Privacy-Aware Recommendation with Private-Attribute Protection using Adversarial Learning

Ghazaleh Beigi, Ahmadreza Mosallanezhad, Ruocheng Guo, Hamidreza Alvari, Alexander Nou, Huan Liu
Computer Science and Engineering, Arizona State University, Tempe, Arizona
{gbeigi,amosalla,rguo12,halvari,asnou,huan.liu}@asu.edu

ABSTRACT

Recommendation is one of the critical applications that helps users find information relevant to their interests. However, a malicious attacker can infer users' private information via recommendations. Prior work obfuscates user-item data before sharing it with recommendation system. This approach does not explicitly address the quality of recommendation while performing data obfuscation. Moreover, it cannot protect users against private-attribute inference attacks based on recommendations. This work is the first attempt to build a *R*ecommendation with *A*ttribute *P*rotection (RAP) model which simultaneously recommends relevant items and counters private-attribute inference attacks. The key idea of our approach is to formulate this problem as an adversarial learning problem with two main components: the private attribute inference attacker, and the Bayesian personalized recommender. The attacker seeks to infer users' private-attribute information according to their items list and recommendations. The recommender aims to extract users' interests while employing the attacker to regularize the recommendation process. Experiments show that the proposed model both preserves the quality of recommendation service and protects users against private-attribute inference attacks.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Security and privacy** → **Social network security and privacy**; **Privacy protections**.

KEYWORDS

Privacy-Aware Recommendation; Private-Attribute Protection; Adversarial Learning; Privacy; Utility

ACM Reference Format:
Ghazaleh Beigi, Ahmadreza Mosallanezhad, Ruocheng Guo, Hamidreza Alvari, Alexander Nou, Huan Liu. 2020. Privacy-Aware Recommendation with Private-Attribute Protection using Adversarial Learning. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20), February 3–7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Recommendation systems play an important role in helping users find relevant and reliable information that is of potential interest [29]. These systems build profiles that represent user's interests [8, 28] and recommend relevant items to the users based on the constructed profiles [40]. Despite the effectiveness of recommendation systems, they can be sources of user privacy breach. Existing work has shown that if malicious attackers have access to the system's output and unrestricted auxiliary information about their targets, they are able to extract their entire user-item interactions history [7, 12, 33, 39]. One main reason is that recommendation systems' outputs (i.e., product recommendation) are partially derived from other users' choices (i.e., user-item interactions history). Thus, privacy concerns arise.

One of privacy issues is the re-identification attack where a malicious adversary attempts to infer user's actual ratings by seeking if a target user is in the database [7]. Prior research on privacy preserving recommendation systems has extensively addressed this type of privacy breach. Common techniques include (1) modifying the output of the recommendation system algorithm so that the absence or presence of a single rating or an entire user data is masked (i.e., differential privacy based techniques) [23, 33, 45]; and (2) coarsening the user's interactions history by adding dummy items and ratings such that the adversary cannot deduce the user's actual ratings and preferences (i.e., perturbation based techniques) [32, 38, 41].

Another privacy issue is the disclosure of user private-attribute information through leaked users' interactions history [11, 43]. Private attribute information contains those attributes that users do not wish to disclose such as age, gender, occupation and location. This type of privacy breach is known as the private-attribute inference attack in which the adversary's goal is to infer private attributes of target users given their interactions history. Little has been done to protect users against this attack of private-attribute inference [10, 11, 24, 43] with focus on anonymizing user-item data before publishing it. Data obfuscation comes at the cost of utility loss where utility is defined as the quality of service users receive. The existing work addresses the utility loss by minimizing the amount of changes made to the data [24, 43]. However, in the context of recommendation, the utility loss due to this approach can lead to degraded recommendation results. Moreover, just sharing perfectly obfuscated user-item data with a recommendation system does not necessarily prevent the adversary from inferring users' private information in future when they receive and accept new recommendations (e.g., when purchasing new products).

This research aims to devise a mechanism to counter private-attribute inference attacks in the context of recommendation systems. We propose a privacy-aware *R*ecommender with *A*ttribute

*P*rotection, namely RAP, which offers relevant products in a way that makes any inference of user's private attributes difficult from his interactions history and recommendations. The proposed model seeks to concurrently prevent the leakage of users' private attribute information while retaining high utility for users.

Recommendation while countering private-attribute inference attack can be naturally formulated as a problem of adversarial learning [19]. In our proposed RAP, there are two components: a Bayesian personalized ranking recommender and a private-attribute inference attacker (illustrated in Figure. 1). The private-attribute inference attacker seeks to accurately infer users' private attribute information. The attacker aims to iteratively adapt its model with respect to the existing recommender. The recommender extracts latent representations of users and items for personalized recommendation, and simultaneously utilizes the private-attribute inference attacker to regularize the recommendation process by incorporating necessary constraints to fool the attacker. Therefore, RAP optimizes a composition of two conflicting objectives, modeled as a min-max game between recommender and attacker components. Its objective is to recommend relevant, ranked items to users such that a potential adversary cannot infer their private attribute information.

In essence, we investigate the following research issues: (1) whether we can develop a personalized privacy-aware recommendation system to guard against private-attribute inference attacks; and (2) how we can ensure that the user's private attributes are effectively obscured after receiving personalized recommendation. Our research on these issues results in a novel framework RAP with the following main contributions:

- To the best of our knowledge, this is the first effort in proposing a recommendation system with guarding against the inference of private attribute information while maintaining the user utility.
- The proposed RAP model uses an attacker component that regularizes the recommendation process to protect users against private-attribute inference attack.
- The proposed RAP model is a general framework for recommendation systems. Both of the integrated Bayesian personalized recommender and the private-attribute attacker can be easily replaced by different models designed for specific tasks.
- We conduct experiments on real-world data to demonstrate the effectiveness of RAP. Our empirical results show that RAP preserves user utility and privacy. The results demonstrates that RAP outperforms the state-of-the-art related work and enables an adjustable balance between private-attribute protection and personalized recommendation.

