Hindawi
Complexity
Volume 2017, Article ID 3437854, 9 pages
https://doi.org/10.1155/2017/3437854

Research Article

Privacy-Preserving and Scalable Service Recommendation Based on SimHash in a Distributed Cloud Environment

Yanwei Xu,[1] Lianyong Qi,[1] Wanchun Dou,[2] and Jiguo Yu[1]

[1]*School of Information Science and Engineering, Chinese Academy of Education Big Data, Qufu Normal University, Rizhao 276826, China*
[2]*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China*

Correspondence should be addressed to Lianyong Qi; lianyongqi@gmail.com

Received 4 August 2017; Revised 9 November 2017; Accepted 19 November 2017; Published 18 December 2017

Academic Editor: Chuan Zhou

With the increasing volume of web services in the cloud environment, Collaborative Filtering- (CF-) based service recommendation has become one of the most effective techniques to alleviate the heavy burden on the service selection decisions of a target user. However, the service recommendation bases, that is, historical service usage data, are often distributed in different cloud platforms. Two challenges are present in such a cross-cloud service recommendation scenario. First, a cloud platform is often not willing to share its data to other cloud platforms due to privacy concerns, which decreases the feasibility of cross-cloud service recommendation severely. Second, the historical service usage data recorded in each cloud platform may update over time, which reduces the recommendation scalability significantly. In view of these two challenges, a novel privacy-preserving and scalable service recommendation approach based on SimHash, named $\text{SerRec}_{\text{SimHash}}$, is proposed in this paper. Finally, through a set of experiments deployed on a real distributed service quality dataset *WS-DREAM*, we validate the feasibility of our proposal in terms of recommendation accuracy and efficiency while guaranteeing privacy-preservation.

1. Introduction

With the ever-increasing volume and variety of web services in various web-based communities, it becomes a challenging task to find the web services that a target user is really interested in [1–3]. In this situation, various service recommendation techniques are introduced to alleviate the heavy burden on the service selection decisions of target users, for example, the well-adopted user-based Collaborative Filtering (i.e., UCF). According to traditional UCF, the similar friends of a target user are often employed to make recommendations to the target user [4]. Therefore, similar friend discovery is the key step to the subsequent service recommendation.

Generally, the bases for similar friend discovery, that is, historical service usage data (e.g., service quality observed by users) are centralized; in this situation, it is easy to determine the similar friends of a target user. However, in the age of IoT (Internet of Things), the quality data of various services are often monitored and collected by geographically distributed sensors and stored in different cloud platforms [5]. In this situation, the historical service usage data are not centralized, but distributed. Such a distributed service recommendation scenario calls for data sharing and collaboration between different cloud platforms. However, as work [6] indicates, this kind of cross-platform data sharing may bring additional privacy leakage risk, which decreases the feasibility of cross-cloud service recommendation severely. Besides, for the involved multiple cloud platforms, their volume of service quality data may become increasingly huge with updates over time, which leads to a frequent recalculation of user similarity and hence reduces the recommendation scalability significantly.

In view of these two challenges, a novel privacy-preserving and scalable service recommendation approach based on SimHash, named $\text{SerRec}_{\text{SimHash}}$, is put forward in this paper. Our $\text{SerRec}_{\text{SimHash}}$ can achieve a good recommendation performance in terms of accuracy, efficiency, and privacy-preservation.

Generally, the contributions of this paper are threefold:

(1) To the best of our knowledge, existing research works seldom consider the service recommendation in a distributed cloud environment, as well as the resulting privacy-preservation problems. In this paper, we formalize this privacy-preserving service recommendation problem and clarify its research significance.

(2) We put forward a novel service recommendation approach based on offline SimHash technique [7], named $\text{SerRec}_{\text{SimHash}}$, to protect the private information of most users in different cloud platforms, and meanwhile improve the service recommendation efficiency and scalability.

(3) We conduct a set of experiments based on a real distributed service quality dataset *WS-DREAM* to validate the feasibility of our proposed $\text{SerRec}_{\text{SimHash}}$ approach. Experiment results show that $\text{SerRec}_{\text{SimHash}}$ achieves a good performance in terms of recommendation accuracy and scalability while guaranteeing privacy-preservation.

The rest of the paper is organized as follows. Related work is presented in Section 2. Research motivation is demonstrated in Section 3. In Section 4, we introduce the details of our proposed service recommendation approach $\text{SerRec}_{\text{SimHash}}$. In Section 5, a set of experiments are conducted based on *WS-DREAM* dataset, to validate the feasibility and advantages of our proposal. And finally, in Section 6, we summarize the paper and point out the future research directions.

2. Related Work

Collaborative Filtering (i.e., CF) has become one of the most effective techniques in various recommender systems. User-based CF and item-based CF are brought forth for high-quality service recommendation in [4] and [8], respectively. In order to combine their advantages, a hybrid CF recommendation approach is introduced in [9]. Experiment results show that the hybrid approach improves the recommendation performance. As the quality of a web service often depends on the service execution context (e.g., time, location), time-aware CF and location-aware CF are proposed in [10] and [11], respectively, to improve the accuracy of recommended results. However, the above approaches cannot handle the recommendation problems where historical service usage data are very sparse. In view of this drawback, a belief propagation-based approach is proposed in [12], to find the potential friends of the target user.

