Time-aware Distributed Service Recommendation with Privacy-preservation

Lianyong Qi[1], Ruili Wang[2], Chunhua Hu[3,*], Shancang Li[4], Qiang He[5], Xiaolong Xu[6]

[1] School of Information Science and Engineering, Qufu Normal University, China
[2] Institute of Natural and Mathematical Sciences, Massey University, New Zealand
[3] Key Laboratory of Hunan Province for Mobile Business Intelligence, Hunan University of Commerce, China
[4] Computer Science and Creative Technologies Department, University of the West of England, UK
[5] Faculty of Information and Communication Technologies, Swinburne University of Technology, Australia
[6] School of Computer and Software, Nanjing University of Information Science and Technology, China

{lianyongqi@gmail.com, ruili.wang@massey.ac.nz, huch@hnuc.edu.cn, Shancang.Li@uwe.ac.uk, qhe@swin.edu.au, xlxu@nuist.edu.cn}

Abstract. As a promising way to extract insightful information from massive data, service recommendation has gained ever-increasing attentions in both academic and industrial areas. Recently, the Locality-Sensitive Hashing (LSH) technique is introduced into service recommendation to pursue high recommendation efficiency and the capability of privacy-preservation, especially when the historical service quality (QoS) data used to make recommendation decisions are distributed across different platforms. However, existing LSH-based service recommendation approaches often face the following challenge: they often assume that the QoS data for service recommendation are static and unique, without considering the influence of dynamic context (e.g., time) on QoS. In view of this challenge, we extend the traditional LSH technique to incorporate the time factor and further propose a novel time-aware and privacy-preserving service recommendation approach based on LSH. Finally, we conduct extensive experiments on a large-scale real-world dataset, i.e., *WS-DREAM*, to validate the effectiveness and efficiency of our proposal. The experiment results show that our approach achieves a good tradeoff between recommendation accuracy and efficiency while guaranteeing privacy-preservation.

Keywords: Distributed Service Recommendation, Privacy-preservation, Time, Locality-Sensitive Hashing.

1 Introduction

With the advent of Internet-of-Things (IoT), an ever-increasing number of intelligent sensor devices (e.g., mobile phones, smart watches and GPS navigators) are deployed in human daily activities, industrial production and social interactions, generating considerable amount of data with sparse but valuable information [12, 25, 24]. In this situation, the service recommendation techniques (e.g., Collaborative Filtering) have become a promising way for users to extract insightful information from massive data. Typically, through analyzing the historical QoS (quality of services) data of candidate services and the subjective preferences of a target user, a recommender system can help the target user to find out the appropriate services that he or she prefers in a time-efficient and cost-effective manner; thus the target user's service selection burden can be reduced considerably.

However, in the IoT environment, the historical QoS data for service recommendation decision-makings are often not centralized, but distributed across different platforms or parties (e.g., some QoS data are recorded by Amazon; some are owned by IBM; the rest are owned by other companies) [21, 6, 30]. In this situation, from the perspective of recommender systems, it is necessary to integrate the QoS data recorded by Amazon, IBM and other companies appropriately to pursue more comprehensive and accurate recommended results. However, Amazon, IBM as well as other companies often cannot release their data to the public due to privacy concerns, which impedes the cross-platform data integration and the distributed service recommendation significantly. Therefore, it becomes an essential requirement for a recommender system to protect the private information of users when integrating the distributed QoS data for recommendation.

The Locality-Sensitive Hashing (LSH) [5] technique has recently been introduced into service recommendation to achieve the abovementioned privacy-preservation goal. Typically, the data owners (e.g., Amazon and IBM) first transform their private QoS data into corresponding hash values with little privacy and then release the hash values to the public. Afterwards, a recommender system utilizes the hash values with little privacy to make recommendation decisions; this way, the private information of users can be protected. Besides, as the hash tables can be built offline, the recommendation efficiency would be improved considerably.

However, existing LSH-based service recommendation approaches often assume that the bases of recommendation decisions, i.e., the historical QoS data of candidate services are static and unique, without considering the dynamic influence of context factors (e.g., time) on QoS; therefore, the produced recommended results may be not reasonable and accurate enough.

In view of this challenge, we extend the traditional LSH-based recommendation models to incorporate a time factor. In this paper, we propose a novel time-aware and privacy-preserving service recommendation approach to improve the recommendation accuracy. In summary, the contributions of this paper are three-fold.

(1) We introduce the time factor into LSH-based service recommendation models, in order to adapt the dynamic QoS update of candidate services.

(2) We propose a novel time-aware and privacy-preserving service recommendation approach based on the time-aware LSH to pursue more accurate recommended results.

(3) A wide range of experiments are conducted based on the public QoS dataset, i.e., *WS-DREAM*. Experiment results demonstrate the advantages of our proposal in terms of service recommendation accuracy and efficiency while protecting the private information of users.