2 PROBLEM STATEMENT

Before formally defining our problem, we first describe the notations used in this paper. Let $\mathcal{I} = \{i_1, ..., i_M\}$ denotes items, and $\mathcal{U} = \{u_1, ..., u_N\}$ denotes users. Also, $\mathcal{I}_h$ represents the set of items rated by user h, and $\mathcal{R}_h$ is set of items recommended to h. $\mathcal{P} = \{p_1, ..., p_T\}$ denotes a set of T private attributes (e.g., age, gender). $\mathbf{R}$ represents user-item rating matrix. The goal is to recommend products to people that would be interesting for them. However, we want to protect people's privacy against a malicious adversary who attempts to infer their private attribute information according to the user's list of items information. Items list $\mathcal{S}_h$ for each user

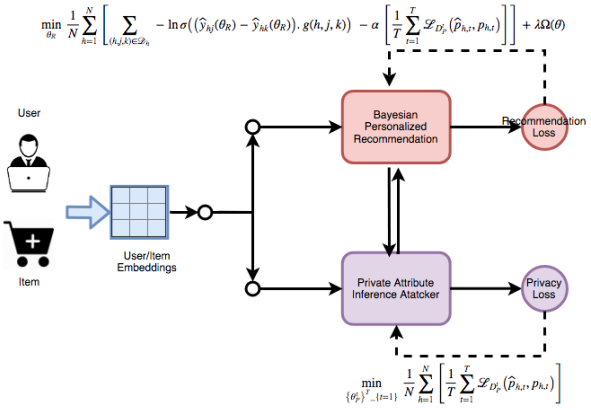


Figure 1: The architecture of Recommendation with Protection (RAP) with two components: a Bayesian personalized recommender and a private-attribute inference attacker.

h is union of his previously rated and newly recommended items, i.e., $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$. In particular, the malicious attacker has a framework which takes a target user's interactions and infers the user's private attribute:

Problem 1. *We aim to learn a function f that can recommend interesting and relevant products $\mathcal{R}_h$ to each user u_h such that, 1) the adversary cannot infer the targeted user's private attribute information $\mathcal{P}$ from the user's list of items information, $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$ and 2) the set of recommended items $\mathcal{R}_h$ is interesting for the user. The problem can be formally defined as: $\mathcal{R}_h = f(\mathcal{I}_h, \mathbf{R}, \mathcal{P})$*

Note that, the goal is to protect users against a malicious adversary who has access to the users' items list, but not against the recommender which is trusted.

3 RELATED WORK

Explosive growth of the Web has raised numerous challenges for online users including disinformation spread [1–4] and threats to users' privacy [7, 9]. Addressing user privacy issues has been studied from different aspects such as textual information [10, 11], web browsing histories [6], private-attributes disclosure [11, 24] and recommendation systems [32, 45] (for a comprehensive survey refer to [7]). Our work is related to a number of research which we discuss below while we elaborate on the differences between our work and them.

Privacy and Recommendation Systems. Existing privacy preserving works in recommendation systems focus on protecting users against re-identification attacks in which an adversary tries to infer a targeted user's actual ratings and investigate if the target is in the database. They could be categorized into differential privacy based [23, 26, 33, 45] and perturbation based [32, 38, 41] approaches. Some methods utilize differential privacy strategy [14] to modify the answers of the recommendation algorithm so the the presence of a user's data (either a single user-item rating or entire user's history) is masked by increasing the chance that two arbitrary records have close probabilities to generate the same noisy data. McSherry et al. [33] utilize differential privacy to construct private covariance

matrices to be further used by recommender. Another work [26] clusters users w.r.t. the social relations and generates differentially private average of users' preferences in each cluster. Hua et al. [23] propose a private matrix factorization which adds noise to item vectors to make them differentially private. Bassily et al. [5] modify user-item ratings data to satisfy differential privacy and then share it with recommender. Another work [45] makes items list differentially private and then sends it to recommender. Perturbation based techniques obfuscate user's interactions history by adding fake items and ratings to it. Rebollo et al. [41] propose an information theoretic based privacy metric and then find the obfuscation rate for generating forged user profiles so that the privacy risk is minimized. Similarly, [37] proposes to add or remove items and ratings from user profiles minimize privacy risk. Polat et al. [38] use a randomized perturbation technique by sharing disguised z-score for items a given user have rated. In another work [32], similar users are grouped to each other. Aggregated ratings of the users within the same group is then used to estimate a group preference vector. Similar to [38], randomness is then added to the preference vector to be shared with the recommender.

Attribute Inference Attacks and Defenses Private-attribute inference attack focuses on inferring users' private attribute information from their publicly available information. These attacks could be categorized into three groups.A group of these attacks leverages a target user's friends' information [18, 21, 31] and community membership information [34, 44] to infer target's private attributes. Second group of these attacks are those works which leverage users' behavioral information such as movie-rating behavior [43] and Facebook likes [30] to infer their private attribute information. The third group of works exploits both friend and behavioral information [16, 17, 25]. Gong et al. [16, 17] make a social-behavior-attribute network in which all users' behavioral and friendship information is integrated in a unified framework. Private attributes are then inferred through a vote distribution attack model. Another work [25] incorporates structural and behavioral information from users who do not have the attribute in the training process, i.e. negative training samples.

Little work focuses on protecting users against private-attribute inference attacks [24, 43]. In [43], a predefined number of dummy items is added to each user's profile which are negatively correlated with his actual attributes before publishing anonymized user-item ratings data. In a recent paper [24], after a value is sampled for the given private attribute w.r.t. a certain probability distribution which is different from the user's actual attribute, the minimum noise is found and added to the user-item data via adapting evasion attacks such that the malicious attacker predicts the sampled attribute value as the user's private attributes.

Our work is different from the existing works. First, existing privacy preserving recommendation systems do not specifically target the private-attribute inference attacks. Second, existing defenses against this attack [24, 43] address the utility loss by minimizing the amount of changes made to the data. However, in scope of recommendation systems, this approach can mean neglecting the quality of received services, i.e., poor recommendation results. Third, sharing anonymized data with recommender does not preclude the malicious attacker to infer private attribute information when users receive new recommendations. All of these limitations arises the need for having a recommendation system guarding against the inference of private attribute while maintaining the user utility.

4 RECOMMENDATION WITH ATTRIBUTE PROTECTION (RAP)

Our proposed recommendation framework, RAP, aims to concurrently recommend interesting items to users and protect them against private attribute leakage. The entire model is illustrated in Figure. 1. This framework consists of two major components, 1) a Bayesian personalized recommender, and 2) a private-attribute inference attacker. The personalized ranking recommender D_R aims to extract users' actual preferences and recommend relevant items to them. The private-attribute inference attacker D_P seeks to develop a model which can deduce users' private information w.r.t. the existing recommendation system. Recommendation component then utilizes D_P to guide the recommendation process by ensuring that the union of previously rated and newly recommended items does not leak user's attributes and further fools the adversary in D_P. Inspired by adversarial machine learning, we model this objective as a min-max game between two components, i.e. attacker D_P seeks to maximize its gain and recommender D_R aims to minimize both its recommendation loss and attacker D_P's gain. The final output of RAP for each user, is a list of top-K items which are interesting yet safe for them.

