Covering the Sensitive Subjects to Protect Personal Privacy in Personalized Recommendation

Zongda Wu, Guiling Li, Qi Liu, Guandong Xu, and Enhong Chen, *Senior Member, IEEE*

Abstract—Personalized recommendation has demonstrated its effectiveness in improving the problem of information overload on the Internet. However, evidences show that due to the concerns of personal privacy, users' reluctance to disclose their personal information has become a major barrier for the development of personalized recommendation. In this paper, we propose to generate a group of fake preference profiles, so as to cover up the user sensitive subjects, and thus protect user personal privacy in personalized recommendation. First, we present a client-based framework for user privacy protection, which requires not only no change to existing recommendation algorithms, but also no compromise to the recommendation accuracy. Second, based on the framework, we introduce a privacy protection model, which formulates the two requirements that ideal fake preference profiles should satisfy: (1) the similarity of feature distribution, which measures the effectiveness of fake preference profiles to hide a genuine user preference profile; and (2) the exposure degree of sensitive subjects, which measures the effectiveness of fake preference profiles to cover up the sensitive subjects. Finally, based on a subject repository of product classification, we present an implementation algorithm to well meet the privacy protection model. Both theoretical analysis and experimental evaluation demonstrate the effectiveness of our proposed approach.

Index Terms—Personalized Recommendation, Personal Privacy, Sensitive Subject, Feature Distribution

1 Introduction

The rapid development of the Internet results in the explosive growth of information quantity, leading to the serious problem of information overload, and thus greatly reducing the using efficiency of information. Personalized recommendation, which can guide users to discover the information that they really need by means of the record analysis of user personal preferences, is considered to one of the most effective tools to solve the problem of information overload [**?**], [**?**], [**?**]. Presently, personalized recommendation has achieved great success in many application fields (typically, e-commerce). Almost all the large-scale e-commerce sites (such as Amazon and Jingdong) have introduced personalized recommendation to a variable extent.

In general, a complete personalized recommendation system consists of three parts [**?**], [**?**], i.e., (1) a **behavior record** component that collects user's personal information, (2) a **preference analysis** component that analyzes user personal preferences, and (3) a **recommendation algorithm** component. In a personalized recommendation system, the recommendation algorithm is the core component, which aims to find out the products that best meet user preferences from a database of products. Presently, there exist many kinds of recommendation algorithms, typically including collaborative filtering [**?**], [**?**], [**?**], content-based recommendation [**?**], [**?**], and network-based recommendation [**?**], [**?**]. In general, the better the accuracy of personalized recommendation, the more users' personal information a recommendation algorithm needs to master. However, the collection and analysis of users' personal information will lead to users' concerns on personal privacy, resulting in negative impacts on the development of personalized recommendation: it not only reduces the willingness of users to use the service of personalized recommendation, but also makes users no longer willing to supply accurate personal information, thereby, reducing the accuracy of personalized recommendation. Therefore, personalized recommendation would lose the confidence and support of the users, if it cannot strengthen the protection of users' personal privacy. In fact, user privacy concerns have become one major barrier for the development and application of personalized recommendation, as pointed out in [**?**], [**?**], [**?**].

1.1 Motivations

In order to protect personal privacy in personalized recommendation, many approaches have been proposed, specifically including: data obfuscation, data transformation, anonymization etc. (1) The basic idea of **data obfuscation** techniques is to use fake or general data to obfuscate the data related to the sensitive preferences contained in users' preference profiles [**?**], [**?**], [**?**]. This kind of techniques might lead to poor recommendation accuracy due to its change to user preference profiles. (2) In **data transformation** techniques, users' personal data need to be transformed (e.g., using noise addition or data perturbation) [**?**], [**?**], [**?**], before being used for personalized recommendation. Generally, this kind of techniques can only be applied to collaborative

- *Z. Wu is with the Oujiang College, Wenzhou University, Wenzhou 325035, Zhejiang, China. E-mail: zongda1983@163.com*
- *G. Li is with the School of Computer Science, China University of Geosciences, Wuhan, China.*
- *G. Xu is with the Faculty of Engineering and IT, University of Technology, Sydney, Australia*
- *Q. Liu and E. Chen are with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China*

Manuscript received 2016; revised 2016.

filtering algorithms. Moreover, it has been demonstrated that effective data transformation would not lead to a negative impact on the accuracy of collaborative filtering recommendation. However, since the recommendation results are fully visible to the untrusted server-side, it is possible for an attacker on the server-side to guess the genuine user preferences conversely by analyzing the recommendation results, thus, leading to the disclosure of personal privacy. (3) **Anonymization** has been widely applied to personal privacy protection [**?**], [**?**], which allows users to use a system without the need to expose their identity information. However, as pointed out in [**?**], [**?**], it is very necessary to confirm the true identity for each user in a recommendation system. Therefore, this kind of techniques cannot satisfy the requirement of the practical application of personalized recommendation.

Based on the above, we conclude that to supply an effective personalized recommendation service, it is required for a privacy protection approach to satisfy the following three requirements. (1) Ensuring the **security** of user sensitive preferences (i.e., the preference information that users are not willing to expose). Specifically, it should be difficult for an attacker not only to identify the user sensitive preferences from users' personal behavior (or data), but also to guess the user sensitive preferences conversely through analyzing the results returned from the recommendation algorithm. The former can be achieved by both data obfuscation and data transformation. However, the latter cannot be achieved by data transformation since it ensures the accuracy of recommendation. (2) Ensuring the **accuracy** of the user final recommendation results, i.e., the recommendation results that users receive finally should be as consistent as possible (or the same), before and after the privacy protection approach is introduced. (3) Ensuring the **efficiency** of personalized recommendation, i.e., the introduction of privacy protection should not lead to a serious effect on the execution efficiency of a personalized recommendation service.

1.2 Contributions

In this paper, we aim to propose an effective approach to protect user's personal privacy in personalized recommendation. The approach should address all the problems mentioned above, i.e., under the precondition of not changing existing recommendation algorithms, it can not only effectively prevent the untrusted server-side from identifying the user sensitive preferences from personal data or recommendation results, but also ensure the accuracy of recommendation results and the efficiency of a personalized recommendation service. The basic idea of the approach is to construct a group of fake preference profiles, so as to cover up the user sensitive subjects, and thus to protect user personal privacy. Specifically, the contributions of this paper are threefold.

First, we present a client-based system framework to protect user sensitive preferences in personalized recommendation. Under the system framework, we move the behavior record component to a trusted client, making that user preference profiles would be generated in the trusted client. Then, the client constructs a group of fake preference profiles, and submits them together with the genuine user preference profile to the server-side for personalized recommendation. Thus, the recommendation results from the server-side would be no longer accurate (since including those corresponding to the fake profiles), which makes it difficult for an attacker to identify the user's sensitive preferences from the recommendation results. Finally, the client discards all the recommendation results that correspond to the fake preference profiles, so only the recommendation result that corresponds to the genuine preference profile is returned to the user, consequently, ensuring the accuracy of personalized recommendation.

Second, based on the system framework, the paper introduces a privacy model for user sensitive preference protection. The model formulates the requirements that the fake preference profiles should satisfy so as to protect the sensitive preferences effectively, i.e., fake profiles should have similar features with the genuine profile, and irrelevant subjects with the sensitive preferences. The feature similarity makes it difficult for an attacker to identify the genuine user preference profile, even if the attacker captures all the preference profiles. The subject irrelevance results in that the exposure degree of the sensitive preferences on the server-side can be effectively reduced by the fake profiles, thereby, ensuring the security of users' sensitive preferences.

Finally, according to the system framework and the privacy model mentioned above, based on a subject repository of product classification, we present an implementation algorithm that runs on a trusted client. The algorithm can well meet the requirements of user privacy protection in personalized recommendation, i.e., it can construct a group of fake preference profiles that well meet the privacy model. In addition, we have demonstrated the effectiveness of the privacy model and its implementation algorithm through theoretical analysis and experimental evaluation.

The rest of this paper is organized as follows. Section 2 briefly reviews the related work. Section 3 presents a system framework for user privacy protection in personalized recommendation, as well as a related attack model. Section 4 formulates a privacy model on the protection of user sensitive preferences, presents an implementation algorithm to well meet the privacy model, and theoretically analyzes the effectiveness of the proposed approach. Section 5 experimentally evaluates the approach. Finally, we conclude this paper in Section 6.

2 Related Works

Depending on the recommendation algorithms, recommendation systems can be divided into three main categories: (1) collaborative filtering [**?**], [**?**], [**?**], which is the process of filtering products based on the similarity computation of users' previous preference products; (2) content-based recommendation [**?**], [**?**], which recommends products for a user based on the similarity between the user preferences and the product descriptions; and (3) social network-based recommendation [**?**], [**?**], which is an extension of collaborative filtering, and measures the similarity of users using a social network analysis technique. In general, a recommendation algorithm has to run on an untrusted server-side, and the better the recommendation accuracy, the more users' personal information the algorithm needs to master,

consequently, leading to users' serious concerns on personal privacy [?], [?].

In order to protect user privacy in personalized recommendation, many approaches have been proposed. In this section, we briefly review and analyze these approaches, specifically, including: data obfuscation, data transformation, anonymization etc.

2.1 Data Obfuscation

The basic idea of data obfuscation techniques is to leverage fake data or general data to obfuscate the data related to the sensitive preferences contained in user preference profiles. In order to protect the genuine intention hidden in a user query, the paper [?] proposes to inject the false keywords into the user query. Then, similar approaches are also proposed in the literature [?], [?], but they allow a user to define his own privacy requirements, i.e., to define the subjects that the user wants to protect, and the degree of protection. Aiming at personalized advertisement recommendation, the paper [?] presents a client-based approach to user privacy protection, which is based on the comprehensive consideration of user privacy (i.e., the privacy level that a user is willing to share with the server-side) and network traffic (i.e., the number of ads returned to a mobile phone) to select relevant ads for a user. Aiming at personalized web search, the paper [?] designs a user preference protection approach. It builds a hierarchical structure of user preferences on the client, where nodes of high level are used to store general preference subjects, while other nodes of low level are used to store special subjects. Then, some general subjects are selected to replace sensitive special subjects, so as to protect the user sensitive preferences. Afterwards, similar approaches are presented in [?], [?], which also propose to cover up the user interested preferences using more general preferences. However, this kind of techniques certainly will reduce the recommendation accuracy due to its change to user preference profiles, namely, whose privacy protection is based on a compromise on recommendation performance.

