useful information than in Fed-NCF**, since user embeddings in Fed-LightGCN aggregate information from item embedding via convolution operation. As a result, only using public parameters to infer user interaction records becomes harder. In Appendix A, we further show the embeddings' deviation from their initial values. The results support our explanation. The above observation motivates us to design our effective IMIA defender (see Section 3.3), which attempts to limit the learning ability of public parameters and enforce private parameters to learn more patterns.

4.6 Effectiveness of LDP Against IMIA

As the most classical and widely used privacy-preserving approach, LDP can effectively prevent attribute inference attacks on FedRecs [45]. Here, we conduct this experiment to study whether LDP can defend against the new inference attack IMIA. Table 4 presents the results of LDP with different noise scales against IMIA attacks. $\lambda = 0.0$

Table 4: The result of Local Differential Privacy (LDP) against our IMIA attacker. F1 is the attacker's performance, and the lower scores (↓) are better. Hit@10 (↑) measures recommendation performance, and the higher scores are better.

Model	Dataset	Noise Scale							
		λ =0.0		λ =0.001		λ =0.01		λ =0.1	
		F1↓	Hit@10↑	F1↓	Hit@10↑	F1↓	Hit@10↑	F1↓	Hit@10↑
Fed-NCF	ML-100K	0.5928	**0.3690**	0.5474	0.3308	0.3954	0.2958	**0.2520**	0.1696
	Steam-200K	0.6707	**0.6645**	0.6012	0.5901	0.3334	0.4524	**0.2199**	0.2224
	Amazon	0.6516	**0.2176**	0.6260	0.1984	0.2933	0.1505	**0.2126**	0.1217
Fed-LightGCN	ML-100K	0.3900	**0.4072**	0.3786	0.3923	0.2816	0.3658	**0.2357**	0.3138
	Steam-200K	0.6007	**0.6943**	0.5690	0.6957	0.3392	0.6890	**0.2188**	0.5123
	Amazon	0.4328	**0.1796**	0.3483	0.1717	0.2642	0.1720	**0.2209**	0.1562

Table 5: The result of our defender against IMIA. The best results on each dataset are bold.

Model	Dataset	Constraint Scale									
		μ =0.0		μ =0.1		μ =0.4		μ =0.7		μ =1.0	
		F1↓	Hit@10↑	F1↓	Hit@10↑	F1↓	Hit@10↑	F1↓	Hit@10↑	F1↓	Hit@10↑
Fed-NCF	ML-100K	0.5928	0.3690	0.2638	0.3605	**0.2140**	**0.3743**	0.2166	0.3563	0.2145	0.3531
	Steam-200K	0.6707	**0.6645**	0.3888	0.6005	0.2667	0.6011	0.2213	0.5960	**0.2058**	0.5960
	Amazon	0.6516	**0.2176**	0.4761	0.2142	0.3368	0.2129	**0.3079**	0.2126	0.3240	0.2121
Fed-LightGCN	ML-100K	0.3900	0.4072	0.2130	**0.4082**	0.1892	0.3891	0.1811	0.3796	**0.1741**	0.3870
	Steam-200K	0.6007	**0.6943**	0.4730	0.6584	0.4620	0.5830	0.4205	0.5582	**0.2246**	0.5472
	Amazon	0.4328	0.1796	**0.2281**	**0.1920**	0.2847	0.1821	0.3231	0.1704	0.3308	0.1615

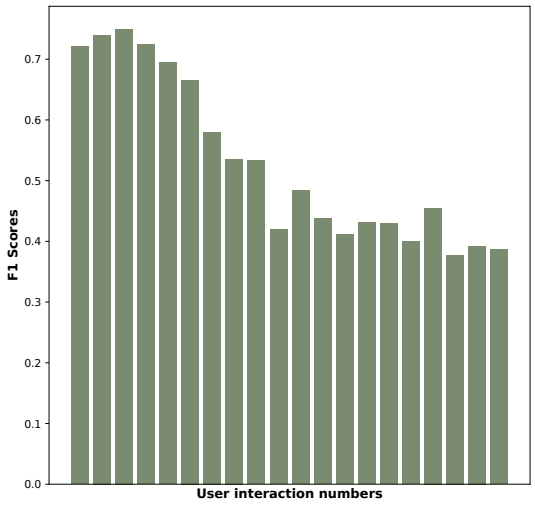


Figure 3: IMIA attacker performance for users with different number of interactions on MovieLens-100K.