4.1 Bayesian Personalized Recommendation

In this section, we propose a new Bayesian personalized recommendation model. The proposed model structure is shown in Fig. 2. This model first extracts users and items latent embeddings and then utilizes learning to rank approach to recommend items to users.

Learning to rank methods have been introduced to optimize recommendation systems toward personalized ranking. Inspired by recent success of Bayesian Personalized Ranking (BPR) [42] in image and friend recommendation systems [13, 35], we choose BPR aver other approaches. The idea behind BPR is that observed user-item interactions should be ranked higher than unobserved ones. Learning from implicit feedback, BPR goal is to maximize the margin between an observed user-item interaction and its unobserved counterparts. In particular, BPR behavior could be interpreted as a classifier in which given a positive triplet instance of user h and items j and k, (h, j, k), it determines whether the user-item interaction (h, j) should have a higher rank score than (h, k).

This recomemndation component has three inputs, the user h and items j and k.We denote the user and items indices by a tuple of vectors (u_h, i_j, i_k) which are one-hot encodings of users and items. Since there are N users and M items, the dimensions of u_h, i_j, and i_k are M, N and N, respectively. Following the input layer, each input layer is fully connected to the corresponding embedding layer to learn the latent representation of the users and items, $\mathbf{q}_h \in \mathbb{R}^d$, $\mathbf{p}_j \in \mathbb{R}^d$, where d is the number of dimensions. The embedding dimensions for both users and items are the same:

$$\mathbf{q}_h = \mathbf{W}_h u_h,\ \mathbf{p}_j = \mathbf{W}_j i_j,\ \mathbf{p}_k = \mathbf{W}_k i_k \tag{1}$$

where $\mathbf{W}_h$, $\mathbf{W}_j$ and $\mathbf{W}_k$ are embedding matrices for users and items. In the next layer, user and item embedding vectors are passed to the hidden layers H_h, H_j, and H_k for further calculations. For example,

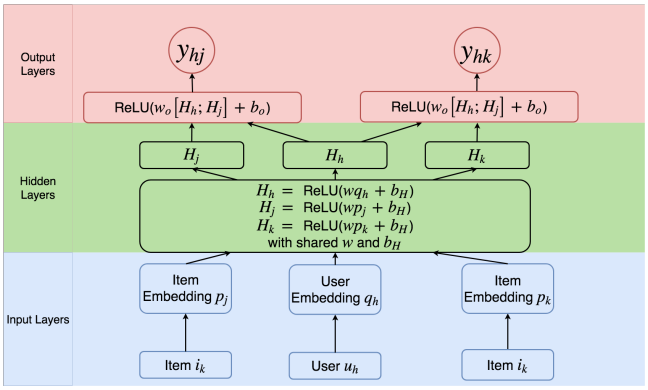


Figure 2: Overview of the Bayesian personalized recommendation component.

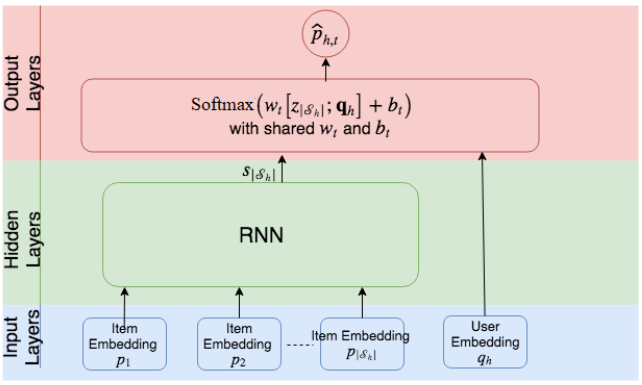


Figure 3: Overview of the private-attribute inference attacker component for one attribute.

the hidden layer produces H_h for user h as:

$$H_h = ReLU(wq_h + b_H) \tag{2}$$

where $ReLU$ is simply defined as $ReLU(x) = \max(0, x)$ and w and b_H are the weights and bias for units, respectively.

Using H_h, H_j, and H_k, the next layer produces the user's preference $\hat{y}_{hj}$, $\hat{y}_{hk}$ toward items j and k, respectively. For example:

$$\hat{y}_{hj} = ReLU(w_o[H_h; H_j] + b_o) \tag{3}$$

where b_o is the bias parameter in the output layer. The activation function is $RelU$ function and $[.; .]$ represents concatenation. Note that due to the model simplicity, all users share the same latent representation learning parameters $\{w, b_H\}$ and $\{w_o, b_o\}$ in the hidden layer and output layer, respectively.

We use BPR to learn how to rank in the problem of recommendation. The final objective function is to minimize the following loss function w.r.t. θ_R:

$$\mathcal{L}_{D_R} = \frac{1}{N}\sum_{h=1}^{N}\sum_{(h,j,k)\in\mathcal{D}_h} -\ln\delta((\hat{y}_{hj}(\theta_R) - \hat{y}_{hk}(\theta_R)).g(h,j,k)) + \lambda_{\theta_R}\|\theta_R\|^2 \tag{4}$$

where, $g(h, j, k)$ is the ground truth value for our model training:

$$g(h,j,k) = \begin{cases} 1, & \text{if user } u_h \text{ prefers item } i_j \text{ over item } i_k \\ -1, & \text{otherwise} \end{cases} \tag{5}$$

where set $\mathcal{D}_h = \{(h, j, k)|j \in \mathcal{I}_h \text{ and } k \in \mathcal{I} / \mathcal{I}_h\}$ also denotes the training pairwise instances in which $\mathcal{I}$ and $\mathcal{I}_h$ represent the whole set of items and the set of items rated by user u, respectively. Moreover, y_{hj} is the actual rating that user h gives to item j. θ_R is also defined as $\theta_R = \{\mathcal{W}_\mathcal{U}, \mathcal{W}_\mathcal{I}, w, b_H, w_o, b_o\}$ such that $\mathcal{W}_\mathcal{U} = \{\mathbf{W}_1, .., \mathbf{W}_N\}$ and $\mathcal{W}_\mathcal{I} = \{\mathbf{W}_1, .., \mathbf{W}_M\}$ represent the set of embedding matrices for N users and M items, respectively. The proposed model considers the recommendation problem as a binary classification problem to ensure that the pairwise preference relations hold.