2.2 Data Transformation

In data transformation techniques, users' personal data need to be transformed (e.g., by noise addition or data perturbation) [?], [?], [?], before being used for personalized recommendation. Generally, this kind of techniques can only be applied to collaborative filtering algorithms. Random perturbation technique (RPT) is a frequently-used approach for data transformation [?], [?]. Its basic idea is to attach a random data (r) to the user sensitive data (a) so that what an attacker can see is ($a + r$), i.e., submit the user sensitive data together with the additional random data to the server for personalized recommendation, so that the server cannot see the true user data. When the user data quantity is large enough, by using the overall user data for collaborative filtering recommendation, we can still obtain a relatively accurate recommendation result. Thus, RPT can ensure not only the security of user privacy, but also the recommendation accuracy. A similar method is proposed in [19] to protect the personal privacy of data mining. The paper [?] designs a collaborative filtering recommendation system based on the discrete wavelet transform (DWT) technique and random perturbation technique. The paper [?] proposes to write several well-designed "predictive scores" into a user-product scoring matrix (that is the input of a collaborative filtering algorithm), so as to perturb the true user scoring information and thus protect personal privacy. The paper [?] has evaluated the effect of data transformation on the accuracy of collaborative filtering recommendation. The results show that effective data transformation would not lead to a negative impact on the accuracy of collaborative filtering recommendation.

It can be seen that this kind of techniques can ensure not only the accuracy of recommendation results to a certain extent, but also the security of a sensitive preference in its user preference profile effectively. However, the accuracy of a recommendation result leads to that many products relevant to the user sensitive preferences are generally contained in the recommendation result. Since the recommendation result is fully visible to the untrusted server-side, it is possible for an attacker on the server-side to guess the genuine user preferences conversely through analyzing the recommendation result, consequently, leading to the disclosure of personal privacy.

2.3 Anonymization

Anonymization is a kind of widely used approaches in privacy protection. It allows users to use a system without the need to expose their identity information. Anonymization, due to the non-complexity of its processing, can be easily applied to a personalized recommendation system, and has been widely used in many systems to protect user personal privacy, such as [?], [?], [?] and [?]. However, there have been many questions about the practicality of using anonymization for privacy protection in personalized recommendation. The papers [?], [?] present the shortages of anonymization to user privacy protection, and demonstrate the results by using experiment evaluations. Anonymization increases the possibility that a user submits useless random data, thereby, decreasing the quality of user personal data. Moreover, anonymization also makes the system easier to be attacked by competitors. For example, a company can submit a large number of fake data in a recommendation system to promote its own products to obtain more opportunities of recommendation. Thus, it is necessary to confirm the true identity for each user in a recommendation system. At present, most of personalized recommendation systems require users to provide the basic information that can identify their personal identities. Therefore, this kind of techniques cannot satisfy the requirement of the practical application of personalized recommendation.

3 Problem Statement

In this paper, we study an approach for protecting user sensitive subjects in a personalized recommendation system. According to the motivations presented in Section 1.1, the approach has to meet the following four requirements. (1) It does not change the existing structure of a recommendation algorithm. (2) It does not compromise the accuracy of the final recommendation. (3) It ensures the security of user

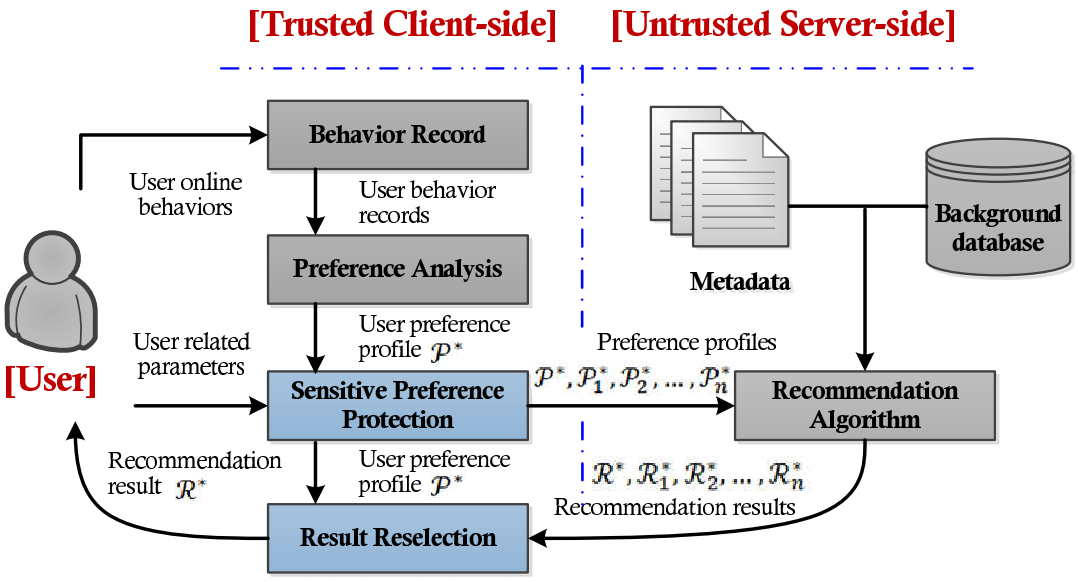


Fig. 1. The system framework for the protection of user sensitive preferences in a personalized recommendation service, where the blue components "sensitive preference protection" and "result reselection" are introduced newly.

preferences, making it difficult for an attacker not only to identify the sensitive subjects from a user preference profile, but also to guess the sensitive subjects conversely from the recommendation result. (4) It does not lead to a serious effect on the execution efficiency of a personalized recommendation service. In this section, we present the system model used in our approach, and then discuss the attack model based on the system model.

3.1 System Model

Here, user sensitive preferences are referred as the personal preferences that users are unwilling to be seen or analyzed by attackers. Fig. 1 shows the system framework used by this paper for the protection of user sensitive preferences in a personalized recommendation service, which consists of an untrusted server-side and many trusted client-sides. The basic process flow of the system framework is presented as follows.

- Under the client-based architecture, the user behavior record component and the preference analysis component are moved from the server to a client. Thus, the client (instead of the server) collects and analyzes user behaviors to generate a user preference profile $\mathcal{P}^*$.
- In the client, the newly-introduced component of sensitive preference protection constructs a group of fake preference profiles $\mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*$ based on the user preference profile $\mathcal{P}^*$, after taking into consideration the requirements of security, accuracy and efficiency. Then, the fake preference profiles are submitted together with the genuine user preference profile to the server-side, as the input of the personalized recommendation algorithm.
- In the client, the newly-introduced result reselection component selects the recommendation result $\mathcal{R}^*$, which corresponds to the user preference profile $\mathcal{P}^*$, from all the recommendation results $\mathcal{R}^*, \mathcal{R}_1^*, \mathcal{R}_2^*, ..., \mathcal{R}_n^*$ that are returned by the recommendation algorithm on the server-side. Then, the component returns $\mathcal{R}^*$ to the user, while discarding the other recommendation results $\mathcal{R}_1^*, \mathcal{R}_2^*, ..., \mathcal{R}_n^*$.

Based on the system framework in Fig. 1, we conclude as follows. On the one hand, the results outputted by the recommendation algorithm component in the server-side, are no longer equal to the true user recommendation result (i.e., the result before the introduction of privacy protection). They contain the recommendation results corresponding to the fake preference profiles. Thus, it is difficult to immediately identify the user sensitive preferences from the recommendation results. On the other hand, the results outputted by the recommendation algorithm are certainly a superset of the true recommendation result, thereby ensuring that the user can obtain an accurate recommendation. In addition, the system framework requires no change to the existing personalized recommendation algorithm, so it is transparent for both the user on the client and the recommendation algorithm component running on the server-side.

However, from Fig. 1, it can also be seen that the fake preference profiles generated by the component of sensitive preference protection play an important role in the framework, i.e., their quality is the key to user privacy protection. Generally, the fake preference profiles generated randomly are easy to be ruled out, thus failed to cover up the sensitive preferences contained in a user preference profile. This is because the features of user preferences are generally regularly distributed (e.g., a user is interested in one or several fixed subjects for a period of time), while randomly generated preference profiles are not (which may be evenly related to a large number of subjects). Thus, an attacker can easily detect fake preference profiles according to their different feature distribution. In addition, the fake preference profiles should be not related to the user sensitive preferences. For example, suppose that a sensitive preference related to a user preference profile is the subject "sporting goods". Then,

it is not appropriate to generate a group of fake profiles that also contain the sensitive subject "sporting goods" or other highly relevant subjects, because at this time, an attacker can immediately draw a conclusion that the user is interested in "sporting goods", without ruling out the fake profiles. To this end, fake preference profiles generated by the sensitive preference protection component should meet the following two requirements: (1) ensuring the security of user sensitive preferences on the untrusted server-side, i.e., reducing the exposure degree of user sensitive preferences on the server-side, and hence the probability of an attacker to detect them; and (2) exhibiting highly-similar feature distribution with the user preference profile, so as to make it difficult for an attacker to rule out the fake profiles, thus, hiding the user profile effectively.

3.2 Attack Model

In the system framework, the server-side is not trusted, which is considered as the biggest potential attacker. Assume that the attacker has taken control of the server (i.e., the attacker may be a hacker who breaks the server, or an administrator who works on the server). Thus, the proposed approach to user privacy protection needs to prevent the server from identifying the sensitive preferences related to a user preference profile. From the system framework shown in Fig. 1, we can see that the attacker can obtain not only all the preference profiles submitted by the client, but also all the recommendation results generated by the personalized recommendation algorithm. Thus, we need to prevent the attacker from identifying the user sensitive preferences not only from the preference profiles, but also from the recommendation results. In addition, because of taking control of the server, the attacker has a powerful capability, which masters the database of all the products and the repository of product classification, and takes charge of executing the personalized recommendation algorithm. Unfortunately, the attacker might also know the existence of the sensitive preference protection algorithm deployed on the client, and obtain a copy of the algorithm. Hence, the attacker can input each of the mastered preference profiles to the privacy protection algorithm, and then observe the output results to guess the user preference profile.