means FedRecs without LDP. The results indicate that with subtle noise (e.g. $\lambda = 0.001$), LDP cannot well protect user interaction data. Adding more noises (e.g. $\lambda = 0.1$) can defend against our IMIA attacker, however, stronger noises severely degenerate the recommendation performance of FedRecs.

To measure how much recommendation performance LDP needs to sacrifice to effectively defend the attacker, we calculate $\frac{|\Delta F1|}{|\Delta Hit@10|}$ for the LDP which degenerates the IMIA attacker's performance to the level of Random Attack. Intuitively, $\frac{|\Delta F1|}{|\Delta Hit@10|}$ measures the change ratio of the attacker's performance and recommendation performance. Lower scores represent that the defender has to sacrifice more recommendation performance to reduce the attacker's threat. Table 6 shows that LDP would sacrifice too much recommendation performance to alleviate IMIA threats. As a result, **LDP is not cost-effective to defend against IMIA**.

Table 6: Comparison of $\frac{|\Delta F1|}{|\Delta Hit@10|}$ for LDP and our defender. Higher scores represent the more cost-effective defense. NCF and LightGCN are short for "Fed-NCF" and "Fed-LightGCN".

Defense	ML-100K		Steam-200K		Amazon	
	NCF	LightGCN	NCF	LightGCN	NCF	LightGCN
LDP	1.70	1.65	1.01	2.09	4.57	9.05
ours	**71.47**	**10.68**	**6.78**	**2.55**	**68.74**	**16.50**

4.7 Effectiveness of IMIA Defender

Since LDP cannot effectively mitigate IMIA threats, we propose a novel defense mechanism against the IMIA attack. The results of our defender against IMIA are shown in Table 5 where we vary the values of the hyper-parameter μ from 0.0 to 1.0, and $\mu = 0.0$ represents the vanilla FedRecs. With our defense method, the attacker's performance is reduced to the level of random guesses in all cases. Meanwhile, the recommender's performance is even improved in some cases (e.g., Fed-NCF on ML-100K, Fed-LightGCN on ML-100K, and Amazon Cell Phone) due to the regularization effect of the constraint term in the loss function, which indicates that

when restricting the updates of public parameters, the recommendation models can still achieve good recommendation performance by enforcing private parameters to learn more patterns.

Table 6 shows the comparison between LDP and our defender. The higher scores represent that the defender invalids the IMIA attacker with less performance loss. As we can see, in all cases, our defender is more cost-effective than LDP. Specifically, our defender's $\frac{|\Delta F1|}{|\Delta Hit@10|}$ scores for Fed-NCF on MovieLens-100K and Amazon Cell Phone are nearly 40 times and 15 times higher than LDP. In conclusion, our defender provides a more cost-effective solution against IMIA than LDP.

4.8 Attack with More Prior Knowledge

As mentioned in Section 3.2, to make the threat more realistic, we strictly restrict the curious server's prior knowledge with only uploaded parameters and some hyper-parameters such as learning rate and sampling ratio. In this section, we explore one possible prior knowledge that the server may have chances to access: the popularity information of items. Although the popularity information is not always accessible, it is still available in many scenarios. In this part, we assume that the server knows the top 10% popular items. Based on the popularity information, instead of randomly assigning ratings to items at the initial phase, the server assigns positive ratings to popular items with a higher probability. Fig. 4 shows that with the item popularity information, the IMIA attacker's performance is improved in most cases.