However, the above approaches all assume that the service recommendation bases, that is, historical service usage data, are centralized, without considering the distributed service recommendation scenarios as well as the resulting privacy leakage risk. In view of this drawback, the authors in [13] suggest that a user should release only a small portion of his/her observed service quality data to the public so that the remaining majority of user-service quality data are secure. However, the released small portion of data can still reveal part of a user's private information. In order to protect user privacy completely, the data obfuscation technique is adopted in [14] to hide the real service quality data by adding an obfuscated data item. However, as the service quality data used to make service recommendations have been obfuscated, the recommendation accuracy is decreased accordingly; besides, additional time cost is brought by the adopted data obfuscation operation. Similarly, a segment-based data hiding approach is introduced in [15], where each piece of user-service quality data is divided into several data segments, and then the data segments are employed to calculate user similarity approximately and make further service recommendation. However, there are still two shortcomings in this approach. First, the data segmentation process often takes much time, which decreases the recommendation efficiency heavily. Second, it fails to protect some important privacy information appropriately, for example, the information of the service intersection commonly invoked by two users. Locality-sensitive hashing technique is recruited in [16] to protect and realize the privacy-preservation purpose; however, only partial private information of users can be protected very well.

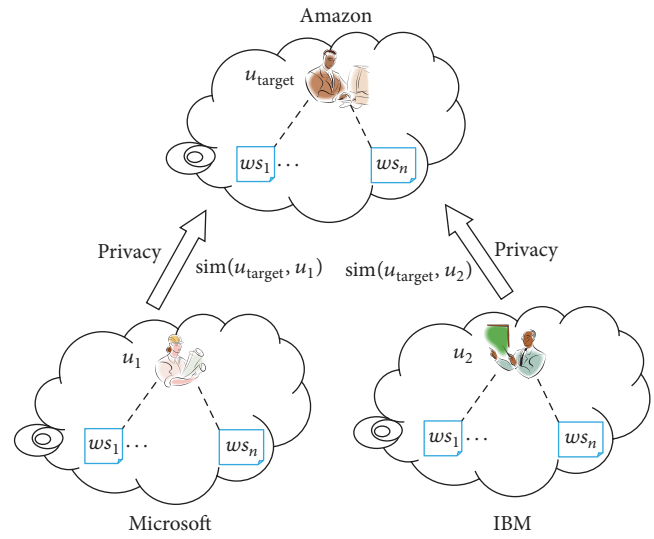

Figure 1: Cross-cloud service recommendation: an example.

In view of the drawbacks of existing approaches, a novel privacy-preserving and scalable service recommendation approach based on SimHash, that is, $\text{SerRec}_{\text{SimHash}}$, is proposed in this paper, to cope with the service recommendation problems in the distributed cloud environment. Next, an example is presented in Section 3 to further demonstrate the research motivation of our paper.

3. Research Motivation

An intuitive example is presented in Figure 1 to motivate our paper. Here, u_{target} denotes a target user to whom *Amazon* platform intends to recommend services; u_1 and u_2 are two users whose observed service quality data are recorded in *Microsoft* and *IBM* platforms, respectively; $\{ws_1, \ldots, ws_n\}$ are the candidate services for recommendation. Specifically, if a

Step 1 (buliding user indexes offline based on SimHash). For each user $u_i \in U$, calculate his/her hash value $H(u_i)$ offline based on SimHash. Then $H(u_i)$ is regarded as the index for u_i.
Step 2 (finding "probably similar" friends of the target user). According to the same hash function adopted in Step 1, calculate user index for u_{target}, that is, $H(u_{\text{target}})$. If the Hamming Distance between $H(u_{\text{target}})$ and $H(u_i)$ is smaller than 3, then u_i is considered as a "probably similar" friend of u_{target}.
Step 3 (finding "really similar" friends of the target user). For a "probably similar" friend u_i obtained in Step 2, calculate his/her similarity with u_{target}; if the similarity is larger than a threshold P, then u_i is a "really similar" friend of u_{target}.
Step 4 (service recommendation). According to u_{target}'s "really similar" friends derived in Step 3, predict the quality of services never invoked by u_{target} and recommend the quality-optimal services to u_{target}.

Box 1: Four steps of service recommendation approach $\text{SerRec}_{\text{SimHash}}$.

user has never invoked a service, the corresponding service quality data is null.

Next, according to traditional UCF, the first step is to calculate user similarity $\text{sim}(u_{\text{target}}, u_1)$ and $\text{sim}(u_{\text{target}}, u_2)$ so as to determine the similar friends of u_{target}. However, the above user similarity calculation process involves the cross-platform collaborations and hence faces the following two challenges:

(1) Generally, *Microsoft* and *IBM* are not willing to share their recorded service quality data to *Amazon* due to privacy concerns, which decreases the feasibility of cross-cloud user similarity calculation and subsequent service recommendation severely.

(2) In *Amazon*, *Microsoft*, and *IBM*, the volume of service quality data may become increasingly huge with updates over time; in this situation, the collaboration efficiency and scalability are often reduced significantly and hence cannot satisfy the quick recommendation requirements from target users.

In view of these two challenges, a privacy-preserving and scalable service recommendation approach, that is, $\text{SerRec}_{\text{SimHash}}$, is proposed in this paper, which will be introduced in detail in the next section.

4. A SimHash-Based Service Recommendation Approach

In this section, a privacy-preserving and scalable approach, that is, $\text{SerRec}_{\text{SimHash}}$, is proposed to handle the distributed service recommendation problems. The main idea behind $\text{SerRec}_{\text{SimHash}}$ is: the users who have invoked the most common services can be regarded as "probably similar" friends [17]; therefore, we first utilize SimHash to look for a small number of "probably similar" friends of the target user, in a privacy-preserving and scalable way; afterwards, we determine the target user's "really similar" friends from the "probably similar" ones; finally, we make service recommendations to the target user based on the preferences of his/her "really similar" friends.