4 Proposed Approach

Based on the system model and the attack model presented in Section 3, in this section, we propose our approach so as to protect the user sensitive preferences. First, based on the system model, we define a privacy model, which formulates the requirements that the fake preference profiles should satisfy so as to effectively protect the user sensitive preferences. Second, based on a subject repository of product classification, we describe an implementation algorithm for the privacy model, to generate a group of fake profiles that have similar feature distribution but irrelevant sensitive subjects with the user preference profile. Finally, we analyze the security of our proposed approach, and compare our approach with other state-of-art ones in terms of security, efficiency, usability and accuracy. In Table 1, we describe key symbols used in this paper.

TABLE 1
Symbols and their meanings

Symbols	Meanings
$\mathcal{P}$	A set of all the products
$\mathcal{P}^*$	A set of user preference products, i.e., $\mathcal{P}^* \subseteq \mathcal{P}$
$\mathcal{G}$	A set of all the subjects
$\mathcal{G}^*$	A set of user preference subjects, i.e., $\mathcal{G}^* \subseteq \mathcal{G}$
$\mathcal{G}_k^*$	A set of user preference subjects with the level k, i.e., $\mathcal{G}_k^* \subseteq \mathcal{G}^*$
$\mathcal{G}^\dagger$	A set of user sensitive preference subjects, i.e., $\mathcal{G}^\dagger \subseteq \mathcal{G}^*$
k^m	The maximum of levels for all the subjects, i.e., $k^m = \max_{g \in \mathcal{G}}\{level(g)\}$
P	The product feature distribution vector, which corresponds to $\mathcal{P}^*$
G^k	The subject feature distribution vector, which corresponds to $\mathcal{G}_k^*$

4.1 Privacy Model

Below, based on the system architecture shown in Fig. 1, we define a privacy model for user sensitive preference protection. As seen from Fig. 1, a user preference profile is an important data structure, which is not only the output of the preference analysis component, but also the input of the sensitive preference protection component and the recommendation algorithm component. The organization structure of a user preference profile is mainly restricted by the recommendation algorithm, i.e., recommendation algorithms of different types will lead to different profile structures. In this paper, we aim to study widely-used collaborative filtering algorithms (whose input is a user product preference scoring matrix) [**?**]. To this end, a user preference profile can be viewed as a set of user preference products, where each product has been scored by a user (the higher preference score a product has, the more the user is interested in), i.e., we can define a user preference product set to represent a user preference profile.

Definition 1 (Preference Product Set). A user preference product set is a set of all the products that a user is interested in. It can be formulated as $\mathcal{P}^* = \{\, p \,|\, p \in \mathcal{P} \wedge score(p) \neq 0\}$, wherein, $\mathcal{P}$ denotes a set of all the products; and $score(p)$ denotes the preference score presented by the user for the product p.

It can be seen that a preference product set is composed of the products whose scores are not equal to zero. Actually, it is easy to obtain the preference product set for a user, based on the scoring values of all the products calculated by the behavior record component and the preference analysis component. In the background database of products of a personalized recommendation system, there is a tree structure organized based on the levels of subjects, so as to manage the products. For example, the product "Lenovo K2450" step-by-step belongs to the subjects "Lenovo", "notebook computer", "computer" and "IT and digit". A sample of a subject tree is shown in Fig. 2, where each leaf node represents a product, and each non-leaf node

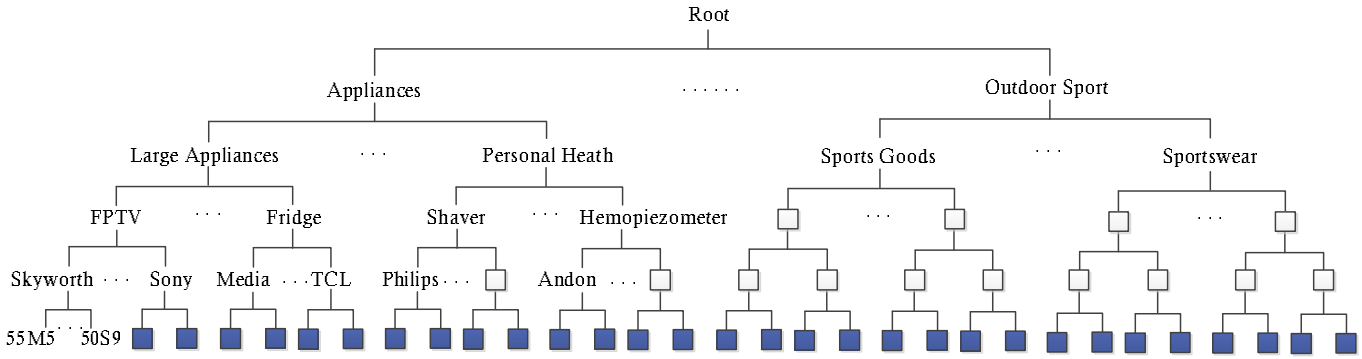


Fig. 2. A sample of a subject tree, where the blue nodes represent products, and the others represent subjects.

represents a subject. Thus, with the help of a subject tree, we can further compute the preference score of the user to each subject, based on the preference scores of all the products.

Definition 2 (Subject Score). A user subject score, which denotes the preference degree of the user to a subject g ($g \in \mathcal{G}$), can be represented as $score(g)$, where $\mathcal{G}$ denotes a set of all the subjects. The subject score can be computed based on the user preference scores for all the products, i.e., it can be computed iteratively as follows.

$$score(g) = \sum_{p \in \mathcal{P}(g)} \frac{score(p)}{\rho_p} + \sum_{g' \in \mathcal{G}(g)} \frac{score(g')}{\rho_g} \quad (1)$$

wherein, $\mathcal{P}(g)$ denotes all the products that directly belong to the subject g; $\mathcal{G}(g)$ denotes all the sub-subjects that directly belong to the subject g; and the parameters ρ_p and ρ_g denote a product attenuation coefficient and a subject attenuation coefficient, respectively.

Below, we take the subject tree in Fig. 2 as an example to show how the subject score is calculated. To simplify the presentation, we set $\rho_p = \rho_g = 2$, and assume that $score(\text{“55M5”}) = score(\text{“50S9”}) = 1$ and the scores of the rest products are all equal to 0. Based on Definition 2, we have that $score(\text{“Skyworth”}) = 1$ (where $\mathcal{G}(\text{“Skyworth”}) = \oslash$) and $score(\text{“FPTV”}) = 0.5$ (where $\mathcal{P}(\text{“FPTV”}) = \oslash$). Further, we have that $score(\text{“Large Appliances”}) = 0.25$ and $score(\text{“Appliances”}) = 0.125$.

From Fig. 2, we observe that there are a number of inclusion relations between subjects. For example, the subject “Fridge” belongs to “Large Appliances”, and “Large Appliances” belongs to “Appliances”. Thus, each product subject is associated with a level. We stipulate that the higher the level, the more special a subject (e.g., “Media”), and the lower the level, the more general a subject (e.g., “Appliances”). Below, we use $level(g)$ to denote the level of a subject $g \in \mathcal{G}$, and k^m to denote the maximum subject level, i.e., $k^m = \max_{g \in \mathcal{G}} \{level(g)\}$.

Definition 3 (Preference Subject Set). A user preference subject set is a set of all the subjects that a user is interested in. It is formulated as $\mathcal{G}^* = \{g \,|\, g \in \mathcal{G} \wedge score(g) \geq \tau_g\}$, where $\mathcal{G}$ denotes a set of all the subjects, and τ_g denotes a preset threshold.

A preference subject set is composed of all the products whose preference scores are greater than the threshold τ_g (the subjects whose scores are less than the threshold, are considered as meaningless). It can be seen that a preference subject set $\mathcal{G}^*$ has to be constructed based on a preference product set $\mathcal{P}^*$, i.e., $\mathcal{G}^*$ corresponds to $\mathcal{P}^*$.

Definition 4 (Preference Subject Set with a Level). A user preference subject set with the level k consists of all the subjects whose levels are equal to k in the preference subject set $\mathcal{G}^*$ ($0 < k \leq k^m$). Formally, it can be formulated as $\mathcal{G}_k^* = \{g \,|\, g \in \mathcal{G}^* \wedge level(g) = k\}$.

Obviously, we have $\mathcal{G}^* = \bigcup_{k=1}^{k^m} \mathcal{G}_k^*$, so based on a preference product set $\mathcal{P}^*$, we can obtain a group of preference subject sets of different levels, i.e., $\mathcal{G}_1^*, \mathcal{G}_2^*, ..., \mathcal{G}_{k^m}^*$. Now, we can use a subject to indicate a user sensitive preference, called a **user sensitive subject** (denoted by $g^\dagger$). A user sensitive subject $g^\dagger$ indicates a subject of products that a user is unwilling to be known by an attacker, which can be assigned by the user in advance. Given a preference product set $\mathcal{P}^*$, based on Definition 3, we can obtain a preference subject set $\mathcal{G}^*$. Then, using $\mathcal{G}^*$ as an intermediate reference, we can further compute the degree of exposure of a user sensitive subject $g^\dagger$ in a user preference product set $\mathcal{P}^*$ (i.e., a user preference profile). Below, we define the significance to represent the exposure degree of a user sensitive subject.