The reminder of this article is structured as follows. Section 2 reviews the existing approaches for service recommendation. In Section 3, we formulate the time-aware and privacy-preserving service recommendation problem in the distributed environment and then motivate our paper through an intuitive example. In Section 4, we introduce the details of our suggested recommendation approach. Experiments are presented in Section 5. Finally, in Section 6, we summarize the whole paper and discuss some potential research directions in the future.

2 Related Work

In this paper, we mainly focus on the time-aware service recommendation problems with privacy-preservation. Therefore, we review the related work from the following two aspects.

(1) Time-aware service recommendation

Many researchers have investigated the influence of the time factor on the service quality prediction and service recommendation. In [34], the authors calculate the user similarity or item similarity based on a three-dimensional user-service-time QoS matrix and then utilize the time-aware similarity to predict the missing QoS data and then make service recommendation. In [37], a matrix factorization technique is used to decompose the user-service QoS matrix into the product of two matrices, i.e., the user-time matrix and time-service matrix; afterwards, the missing quality data are predicted based on the obtained two matrices. In [27], the authors formulate the time-aware QoS prediction problem as a generic regression problem; through minimizing the gap between the real QoS and predicted QoS, a QoS regression model is derived and then employed to perform service quality prediction. The correlation between API popularity and time is investigated in [39] where only the popular web APIs at current time slots are recommended to the app developers to create promising mashups. In [19, 10, 18], the user similarity or item similarity are assigned a time-aware coefficient to quantify the influence of the time factor on similarity-based QoS prediction.

The above approaches all consider the important role of the time factor in service recommendation; however, they still have several shortcomings. Firstly, existing time-aware recommendation approaches seldom consider privacy disclosure risks when the distributed QoS data are required to be merged together to make comprehensive service recommendation decisions. Secondly, existing approaches often fall short in offering high efficiency and scalability because the similarity calculation or model training need to be repeated when the QoS data increase or are updated frequently.

(2) Privacy-preserving service recommendation.

The distributed property of data often leads to security and privacy concerns [9, 16, 35, 3, 36, 8, 28, 7]. Anonymity is an effective technique to protect the sensitive information of users. In [15], K-anonymity technique is adopted to confuse the real QoS data of services. Although the anonymous QoS data contain little private information, the data availability after anonymization would be reduced accordingly; therefore, the recommendation accuracy would be decreased accordingly. To protect the user privacy contained in QoS data, in [4], the authors advise to partially publish or release QoS data at a small portion; however, the potential attackers can still extract the private information hidden in the partially released QoS data through various machine learning techniques. Considering this drawback, data obfuscation technique is employed in [40] where the real QoS data are obfuscated by adding a random value and then the obfuscated QoS data are utilized to calculate user similarity and make appropriate recommendations. However, as the obfuscated QoS data instead of the real QoS data are used to make recommendations, the accuracy of recommended results is reduced to some extent. In [13], each piece of QoS data is randomly divided into several segments that are recorded by different users; afterwards, the distributed QoS segments (with only partial private information of a user) are merged together to calculate user similarity approximately. However, this privacy-aware recommendation approach has two drawbacks. Firstly, much communication cost will be incurred when the QoS segments from different users are merged. Secondly, this approach may still reveal some privacy of users, e.g., the services co-executed by different users.

Due to the inherent property of "similarity retention" (i.e., two neighboring points are still neighbors after hash), LSH has recently been employed to protect the sensitive information of users (e.g., the quality of services invoked by a user) in service recommendation. In [20], the authors first transform the sensitive QoS data into less-sensitive user indices offline; afterwards, the user indices are used to search for the similar neighbors of a target user and then make corresponding recommendations. This way, the service recommendation process can be finished in a time-efficient and privacy-preserving manner. However, this LSH-based service recommendation approach still faces the following challenge, i.e., it does not consider the dynamic fluctuation of QoS data incurred by in a varied context environment (e.g., time).

With the above reviews and analyses, existing service recommendation approaches seldom consider both the time-aware QoS variation and the capability of privacy-preservation simultaneously. In view of this challenge, in this paper, we propose a novel time-aware service recommendation approach with privacy-preservation.

3 Formulation and Motivation

3.1 Problem formulation

To facilitate the following discussions, some symbols used in this paper are formulated as below. For simplicity, only one quality dimension q of services (e.g., the *response time*) is considered in this paper.