Concretely, $\text{SerRec}_{\text{SimHash}}$ consists of the four steps in Box 1. Here, u_{target} denotes a target user, U is the user set in multiple involved cloud platforms, $\{ws_1, \ldots, ws_n\}$ is the candidate service set, and $H(u)$ denotes the hash value of user u based on SimHash.

Step 1 (building user indexes offline based on SimHash). For each user $u_i \in U$, according to his/her historical service invocation records, we can build his/her index offline, denoted by $H(u_i)$, based on SimHash technique (see Figure 2). Here, m and n denote the number of users and number of services, respectively. Next, we introduce how to obtain $H(u_i)$.

First, for each service $ws_j \in WS$, we can generate a random r-dimensional 0-1 vector v_j where $r = \lceil \log_2^n \rceil$ (here, $\lceil x \rceil$ means the upper integer of x; e.g., $\lceil 3.4 \rceil = 4$). Considering the example in Figure 2, $r = 6$ and the 0-1 vector corresponding to service ws_1, that is, $v_1 = (0, 0, 0, 0, 0, 1)$ holds. Then according to the historical service invocation records, u_i can be denoted by a n-dimensional vector $h_1(u_i) = (V_1, \ldots, V_n)$ in (1).

$$V_j = \begin{cases} v_j & \text{if } u_i \text{ has invoked } ws_j \text{ before} \\ \text{Null} & \text{if } u_i \text{ has never invoked } ws_j \text{ before.} \end{cases} \quad (1)$$

Next, in vector $h_1(u_i)$, we drop the dimensions with null value and replace value "0" by value "–1", after which a new vector $h_2(u_i)$ is achieved (see Figure 2(2)). Then for the derived $n * r$ (at most) matrix corresponding to vector $h_2(u_i)$, we calculate the sum of its each column. Afterwards, we obtain a new vector $h_3(u_i)$ (see Figure 2(3)), where the positive and negative values are replaced by "1" and "0", respectively, after which r-dimensional 0-1 vector $H(u_i)$ (see Figure 2(4)) is obtained. Then according to SimHash theory [6], $H(u_i)$ can be regarded as the index for user u_i. This way, we can build indexes for all the users in set U.

For a user, his/her historical service invocation data are recorded by a certain cloud platform (e.g., *Amazon* or *Microsoft* or *IBM* in Figure 1); therefore, the user index can be built offline beforehand by the cloud platform so as to

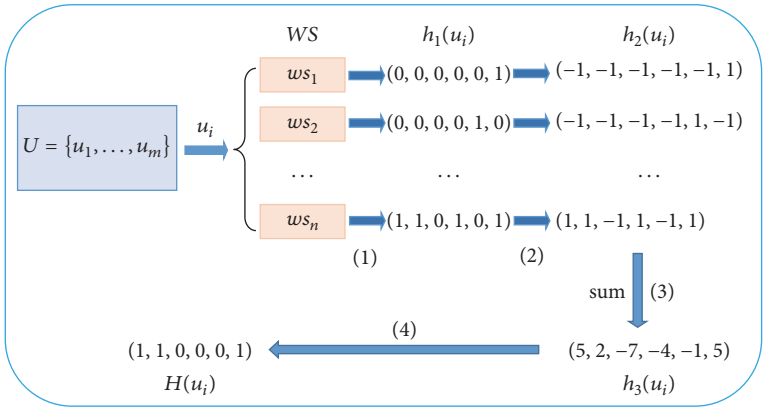

Figure 2: User indexes building procedure in Step 1: an example.

reduce the time cost. Besides, through SimHash, each user u_i is encapsulated into a less-sensitive user index $H(u_i)$, without revealing his/her sensitive information (e.g., *whether he/she has invoked a service or not*, *a service's running quality observed by him/her*) to other platforms. Therefore, user privacy is protected.

Step 2 (finding "probably similar" friends of the target user). According to same hash functions adopted in Step 1, we calculate the index for the target user, that is, $H(u_{\text{target}})$. Next, we calculate the Hamming Distance between $H(u_{\text{target}})$ and $H(u_i)$ $(1 \le i \le m)$, denoted by $D(H(u_{\text{target}}), H(u_i))$. Concretely, suppose $H(u_{\text{target}})$ and $H(u_i)$ are denoted by r-dimensional vectors $(v_{\text{target-}1}, \ldots, v_{\text{target-}r})$ and $(v_{i\text{-}1}, \ldots, v_{i\text{-}r})$, respectively. Then $D(H(u_{\text{target}}), H(u_i))$ can be calculated by (2), where a_k is a Boolean value calculated by (3). Here, symbol "$\oplus$" denotes the XOR operation.

$$D\left(H\left(u_{\text{target}}\right), H\left(u_i\right)\right) = \sum a_k \quad (1 \le k \le r) \tag{2}$$

$$a_k = v_{\text{target-}k} \oplus v_{i\text{-}k} = \begin{cases} 1 & \text{if } v_{\text{target-}k} \ne v_{i\text{-}k} \\ 0 & \text{if } v_{\text{target-}k} = v_{i\text{-}k}. \end{cases} \tag{3}$$

According to SimHash [6], if $D(H(u_{\text{target}}), H(u_i)) < 3$ holds, then we can conclude that the services invoked by u_{target} and u_i are approximately the same. In other words, u_i can be regarded as a "probably similar" friend of u_{target} and then put into set Prob_Sim_Friend(u_{target}). Moreover, the size of Prob_Sim_Friend(u_{target}), that is, |Prob_Sim_Friend(u_{target})|, is often small ($\ll m$) due to the nature of SimHash.