Definition 5 (Sensitive Subject Significance). Given any sensitive subject $g^\dagger$ and a user preference product set $\mathcal{P}^*$, let k be the level of the subject $g^\dagger$ (i.e., $k = level(g^\dagger)$), and $\mathcal{G}_k^*$ be a preference subject set with the level k, which is obtained based on the set $\mathcal{P}^*$. Then, the significance of the sensitive subject $g^\dagger$ in the preference product set $\mathcal{P}^*$ can be defined as follows.

$$sig(g^\dagger, \mathcal{P}^*) = \left(\sum_{g \in \mathcal{G}_k^*} score(g) \right)^{-1} score(g^\dagger) \quad (2)$$

Given several preference product sets $\mathbf{P} = \{\mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*\}$, let $\mathcal{G}_{ik}^*$ be a user preference subject set with the level k, which is obtained based on the set $\mathcal{P}_i^* \in \mathbf{P}$. Then, the significance

of the sensitive subject $g^\dagger$ in these preference product sets can be defined as follows.

$$sig(g^\dagger, \mathbf{P}) = \left(\sum_{\mathcal{P}_i^* \in \mathbf{P}} \sum_{g \in \mathcal{G}_{ik}^*} score(g) \right)^{-1} score(g^\dagger) \quad (3)$$

How to effectively protect each sensitive subject related to the product set $\mathcal{P}^*$ is the key to user privacy protection. According to the system model and the attack model mentioned in Section 3, when an attacker does not know the user sensitive subjects, he/she can only guess by analyzing the preference product sets $\mathcal{P}^*, \mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*$ submitted from a client-side. Obviously, given a sensitive subject, the greater its significance in the preference product sets, the more likely the attacker guesses it. To this end, we can use the significance of each sensitive subject to measure the exposure risk of user personal privacy.

According to Definition 5, the sensitive preference protection component can construct fake preference product sets $\mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*$ for the user preference product set $\mathcal{P}^*$, so as to decrease the significance of each user sensitive subject related to $\mathcal{P}^*$, and hence the probability of exposing the sensitive subjects. However, the precondition of the above idea is that the features of the fake preference product sets have to be highly similar to that of the genuine user preference product set, so as to make them difficult to be ruled out by an attacker. To this end, we below define the feature distribution of a preference product set.

Definition 6 (Product Feature Distribution). Given a preference product set $\mathcal{P}^*$, its feature distribution can be described using the following vector:

$$P = (score(p_1), score(p_2), ..., score(p_n))$$

wherein, $p_i \in \mathcal{P}^*$ $(i = 1, 2, ..., n)$, $score(p_i) \leq score(p_{i+1})$ $(i = 1, 2, ..., n-1)$, and $n = |\mathcal{P}^*|$.

Definition 7 (Subject Feature Distribution). Given a preference subject set $\mathcal{G}_k^*$ with the level k, its feature distribution can be described using the following vector:

$$G^k = (score(g_1), score(g_2), ..., score(g_n))$$

wherein, $g_i \in \mathcal{G}_k^*$ $(i = 1, 2, ..., n)$, $score(g_i) \leq score(g_{i+1})$ $(i = 1, 2, ..., n-1)$, and $n = |\mathcal{G}_k^*|$.

Now, for any preference product set $\mathcal{P}^*$, we can obtain a product feature vector P, and a group of subject feature vectors, $G^1, G^2, ..., G^{k^m}$. Then, we can further define the feature similarity between any two product sets, which is measured by the similarity of the product feature vectors of the two product sets, and the similarities of the subject feature vectors.

Definition 8 (Feature Similarity). The feature similarity between two product sets can be measured by the similarity of the product feature vectors of the two product sets, and the similarities of the subject feature vectors. Given any two preference product sets $\mathcal{P}_1^*$ and $\mathcal{P}_2^*$, we use P_1 and P_2 to denote their product feature vectors, and G_1^k and G_2^k to denote their subject feature vectors with the level k $(k = 1, 2, ..., k^m)$. Then, the feature similarity between $\mathcal{P}_1^*$ and $\mathcal{P}_2^*$ is measured as follows (where $dist$ denotes the Euler distance between two vectors):

$$\begin{aligned} sim(\mathcal{P}_1^*, \mathcal{P}_2^*) &= a_0 \cdot \underbrace{sim(P_1, P_2)}_{\text{product similarity}} \\ &\quad + \sum_{k=1}^{k^m} a_k \cdot \underbrace{sim(G_1^k, G_2^k)}_{\text{subject similarity}} \\ &= \frac{a_0}{dist(P_1, P_2) + 1} \\ &\quad + \sum_{k=1}^{k^m} \frac{a_k}{dist(G_1^k, G_2^k) + 1} \end{aligned} \quad (4)$$

In the above formula, the parameters $(a_0, a_1, ..., a_{k^m})$ are used to balance different kinds of similarities of feature vectors. In the subsequent experiments, we will simply set them to $\frac{1}{(k^m+1)}$ uniformly. It should be pointed out that the feature vectors P_1 and P_2 (or G_1^k and G_2^k) may be not of the same dimensionality. At this time, we will fill the feature vector of smaller size with zeros, so as to calculate the similarity between them. Now, based on Definition 5 (i.e., sensitive preference significance) and Definition 8 (i.e., feature distribution similarity), we can further formulate the requirements that the fake preference product sets generated by the component of sensitive preference protection have to satisfy, so as to prevent an attacker from guessing the user sensitive subjects.

Definition 9 (Sensitive Subject Protection). Given a user preference product set $\mathcal{P}^*$, a group of user sensitive subjects $\mathcal{G}^\dagger$, and a group of fake preference product sets $\mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*$, if the fake product sets meet the following two requirements, then it is deemed that they can be used to effectively protect the sensitive subjects $\mathcal{G}^\dagger$.

- **Ensuring the security of the sensitive subject**: based on the fake preference product sets, the significance of each sensitive subject $g^\dagger \in \mathcal{G}^\dagger$ can be effectively decreased, i.e.,

  $$\forall g^\dagger \in \mathcal{G}^\dagger \rightarrow \frac{sig\left(g^\dagger, \{\mathcal{P}^*, \mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*\}\right)}{sig(g^\dagger, \mathcal{P}^*)} \leq \mu_p$$

  wherein, $0 < \mu_p < 1$. This condition ensures the exposure degree (i.e., the significance) of each sensitive subject in the preference product sets can be decreased effectively, consequently, making it difficult for an attacker to discover the sensitive subjects.
- **Ensuring the similarity of the feature distribution**: the feature of each fake preference product set should be similar to that of the user preference product set, i.e.,

  $$\forall \mathcal{P}_i^* \in \{\mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*\} \rightarrow sim(\mathcal{P}^*, \mathcal{P}_i^*) \geq \mu_o$$

  wherein, $0 < \mu_o < 1$. This condition makes it difficult for an attacker to rule out the fake preference product sets, thereby, hiding the genuine user preference product set effectively.

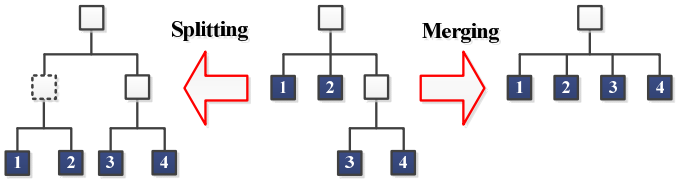


Fig. 3. The two preprocessing ways for a subject tree, where the left denotes a splitting operation, and the right denotes a merging operation.

Now, our objective is to design and implement an effective algorithm for the component of sensitive preference protection, so as to construct a group of fake preference product sets that well satisfy the requirements presented in Definition 9.

4.2 Implementation Algorithm

Below, we present an implementation algorithm for the privacy model mentioned above. The preference analysis component calculates the preference score for each product by analyzing user online behaviors, and constructs a preference product set. Then, based on the score of the user to each product, the sensitive preference protection component calculates the preference score of the user to each subject (i.e., Definition 2), with the help of a hierarchical subject tree. As a result, we can further compute the significance of each sensitive subject (i.e., Definition 5) and the feature distribution similarity (i.e., Definition 8). Therefore, in the privacy model, the subject tree is a very important data structure. A subject tree has the following characteristics: (1) each leaf node represents a product; (2) each non-leaf node represents a subject; (3) each product is contained in a subject; and (4) each subject is contained in another subject (except for the root subject). In the background product database of a personalized recommendation system, there generally exists a hierarchical subject tree similar to that shown in Fig. 2; even if not, it can be constructed with the help of external product subject classification knowledge base, such as Wikipedia [**?**], ODP [**?**] and WordNet [**?**]. Therefore, we can assume that the hierarchical subject tree is pre-existent in the algorithm implementation of the privacy model. However, the depths of leaf nodes of the subject tree may be different from each other, but we note that such difference is very small. Thus, to make all the leaf nodes of a hierarchical subject tree with the same depth (i.e., all equal to $k^m + 1$) so as to facilitate the algorithm implementation, we can preprocess the subject tree by using the following two ways: (1) splitting some leaf nodes of smaller depth, and constructing their parent node; and (2) merging some leaf nodes of bigger depth, and deleting their parent node. Fig. 3 illustrates the above two preprocessing ways. In addition, we also load the classification subject tree into the memory in advance, so as to improve the running efficiency of the algorithm.

Algorithm 1 details the implementation of our approach to sensitive subject protection. Generally, the number of the user sensitive subjects and the number of the user preference subjects are both small (i.e., $|\mathcal{G}^{\dagger}| \ll |\mathcal{G}|$ and $|\mathcal{G}^{*}| \ll |\mathcal{G}|$), thus in Line 13 of Algorithm 1, we assume that $|\mathcal{A}^{*}| \leq |\mathcal{A}|$, so as to simplify the presentation of the algorithm. In the WHILE loop (Lines 6-9) of Algorithm 1, a procedure call operation (i.e., "SearchFakeProducts" at Line 8) would obtain a fake product set $\mathcal{P}_i^{*}$. Thus, after the WHILE loop, a group of fake product sets $\mathbf{P}$ would be generated, based on which, the significance of the user sensitive subjects (see the WHILE condition) can be decreased effectively. In the procedure "SearchFakeProducts", the parameter k is used to denote the level of the currently processed user subjects in $\mathcal{A}^{*}$. If $k = k^m$ (i.e., $\mathcal{A}^{*} \subseteq \mathcal{G}_{k^m}^{*}$), it indicates that the child nodes of each subject in $\mathcal{A}^{*}$ are leaf nodes that denote products. At this time, from $\mathcal{P}$, we randomly search a group of fake products that have the same product feature distribution with the user products belonging to $\mathcal{A}^{*}$ (Lines 20-23). If $k < k^m$, it indicates that the child nodes of each subject in $\mathcal{A}^{*}$ are non-leaf nodes that denote subjects. At this time, we search a group of fake subjects that have the same subject feature distribution with the user subjects (Line 17), and then, recursively call the procedure "SearchFakeProducts" to process the user subjects at the next level $(k+1)$ (Line 18). Finally, this results in that each fake product set constructed by "SearchFakeProducts" has highly similar overall feature distribution with the user product set.