(1) u_{target}: a target user to whom a recommender system plans to recommend its services.
(2) $WS = \{ws_1, \dots, ws_n\}$: the set of candidate recommended services for u_{target}.
(3) $PF = \{pf_1, \dots, pf_N\}$: the set of distributed platforms that record the QoS data of services in set WS.
(4) $U = \{u_1, \dots, u_m\}$: the set of users who have ever invoked services in set WS.
(5) $T = \{t_1, \dots, t_p\}$: the set of time slots when a service is invoked by a user.
(6) $q_{i,j,k}$: the QoS value of q for user u_i ($1 \le i \le m$) and service ws_j ($1 \le j \le n$) at time slot t_k ($1 \le k \le p$). Specifically, if u_i did not invoke ws_j at time slot t_k, then $q_{i,j,k} = 0$ holds.

With the above formal symbols, we can formulate the time-aware and privacy-preserving service recommendation problem as follows: according to the time-aware QoS data $q_{i,j,k}$ distributed in different platforms in the set of PF, a recommender system searches for the appropriate services from candidates in the set of WS and recommends them to the target user u_{target}, during which the real values of $q_{i,j,k}$ will not be revealed to the recommender system. In this paper, we will propose a novel approach to tackle this specific recommendation problem.

3.2 Paper motivation

To facilitate the understanding of readers, we introduce an intuitive example (in Fig.1) to illustrate the motivation of this paper.

In Fig.1, in this example, the historical QoS data for recommendation are distributed in two platforms: *Amazon* and *IBM*. The *Amazon* platform records the QoS data generated by $\{u_{target}, \ldots\}$, while the *IBM* platform records the QoS data generated by $\{u_1, \ldots\}$. Assume that there are totally n candidate services $\{ws_1, \ldots, ws_n\}$. Each (u_i, ws_j) pair is corresponding to p QoS values $\{q_{i,j,1}, \ldots, q_{i,j,p}\}$. Then for the recommender system, it should fuse or integrate the QoS data across *Amazon* and *IBM*, to search for the similar neighbors of u_{target} in a privacy-preserving way and then pursue more comprehensive and accurate recommended results.

However, existing recommendation approaches seldom consider the time-aware QoS values $\{q_{i,j,1}, \ldots, q_{i,j,p}\}$ $(1 \le i \le m, 1 \le j \le n, 1 \le k \le p)$ and the capability of privacy-preservation simultaneously. Considering this drawback, a novel service recommendation approach is proposed in this paper, which will be introduced in detail in the next section.

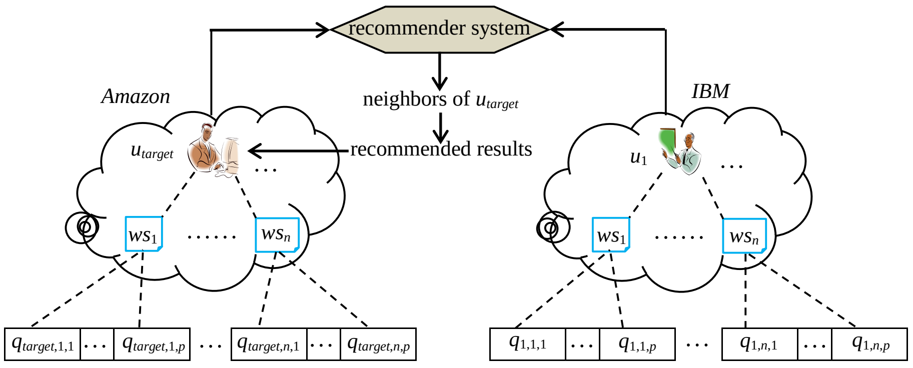
recommender system
Amazon
neighbors of u_{target}
IBM
u_{target}
recommended results
u_1
ws_1
ws_n
ws_1
ws_n
$q_{target,1,1}$ … $q_{target,1,p}$ … $q_{target,n,1}$ … $q_{target,n,p}$
$q_{1,1,1}$ … $q_{1,1,p}$ … $q_{1,n,1}$ … $q_{1,n,p}$

Fig.1 Time-aware and privacy-preserving service recommendation in a distributed environment: an intuitive example.

4 Time-aware and Privacy-preserving Service Recommendation Approach based on LSH: *SerRec*$_{time\text{-}LSH}$

In this section, we introduce a novel time-aware and privacy-preserving service recommendation approach based on LSH, named *SerRec*$_{time\text{-}LSH}$. The basic idea of our proposed *SerRec*$_{time\text{-}LSH}$ approach is: according to the LSH technique, we first transform the sensitive and dynamic QoS data into less-sensitive user indices; afterwards, we utilize the less-sensitive user indices to determine the similar time slots of u_{target}; finally, we predict the missing QoS data of u_{target} at a certain time slot based on the QoS data at the similar time slots and then recommend appropriate services to u_{target}. This way, the dynamic QoS data of services are considered and the private information of users is also protected. Concretely, our approach consists of the following three steps, where u_{target} denotes a target user and q is a quality dimension of services.

Step-1: Build a time-aware user index table based on LSH. According to the dynamic service QoS data observed by users, generate less-sensitive and time-aware user index table $Table_{index}$ based on the LSH technique.