Step 3 (finding "really similar" friends of the target user). The users in set Prob_Sim_Friend(u_{target}) (obtained in Step 2) are only "probably similar" friends of the target user, not necessarily "really similar" friends. Considering this point, in this step, we further determine the "really similar" friends of the target user from set Prob_Sim_Friend(u_{target}). Concretely, for any $u_i \in$ Prob_Sim_Friend(u_{target}), we calculate his/her similarity with u_{target}, that is, Sim(u_{target}, u_i), based on Pearson Correlation Coefficient (PCC) [18] in (4) (as |Prob_Sim_Friend(u_{target})| is often small, only a small number of users take part in the user similarity calculation process in (4); as a consequence, we can protect the private service quality data observed by the remaining majority of users).

In (4), symbol I denotes the service intersection invoked by u_{target} and u_i; q is a quality dimension of web services, for example, *response time*; $q_{\text{target-}j}$ and $q_{i\text{-}j}$ represent service ws_j's quality values over dimension q observed by u_{target} and u_i, respectively; $\overline{q_{\text{target}}}$ and $\overline{q_i}$ denote u_{target}'s and u_i's average quality values over dimension q of all the services invoked by u_{target} and u_i, respectively. Specifically, if the service intersection I = Null, Sim(u_{target}, u_i) = 0 holds. Moreover, if condition in (5) holds, u_i can be regarded as a "really similar" friend of u_{target} and put into set Real_Sim_Friend(u_{target}). Here, symbol P is a predefined similarity threshold ($0.5 \le P \le 1$).

$$\begin{aligned} &\text{Sim}\left(u_{\text{target}}, u_i\right) \\ &= \frac{\sum_{ws_j \in I}\left(q_{\text{target-}j} - \overline{q_{\text{target}}}\right) * \left(q_{i\text{-}j} - \overline{q_i}\right)}{\sqrt{\sum_{ws_j \in I}\left(q_{\text{target-}j} - \overline{q_{\text{target}}}\right)^2} * \sqrt{\sum_{ws_j \in I}\left(q_{i\text{-}j} - \overline{q_i}\right)^2}} \end{aligned} \tag{4}$$

$$\text{Sim}\left(u_{\text{target}}, u_i\right) \ge P. \tag{5}$$

Step 4 (service recommendation). For all the users u_i in set Real_Sim_Friend(u_{target}) (obtained in Step 3), we rank them by Sim(u_{target}, u_i) (see (4)) in descending order and return the Top 3 (at most) similar friends (denoted by set $U_{\text{top-3}}$) of the target user. Afterwards, for each service never invoked by the target user, denoted by ws_j, we predict its quality over dimension q observed by u_{target}, that is, $q_{\text{target-}j}$, by (6), where $u_i \in U_{\text{top-3}}$ and $q_{i\text{-}j}$ represents service ws_j's quality value over dimension q observed by u_i. Finally, we select the service with the optimal predicted quality and recommend it to the target user, so as to finish the whole service recommendation process.

$$q_{\text{target-}j} = \frac{\sum_{u_i \in U_{\text{top-3}}} \text{Sim}\left(u_{\text{target}}, u_i\right) * q_{i\text{-}j}}{\sum_{u_i \in U_{\text{top-3}}} \text{Sim}\left(u_{\text{target}}, u_i\right)}. \tag{6}$$

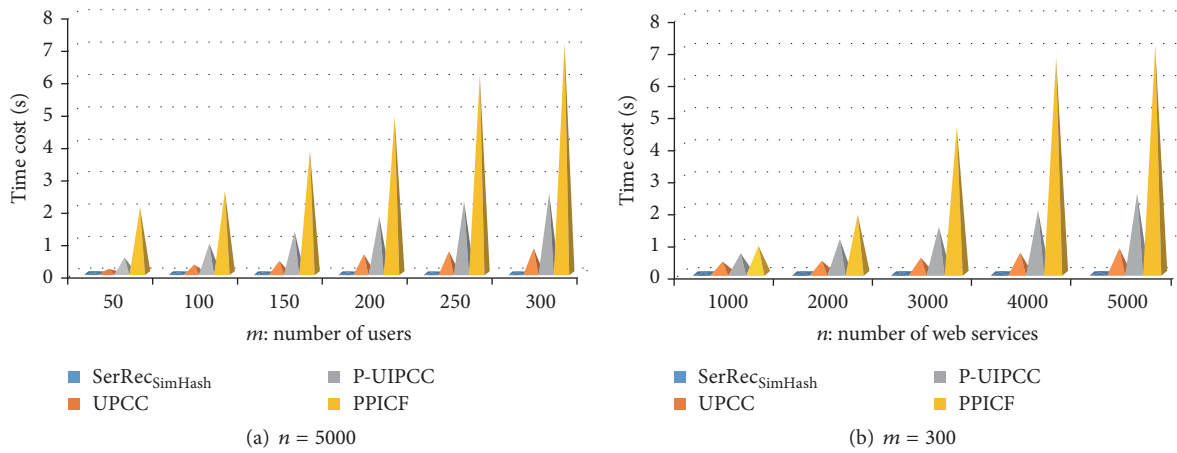

Figure 3: Recommendation efficiency comparison of four approaches.