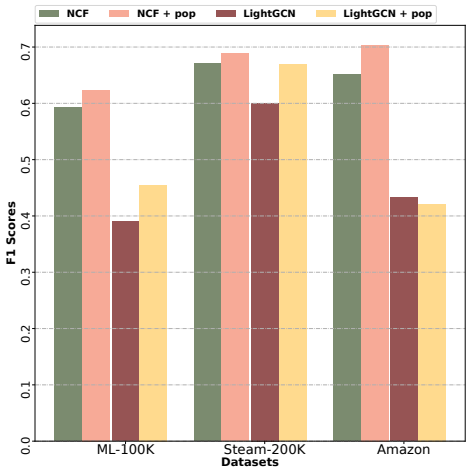


Figure 4: IMIA with popularity information. NCF and LightGCN are short for "IMIA for Fed-NCF" and "IMIA for Fed-LightGCN". "pop" means popularity information.

5 RELATED WORK

In this section, we mainly introduce the related works of attacks against federated learning and attacks against federated recommender systems. The recent progress of recommender systems, federated recommender systems, federated learning, and local differential privacy can be referred to [21, 31, 34, 39, 44].

5.1 Attack against Federated Learning

Recently, varieties of attacks were proposed to access privacy risks in federated learning (FL) [21, 28]. These attacks include threats such as model inversion [48], attribute inference [3], and membership inference. In this paper, we mainly discuss membership inference attacks. Nasr et al. [25] took the first comprehensive study of class-level membership inference attack in FL under both white-box and black-box settings. Then, many works took further steps to study more fine-grained membership inference attacks, e.g. [12, 26, 30, 36, 49]. However, existing membership inference attacks cannot be used in FedRec because of the major differences mentioned in Section 1.

5.2 Attack against Federated Recommendation

Zhang et al. [45] conducted the first analysis of FedRec's privacy-preserving, however, their work only reveals attribute-level leakage risks. Some research discussed the user rating privacy issue of FedRec with explicit feedback [16–19], but the interaction privacy issue of FedRec with implicit feedback is another pair of shoes. Other attack methods [47] aim to promote/demote item's rank, which cannot reveal the privacy issue of FedRecs. As a result, the privacy issue of FedRecs is still under explored. Besides, the defense method for improving federated recommendation's privacy protection is also under explored [42].

6 CONCLUSION

In this paper, we perform the first study of interaction-level membership inference attacks (IMIA) in federated recommender systems (FedRecs) to reveal the privacy issue of user-item interactions. We first design an attacker from the curious-but-honest server side. The attacker infers the target user's private interaction based on its uploaded public parameters by iteratively training shadow models on shadow datasets. We implement IMIA attack with two commonly used FedRecs on three real-world datasets. The experimental results validate the threats of IMIA for FedRecs. Furthermore, we find that the classical privacy-preserving method, LDP, cannot effectively defend against our attack. In light of this, we propose a novel defender to mitigate IMIA threats with imperceptible influence on the recommendation performance.

ACKNOWLEDGMENTS

This work is supported by Australian Research Council Future Fellowship (Grant No. FT210100624), Discovery Project (Grant No. DP190101985).

REFERENCES

[1] Muhammad Ammad-Ud-Din, Elena Ivannikova, Suleiman A Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and Adrian Flanagan. 2019. Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888* (2019).

[2] Germán Cheuque, José Guzmán, and Denis Parra. 2019. Recommender systems for Online video game platforms: The case of STEAM. In *Companion Proceedings of The 2019 World Wide Web Conference*. 763–771.

[3] Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. 2018. Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 619–633.

[4] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems* 33 (2020), 16937–16947.

[5] Carlos A Gomez-Uribe and Neil Hunt. 2015. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)* 6, 4 (2015), 1–19.

[6] Elizabeth Liz Harding, Jarno J Vanto, Reece Clark, L Hannah Ji, and Sara C Ainsworth. 2019. Understanding the scope and impact of the California Consumer Privacy Act of 2018. *Journal of Data Protection & Privacy* 2, 3 (2019), 234–253.

[7] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[8] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)* 28, 1 (1979), 100–108.