In Algorithm 1, although there exists a recursive call for the procedure "SearchFakeProducts" (Line 8), the execution number of the recursive operations at the bottom level (Lines 21-23) for constructing fake products is equal to the size of the user product set, i.e., equal to $|\mathcal{P}^{*}|$; and the execution number of the other recursive operations (Line 17) for constructing fake subjects is equal to the size of the user subject set, i.e., equal to $|\mathcal{G}^{*}|$. In addition, since the fake product sets are irrelevant to the user sensitive subjects (see Line 4), the execution number of the WHILE loops should be approximately equal to $\lfloor \frac{1}{\mu_p} \rfloor$, i.e., the number of fake product sets constructed by Algorithm 1 is approximately equal to $\lfloor \frac{1}{\mu_p} \rfloor$. Thus, the time complexity of Algorithm 1 is equal to $O\left(\lfloor \frac{1}{\mu_p} \rfloor \cdot (|\mathcal{P}^{*}| + |\mathcal{G}^{*}|)\right)$, which is a relatively ideal polynomial time complexity, and thus does not cause a serious effect on the execution efficiency of a personalized recommendation service.

In addition, since the number of fake product sets constructed by Algorithm 1 for a user preference product set is approximately equal to $\lfloor \frac{1}{\mu_p} \rfloor$, in the next experiment section, the number of fake product sets will be used as an input parameter of Algorithm 1 (instead of μ_p), so as to simplify the presentation for the experimental results.

4.3 Effectiveness Analysis

Based on the system model given in Section 3.1, it can be seen that our proposed approach to user privacy protection not only requires no change to an existing recommendation algorithm, but also requires no compromise on the accuracy of recommendation results. In the approach, the threshold μ_p is used to control the significance of the sensitive subjects, and the smaller the threshold value is, the lower the risk of the sensitive subjects are exposed. In addition, a personalized recommendation service would input $(n+1)$ preference product sets, and output $(n+1)$ recommendation results (where $n \approx \lfloor \frac{1}{\mu_p} \rfloor$). Thus, if we ignore the running time of the sensitive preference protection algorithm itself

Algorithm 1: Protecting the User Sensitive Preferences

Input: (1) $\mathcal{P}^*$, a user preference product set (i.e., a user preference profile); (2) $\mathcal{G}^\dagger$, the user sensitive subjects (i.e., the sensitive preferences); and (3) related parameters (e.g., μ_p).
Output: $\mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*$, a group of fake product sets (i.e., fake preference profiles).

1 **begin**
2 From a set of all the subjects $\mathcal{G}$, select the subject sets with the levels $1, 2, ..., k^m$ (k^m denotes the maximum of subject levels), respectively, denoted by $\mathcal{G}^1, \mathcal{G}^2, ..., \mathcal{G}^{k^m}$, i.e., $\forall g \in \mathcal{G}^k \rightarrow level(g) = k$ ($k = 1, 2, ..., k^m$);
3 From a set of all the user preference subjects $\mathcal{G}^*$, select the subject sets with the levels $1, 2, ..., k^m$, respectively, denoted by $\mathcal{G}_1^*, \mathcal{G}_2^*, ..., \mathcal{G}_{k^m}^*$;
4 **foreach** $\mathcal{G}^k \in \{\mathcal{G}^1, \mathcal{G}^2, ..., \mathcal{G}^{k^m}\}$ **do set** $\mathcal{G}^k = \mathcal{G}^k - \mathcal{G}^\dagger$; `// Remove all the sensitive subjects from` $\mathcal{G}^k$
5 **set** $\mathbf{P} = \oslash$; `//` $\mathbf{P}$ `is used to store the generated fake product sets`
6 **while** $\exists g^\dagger \in \mathcal{G}^\dagger \rightarrow \mu_p \cdot sig(g^\dagger, \mathcal{P}^*) < sig(g^\dagger, \{\mathcal{P}^*\} \cup \mathbf{P})$ **do**
7 **set** $\mathcal{P}_i^* = \oslash$; `// Set an empty fake product set`
8 **call** SearchFakeProducts($\mathcal{G}^1$, $\mathcal{G}_1^*$, 1, $\mathcal{P}_i^*$);
9 **set** $\mathbf{P} = \mathbf{P} \cup \{\mathcal{P}_i^*\}$; `// Generate a fake product set`
10 **return** $\mathbf{P}$; `// Output all the generated fake product sets`

11 **Procedure** SearchFakeProducts($\mathcal{A}$ **in**, $\mathcal{A}^*$ **in**, k **in**, $\mathcal{P}_i^*$ **in & out**) `//` k `denotes the level of the subjects in` $\mathcal{A}$
12 **begin**
13 Select $|\mathcal{A}^*|$ subjects from $\mathcal{A}$ randomly to form a fake subject set $\mathcal{A}^\#$; `// Here, we assume that` $|\mathcal{A}^*| \leq |\mathcal{A}|$
14 Pair the subjects in $\mathcal{A}^*$ and $\mathcal{A}^\#$ randomly (below, we assume $g^* \in \mathcal{A}^*$ paired with $g^\# \in \mathcal{A}^\#$);
15 **if** $k < k^m$ **then** `// If the current subject level is not the highest`
16 **foreach** $g^* \in \mathcal{A}^*$ **do**
17 Let $\mathcal{B}^*$ be all the subjects in $\mathcal{G}_{k+1}^*$ that belong to g^*, and $\mathcal{B}^\#$ be all the subjects in $\mathcal{G}^{k+1}$ that belong to $g^\#$;
18 **call** SearchFakeProducts($\mathcal{B}^\#$, $\mathcal{B}^*$, $k+1$, $\mathcal{P}_i^*$) ; `// A recursive procedure call`
19 **else** `// If the current level is the highest, i.e., the child nodes are products`
20 **foreach** $g^* \in \mathcal{A}^*$ **do**
21 Let $\mathcal{B}^*$ be all the products in $\mathcal{P}^*$ that belong to g^*, and $\mathcal{B}^\#$ be all the products in $\mathcal{P}$ that belong to $g^\#$;
22 **foreach** $p \in \mathcal{B}^*$ **do** Select a fake product p' randomly from $\mathcal{B}^\#$, and **set** $score(p') = score(p)$ Add all the scored fake products from $\mathcal{B}^\#$ into the fake product set $\mathcal{P}_i^*$; `//` $\mathcal{P}_i^*$ `is an output parameter`

on the client-side, the running time of a personalized recommendation service will be increased to $(n + 1)$ times, after the introduction of the preference protection algorithm. As a result, the decrease degree of the running performance of personalized recommendation caused by our approach has a linear positive correlation with the level of user privacy protection, i.e., the approach has little impact on the running performance of personalized recommendation. Next, we analyze the security of our approach. We assume that an attacker on the server-side has mastered the whole database of products and the hierarchical subject tree, and obtained a copy of the sensitive preference protection algorithm. What can the attacker deduce about the user sensitive subjects $\mathcal{G}^\dagger$, according to the preference product sets $\mathbf{P} = \{\mathcal{P}^*, \mathcal{P}_1^*, \mathcal{P}_2^*, ..., \mathcal{P}_n^*\}$? Here, we take the following three cases into consideration.

(1) Under the precondition of not identifying out the genuine user preference product set $\mathcal{P}^*$ from $\mathbf{P}$, can the attacker guess the sensitive subjects $\mathcal{G}^\dagger$ immediately? At this time, since the attacker does not know which one in $\mathbf{P}$ is the genuine product set, he/she can only first obtain all the subjects related to each product set in $\mathbf{P}$ (calculated based on Definition 3), and then guess which ones are the user sensitive subjects. Since the significance of each sensitive subject $g^\dagger \in \mathcal{G}^\dagger$ has been reduced greatly in $\mathbf{P}$ (see the experimental results in Section 5), the possibility of guessing $g^\dagger$ would become small. Therefore, it is difficult for the attacker to guess the sensitive subjects $\mathcal{G}^\dagger$, if not finding out the user preference product set $\mathcal{P}^*$ in advance.

(2) Can the attacker find out the genuine user preference product set $\mathcal{P}^*$ from $\mathbf{P}$? At this time, the attacker can only analyze the features of all the product sets in $\mathbf{P}$ to guess which one is the user preference product set. Because the fake product sets constructed by our approach are of the same product feature distribution with the user product set, it is difficult for the attacker to distinguish the genuine product set according to the product feature distribution. In addition, the hierarchical subject tree related to the background database of products is visible to the attacker, so the attacker can obtain all the subject sets related to each product set in $\mathbf{P}$. However, since these subject sets also have identical feature distribution to each other (see the experimental results in Section 5), it is also difficult for the attacker to distinguish the genuine product set according to the subject feature distribution.

(3) Under the precondition of obtaining a copy of the sensitive preference protection algorithm (i.e., Algorithm 1), can the attacker guess the user preference product set $\mathcal{P}^*$? At this time, the attacker can in turn input each product set $\mathcal{P}_i^* \in \mathbf{P}$, and then test whether the sensitive preference protection algorithm outputs the other product sets $\mathbf{P} \setminus \{\mathcal{P}_i^*\}$. If successfully, then it indicates that $\mathcal{P}_i^*$ is the user product set. However, such an attempt will not succeed, because all the fake products and their subjects are randomly selected (see Lines 13 and 22 in Algorithm 1), i.e., the same input will lead to different output.