5. Experiments

5.1. Experiment Configurations. In this section, a set of experiments are deployed on *WS-DREAM* dataset [19] to validate the feasibility of our proposed recommendation approach SerRec$_{SimHash}$. *WS-DREAM* is a real-world service quality (e.g., *throughput*) set obtained from 339 users on 5825 web services from different countries. To simulate the recommendation scenario that we focus on in this paper (i.e., recommendation in a distributed cloud environment), each country is regarded as a cloud platform.

We compare our approach with a benchmark approach *UPCC* [20] and another two up-to-date privacy-preserving recommendation approaches, that is, *P-UIPCC* [14] and *PPICF* [15]. Many works, for example, [21–23], consider the *time cost* and the *MAE* as the evaluation criteria; likewise, we also adopt these two criteria in this paper (in our SerRec$_{SimHash}$ approach, most user privacy information, e.g., *whether a user has invoked a service or not* and *service quality observed by a user*, can be protected by the intrinsic nature of SimHash; therefore, we will not evaluate the capability of privacy-preservation of our proposal here).

(1) *Time cost*: the consumed time for recommending a web service to the target user, which can be used to measure the recommendation efficiency and scalability.

(2) *MAE*: the difference between the predicted quality and real quality of recommended services (the smaller the better), which can be used to measure the recommendation accuracy.

The density of user-service quality matrix is set at 3% and the experiments are conducted on a Lenovo laptop with 2.40 GHz processor and 12.0 GB RAM. The laptop is running under Windows 10 and JAVA 8. Each experiment is repeated 10 times and the average experiment results are reported.

5.2. Experiment Results and Analyses. Concretely, the following four profiles are tested and compared, respectively. Here, m and n denote the number of users and number of web services, respectively; user similarity threshold $P = 0.5$ holds.

Profile 1: Recommendation Efficiency Comparison. In this profile, we test the time cost of our proposal with respect to m and n and compare it with the remaining three approaches. The experiment parameters are set as follows: m is varied from 50 to 300; n is varied from 1000 to 5000. The concrete experiment results are shown in Figure 3 ($n = 5000$ holds in Figure 3(a) and $m = 300$ holds in Figure 3(b)).

As can be seen from Figure 3(a), the time costs of *UPCC*, *P-UIPCC*, and *PPICF* approaches all increase approximately linearly with the growth of m; this is because more time is needed to calculate user similarities when the number of users, that is, m, becomes larger, while our proposed SerRec$_{SimHash}$ approach outperforms those three ones in terms of time cost, as most jobs (e.g., user indexes building) can be finished offline before a service recommendation request arrives. Furthermore, after the hashing process, only a few "probably similar" friends of the target user are obtained; as a consequence, little time is taken to find the "really similar" friends of the target user from the small number of "probably similar" friends. Due to the above two reasons, the recommendation efficiency and scalability of our proposed SerRec$_{SimHash}$ approach are improved significantly. Similar comparison results can be observed from Figure 3(b), whose reasons are the same as those in Figure 3(a) and will not be discussed repeatedly.

Profile 2: Recommendation Accuracy Comparison. Accuracy is a key criterion to evaluate the quality of a recommender system. Therefore, in this profile, we test the *MAE* (the smaller the better) of our proposal and compare it with the remaining three approaches. The experiment parameters are set as follows: m is varied from 50 to 300; n is varied from 1000 to 5000. The experiment results are presented in Figure 4 ($n = 5000$ holds in Figure 4(a) and $m = 150$ holds in Figure 4(b)).

As Figure 4 shows, the recommendation accuracy values of *P-UIPCC* and *PPICF* approaches are often low (i.e., *MAE* values are high), as many approximate operations are

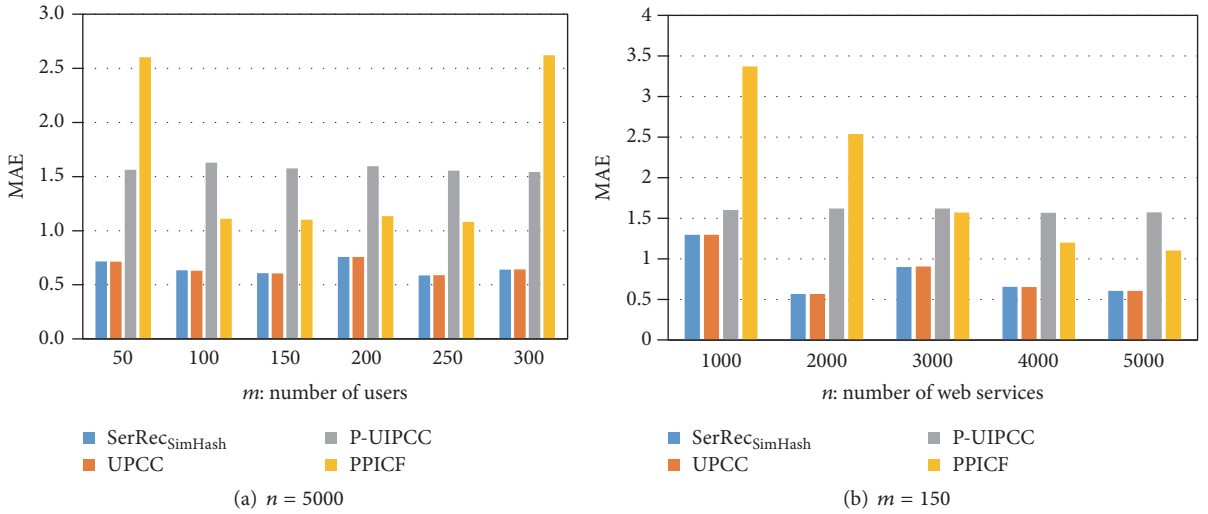

Figure 4: Recommendation accuracy comparison of four approaches.