After training the recommendation model, given a user h, for every item j that the user has not rated, i.e., $j \in \{\mathcal{I} / \mathcal{I}_h\}$, his preference score $\hat{y}_{hj}$ is predicted by the recommender. In order to calculate the preference score $\hat{y}_{hj}$, we pass the tuple (h, j, j) to the recommender, and get $\hat{y}_{hj}$ and $\hat{y'}_{hj}$ as the model's output. The final preference score of user h toward item j is calculated as $\hat{y}_{hj} = 0.5(\hat{y}_{hj} + \hat{y'}_{hj})$. All of the unrated items will be then sorted based on their preference scores descendingly and the top-K items are then returned as the recommendation $\mathcal{R}_h$ to the user.

4.2 Training an Attacker against Inferring Private Attribute Information

The goal of our model is to recommend ranked items to users such that any potential adversary cannot infer users' private attribute information such as age, gender and occupation. However, a challenge is that the recommendation system does not know the malicious attacker's model. To address this challenge, we add a private-attribute inference attacker D_P component to our model which seeks to learn a classifier that can accurately identify the private information of users from their previous interactions. Then, we leverage this component to regularize the recommendation process by incorporating necessary constraints in order to fool the adversary D_P and further avoid the leakage of private attributes after recommendation. This part is discussed in details in Section. 4.3.

The goal of the private-attribute attacker is now to predict target user h's private attribute information by leveraging the information of his latent representation as well as the the latent representation of his items list. The user h's items list $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$ includes both items $\mathcal{I}_h$ that user has rated previously and new recommended items $\mathcal{R}_h$. Given T private attributes (e.g., age, gender), the set of $\{\theta_{P^t}\}_{t=1}^{T}$ represents all the parameters included in the private-attribute inference attacker component D_P. The output of the private attribute attacker component for user h w.r.t. t-th private attribute is the probability that user h has t-th attribute.

We use $p_{h,t}$ to represent the actual value for user h's t-th private attribute. The structure of private attribute inference attacker is represented in Fig.3. The input to this model for each user h is the latent embedding representations of each item p_j in his items list $\mathbf{p}_j \in \mathcal{S}_h, j = 1, 2, ..., |\mathcal{S}_h|$ and h's latent embedding representation $\mathbf{q}_h$. Given the input, the items embeddings are passed to a single-layer recurrent neural network (RNN) and the output of RNN ($z_{|\mathcal{S}_h|}$) is then concatenated with user's embedding. The last layer produces

the predicted t-th sensitive attribute for user h, $\hat{p}_{h,t}$:

$$\hat{p}_{h,t} = softmax(w_t[z_{|\mathcal{S}_h|}; \mathbf{q}_h] + b_t) \tag{6}$$

where $[.;.]$ represents concatenation. Also, w_t and b_t are the weights and bias for units, respectively and are shared among all users due to the model simplicity. We then minimize the private-attribute inference attacker component loss function $\mathcal{L}_{D_P^t}$ for all private attributes by seeking the optimal parameters $\{\theta_P^t\}_{t=1}^T$. The objective function for all users can be formally written as follows:

$$\mathcal{L}_{D_P} = \frac{1}{N}\sum_{h=1}^{N}\left[\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{D_P^t}(\hat{p}_{h,t}, p_{h,t})\right] \tag{7}$$

where $\mathcal{L}_{D_P^t}$ denotes the cross entropy loss for t-th private attribute.

4.3 Adversarial Learning for Recommendation with Private-Attribute Protection

Thus far, we have discussed how we 1) learn users and items representations to recommend ranked items to each user based on his personalized preferences; and 2) train an attacker which can accurately infer a target user's private attribute information given a list of his rated items and received recommendations. We stress that the adversary always has the upper hand and adapts his private-attribute inference attack in order to minimize his inference loss w.r.t. the existing recommendation system. The final objective is thus to recommend relevant ranked items to users such that a potential adversary cannot infer their private attribute information. To achieve two goals together, we design an optimization problem to minimize the recommendation loss of our model *and* maximize the inference loss of a determined attacker who adaptively minimizes his loss. Inspired by the idea of adversarial learning, we model this optimization as a min-max game between two components, Bayesian personalized recommender and private-attribute attacker.

In our proposed model, the adversary tries to adapt itself and gets the maximum gain, while the recommendation system seeks to recommend ranked items to users. The recommended items not only align well with the users' preferences, but also minimize the adversary's gain. We reformulate the objective function of the recommendation system as minimizing attacker's gain and recommendation loss simultaneously:

$$\underbrace{\min_{\theta_R}\left(\mathcal{L}_{D_R} \overbrace{-\alpha \max_{\{\theta_P^t\}_{t=1}^T}\mathcal{L}_{D_P}}^{\text{private-attribute attacker}}\right)}_{\text{privacy-aware recommendation system}} \tag{8}$$

The inner part learns the most determined adversary which adaptively minimizes its loss regarding private-attribute inference given the users and items information. The outer part seeks to both minimize the recommendation loss and fool the given adversary. The parameter α controls the contribution of the private-attribute inference attacker in the learning process. Objective function in Eq. 8 can be written as follows:

$$\min_{\theta_R}\max_{\{\theta_P^t\}_{t=1}^T}\left(\frac{1}{N}\sum_{h=1}^{N}\left[\sum_{(h,j,k)\in\mathcal{D}_h} -\ln\delta((\hat{y}_{hj}(\theta_R) - \hat{y}_{hk}(\theta_R)).g(h,j,k))\right.\right. \tag{9}$$

$$\left.\left. -\alpha\left[\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{D_P^t}(\hat{p}_{h,t}, p_{h,t})\right]\right] + \lambda\Omega(\theta)\right)$$

where $\theta = \{\theta_R, \{\theta_P^t\}_{t=1}^T\}$ is the set of all parameters to be learned, $\Omega(\theta)$ is the L_2 norm regularizer on the parameters, and λ is a scalar to control the contribution of the regularization $\Omega(\theta)$.

Algorithm 1 The Learning Process of RAP model

Input: Items set $\mathcal{I}$, training user data $\mathcal{U}$, training user-item matrix data $\mathbf{R}$, batch size b, θ_R, $\{\theta_P^t\}_{t=1}^T$, α, λ and K.
Output: Trained recommendation with protection RAP.