Step-2: Determine the similar time slots of u_i based on the user index table. According to the user index table $Table_{index}$, determine the similar time slots of u_i from $T = \{t_1, \ldots, t_p\}$, denoted by similarity matrix *SIM_matrix* (u_i).

Step-3: Service recommendation based on the QoS data observed by u_{target} at similar time slots. According to the QoS data observed by u_{target} at the similar time slots in set *SIM_matrix* (u_{target}), predict the missing QoS data of services never invoked by u_{target} and then recommend the optimal services to u_{target}.

Fig.2 Three steps of the proposed *SerRec*$_{time\text{-}LSH}$ approach

Step-1: Build a time-aware user index table based on LSH.

As Fig.3 shows, in the dynamic service running environment, a QoS value $q_{i,j,k}$ is corresponding to a point in three-dimensional space constituted by user (i.e., i), service (i.e., j) and time (i.e., k). In this situation, the QoS data for user u_i can be represented by a service-time matrix as specified in (1), where each row depicts the QoS values of n services invoked by u_i at a time slot, and each column depicts the QoS values of a service invoked by u_i at p time slots. Please note that $q_{i,j,k} = 0$ if u_i does not invoke ws_j at time slot t_k.

$$Q(u_i) = \begin{bmatrix} q_{i,1,1} & \cdots & q_{i,n,1} \\ \vdots & \ddots & \vdots \\ q_{i,1,p} & \cdots & q_{i,n,p} \end{bmatrix} \qquad \textbf{(1)}$$

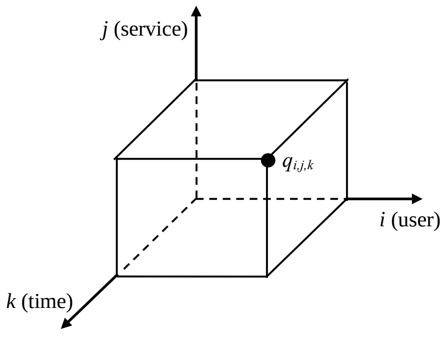

Fig.3 Dynamic QoS representation in three-dimensional space.

Next, we utilize the QoS matrix in (1) to build a time-aware index for user u_i based on the LSH technique. Generally, Pearson Correlation Coefficient (PCC) [22] is often used as the user similarity measurement in the traditional collaborative filtering-based service recommendation approaches. Therefore, to guarantee the property of "similarity retention" of the LSH technique, we utilize the LSH function [11] corresponding to the PCC distance to realize the transformation from the sensitive QoS data in (1) to less-sensitive user indices. Concretely, for each row of the QoS matrix in (1), the LSH function in (2) is adopted. Here, $\overrightarrow{Q(u_i)_k}$ denotes the n-dimensional vector corresponding to k-th row of matrix $Q(u_i)$; $\vec{v} = (v_1, \ldots, v_n)$ is an n-dimensional vector where v_j $(1 \le j \le n)$ is randomly selected from the range [-1, 1]; operation "$a \circ b$" means the dot product of vectors a and b.

$$h(\overrightarrow{Q(u_i)_k}) = \begin{cases} 1 & \text{if } \overrightarrow{Q(u_i)_k} \circ \vec{v} > 0 \\ 0 & \text{if } \overrightarrow{Q(u_i)_k} \circ \vec{v} \le 0 \end{cases} \qquad \textbf{(2)}$$

Thus through the hash mappings in (2), the vector corresponding to the k-th row of matrix $Q(u_i)$, i.e., $\overrightarrow{Q(u_i)_k}$ is transformed into a Boolean value $h(\overrightarrow{Q(u_i)_k})$ in (2). Repeat the above the hash mapping process until the original QoS matrix $Q(u_i)$ in (1) is transformed into the p-dimensional Boolean vector $h(Q(u_i))$ in (3) where "a^T" means the transpose of vector a.

$$h(Q(u_i)) = (h(\overrightarrow{Q(u_i)_1}), \ldots, h(\overrightarrow{Q(u_i)_p}))^T \qquad \textbf{(3)}$$

However, LSH is a probability-based approximate neighbor search technique; therefore, only one LSH function in (2) often cannot guarantee the "similarity retention" property of LSH. Considering this drawback, multiple LSH functions are used in our hash mapping process. Concretely, we randomly generate r LSH functions $h_1(.)$, …, $h_r(.)$ (see (2)) and then utilize them to realize the transformation from $Q(u_i)$ in (1) to $h(Q(u_i))$ in (3). Afterwards, we obtain a $p*r$ hash value matrix, denoted by $H(Q(u_i))$ in (4). Thus, $H(Q(u_i))$ can be regarded the time-aware user index for u_i, which is often less sensitive compared to the original $Q(u_i)$.