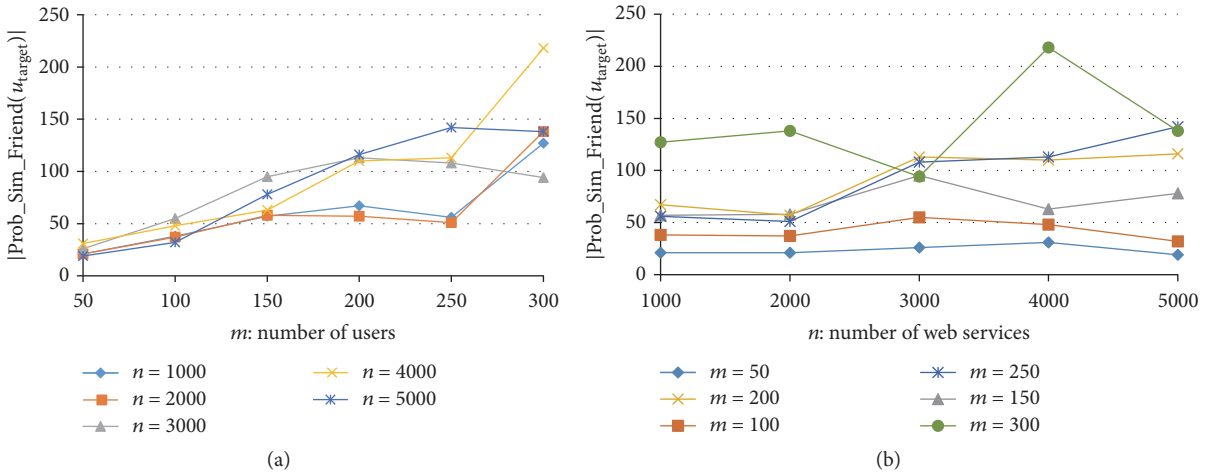

Figure 5: |Prob_Sim_Friend(u_{target})| with respect to m and n.

recruited in these two approaches to protect the user privacy, for example, data obfuscation technique adopted in *P-UIPCC* approach and data segmentation-merging technique recruited in *PPICF* approach. These techniques on one hand can protect the privacy information of users effectively and on the other hand decrease the accuracy of recommended results, while our proposed SerRec$_{SimHash}$ approach achieves the approximate service recommendation accuracy as the benchmark approach *UPCC*, as the SimHash technique adopted in SerRec$_{SimHash}$ can guarantee finding the "really similar" friends of a target user with high probability and thereby can achieve a high recommendation accuracy.

Profile 3: Number of "Probably Similar" Friends of the Target User in SerRec$_{SimHash}$ with respect to m and n. In our SerRec$_{SimHash}$ approach, a small number of "probably similar" friends (the number is |Prob_Sim_Friend(u_{target})|) of a target user are obtained. In this profile, we test the relationship between |Prob_Sim_Friend(u_{target})| and m and n. Experiment parameters are set as follows: m is varied from 50 to 300; n is varied from 1000 to 5000. The concrete experiment results are presented in Figure 5.

As Figure 5(a) shows, the value of |Prob_Sim_Friend(u_{target})| increases approximately linearly with the growth of m; this is because it is more probable to find a "probable friend" of the target user when the candidate user space becomes larger. As Figure 5(b) shows, the value of |Prob_Sim_Friend(u_{target})| increases relatively slowly when n rises, whose reasons are twofold. First, more valuable recommendation information is available when the number of services, that is, n, increases; as a consequence, more "probably similar" friends of the target user can be found by our proposed SerRec$_{SimHash}$ approach. Second, due to the intrinsic nature of SimHash technique adopted in our SerRec$_{SimHash}$ approach, the number of services, that is, n, does not influence the finding process of "probably similar"