1: **repeat**
2: Create a mini-batch $\mathcal{U}_b$ of b users with their private-attribute and item-rating information from $\mathcal{U}$
3: Train the recommendation with attribute protection via Eq. 10 w.r.t. θ_R
4: For each user h in $\mathcal{U}_b$, calculate the top-K recommended items $\mathcal{R}_h$
5: Train the private-attribute inference attacker D_P (i.e., $\{\theta_P^t\}_{t=1}^T$) via Eq. 7 given the users' information including their list of items information, i.e., $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$
6: **until** Convergence

4.4 Optimization Algorithm

The optimization process is illustrated in Algorithm 1. First, we create a mini-batch sample $\mathcal{U}_b$ of b users from the training data and serve their private attribute and item-rating information to the model. Next, we train the Bayesian personalized recommender D_R according to the Eq. 10 w.r.t. θ_R in Line 3. Then, for each user h in $\mathcal{U}_b$ we calculate the top-K recommended items $\mathcal{R}_h$ and accordingly make his list of items, $\mathcal{S}_h$. The private-attribute inference attacker component is then trained according to the users and item embeddings information using Eq. 7 in Line 5. After training RAP, for each user h, a list of top-K items $\mathcal{R}_h$ will be returned as recommendation.

5 EXPERIMENTS

In this section we conduct experiments to evaluate the efficiency of the proposed framework in terms of both privacy and quality of the recommendation. We aim to answer the following questions:

- **Q1** - *Privacy*: How does RAP perform in preventing leakage of users' private information?
- **Q2** - *Utility*: How does RAP perform in recommending relevant items to users?
- **Q3** - *Utility-Privacy Relation*: Does the improvement in privacy result in sacrificing the utility of recommendation system?

To answer the first question (**Q1**), we examine our model against different private information with different distributions, such as age, gender, and occupation. Then, we evaluate the effectiveness of RAP in preventing leakage of users' private information given

union of users' previously rated and newly recommended items. Addressing leakage of private attribute information may result in recommendation performance deterioration. Therefore, to answer the second question (**Q2**), we examine the performance of RAP in terms of the quality of the recommendation. Finally, to answer the third question (**Q3**), we investigate the loss in recommendation performance when enhancing privacy of users.

5.1 Data

We use publicly available data MovieLens [20]. This dataset includes 100,000 ratings by 943 users on 1,682 movies. Each user has rated at least 20 movies and the rating scores are between 1 and 5. In the collected dataset, each user is associated with three private attributes, gender (male/female), age, and occupation. For this paper, we follow the setting of [22] and categorize age attribute into three groups, over-45, under-35, and between 35 and 45. In total, 21 possible occupations have been considered for this data. The average number of rated items for each user is 129.

5.2 Experimental Setting

Here, we first explain how we design experiments to evaluate utility and privacy. Then, we discuss evaluation metrics and baselines.

Implementation Details: The parameters for recommendation and attacker components are determined through grid search. For the Bayesian personalized ranking recommendation component, we set the dimension of first layer as $d = 70$. Accordingly, size of user and item embedding vectors is $d = 70$. The dimension of hidden layer is also set as 20. For the private-attribute inference attacker component, we use single layer RNN with the dimension of input layer set as $d = 70$. User and item embeddings are then passed from recommendation component to the attacker component. The dimension of hidden layer is set as 100. The parameters α and λ are also determined through cross-validation, $\alpha = 1$ and $\lambda = 0.01$.

We initialize the weight matrices in both components with random values uniformly distributed in $[0, 1]$. The error gradient is back propagated from output to input and parameters in each layer are updated. The optimization algorithm used for gradient update is Adam's algorithm [27]. The loss generally converges after 20 epochs. The batch size we use in experiments is $b = 32$.

Recommendation Evaluation: We evaluate the performance of recommendation by examining the quality of recommended items for all users. We follow the setting of [24] to set-up the experimental settings. To do so, we split the data for train and test as follows. For each user h in the data, we randomly select l rated items for test set and the remaining $n_h - l$ items for training set, where n_h is the number of rated items for user h. We set the item rating for those in the test set as zero. We vary the value of l as $\{35, 40, 45\}$. Then, the top-K items are then returned to each user as the recommendation. Note that we assume RAP has access to the users' private attribute information during the training process.

Private-Attribute Evaluation: We evaluate privacy of users in terms of their robustness against the malicious attribute inference attacks in which the adversary's goal is to infer users' private attributes. In particular, the malicious attacker learns a multi-class classifier which takes a target user h list of items information, i.e. $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$, where $\mathcal{I}_h$ is set of h's rated items and $\mathcal{R}_h$ is set of items recommended to h. The adversary then infers the user's private attributes, i.e., gender, age, and occupation.

We use a Neural Network (NN) model as the adversary's classifier. Note that RAP is not aware of the adversary's model. In this attack, the adversary deploys a feed-forward network with a single hidden layer to perform the attack. The input to this model is one-hot encoding of each user, $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$. Since there are M items in the dataset, the dimension of input vector is M. The input layer is then fully connected to the hidden layer with dimension of hidden state set as 100 and a $softmax$ layer used as the output layer. The dimension of the hidden layer is determined through grid search. We note that Gong et al. [36] also proposed an attribute inference attack which leverages both social friends and rating behavior. However, their attack is not applicable to our problem as we focus on leveraging only user-item rating information.

We follow the setting of [24] to set-up the experiments. We split the data to train and test sets by sampling 80% of the users in the dataset uniformly at random as the training set and use the remaining users as testing set. We assume that the users in the training set has publicly disclosed their private information while the users in the testing set keep those attribute information private. Then, for each user in the test set, we randomly select l rated items and remove them from the user's rating history by setting the their rating as zero. We keep the user-item ratings for users in the training set intact (i.e., original user-item ratings). Trained RAP model is deployed on the users in the test set and top-l recommended items $\mathcal{R}_h$ are added to the users' previously rated items $\mathcal{I}_h$, in order to make $\mathcal{S}_h = \{\mathcal{I}_h \cup \mathcal{R}_h\}$. We vary value of l as $\{35, 40, 45\}$.

The adversary's classifier is trained on the training set and evaluated on the users in the test set. Note that we assume that the malicious attacker knows the original intact user-item interactions for those users in the training set and seeks to predict private attribute information of the users in the test set, given their $\mathcal{S}_h$. We evaluate a malicious attack for each private attribute.