$$H(Q(u_i)) = \begin{bmatrix} h_1(\overrightarrow{Q(u_i)_1}) & \cdots & h_r(\overrightarrow{Q(u_i)_1}) \\ \vdots & \ddots & \vdots \\ h_1(\overrightarrow{Q(u_i)_p}) & \cdots & h_r(\overrightarrow{Q(u_i)_p}) \end{bmatrix} \qquad \textbf{(4)}$$

For each user u_i in set U, we repeat the above process to build his/her time-aware index $H(Q(u_i))$ in (4); afterwards, the users as well as their respective index values form a hash table (i.e., user index table), denoted by $Table_{index}$. Thus, for a platform that is willing to but dares not share its data with other platforms, it can publish the less sensitive user index table $Table_{index}$ to the public; this way, the private information of users can be protected very well.

Step-2: Determine the similar time slots of u_i based on the user index table.

According to Step-1, in the user index table $Table_{index}$, each user $u_i \in U$ is corresponding to a $p*r$ index matrix $H(Q(u_i))$ (here, p is the number of time slots). Next, we determine the similar time slots of u_i based on $Table_{index}$. Concretely, to reduce the computational cost, for the $p*r$ matrix $H(Q(u_i))$ in (4), we treat its each row as a r-dimensional 0-1 string and then convert the 0-1 string into its corresponding decimal number. For example, if $r = 5$, then the "01101" string will be converted into decimal number "13". Thus we transform the $p*r$ matrix $H(Q(u_i))$ in (4) into the p-dimensional column vector $H(Q(u_i))_{decimal}$ in (5).

$$H(Q(u_i))_{decimal} = \begin{bmatrix} A_1(u_i) \\ \vdots \\ A_p(u_i) \end{bmatrix} \tag{5}$$

Next, for user u_i, we compare its hash values at p time slots, i.e., $A_1(u_i)$, …, $A_p(u_i)$. If $A_{k1}(u_i) = A_{k2}(u_i)$ ($k_1 < k_2$), then we denote $sim_{k1,k2}(u_i) = 1$ where $sim_{k1,k2}(u_i)$ means the similarity between hash values of u_i at the k_1-th and k_2-th time slots; otherwise, $sim_{k1,k2}(u_i) = 0$. Thus, for user u_i, we can obtain a $p*p$ symmetric matrix $Sim\_matrix\ (u_i)$ as in (6) where each entry is a Boolean value (specifically, $sim_{k,k}(u_i) = 0$ as the similarity between hash values of u_i at an identical time slot makes no much sense). Then the mappings from u_i to $Sim\_matrix\ (u_i)$ form a hash table (denoted by HT); moreover, as the QoS data of services are already recorded by different platforms, the hash table HT can be created offline by the corresponding platforms.

LSH is actually a probability-based approximate neighbor search technique; therefore, for user u_i, one hash table is often not enough to evaluate the similarity between the QoS values at different time slots. Considering this, we create L hash tables offline, i.e., HT_1, …, HT_L and accumulate their corresponding similarity matrices $Sim\_matrix\ (u_i)_1$, …, $Sim\_matrix\ (u_i)_L$. Afterwards, a new similarity matrix for user u_i, i.e., $SIM\_matrix\ (u_i)$ is obtained as in (7) where each entry $sim_{k1,k2}(u_i) \in [0, L]$ holds. Furthermore, a larger $sim_{k1,k2}(u_i)$ often means higher probability that the QoS values at the k_1-th and k_2-th time slots are similar; in other words, the QoS value at the k_1-th time slot is more suitable to be utilized to predict the missing QoS value at the k_2-th time slot, vice versa.

$$Sim\_matrix\ (u_i) = \begin{bmatrix} sim_{1,1}(u_i) & \cdots & sim_{1,p}(u_i) \\ \vdots & \ddots & \vdots \\ sim_{p,1}(u_i) & \cdots & sim_{p,p}(u_i) \end{bmatrix} \tag{6}$$

$$SIM\_matrix\ (u_i) = Sim\_matrix\ (u_i)_1 + \ldots + Sim\_matrix\ (u_i)_L \tag{7}$$

Step-3: Service recommendation based on the QoS data of u_{target} at similar time slots.