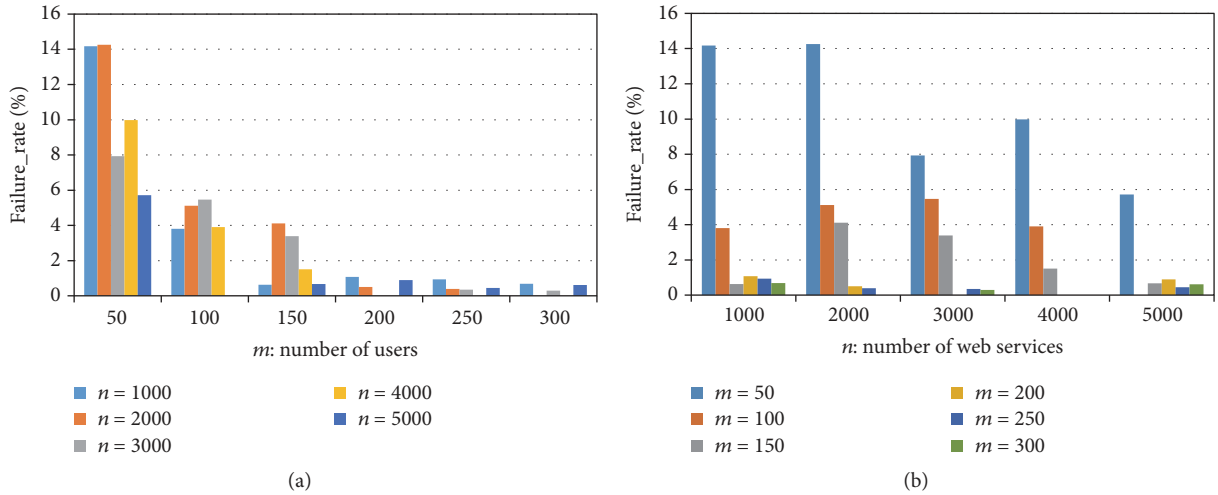

FIGURE 6: Recommendation failure rate of $\text{SerRec}_{\text{SimHash}}$ with respect to m and n.

friends directly in our proposal and, hence, the influence of parameter n stressed on $|\text{Prob\_Sim\_Friend}(u_{\text{target}})|$ is not so obvious as that in Figure 5(a).

Profile 4: Recommendation Failure Rate of $\text{SerRec}_{\text{SimHash}}$ with respect to m and n. The SimHash technique adopted in this paper is essentially a kind of probability-based similar neighbor finding approach [24]. Therefore, our proposed $\text{SerRec}_{\text{SimHash}}$ approach may fail to return any recommended result in certain situations, that is, a failure occurs. Considering this point, in this profile, we test the recommendation failure rate of $\text{SerRec}_{\text{SimHash}}$ with respect to m and n. Concretely, failure rate can be measured by the equation in (7), where $\text{Num}_{\text{success}}$ and Num_{fail} represent the number of successful service recommendations and the number of failed service recommendations, respectively. The concrete experiment parameters are set as follows: m is varied from 50 to 300; n is varied from 1000 to 5000. The experiment results are shown in Figure 6.

$$\text{failure\_rate} = \left(\frac{\text{Num}_{\text{fail}}}{(\text{Num}_{\text{success}} + \text{Num}_{\text{fail}})} \right) * 100\%. \quad (7)$$

As Figure 6(a) shows, the failure rate of $\text{SerRec}_{\text{SimHash}}$ approach decreases with the growth of m; this is because it is more probable to find the "probably similar" friends of a target user when the candidate space of users becomes larger. Moreover, the failure rate approaches 0 when m is large enough, for example, when m = 200, 250, or 300. Figure 6(b) shows the relationship between failure rate of $\text{SerRec}_{\text{SimHash}}$ and the number of services, that is, n. As indicated in Figure 6(b), the failure rate approximately drops with the growth of n; this is because when the number of services increases, the probability that two users have invoked the common services grows accordingly, and hence it is more probable to find the "probably similar" friends of a target user. Furthermore, as can be seen from Figure 6(b), the failure rate of $\text{SerRec}_{\text{SimHash}}$ approach approaches 0 when n is large enough, for example, when n = 5000.

5.3. Shortcoming Analyses. In terms of the experiment results, we can conclude that $\text{SerRec}_{\text{SimHash}}$ approach achieves a good tradeoff among the recommendation accuracy, efficiency, and failure rate while guaranteeing privacy-preservation. However, other evaluation criteria are not discussed in depth, such as the well-known consistency criterion (e.g., the inferred friend consistency) suggested in work [25]. Besides, as [26] indicates, weight plays an important role in the final evaluation results; however, we do not consider the weight of found friends in this paper for simplicity.

6. Conclusions and Future Work

In the distributed cloud environment, a cloud platform is often not willing to share its recorded user-service invocation data with other cloud platforms due to privacy concerns, which decreases the feasibility of cross-cloud collaborative service recommendation severely. Besides, the user-service invocation data recorded by each cloud platform may update over time, which reduces the recommendation scalability significantly. In view of these two challenges, a novel privacy-preserving and scalable service recommendation approach based on SimHash, that is, $\text{SerRec}_{\text{SimHash}}$, is put forward in this paper. To validate the feasibility of our proposal, we conduct a set of experiments based on a real distributed service quality dataset *WS-DREAM*. Experiment results show that $\text{SerRec}_{\text{SimHash}}$ outperforms the other up-to-date approaches in terms of recommendation accuracy and efficiency while guaranteeing privacy-preservation.

As work [27] indicates, SimHash is essentially a probability-based search technique and, hence, failure is inevitable in certain situations. Considering this point, in the future, we will continue to refine our proposal so as to further decrease the recommendation failure rate and boost the recommendation robustness. Besides, due to the inherent shortcoming of various hash-based privacy-preservation techniques suggested in [28], it is hard to evaluate the privacy-preservation performance of our proposal. In the future, we hope to find well-adopted technical criteria

to evaluate the effectiveness of our proposal in terms of privacy-preservation. Moreover, work [29] proposes to utilize the semantic information to improve the retrieval performance; likewise, we hope to refine our work by adding more semantic information in the future.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper is partially supported by the Natural Science Foundation of China (no. 61402258, no. 61672276, no. 61373027, and no. 61672321), Key Research and Development Project of Jiangsu Province (no. BE2015154, no. BE2016120), and Open Project of State Key Laboratory for Novel Software Technology (no. KFKT2016B22).

References

[1] Y. Xia, M. Zhou, X. Luo, Q. Zhu, J. Li, and Y. Huang, "Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 1, pp. 162–170, 2015.