In summary, it is difficult for the attacker to identify the user sensitive preferences (the sensitive subjects) from

TABLE 2
The comparison of effectiveness, where the security 1 denotes the security of the sensitive subjects in preference profiles, and the security 2 denotes the security of the sensitive subjects in recommendation results

Candidates	Our approach	Anonymization	Data obfuscation	Data transformation
Security 1	Good	Good	Good	Good
Security 2	Good	Good	Not Good	Not Good
Accuracy	Good	Good	Not Good	Not Good
Usability	Good	Not Good	Good	Good
Efficiency	Not Good	Not Good	Good	Good

a preference profile submitted by a user from a client-side. For the same reason, although the recommendation result $\mathcal{R}^*$ contains the products corresponding to the user sensitive preferences, the attacker cannot from $\{\mathcal{R}^*, \mathcal{R}_1^*, \mathcal{R}_2^*, ..., \mathcal{R}_n^*\}$ guess which one is the recommendation result $\mathcal{R}^*$ corresponding to the user preference product set $\mathcal{P}^*$, so also cannot further deduce reversely the user sensitive preferences based on $\mathcal{R}^*$. In short, our approach to user sensitive preference protection can ensure the security of user sensitivity preferences effectively, i.e., it is difficult for an attacker to identify the user sensitive preferences not only from the input of the recommendation algorithm (i.e., from preference profiles), but also from the output of the recommendation algorithm (i.e., from recommendation results).

In addition, from the related work presented in Section 2, we see that: (1) sensitive data obfuscation cannot ensure the accuracy of personalized recommendation results; (2) data transformation cannot ensure the security of user personal privacy, i.e., an attacker can guess the user sensitive preferences reversely according to recommendation results; and (3) anonymization requires to change the framework of an existing personalized recommendation system, resulting in a poor usability. Table 2 shows the effectiveness comparison of our approach to the state-of-the-art ones, where: (1) the **security** is "good", if and only if the related security problem has been considered by the approach, and a good solution has been proposed; (2) the **accuracy** is "good", if and only if the recommendation result is the same before and after the approach is introduced; (3) the **usability** is "good", if and only if the approach is transparent for both the user and the recommendation algorithm; and (4) the **efficiency** is "good", if and only if the recommendation efficiency is the same before and after the approach is introduced, if we ignore the running efficiency of the approach itself. From Table 2, we observe that our proposed approach obtains better comprehensive performance than the others in terms of security, accuracy, usability and efficiency.

5 Experiment Evaluation

From the effectiveness analysis in Section 4.3, it can be seen that the effectiveness of our approach on user sensitive subject protection is dependent on the generated fake product sets, i.e., dependent on whether the fake product sets can effectively reduce the significance of the sensitive subjects, and have highly similar feature distribution with the user product set (so as to hide the user preference profile). In this section, we will evaluate the effectiveness of the fake product sets by experiments. First, we describe the experimental setup. Second, we present the experimental results in terms of the sensitive subject significance and the feature distribution similarity. Finally, the additional time and space overheads caused by our approach are also analyzed.

5.1 Experimental Setup

Before the experiments, we briefly describe the experimental setup, including the reference dataset, the construction of user preference product sets, algorithm candidates and system resource configuration.

(1) Reference dataset: The data used in the experiments are mainly collected from Jingdong[1], which is one of the most famous e-commerce platforms. First, we obtain all the subjects at the foremost three levels of the Jingdong product classification structure[2]. Second, we use a webpage program to automatically open each subject at the level 3, thereby, obtaining all the subjects at the level 4 (in Jingdong, the subjects at the level 4 are the highest among all the subjects, which correspond to various product brands). Third, we use a webpage program to further open each subject at the level 4, to obtain all the products (here, we only obtain the top 10 products for each subject). Finally, a classification subject tree (including a root node and a large number of leaf nodes representing products) is constructed, which consists of 20,751 subjects and 198,410 products. In addition, we also optimize the subject tree in advance (e.g., sort all the products and all the subjects at the same level), consequently, enabling Algorithm 1 to access subjects and products efficiently (Lines 13 and 22).

(2) Preference product sets: Based on the classification subject tree, we construct a group of user preference product sets randomly, in which the number of products in each product set, the number of the preference subjects related to each product set, the level of each preference subject, the number of sensitive subjects related to each product set, and the level of each sensitive subject, are all experimental parameters, i.e., they can be adjusted dynamically. In the experiments, by adjusting these parameters, we construct a large number of user preference product sets with different feature distributions, used as the input of each algorithm candidate. In addition, we simply set the threshold τ_g of Definition 3 to be 0 in our experiments.

(3) Algorithm candidates: We benchmark our approach (called **Privacy** below) against the random approach (called **Random**, where the products in each fake set are randomly

1. Jingdong – http://www.jd.com
2. All the subjects in Jingdong – http://www.jd.com/allSort.aspx

selected from the product database, the preference scores of the fake products are also randomly set, but the size of a fake product set is equal to the size of a genuine user product set). Here, **Random** is only used as the baseline approach. In the experiments, we do not compare against other algorithms mentioned in the related work section, since these algorithms are proposed under different system frameworks or privacy models, and are not comparable to our approach. Instead, we have analyzed the advantages and disadvantages of these algorithms in Section 4.3.

(4) System resource configuration: In our experiments, all the algorithms are implemented by using the Java programming language. The experiments are performed on a Java Virtual Machine (version 1.7.0_07) with an Intel i7-5500U CPU and 2 GB of maximum working memory.

5.2 Feature Distribution Similarity

In the first group of experiments, we aim to evaluate the feature distribution similarity between genuine user product sets and fake product sets produced by our approach. Here, we use the metric "feature distribution similarity", which is developed based on Definition 8, and used to measure the effectiveness of fake profiles to hide genuine user profiles. Given an algorithm candidate $\mathcal{AC}$ (i.e., it may be **Privacy** or **Random**) and a user product set $\mathcal{P}^*$, let $\mathbf{P}$ denote a group of fake product sets generated by $\mathcal{AC}$ for $\mathcal{P}^*$, P_i denote the product vector of $\mathcal{P}_i^* \in \mathbf{P}$, and G_i^k denote the subject vector with the level k ($k = 1, 2, ..., k^m$) for $\mathcal{P}_i^*$. Then, the similarity metric for the candidate $\mathcal{AC}$ can be formulated as

$$\textbf{ProductSim}(\mathcal{AC}) = \min_{\mathcal{P}_i^* \in \mathbf{P}} \{sim(P_i, P)\} \quad (5)$$

$$\textbf{SubjectSim}_k(\mathcal{AC}) = \min_{\mathcal{P}_i^* \in \mathbf{P}} \{sim(G_i^k, G^k)\} \quad (6)$$

$$\textbf{OverallSim}(\mathcal{AC}) = \frac{\textbf{ProductSim}(\mathcal{AC})}{k^m + 1} + \sum_{k=1}^{k^m} \frac{\textbf{SubjectSim}_k(\mathcal{AC})}{k^m + 1} \quad (7)$$

A higher value is better, because it means that the fake product sets have more similar feature distribution as the user product set, consequently, making it difficult for an attacker to identify the user product set from $\mathbf{P} \cup \{\mathcal{P}^*\}$.

In the experiments, the number of products contained in each user product set (i.e., the size of $\mathcal{P}^*$) is set to 200-1000, and the level of each preference subject is set to 1-4, and the number of preference subjects (i.e., the size of $\mathcal{G}^*$) is set to 85 (where $|\mathcal{G}_1^*| = 1, |\mathcal{G}_2^*| = 4, |\mathcal{G}_3^*| = 16, |\mathcal{G}_4^*| = 64$). The experiment results are shown in Fig. 4, where the value of each point is from the average of 10 running results. In Fig. 4, the caption of each subfigure denotes the feature similarity metric used in the experiment (**ProductSim**, $\textbf{SubjectSim}_k$ or **OverallSim**). In addition, the X axis denotes the number of product in each user product set, i.e., the size of $\mathcal{P}^*$; the Y axis denotes the feature similarity between user product sets and the fake product sets produced by an algorithm candidate; and **Privacy** [n] (n is equal to 2, 4 or 6) denotes the number of fake product sets constructed using **Privacy** for each user product set, and **Random** [n] denotes the number of fake product sets constructed using **Random**.

From Fig. 4, it can be seen that, as expected, the fake product sets constructed using **Privacy** exhibit a much better feature distribution similarity (including the product similarity, the subject similarity and the overall similarity), compared to those constructed using **Random**. Specifically, the similarity of the fake product sets from **Privacy** is close to 1.0, i.e., both have nearly the same feature distribution; and the similarity almost remains unchanged, with the changing of the number of fake product sets, and the number of products in each fake product set. Moreover, the overall similarity of the fake product sets from **Random** is less than 0.2, and obviously smaller than that from **Privacy**; and the similarity is decreased with the increasing of the number of fake product sets, and the number of products in each fake product set.

Based on the above experimental analysis, we conclude that the fake product sets produced by our approach have a highly similar feature distribution with the genuine user product sets, making it difficult for an attacker to rule out the fake product sets based on the feature distribution, i.e., the genuine user product sets (the user preference profiles) can be hidden effectively by using our approach.