[9] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. 507–517.

[10] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[12] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, and Xuyun Zhang. 2021. Source inference attacks in federated learning. In *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1102–1107.

[13] Mubashir Imran, Hongzhi Yin, Tong Chen, Nguyen Quoc Viet Hung, Alexander Zhou, and Kai Zheng. 2022. ReFRS: Resource-efficient Federated Recommender System for Dynamic and Diversified User Preferences. *ACM Transactions on Information Systems (TOIS)* (2022).

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Shyong K Lam, Dan Frankowski, John Riedl, et al. 2006. Do you trust your recommendations? An exploration of security and privacy issues in recommender systems. In *International conference on emerging trends in information and communication security*. Springer, 14–29.

[16] Feng Liang, Weike Pan, and Zhong Ming. 2021. Fedrec++: Lossless federated recommendation with explicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4224–4231.

[17] Guanyu Lin, Feng Liang, Weike Pan, and Zhong Ming. 2020. Fedrec: Federated recommendation with explicit feedback. *IEEE Intelligent Systems* 36, 5 (2020), 21–30.

[18] Zhaohao Lin, Weike Pan, and Zhong Ming. 2021. FR-FMSS: federated recommendation via fake marks and secret sharing. In *Fifteenth ACM Conference on Recommender Systems*. 668–673.

[19] Zhaohao Lin, Weike Pan, Qiang Yang, and Zhong Ming. 2022. A Generic Federated Recommendation Framework via Fake Marks and Secret Sharing. *ACM Transactions on Information Systems (TOIS)* (2022).

[20] Zhiwei Liu, Liangwei Yang, Ziwei Fan, Hao Peng, and Philip S Yu. 2022. Federated social recommendation with graph neural network. *ACM Transactions on Intelligent Systems and Technology (TIST)* 13, 4 (2022), 1–24.

[21] Lingjuan Lyu, Han Yu, and Qiang Yang. 2020. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133* (2020).

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[23] Andriy Mnih and Russ R Salakhutdinov. 2007. Probabilistic matrix factorization. *Advances in neural information processing systems* 20 (2007).

[24] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragos, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. 2020. Fedfast: Going beyond average for faster training of federated recommender systems. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1234–1242.

[25] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*. IEEE, 739–753.

[26] Quoc Viet Hung Nguyen, Chi Thang Duong, Thanh Tam Nguyen, Matthias Weidlich, Karl Aberer, Hongzhi Yin, and Xiaofang Zhou. 2017. Argument discovery via crowdsourcing. *The VLDB Journal* 26 (2017), 511–535.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[28] Nuria Rodríguez-Barroso, Daniel Jiménez López, M Victoria Luzón, Francisco Herrera, and Eugenio Martínez-Cámara. 2022. Survey on federated learning threats: concepts, taxonomy on attacks and defences, experimental study and challenges. *Information Fusion* (2022).

[29] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks* 20, 1 (2008), 61–80.

[30] Anshuman Suri, Pallika Kanani, Virendra J Marathe, and Daniel W Peterson. 2022. Subject Membership Inference Attacks in Federated Learning. *arXiv preprint arXiv:2206.03317* (2022).

[31] Huynh Thanh Trung, Tong Van Vinh, Nguyen Thanh Tam, Hongzhi Yin, Matthias Weidlich, and Nguyen Quoc Viet Hung. 2020. Adaptive network alignment with unsupervised and multi-order convolutional networks. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 85–96.

[32] Paul Voigt and Axel Von dem Bussche. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10, 3152676 (2017), 10–5555.

[33] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. 2019. Collecting and analyzing multidimensional data with local differential privacy. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 638–649.

[34] Qinyong Wang, Hongzhi Yin, Tong Chen, Zi Huang, Hao Wang, Yanchang Zhao, and Nguyen Quoc Viet Hung. 2020. Next point-of-interest recommendation on resource-constrained mobile devices. In *Proceedings of the Web conference 2020*. 906–916.