Next, we predict the missing QoS data of u_{target} at the k_2-th time slot based on the QoS data of u_{target} at the k_1-th time slot, if these two time slots are similar. Concretely, we set a similarity threshold for $sim_{k1,k2}(u_{target})$ in (7). If $sim_{k1,k2}(u_{target})$ is larger than the threshold, then for each service ws_j never invoked by u_{target}, its missing QoS value of criterion q at the k_2-th time slot (i.e., $q_{target,j,k2}$) can be predicted by the equation in (8). Here, $Sim\_set(t_{k2})$ denotes the set of time slots which are similar to the k_2-th time slot (see (9)); $|Sim\_set(t_{k2})|$ means the size of set $Sim\_set(t_{k2})$. Finally, we select the services with the highest predicted value $q_{target,j,k2}$, denoted by $ws_{optimal}$, and recommend them to the target user.

$$q_{target,\,j,\,k2} = \frac{1}{|Sim\_set(t_{k2})|} * \sum_{t_{k1} \in Sim\_set(t_{k2})} q_{i,\,j,\,k1} \qquad \textbf{(8)}$$

$$Sim\_set(t_{k2}) = \{\, t_{k1} \mid sim_{k1,k2}(u_{target}) \geq \text{threshold} \} \qquad \textbf{(9)}$$

Through the abovementioned Step-1~Step-3 of our proposed $SerRec_{time\text{-}LSH}$ approach, we can make time-aware and privacy-preserving service recommendations in a distributed environment. More formally, the pseudo code of $SerRec_{time\text{-}LSH}$ approach is presented as follows.

```
Algorithm-1: SerRec_time-LSH

Inputs: $u_{target}$: a target user
        $WS = \{ ws_1, \ldots, ws_n \}$: web service set
        $U = \{ u_1, \ldots, u_m \}$: user set
        $T = \{ t_1, \ldots, t_p \}$: time slot set
        $q_{i,j,k}$: QoS value of $q$ for user $u_i$ and service $ws_j$ at time slot $t_k$
Output: $ws_{optimal}$: optimal services that are recommended to $u_{target}$

1  For $x$ =1 to $r$ do  // generate $r$ hash functions $h_1(.), \ldots, h_r(.)$
2    For $j$ =1 to $n$ do
3      $v_j$ = random [-1, 1]
4    End For
5    $h_x(.) = (v_1, \ldots, v_n)$
6    For each $u_i \in U$ do
7      generate QoS matrix $Q(u_i)$ according to (1)
8      For $k$ =1 to $p$ do
9        $h(\overrightarrow{Q(u_i)_k}) = Q(u_i)_k * h_x(.)$  // LSH mappings
10     End For
11     $h(Q(u_i)) = (h(\overrightarrow{Q(u_i)_1}), \ldots, h(\overrightarrow{Q(u_i)_p}))^T$
12   End For
13 End For
14 For each $u_i \in U$ do
15   generate $H(Q(u_i))$ according to (4)
16 End For
17 generate $Table_{index}$ based on all the "$u_i \rightarrow H(Q(u_i))$" pairs
18 For each $u_i \in U$ do
19   For $k$ =1 to $p$ do
20     transform $H(Q(u_i))_k$ into corresponding decimal number $A(u_i)_k$
21   End For
22   If $A(u_i)_{k1} = A(u_i)_{k2}$
```

```
23         Then $sim_{k1,k2}(u_i) = 1$
24         Else $sim_{k1,k2}(u_i) = 0$
25     End If
26 End For
27 Create a hash table $HT$ based on "$u_i \rightarrow Sim\_matrix\ (u_i)$" mappings
28 Repeat line 18-27 to create $L$ hash tables $HT_1, \ldots, HT_L$
29 Calculate $SIM\_matrix\ (u_i)$ based on the accumulation operation in (7)
30 Set a similarity threshold $threshold$
31 Set a time slot $t_{k2}$ at which the QoS value needs to be predicted
32 For $k_1$ =1 to $p$ do
33     If $k_1 < k_2$ and $sim_{k1,k2}(u_{target}) \geq threshold$
34         Then $q_{target, j, k2}$ is predicted by (8)-(9)
35     End If
36 End For
37 $ws_{optimal} = \{ws_j \mid q_{target, j, k2} = \text{optimal}\ (q_{target, j, k2})\}$  // optimal services are selected
38 Return $ws_{optimal}$ to $u_{target}$
```

5 Experiments

To validate the feasibility of $SerRec_{time\text{-}LSH}$ approach, we conduct a set of experiments based on a real-world time-aware QoS dataset. Concretely, we introduce the experiment settings in Subsection 5.1 and analyze the experiment results in Subsection 5.2.

5.1 Experiment Dataset and Configurations

Our experiments are deployed on a distributed web service QoS dataset, i.e., *WS-DREAM* [38], which was tested and collected by Dr. Zibin Zheng in 2014. The *WS-DREAM* dataset consists of the QoS values of 4532 real-world web services invoked by 142 users at 64 different time slots. Therefore, the dataset is suitable for testing the performance of our suggested time-aware distributed service recommendation approach, i.e., $SerRec_{time\text{-}LSH}$.

Concretely, we test the following two criteria (due to the inherent property of LSH, the capability of privacy-preservation of our proposal is not measured here; the security that is crucial in many systems [23, 32, 1, 17, 14] is out of the scope of this paper):

(1) RMSE (Root Mean Square Error, the smaller the better): measure the accuracy of finally derived recommended results.