[2] N. Zhang, J. Wang, and Y. Ma, "Mining domain knowledge on service goals from textual service descriptions," *IEEE Transactions on Services Computing*, no. 99, article 1, 2017.

[3] J. Wang, Z. Zhu, J. Liu, C. Wang, and Y. Xu, "An approach of role updating in context-aware role mining," *International Journal of Web Services Research*, vol. 14, no. 2, pp. 24–44, 2017.

[4] H. Rong, S. Huo, C. Hu, and J. Mo, "User similarity-based collaborative filtering recommendation algorithm," *Journal on Communication*, vol. 35, no. 2, pp. 16–24, 2014.

[5] L. Qi, H. Xiang, W. Dou, C. Yang, Y. Qin, and X. Zhang, "Privacy-preserving distributed service recommendation based on locality-sensitive hashing," in *Proceedings of the 2017 IEEE International Conference on Web Services (ICWS)*, pp. 49–56, Honolulu, Hawaii, USA, June 2017.

[6] J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun, and Y. Xiang, "Block design-based key agreement for group data sharing in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, no. 99, article 1, 2017.

[7] Q. Jiang and M. Sun, "Semi-supervised SimHash for efficient document similarity search," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 93–101, 2011.

[8] K.-Y. Chung, D. Lee, and K. J. Kim, "Categorization for grouping associative items using data mining in item-based collaborative filtering," *Multimedia Tools and Applications*, vol. 71, no. 2, pp. 889–904, 2014.

[9] C. Jiang, R. Duan, H. K. Jain, S. Liu, and K. Liang, "Hybrid collaborative filtering for high-involvement products: a solution to opinion sparsity and dynamics," *Decision Support Systems*, vol. 79, pp. 195–208, 2015.

[10] X. Wang, J. Zhu, Z. Zheng, W. Song, Y. Shen, and M. R. Lyu, "A spatial-temporal qoS prediction approach for time-aware web service recommendation," *ACM Transactions on the Web (TWEB)*, vol. 10, no. 1, article 7, 2016.

[11] C. Yu and L. Huang, "A Web service QoS prediction approach based on time- and location-aware collaborative filtering," *Service Oriented Computing and Applications*, vol. 10, no. 2, pp. 135–149, 2016.

[12] L. Qi, X. Xu, X. Zhang et al., "Structural balance theory-based e-commerce recommendation over big rating data," *IEEE Transactions on Big Data*, no. 99, article 1, 2016.

[13] W. Dou, X. Zhang, J. Liu, and J. Chen, "HireSome-II: towards privacy-aware cross-cloud service composition for big data applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 455–466, 2015.

[14] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "A privacy-preserving qos prediction framework for web service recommendation," in *Proceedings of the IEEE International Conference on Web Services (ICWS '15)*, pp. 241–248, New York, NY, USA, July 2015.

[15] D. Li, C. Chen, Q. Lv et al., "An algorithm for efficient privacy-preserving item-based collaborative filtering," *Future Generation Computer Systems*, vol. 55, pp. 311–320, 2016.

[16] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren, "A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2594–2608, 2016.

[17] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," *IEEE Transactions on Services Computing*, vol. 4, no. 2, pp. 140–152, 2011.

[18] J. L. Rodgers and A. W. Nicewander, "Thirteen ways to look at the correlation coefficient," *The American Statistician*, vol. 42, no. 1, pp. 59–66, 1988.

[19] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world web services," *IEEE Transactions on Services Computing*, vol. 7, no. 1, pp. 32–39, 2014.

[20] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI '98)*, pp. 43–52, Madison, Wis, USA, July 1998.

[21] Y. Xia, M. Zhou, X. Luo, S. Pang, and Q. Zhu, "Stochastic modeling and performance analysis of migration-enabled and error-prone clouds," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 2, pp. 495–504, 2015.

[22] W. Zheng, M. Zhou, L. Wu et al., "Percentile performance estimation of unreliable iaas clouds and their cost-optimal capacity decision," *IEEE Access*, vol. 5, pp. 2808–2818, 2017.

[23] Z. Zhou, Q. J. Wu, and X. Sun, "Encoding multiple contextual clues for partial-duplicate image retrieval," *Pattern Recognition Letters*.

[24] S. Sood and D. Loguinov, "Probabilistic near-duplicate detection using simhash," in *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM '11)*, pp. 1117–1126, Glasgow, UK, October 2011.

[25] Z. Zhou, Y. Wang, Q. M. J. Wu, C.-N. Yang, and X. Sun, "Effective and efficient global context verification for image copy detection," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 48–63, 2017.

[26] Z. Fu, X. Wu, Q. Wang, and K. Ren, "Enabling central keyword-based semantic extension search over encrypted outsourced data," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 2986–2997, 2017.

[27] Z. Fu, J. Shu, J. Wang, Y. Liu, and S. Lee, "Privacy-preserving smart similarity search based on simhash over encrypted data in cloud computing," *Journal of Internet Technology*, vol. 16, no. 3, pp. 453–460, 2015.

[28] Z. Zhou, Q. M. Jonathan Wu, C. Yang, X. Sun, and Z. Pan, "Coverless image steganography based on histograms of oriented gradients-based hashing algorithm," *Journal of Internet Technology*, vol. 18, no. 5, pp. 1177–1184, 2017.

[29] Z. Fu, F. Huang, K. Ren, J. Weng, and C. Wang, "Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1874–1884, 2017.