[35] Qinyong Wang, Hongzhi Yin, Tong Chen, Junliang Yu, Alexander Zhou, and Xiangliang Zhang. 2022. Fast-adapting and privacy-preserving federated recommender system. *The VLDB Journal* 31, 5 (2022), 877–896.

[36] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2512–2520.

[37] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3454–3469.

[38] Chuhan Wu, Fangzhao Wu, Yongfeng Huang, and Xing Xie. 2021. Personalized news recommendation: A survey. *arXiv preprint arXiv:2106.08934* (2021).

[39] Liu Yang, Ben Tan, Vincent W Zheng, Kai Chen, and Qiang Yang. 2020. Federated recommendation systems. In *Federated Learning*. Springer, 225–239.

[40] Mengmeng Yang, Lingjuan Lyu, Jun Zhao, Tianqing Zhu, and Kwok-Yan Lam. 2020. Local differential privacy and its applications: A comprehensive survey. *arXiv preprint arXiv:2008.03686* (2020).

[41] Hongzhi Yin, Weiqing Wang, Hao Wang, Ling Chen, and Xiaofang Zhou. 2017. Spatial-aware hierarchical collaborative deep learning for POI recommendation. *IEEE Transactions on Knowledge and Data Engineering* 29, 11 (2017), 2537–2551.

[42] Wei Yuan, Hongzhi Yin, Fangzhao Wu, Shijie Zhang, Tieke He, and Hao Wang. 2022. Federated Unlearning for On-Device Recommendation. *arXiv preprint arXiv:2210.10958* (2022).

[43] Jingwen Zhang, Jiale Zhang, Junjun Chen, and Shui Yu. 2020. Gan enhanced membership inference: A passive local attack in federated learning. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 1–6.

[44] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

[45] Shijie Zhang and Hongzhi Yin. 2022. Comprehensive Privacy Analysis on Federated Recommender System against Attribute Inference Attacks. *arXiv preprint arXiv:2205.11857* (2022).

[46] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Lizhen Cui, and Xiangliang Zhang. 2021. Graph embedding for recommendation against attribute inference attacks. In *Proceedings of the Web Conference 2021*. 3002–3014.

[47] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Quoc Viet Hung Nguyen, and Lizhen Cui. 2022. Pipattack: Poisoning federated recommender systems for manipulating item promotion. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1415–1423.

[48] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. 2020. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 253–261.

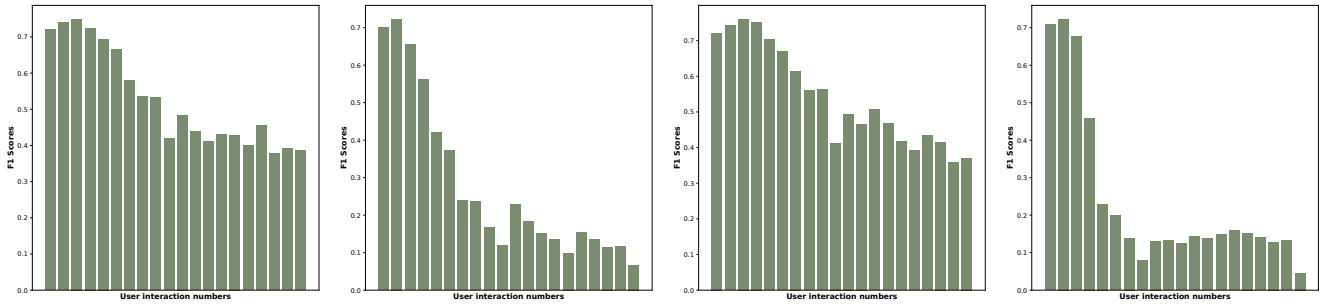
[49] Yanchao Zhao, Jiale Chen, Jiale Zhang, Zilu Yang, Huawei Tu, Hao Han, Kun Zhu, and Bing Chen. 2021. User-Level Membership Inference for Federated Learning in Wireless Network Environment. *Wireless Communications and Mobile Computing* 2021 (2021).