(2) Time cost: measure the recommendation efficiency and scalability (scalability is also a key criterion to evaluate the system performance [2, 29]).

Besides, we compare $SerRec_{time\text{-}LSH}$ approach with the following three approaches:

(1) *Average*: the missing QoS value of a service observed by u_{target} at time slot t_p is predicted based on the service's average QoS values at time slot $t_1, \ldots, t_{p\text{-}1}$;

(2) *Partial-HR* [18]: the missing QoS value of a service observed by u_{target} at time slot t_p is predicted based on the service's QoS values at time slot $t_1, \ldots, t_{p\text{-}1}$, and each QoS value is assigned a time-aware weight;

(3) *SerRec*$_{distri\text{-}LSH}$ [21]: the missing QoS value of a service observed by u_{target} at time slot t_p is predicted based on the average QoS values of the service observed by the similar neighbors of u_{target} at time slot t_p.

The experiments configurations are listed as follows. (1) hardware: 2.60 GHz CPU and 8.0 GB RAM; (2) software: Windows 10 and Python 3.6. Each experiment was carried out 50 times (the reason will be explained in Profile-6) and their average experiment results were adopted finally.

5.2 Experiment Results

In our experiments, totally six profiles are designed, tested and compared, which are presented and analyzed as below. Here, m and n denote the size of user set U and size of service set WS, respectively; r and L represent the number of hash functions and the number of hash tables, respectively (r and L are two key parameters that control the similar neighbor search conditions). In the following profiles, $r = 2$ and $L = 10$ holds.

Profile-1: RMSE comparison of four recommendation approaches.

In this profile, we measure the service recommendation accuracy of four approaches through RMSE. The experiment parameters are set as follows: n is varied from 500 to 4500. Concrete comparison results are demonstrated in Fig.4.

As indicated in Fig.4, the recommendation accuracy of the *Partial-HR* approach is the lowest (i.e., RMSE is the highest); this is because the weighting mechanism adopted in *Partial-HR* is a bit rough and the similarity between different users are not considered. The accuracy of *SerRec*$_{distri\text{-}LSH}$ approach is also not high as this approach does not consider the dynamic variation of service QoS values either. While our suggested *SerRec*$_{time\text{-}LSH}$ approach outperforms the other three approaches in terms of accuracy; this is because *SerRec*$_{time\text{-}LSH}$ not only considers the time-aware QoS variation of services but also recruits the QoS values at similar time slots for missing QoS prediction and service recommendation.

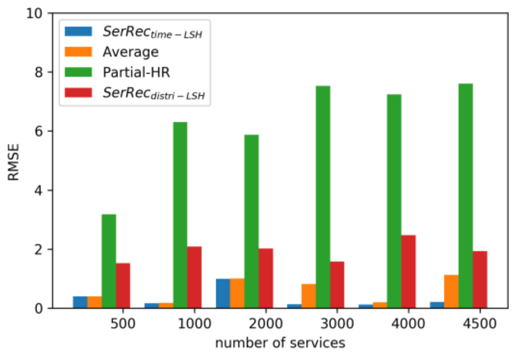

Fig.4 Recommendation accuracy comparison of four approaches.

Profile-2: Time cost comparison of four recommendation approaches.

Next, we measure and compare the efficiency and scalability of four service recommendation approaches. The parameters are set as follows: n is varied from 500 to 4500. The compared experiment results are shown in Fig.5.

As Fig.5 demonstrates, the time costs of four approaches all increase with the growth of *n* (i.e., the number of services), because more computational cost is required when the volume of candidate services becomes larger. Besides, the efficiency of our *SerRec_{time-LSH}* approach is close to that of *SerRec_{distri-LSH}* approach because they both take the same privacy protection strategy (i.e., LSH). However, as Fig.5 shows, our *SerRec_{time-LSH}* performs worse than *Average* and *Partial-HR* in terms of recommendation efficiency and scalability; this is because no additional privacy-preservation operations are adopted in both *Average* and *Partial-HR* approaches.

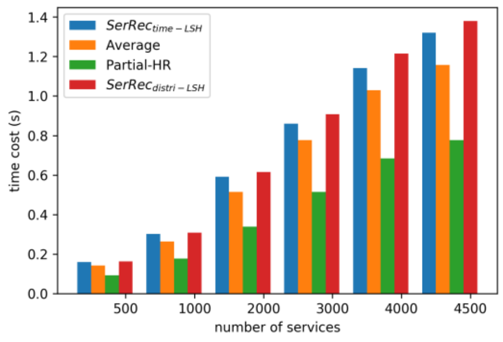

Fig.5 Recommendation efficiency comparison of four approaches.