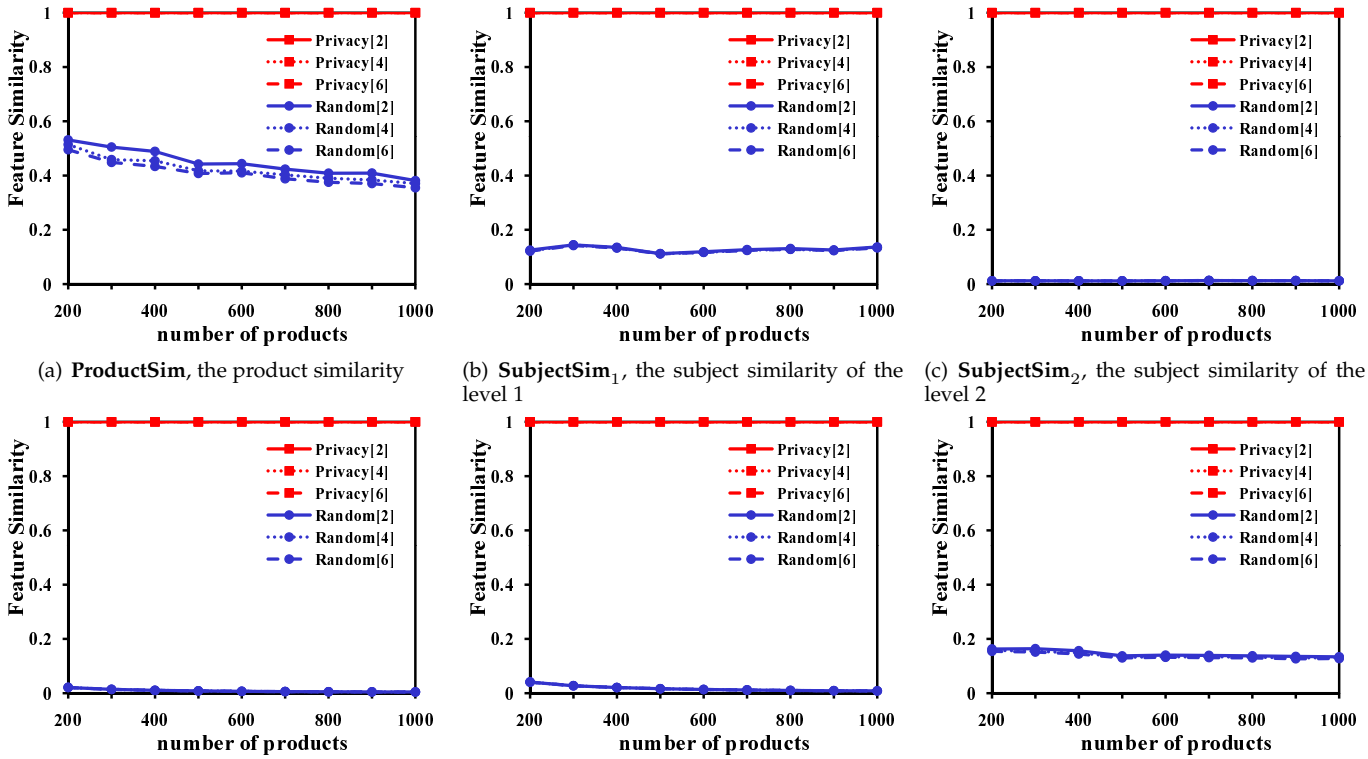
5.3 Sensitive Subject Significance

In the second group of experiments, we aim to evaluate the effectiveness of the fake product sets produced by our approach to cover up the user sensitive subjects (i.e., the significance of the sensitive subjects). Here, we use the metric "sensitive subject significance", which is used to measure the exposure degree of a sensitive subject in the fake product sets. Given an algorithm candidate $\mathcal{AC}$ and a user product set $\mathcal{P}^*$, we use $\mathbf{P}$ to denote a group of fake product sets generated by $\mathcal{AC}$ for $\mathcal{P}^*$, and $\mathcal{G}_k^\dagger$ to denote the sensitive subjects with the level k related to $\mathcal{P}^*$. Then, based on Definition 5, the significance metric for the candidate $\mathcal{AC}$ over the level k can be formulated as

$$\textbf{LevelSig}_k(\mathcal{AC}) = \max_{g^\dagger \in \mathcal{G}_k^\dagger} \frac{sig\left(g^\dagger, \{\mathcal{P}^*\} \cup \mathbf{P}\right)}{sig(g^\dagger, \mathcal{P}^*)} \quad (8)$$

A smaller value is better, because it means the better effectiveness of the fake product sets to cover up the sensitive subjects, consequently, making it difficult for an attacker to guess the sensitive subjects immediately from $\mathbf{P} \cup \{\mathcal{P}^*\}$.

In the experiments, for each user product set, the levels of its sensitive subjects are set to be the same value (which may be 1, 2 or 3), and the number of its sensitive subjects is set to 1 (when the level of the sensitive subjects is set to 1), 4 (when the level is 2) or 8 (when the level is 3). The experimental results are shown in Fig. 5, where: the caption of each subfigure denotes the number of user preference subjects related to each user product set (i.e., $\mathcal{G}_1^*, \mathcal{G}_2^*, \mathcal{G}_3^*$); the X axis denotes the number of fake product sets produced by an algorithm candidate; the Y axis denotes the sensitive subject significance, i.e., the effectiveness of fake product sets to cover up the sensitive subjects; and **Privacy** [n] (n is equal to 1, 2 or 3) denotes the metric $\textbf{LevelSig}_n$(**Privacy**), and **Random** [n] denotes $\textbf{LevelSig}_n$(**Random**). From Fig. 5, it can be seen that the fake product sets constructed using **Privacy** can effectively reduce the significance of the sensitive subjects; and such changing of significance is



(a) **ProductSim**, the product similarity

(b) **SubjectSim**$_1$, the subject similarity of the level 1

(c) **SubjectSim**$_2$, the subject similarity of the level 2

(d) **SubjectSim**$_3$, the subject similarity of the level 3

(e) **SubjectSim**$_4$, the subject similarity of the level 4

(f) **OverallSim**, the overall feature similarity

Fig. 4. The experimental evaluation results for feature distribution similarity

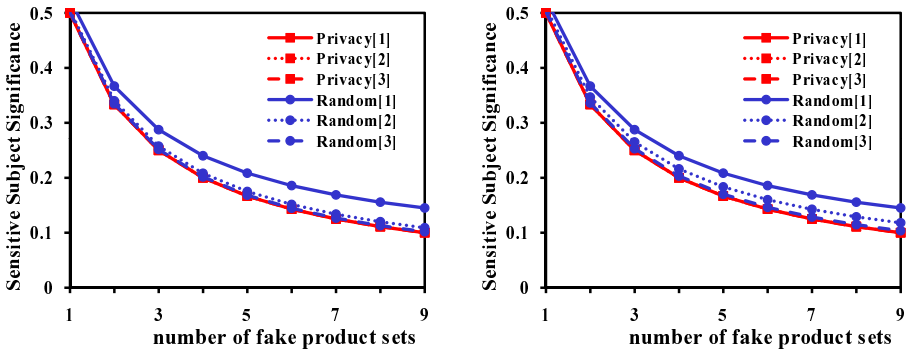
almost linearly negatively related to the number of fake product sets, independently of the number of products in each fake product set and the level of the subjects. Moreover, compared to our approach, the fake product sets constructed using **Random** can also effectively reduce the significance of the sensitive subjects, but their stability is relatively worse (especially, when the level of the preference subjects is set to 1, i.e., **Random** [1]). However, more importantly, based on the first group of experiments, we know that the fake product sets from **Random** exhibit a much worse feature distribution similarity with the genuine user product sets, consequently, making them easy to be ruled out by an attacker, and in turn, failed to protect user sensitive subjects.

Based on the above experimental analysis, we conclude that the fake product sets produced by our approach can effectively reduce the significance of user sensitive subjects, consequently, making it difficult for an attacker to guess the sensitive subjects (the user sensitive preferences) immediately under the precondition of not finding out the genuine user preference profiles.

5.4 Space and Time Overheads

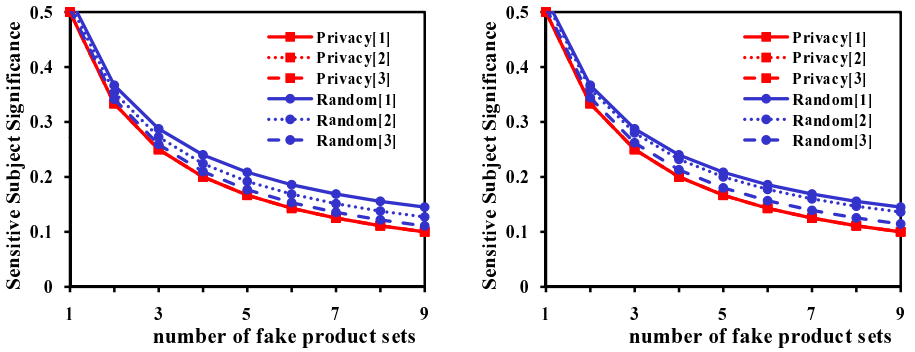
Since we have in advance sorted all the subjects and products in the classification subject tree, the selection process for fake subjects and fake products in Algorithm 1 becomes very efficient. Moreover, the number of products contained in a user preference product set is generally small (at the level of hundreds of products). Therefore, our algorithm has a good running performance. According to the experimental results, our algorithm has almost the same running time with the random algorithm, both less than 1 millisecond, so such a time overhead is negligible. Thus, based on the system framework shown in Fig. 1, we conclude that after the introduction of the sensitive preference protection mechanism, the additional time overhead of a personalized recommendation service is mainly generated by the recommendation for the fake product sets, which is linearly positively related to the number of fake product sets. As a result, when the number of fake product sets is smaller, it does not result in a bigger effect on the running efficiency.

In addition to the time overhead, there is also space overhead. The extra space overhead of our algorithm is mainly from the preload of the subject tree to the main memory. In the subject tree, we only store the product numbers without other product information, so the storage space overhead is not high, especially, when the number of products is not large. In the experiments, we used 20,751 subjects and 198,410 products, which only require several megabytes of the space overhead (about 0.87 MB). In fact, the number of all the products contained in the Jingdong platform is up to ten millions; even so, we only need hundreds of megabytes of space overhead to handle with them. In extreme cases, if we need to deal with a very large database of products (e.g., billions of products), we can use the following strategy to reduce the space overhead: first, for each subject at the highest level (i.e., the parent nodes of leaf nodes), we randomly select a part of its products and load them into the main memory (instead of all the products), so as to reduce the space overhead; and then, at regular intervals, we randomly replace a number of products for each subject.



(a) The sensitive subject significance when $|\mathcal{G}_1^*| = 1$, $|\mathcal{G}_2^*| = 2$ and $|\mathcal{G}_3^*| = 4$

(b) The sensitive subject significance when $|\mathcal{G}_1^*| = 1$, $|\mathcal{G}_2^*| = 4$ and $|\mathcal{G}_3^*| = 16$



(c) The sensitive subject significance when $|\mathcal{G}_1^*| = 1$, $|\mathcal{G}_2^*| = 6$ and $|\mathcal{G}_3^*| = 24$

(d) The sensitive subject significance when $|\mathcal{G}_1^*| = 1$, $|\mathcal{G}_2^*| = 8$ and $|\mathcal{G}_3^*| = 32$

Fig. 5. The experimental evaluation results for sensitivity subject significance

6 Conclusion

In this paper, we proposed an approach for protecting personal privacy for users when using a personalized recommendation service, whose basic idea is to construct a group of fake preference profiles to cover up the sensitive subjects contained in a user preference profile, and in turn protect user personal privacy. We used a client-based system framework that requires not only no change to the existing recommendation algorithms, but also no compromise to the accuracy of recommendation results. Finally, both theoretical analysis and experimental evaluation have demonstrated the effectiveness of our approach: (1) it can generate a group of good-quality fake preference profiles, which not only have high feature distribution similarities with the genuine user preference profile (so as to hide the genuine profile), but also can be used to effectively reduce the risk of exposing the user sensitive subjects; and (2) it does not cause serious performance overheads on either running time or running memory. Therefore, we conclude that our approach can be used to effectively protect users' personal privacy in personalized recommendation.