A DETAILS OF EMBEDDINGS DEVIATION

In Section 3.3, we present the trend of embeddings' deviation on MovieLens-100K. Here, we calculate all FedRecs' embeddings deviation from their converged point to the initial point using L2

Table 7: The average deviation (L2 loss) of embeddings from initial point to the converged model point.

		ML-100K		Steam-200K		Amazon	
	μ	Fed-NCF	Fed-LightGCN	Fed-NCF	Fed-LightGCN	Fed-NCF	Fed-LightGCN
User Embedding	0.0	0.0143	0.4258	0.0884	0.2666	0.0994	0.2252
	0.1	0.0816	0.5493	0.0932	0.3408	0.0996	0.2260
	1.0	0.1580	0.4481	0.0935	0.2935	0.0994	0.2031
Item Embedding	0.0	0.6088	0.6396	0.3144	0.2231	0.0667	0.0717
	0.1	0.0030	0.0653	0.0042	0.0303	0.0060	0.0313
	1.0	0.0004	0.0099	0.0005	0.0058	0.0009	0.0081



(a) F1 on MovieLens-100K with Fed-NCF. **(b) F1 on MovieLens-100K with Fed-LightGCN.** **(c) F1 on Steam-200K with Fed-NCF.** **(d) F1 on Steam-200K with Fed-LightGCN.**

Figure 5: IMIA attacker performance for users with different number of interactions.

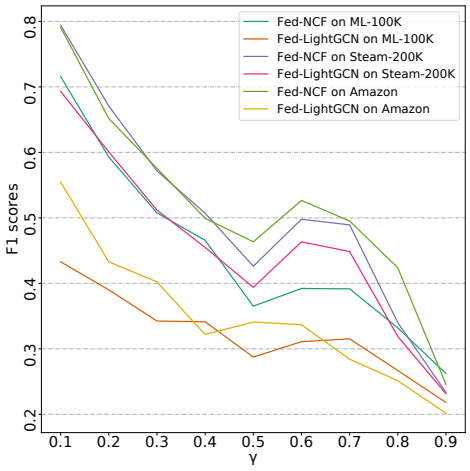


Figure 6: Our IMIA attacker's performance with different values of γ. 0.1 means selecting the top $10\% * |\mathcal{V}_i|$ items as correct guesses according to distance metrics each iteration.

loss. In Table 7, after applying our defense method, the deviation of item embedding is restrained, meanwhile, the user embedding is forced to update more. As a result, more information is encoded in private parameters, rather than in public parameters. Besides, across FedRecs, we can find that the updates of user embeddings are more significant in Fed-LightGCN than in Fed-NCF. This observation is consistent with our argument that "private parameters in Fed-LightGCN are more sufficiently used than in Fed-NCF".

B THE IMPACT OF INTERACTION NUMBER

Fig. 5 is an extension of Fig. 3. We cluster users into 20 groups based on their interaction numbers and report their average F1 score. Since users in Amazon Cell Phone all have fewer interactions, we only visualize the statistics of MovieLens-100K and Steam-200K. As shown in Fig. 5, users with fewer interactions are prone to leak more interaction information. This phenomenon is more obvious in Fed-LightGCN, because by using convolution aggregation, users with more interaction will have more complicated private embeddings, therefore, they are difficult to be attacked by solely relying on public parameters. This observation further implies that to prevent IMIA, we should improve the importance of private parameters.

B.1 The Impact of γ

The hyper-parameter γ denotes the percentage of items whose ratings the attacker is assumed to correctly infer at each iteration. Fig. 6 illustrates the trend of the attacker's performance with different γ on all datasets. Generally, with smaller γ, the attacker achieves better performance. For example, when $\gamma = 0.1$, the attacker achieves nearly 0.8 F1 scores on Fed-NCF and MovieLens-100K, however, when $\gamma = 0.9$, the performance is reduced to lower than 0.3. On the other hand, smaller γ needs more iterations to infer all the target user's interacted items. A desirable γ value should make a good balance between attack effectiveness and attack efficiency.