Evaluation Metrics: We use the following metrics for evaluating RAP performance w.r.t. malicious private-attribute inference (i.e., privacy) and product recommendation (i.e., utility):

- **Private-Attribute Evaluation:** Since distribution of data for different private attribute values is imbalance, we report micro-AUC [15] of the adversary's classifier. Micro-AUC [15] gives a more accurate assessment. Lower AUC demonstrates higher privacy in terms of obscuring private attributes.
- **Recommendation Evaluation:** We use standard metrics that are widely used in other related works [46], i.e., $P@K$ and $R@K$. $P@K$: $P@K$ represents the ratio of test cases which has been successfully recommended in a top-K position in a ranking list to value of K. For each user, we measure $P@K$ as:

$$P@K = \frac{|\{\text{test items}\} \cap \{\text{top-}K \text{ returned items}\}|}{K} \tag{10}$$

$R@K$: $R@K$ defines the ratio of top-K recommended items which are in the test set to the number of items to be recommended in the test. For each user in the data, we measure $R@K$ as follows:

$$R@K = \frac{|\{\text{test items}\} \cap \{\text{top-}K \text{ returned items}\}|}{|\{\text{test items}\}|} \tag{11}$$

We then report the average of $R@K$ and $P@K$ for all users in the dataset and set the number of returned items as $K = 35$.

Model	# test items (l)														
	35					40					45				
	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$
Original	0.7662	0.7050	0.8332	0.156	0.156	0.7662	0.7050	0.8332	0.151	0.172	0.7662	0.7050	0.8332	0.145	0.187
LDP-SH	0.6587	0.6875	0.8076	0.071	0.071	0.6440	0.6777	0.7954	0.062	0.078	0.6398	0.6732	0.7817	0.055	0.081
BlurMe	0.6266	0.6177	0.7614	0.118	0.118	0.6013	0.5949	0.7589	0.109	0.134	0.5884	0.5901	0.7522	0.099	0.150
RAP	**0.6039**	**0.5397**	**0.7319**	**0.152**	**0.152**	**0.5714**	**0.5270**	**0.7315**	**0.147**	**0.168**	**0.5278**	**0.5262**	**0.7312**	**0.142**	**0.183**

Table 1: RAP Performance. Higher $P@K$ and $R@K$ values show higher utility, while lower AUC indicates higher privacy.

Baseline Methods: We compare RAP with the following baselines:

- **Original**: This baseline is a variant of RAP which recommends items for each user without incorporating the private-attribute inference attacker component, i.e., $\alpha = 0$.
- **LDP-SH [5]**: This method adds noise to user-item ratings based on ϵ-differential privacy. It requires categorical data which for our case, each user-item rating can be viewed as categorical data taking values $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$.
- **BlurMe [43]**: This method perturbs user-item ratings before sending to recommendation system. It adds new items to each user's profile that are negatively correlated with the user's actual private attributes and then adds the average rating score to those items. BlurMe needs to be deployed for each attribute separately.

To have a fair comparison between RAP and two baselines, we anonymze the user-item rating data w.r.t. baselines. The noisy manipulated data is used to train the recommendation model. We use matrix factorization model as the recommendation framework for both baselines. The discussed procedure is then used for evaluation.

5.3 Privacy Analysis (Q1)

The results against the malicious private-attribute inference attack (Section 5.2) are demonstrated in Table. 1. We observe that increasing the number of test items (l) results in decrease of AUC score for all frameworks. This is because for each target user h in the test set, l recommended items $\mathcal{R}_h$ have been added to user's item list $\mathcal{S}_h$. Therefore, increase of l can decrease the malicious attacker's chance for correctly inferring users' private attribute information. Moreover, RAP has significantly lower AUC score in comparison to Original for all three private attributes and thus outperforms Original in terms of obscuring users' private attribute information. RAP also has significantly better performance in hiding private information in comparison to LDB-SH. The reason is that LDB-SH aims to achieve a privacy goal that is different from preventing leakage of private information. This confirms that although adding noise and satisfying ϵ-differential privacy can indirectly benefit private attribute leakage, it does not directly target this problem. These results show the importance of private-attribute inference attacker component in obfuscating private information. We also observe that RAP hides more private information rather than BlurMe (lower AUC score). This demonstrates that providing obfuscated user-item rating data to the recommendation system, does not necessarily guarantee preventing future private attribute leakage when user receives (and accordingly buy) more recommended products. Moreover, BlurMe needs to be deployed for each private attribute separately while RAP considers three private attributes all together.

These results confirm the efficiency of RAP in obscuring users' private attribute information and demonstrate that despite the fact that RAP is not aware of the adversary's inference model, it is prepared against the malicious attacker.

5.4 Utility Analysis (Q2)

The results for recommendation task for different methods and different number of test items (l) are shown in Table. 1. We observe that increasing the number of test items (l) results in increasing $R@K$ and decreasing $P@K$ for all methods. Note that the higher the $P@K$ and $R@K$ score values are, the higher recommendation quality is. Another observation is that LDP-SH has the worst performance amongst all methods, i.e., lowest $P@K$ and $R@K$ scores. This is because of the way LDP-SH adds noise to the user data without considering the quality of recommendation service in practice which can result in degraded recommendation results. BlurMe has also lower performance than RAP as it neglects quality of recommendation results. These results confirm the effectiveness of Bayesian personalized recommendation component which helps RAP to take the utility into consideration in practice. Moreover, quality of recommendation results for RAP method is comparable to the Original approach. This means that RAP can accurately capture users' actual preferences and interests (i.e., high utility).

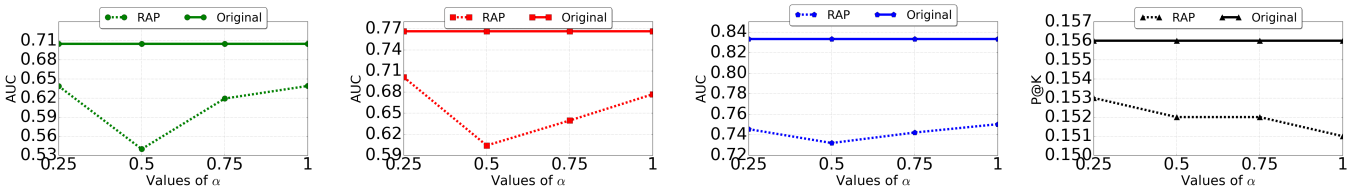
The results confirm the effectiveness of RAP in understanding users' actual preferences and recommending ranked relevant products that are interesting yet safe products to users.