Acknowledgments

We thank anonymous reviewers for their constructive comments. The work is supported by the Zhejiang Provincial Natural Science Foundation of China (No. LY15F020020), and the National Natural Science Foundation of China (Nos. 61202171 and 61303113).

References

[1] Qiang Song, Jian Cheng, Ting Yuan. "Personalized recommendation meets your next favorite". Proc. of ACM Conference on Information and Knowledge Management (CIKM), 2015, pp. 1775–1778

[2] Adem Ozturk, Huseyin Polat. "From existing trends to future trends in privacy-preserving collaborative filtering". Data Mining and Knowledge Discovery, 2015, 5 (6): 276–291

[3] Zhiang Wu, Junjie Wu, Jie Cao et al. "HySAD: A semi-supervised hybrid shilling attack detector for trustworthy product recommendation". Proc. of ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2012, pp. 985–993

[4] Deuk Hee Park, Hyea Kyeong Kim, II Young Choi et al. "A literature review and classification of recommender systems research". Expert Systems with Applications, 2012, 39: 10059–10072.

[5] J. Bobadilla, F. Ortega, A. Hernando et al. "Recommender systems survey". Knowledge-Based Systems, 2013, 46: 109–132

[6] Zibin Zheng, Hao Ma, M. R. Lyu et al. "Qos-aware web service recommendation by collaborative filtering". IEEE Transactions on Services Computing, 2011, 4 (2): 140–152.

[7] F. Cacheda, V. Carneiro, D. Fernndez et al. "Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender Systems". ACM Transactions on the Web, 2011 5 (1): Article 2

[8] J. Bobadilla, A. Hernando, F. Ortega et al. "Collaborative filtering based on significances". Information Sciences, 2012, 185 (1): 1–17

[9] A. B. Barragans-Martinez, E. Costa-Montenegro, J.C. Burguillo et al. "A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition". Information Sciences, 2010, 180 (22): 4290–4311

[10] Silvia Puglisi , Javier Parra-Arnau , Jordi Forn et al. "On content-based recommendation and user privacy in social-tagging systems". Computer Standards & Interfaces, 2015, 41: 17–27

[11] Carrer-Neto, Marła Luisa Hernndez-Alcaraz, Rafael Valencia-Garcła et al. "Social knowledge-based recommender system. Application to the movies domain". Expert Systems with Applications, 2012, 39 (12):10990–11000

[12] Khalid O, Khan M U S, Khan S U et al. "OmniSuggest: A ubiquitous cloud-based context-aware recommendation system for mobile social networks". IEEE Transactions on Services Computing, 2014, 7 (3): 401–414.

[13] Joseph A. Calandrino, Ann Kilzer, Arvind Narayanan et al. "You might also like: Privacy risks of collaborative filtering". Proc. of IEEE Symposium on Security and Privacy (S&P), 2011, pp. 231–247

[14] Jieming Zhu, Pinjia He, Zibin Zheng et al. "A privacy-preserving QoS prediction framework for web service recommendation". Proc. of IEEE International Conference on Web Services (ICWS), 2015, pp. 241-248

[15] HweeHwa Pang , Xuhua Ding , Xiaokui Xiao. "Embellishing text search queries to protect user privacy". Proc. VLDB Endow. 2010, 3 (1–2): 598–607

[16] HweeHwa Pang, Xiaokui Xiao, Jialie Shen. "Obfuscating the topical intention in enterprise text search". Proc. of IEEE International Conference on Data Engineering (ICDE), 2012, pp. 1168–1179

[17] Zongda Wu, Jie Shi, Chenglang Lu et al. "Constructing plausible innocuous pseudo queries to protect user query intention". Information Sciences, 2015, 325: 215 – 226

[18] Huseyin Polat, Wenliang Du. "Privacy-preserving collaborative filtering using randomized perturbation techniques". Proc. of IEEE Conference on Data Mining (ICDM), 2003, pp. 625–628

[19] Feng Zhang, Victor E. Lee, Ruoming Jin. "k-CoRating: Filling up data to obtain privacy and utility". Proc. of AAAI Conference on Artificial Intelligence (AAAI), 2014, pp. 320–327

[20] Yilin Shen, Hongxia Jin. "Privacy-preserving personalized recommendation: An instance-based approach via differential privacy". Proc. of IEEE Conference on Data Mining (ICDM), 2014, pp. 540–549

[21] Alper Bilge, Huseyin Polat. "An improved privacy-preserving DWT-based collaborative filtering scheme". Expert Systems with Applications. 2012, 39: 3841–3854

[22] Lucila Ishitani, Virgilio Almeida, Wagner Meira Jr et al. "Masks: Bringing anonymity and personalization together". IEEE Security and Privacy Magazine, 2003, 1 (3): 18–23

[23] Zhifeng Luo, Shuhong Chen, Yutian Li. "A distributed anonymization scheme for privacy-preserving recommendation systems", Proc. of IEEE Conference on Software Engineering and Service Science (ICSESS), 2013, pp. 491–494

[24] Josyula R. Rao, Pankaj Rohatgi. Can pseudonymity really guarantee privacy?. Proc. of USENIX Security Symposium, 2000, pp. 85–96

[25] Narayanan, A., and Shmatikov, V. Robust de-anonymization of large sparse datasets. Proc. of IEEE Symposium on Security and Privacy (S&P), 2008, pp. 111–125.

[26] Michaela Goetz, Suman Nath. "Privacy-aware personalization for mobile advertising". Proc. of ACM Conference on Computer and Communications (CCS), 2012, pp. 662–673

[27] Yabo Xu, Ke Wang, Benyu Zhang. "Privacy-enhancing personalized web search". Proc. of World Wide Web Conference (WWW), 2007, pp. 591–600

[28] Gang Chen, Bai He, Lidan Shou et al. "UPS: efficient privacy protection in personalized web search". Proc. of ACM SIGIR Conference on Research on Development in Information Retrieval (SIGIR), 2011, pp. 615–624

[29] Lidan Shou, He Bai, Ke Chen et al. "Supporting privacy protection in personalized web search". IEEE Transactions on Knowledge and Data Engineering, 2012, 26 (2): 1–14

[30] Shlomo Berkovsky, Tsvi Kuflik, Francesco Ricci."The impact of data obfuscation on the accuracy of collaborative filtering". Expert Systems with Applications. 2012, 39: 5033–5042

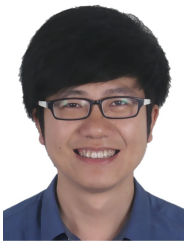
[31] J Vera-Del-Campo, J Pegueroles, J Hernndez-Serrano, et al. "DocCloud: A document recommender system on cloud computing with plausible deniability". Information Sciences, 2014, 258 (3): 387–402

[32] Shang Shang, Yuk Hui, Pan Hui et al. "Beyond personalization and anonymity: towards a group-based recommender system". Proc. of ACM Symposium on Applied Computing (SAC), 2014, pp. 266–273

[33] Guandong Xu, Zongda Wu, Guiling Li et al. "Improving contextual advertising matching by using wikipedia thesaurus knowledge". Knowledge and Information Systems, 2015, 43 (3): 599–631

[34] Babbar R, Metzig C, Partalas I, et al. "On power law distributions in large-scale taxonomies". ACM SIGKDD Explorations Newsletter, 2014, 16 (1): 47–56

[35] Dipasree Pal, Mandar Mitra, Kalyankumar Datta. "Improving query expansion using WordNet". Journal of the Association for Information Science and Technology, 2014, 65 (12): 2469–2478



Zongda Wu is an associate professor in Computer Science at Wenzhou University. He received his Ph.D. degree in Computer Science from Huazhong University of Science and Technology (HUST) in 2009. From 2012 to 2014, he worked as a postdoctoral research fellow with the School of Computer Science and Technology at University of Science and Technology of China (USTC). His research interests are primarily in the area of information retrieval and personal privacy.

Guiling Li is an associate professor in School of Computer Science at China University of Geosciences(Wuhan). She received her Ph.D. degree in Computer Science from Huazhong University of Science and Technology (HUST) in 2012. She is a visiting scholar at University of Illinois at Chicago in 2015. Her research interests are primarily in the area of data mining and knowledge discovery, data management for time series data, social media etc.

Qi Liu is an associate professor in University of Science and Technology of China (USTC). He received his Ph.D. in Computer Science from USTC. His general area of research is data mining and knowledge discovery. He has published prolifically in refereed journals and conference proceedings, e.g., TKDE, TOIS, TKDD, TIST, KDD, IJCAI, ICDM and CIKM. He has served regularly in the program committees of a number of conferences.

Guandong Xu is a research fellow in Advanced Analytics Institute, Faculty of Engineering and Information Technology, University of Technology, Sydney. He obtained his Ph.D. degree in Computer Science from Victoria University in 2008. After that he worked as a postdoctoral research fellow in Centre for Applied Informatics at Victoria University and then postdoc in Department of Computer Science at Aalborg University, Denmark. His research interests include web information retrieval, web mining, web services etc.

Enhong Chen is a professor and a vice dean at School of Computer Science and Technology, University of Science and Technology of China (USTC), IEEE senior member. He received his Ph.D. degree in Computer Science from USTC in 1996. He has been actively involved in the research community by serving as a PC member for more than 50 conferences, such as KDD, AAAI, ICDM and SDM. His research interests include semantic web, machine learning, data mining, web information processing etc.