5.5 Utility-Privacy Relation (Q3)

We compare the privacy and utility results in Table. 1 for all methods. We observe that LDP-SH has the worst results in terms of both preserving privacy and recommendation performance. Another observation is that BlurMe improves privacy compared to the Original method, but it loses utility in terms of recommendation system performance. This is in contrast with the results of RAP, which has outperformed BlurMe and LDP-SH in terms of recommendation and has comparable results with Original. RAP has also achieved the lowest AUC score and therefore highest privacy among all other methods. Comparing RAP with other methods confirms that approaching utility loss by minimizing the amount of data changes results in loss of quality of recommendation system in practice. This is reflected as degraded recommendation results for baseline approaches. Moreover, these results confirm the effectiveness of Bayesian personalized recommendation component in RAP, which helps us to consider quality of recommendation in practice. Results also demonstrate the complementary roles of both

Model	# test items (l)														
	35					40					45				
	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$	Gen	Age	Occ	$P@K$	$R@K$
RAP	0.6039	**0.5397**	**0.7319**	**0.152**	**0.152**	0.5714	**0.5270**	**0.7315**	**0.147**	**0.168**	0.5278	**0.5262**	**0.7312**	**0.142**	**0.183**
RAPAge	0.6450	0.5948	0.7528	0.150	0.150	0.5489	0.5938	0.7522	0.146	0.167	0.5475	0.5909	0.7497	0.141	0.182
RAPGen	**0.5332**	0.6789	0.7558	0.151	0.151	**0.5298**	0.6614	0.7556	0.145	0.166	**0.5211**	0.6415	0.7555	0.141	0.181
RAPOcc	0.6571	0.6949	0.7468	0.147	0.147	0.6485	0.6871	0.7466	0.141	0.161	0.6454	0.6853	0.7438	0.135	0.174

Table 2: Impact of different private-attribute attacker components on RAP in terms of utility and privacy.



(a) Attribute Age **(b) Attribute Gender** **(c) Attribute Occupation** **(d) Recommendation**

Figure 4: Performance results for private-attribute inference attack and recommendation task for different values of α

recommendation and private attribute components which guide each other through both privacy and utility issues. This results in a privacy-aware recommendation system which is prepared for private attribute inference attack and understands users' preferences.

5.6 Impact of Different Components

Here, we investigate the impact of different private attribute components on obscuring users' private information. We define three variants of our proposed framework, i.e., RAPAge, RAPGen, and RAPOcc. In each of these variants, the model is trained with the corresponding private-attribute inference attacker component, e.g. RAPAge is trained solely with age inference attacker component and does not utilize any other private-attribute attackers during training phase. Results are shown in Table. 2. We observe that for gender attribute, RAPGen has the best performance in terms of obscuring gender attribute comparing to the other approaches (i.e., lowest AUC score). This is in contrast to quality of RAPGen performance for recommendation task which is lower than original proposed model RAP. For other private attributes, RAP still outperforms RAPOcc and RAPAge in terms of obscuring age and occupation attributes. Moreover, results show that using one private-attribute attacker compromises the effectiveness model for obfuscating other private attributes. For the recommendation task, we surprisingly observe that using solely one of the private-attribute attackers in training process can result in performance reduction in comparison to RAP in terms of $P@K$ and $R@K$. This means that focusing merely on obscuring one private attribute can result in more recommendation performance degradation.

5.7 Probing Further

RAP has one important parameter α which controls the contribution from private-attribute attacker component. In this section, we probe further to investigate the effect of this parameter by varying it as $\{0.25, 0.5, 0.75, 1\}$. For this experiment, we set the number of test items $l = 35$. We also set the number of top-K returned items as $K = 35$ for calculating $P@K$. Note that $P@K$ and $R@K$ are equal in this scenario as $K = l = 35$. Results are shown in the Fig. 4.

Although α controls the contribution of private-attribute inference attacker component, we surprisingly observe that with the increase of α, the AUC score for attribute inference attack decreases at first up to the point that $\alpha = 0.5$ and then it increases. This means that private information were obscured more accurately at the beginning with the increase of α and less later. Moreover, with the increase of α, the performance of recommendation task decreases, i.e., lower $P@K$. This shows that increasing the contribution of private-attribute attacker component leads to decrease in the quality of recommendation framework. Another observation is that setting $\alpha = 0.25$ leads to improvement in hiding private attribute information in comparison to the results of using Original (or when $\alpha = 0$). This result shows the importance of the RAP's private-attribute attacker component in preserving privacy of users. Another observation is that after $\alpha = 0.5$, continuously increasing α increases the AUC for malicious private-attribute inference attack, i.e., degrades the performance of hiding private information. The reason is that the model could overfit by increasing the value of α and lead to an inaccurate estimation of privacy protection.

6 CONCLUSION

In this paper, we propose an adversarial learning-based recommendation with attribute protection model, RAP, which guards users against private-attribute inference attack while maintaining utility. RAP recommends interesting yet safe products to users such that a malicious attacker cannot infer their private attribute from users' interactions history and recommendations. RAP has two main components, Bayesian personalized recommender, and private-attribute inference attacker. Our empirical results show the effectiveness of RAP in both protecting users against private-attribute inference attacks and preserving quality of recommendation results. RAP also

consistently achieves better performance compared to the state-of-the-art related work. One extension to this work is to study the possibility of extending differential privacy mechanism for this type of attack in recommender systems. It would be also interesting to investigate personalized utility-privacy trade-off by tweaking framework parameters to fit the specific needs of individuals.

ACKNOWLEDGMENTS

This material is based upon the work supported, in part, by NSF #1614576, ARO W911NF-15-1-0328 and ONR N00014-17-1-2605.

REFERENCES

[1] Hamidreza Alvari, Soumajyoti Sarkar, and Paulo Shakarian. 2019. Detection of Violent Extremists in Social Media. In *2019 2nd International Conference on Data Intelligence and Security (ICDIS)*. 43–47.

[2] Hamidreza Alvari, Elham Shaabani, Soumajyoti Sarkar, Ghazaleh Beigi, and Paulo Shakarian. 2019. Less is More: Semi-Supervised Causal Inference for Detecting Pathogenic Users in Social Media. In *Companion Proceedings of The 2019 World Wide Web Conference*. ACM, 154–161.

[3] Hamidreza Alvari, Elham Shaabani, and Paulo Shakarian. 2018. Early identification of pathogenic social media accounts. In *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 169–174.

[4] Hamidreza Alvari and Paulo Shakarian. 2019. Hawkes Process for Understanding the Influence of Pathogenic Social Media Accounts. In *2019 2nd International Conference on Data Intelligence and Security (ICDIS)*. 36–42.

[5] Raef Bassily and Adam Smith. 2015. Local, private, efficient protocols for succinct histograms. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. ACM, 127–135.

[6] Ghazaleh Beigi, Ruocheng Guo, Alexander Nou, Yanchao Zhang, and Huan Liu. 2019. Protecting user privacy: An approach for untraceable web browsing history and unambiguous user profiles. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 213–221.

[7] Ghazaleh Beigi and Huan Liu. 2018. Privacy in social media: Identification, mitigation and applications. *arXiv preprint arXiv:1808.02191* (2018).

[8] Ghazaleh Beigi and Huan Liu. 2018. Similar but different: Exploiting usersâĂŹ congruity for recommendation systems. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer, 129–140.

[9] Ghazaleh Beigi and Huan Liu. 2019. Identifying novel privacy issues of online users on social media platforms by Ghazaleh Beigi and Huan Liu with Martin Vesely as coordinator. *ACM SIGWEB Newsletter* Winter (2019), 4.

[10] Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. 2019. I Am Not What I Write: Privacy Preserving Text Representation Learning. *arXiv preprint arXiv:1907.03189* (2019).

[11] Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. 2019. Privacy Preserving Text Representation Learning. In *Proceedings of the 30th ACM Conference on Hypertext and Social Media*. ACM, 275–276.

[12] Joseph A Calandrino, Ann Kilzer, Arvind Narayanan, Edward W Felten, and Vitaly Shmatikov. 2011. " You Might Also Like:" Privacy Risks of Collaborative Filtering. In *Security and Privacy (SP), 2011 IEEE Symposium on*. IEEE, 231–246.

[13] Daizong Ding, Mi Zhang, Shao-Yuan Li, Jie Tang, Xiaotie Chen, and Zhi-Hua Zhou. 2017. BayDNN: Friend Recommendation with Bayesian Personalized Ranking Deep Neural Network. In *Proceedings of the ACM CIKM*.

[14] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*. Springer, 1–19.

[15] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006), 861–874.

[16] Neil Zhenqiang Gong and Bin Liu. 2016. You Are Who You Know and How You Behave: Attribute Inference Attacks via Users' Social Friends and Behaviors.. In *USENIX Security Symposium*. 979–995.

[17] Neil Zhenqiang Gong and Bin Liu. 2018. Attribute Inference Attacks in Online Social Networks. *ACM Transactions on Privacy and Security (TOPS)* 21, 1 (2018).

[18] Neil Zhenqiang Gong, Ameet Talwalkar, Lester Mackey, Ling Huang, Eui Chul Richard Shin, Emil Stefanov, Elaine Runting Shi, and Dawn Song. 2014. Joint link prediction and attribute inference using a social-attribute network. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 2 (2014).

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[20] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016).

[21] Jianming He, Wesley W Chu, and Zhenyu Victor Liu. 2006. Inferring privacy information from social networks. In *International Conference on Intelligence and Security Informatics*. Springer, 154–165.

[22] Dirk Hovy and Anders Søgaard. 2015. Tagging performance correlates with author age. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 483–488.

[23] Jingyu Hua, Chang Xia, and Sheng Zhong. 2015. Differentially Private Matrix Factorization.. In *IJCAI*. 1763–1770.

[24] J Jia and Gong NZhenqiang. 2018. AttriGuard: A Practical Defense Against Attribute Inference Attacks via Adversarial Machine Learning. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*. USENIX Association.

[25] Jinyuan Jia, Binghui Wang, Le Zhang, and Neil Zhenqiang Gong. 2017. AttriInfer: Inferring user attributes in online social networks using markov random fields. In *Proceedings of the WWW*. 1561–1569.

[26] Zach Jorgensen and Ting Yu. 2014. A Privacy-Preserving Framework for Personalized, Social Recommendations. *EDBT* 582.

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

[28] Joseph A Konstan and John Riedl. 2012. Recommender systems: from algorithms to user experience. *User modeling and user-adapted interaction* 22, 1-2 (2012).

[29] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 447–456.

[30] Michal Kosinski, David Stillwell, and Thore Graepel. 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences* 110, 15 (2013), 5802–5805.

[31] Jack Lindamood, Raymond Heatherly, Murat Kantarcioglu, and Bhavani Thuraisingham. 2009. Inferring private information using social network data. In *Proceedings of WWW*. ACM, 1145–1146.

[32] Zhifeng Luo and Zhanli Chen. 2014. A privacy preserving group recommender based on cooperative perturbation. In *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. IEEE.

[33] Frank McSherry and Ilya Mironov. 2009. Differentially private recommender systems: building privacy into the net. In *Proceedings of SIGKDD*. ACM.

[34] Alan Mislove, Bimal Viswanath, Krishna P Gummadi, and Peter Druschel. 2010. You are who you know: inferring user profiles in online social networks. In *Proceedings of WSDM*. ACM, 251–260.

[35] Wei Niu, James Caverlee, and Haokai Lu. 2018. Neural Personalized Ranking for Image Recommendation. In *Proceedings of the 11th ACM WSDM*.

[36] Gong NZhenqiang and B Liu. 2016. You Are Who You Know and How You Behave: Attribute Inference Attacks via Users' Social Friends and Behaviors. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*. USENIX Association.

[37] Javier Parra-Arnau, David Rebollo-Monedero, and Jordi Forné. 2014. Optimal forgery and suppression of ratings for privacy enhancement in recommendation systems. *Entropy* 16, 3 (2014), 1586–1631.

[38] Huseyin Polat and Wenliang Du. 2003. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *International Conference on Data Mining*. IEEE.

[39] Naren Ramakrishnan, Benjamin J Keller, Batul J Mirza, Ananth Y Grama, and George Karypis. 2001. Privacy risks in recommender systems. *IEEE Internet Computing* 6 (2001), 54–62.

[40] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K Lam, Sean M McNee, Joseph A Konstan, and John Riedl. 2002. Getting to know you: learning new user preferences in recommender systems. In *Proceedings of the 7th international conference on Intelligent user interfaces*. ACM, 127–134.

[41] David Rebollo-Monedero, Javier Parra-Arnau, and Jordi Forné. 2011. An information-theoretic privacy criterion for query forgery in information retrieval. In *International Conference on Security Technology*. Springer, 146–154.

[42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press.

[43] Udi Weinsberg, Smriti Bhagat, Stratis Ioannidis, and Nina Taft. 2012. BlurMe: Inferring and obfuscating user gender based on ratings. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 195–202.

[44] Elena Zheleva and Lise Getoor. 2009. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th international conference on World wide web*. ACM, 531–540.

[45] Xue Zhu and Yuqing Sun. 2016. Differential privacy for collaborative filtering recommender algorithm. In *Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics*. ACM, 9–16.

[46] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*. ACM, 22–32.