Profile-3: RMSE with respect to threshold in our *SerRec_{time-LSH}*

In Step-3 of our proposed *SerRec_{time-LSH}* approach, similarity threshold is a key factor which can influence the recommendation performances. Thus, in this profile, we test the relationship between the recommendation accuracy (i.e., RMSE) and the threshold. In the test, the threshold is varied from 2 to 8 and the concrete experiment results are shown in Fig.6. As Fig.6 indicates, the recommendation accuracy increases (i.e., RMSE drops) with the growth of threshold. This is because a larger similarity threshold often means stricter search condition for similar time slots and correspondingly, the recommendation accuracy is improved.

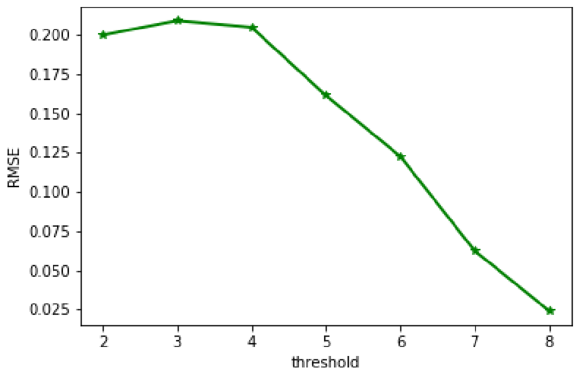

Fig.6 RMSE w.r.t. similarity threshold (*SerRec_{time-LSH}*)

Profile-4: Time cost with respect to threshold in our *SerRec*$_{time\text{-}LSH}$

In this profile, we test the relationship between the time cost and the similarity threshold. Here, the threshold is varied from 2 to 8 and the concrete experiment results are presented in Fig.7. As Fig.7 shows, the time cost gradually decreases with the growth of similarity threshold and becomes gradually convergent when the threshold is larger than 6. This is because when the similarity threshold rises, the "qualified" similar time slots becomes fewer and increasingly stable and therefore, the time cost for QoS prediction and service recommendation gradually decreases and stays approximately stable.

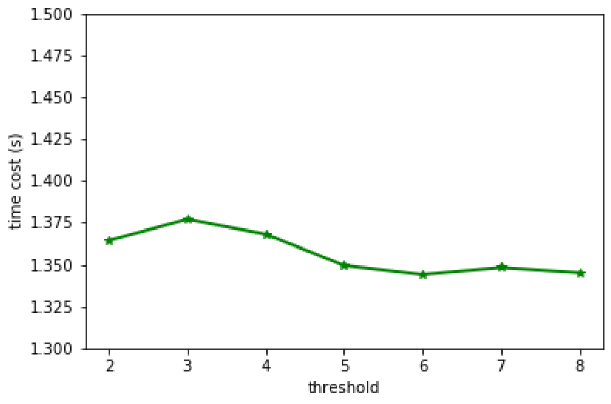

Fig.7 Time cost w.r.t. similarity threshold (*SerRec*$_{time\text{-}LSH}$)

Profile-5: Number of similar time slots with respect to threshold in *SerRec*$_{time\text{-}LSH}$

The similarity threshold plays an important role in the search of similar time slots in our proposed *SerRec*$_{time\text{-}LSH}$ approach. In this profile, we test and analyze their correlations, whose results are presented in Fig.8. As Fig.8 shows, when the threshold grows, the search condition for similar time slots becomes stricter and correspondingly, the "qualified" similar time slots become fewer.

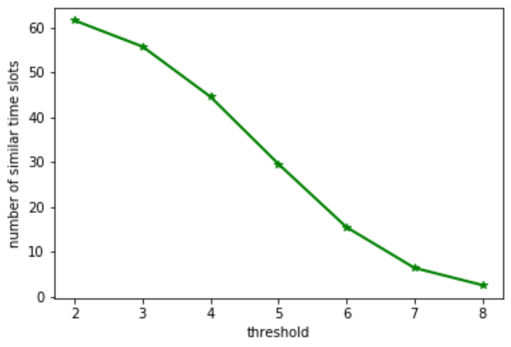

Fig.8 Number of similar time slots w.r.t. similarity threshold (*SerRec*$_{time\text{-}LSH}$)

Profile-6: RMSE convergence with respect to experiment times

As introduced in Subsection 5.1, each experiment was repeated 50 times. The experiment times (i.e., 50) is concluded from the experiment results in Fig.9 (the number of services, i.e., *n* is varied from 500 to 4500). In Fig.9, we test the RMSE convergence of four approaches with respect to the experiment times. As can be seen from the experiment results, when the experiment times approach 50, the RMSE values of four recommendation approaches all become convergent approximately (especially for the *SerRec*$_{distri\text{-}LSH}$ approach). Therefore, in our experiments, each test was repeated 50 times and their average experiment results were adopted finally.

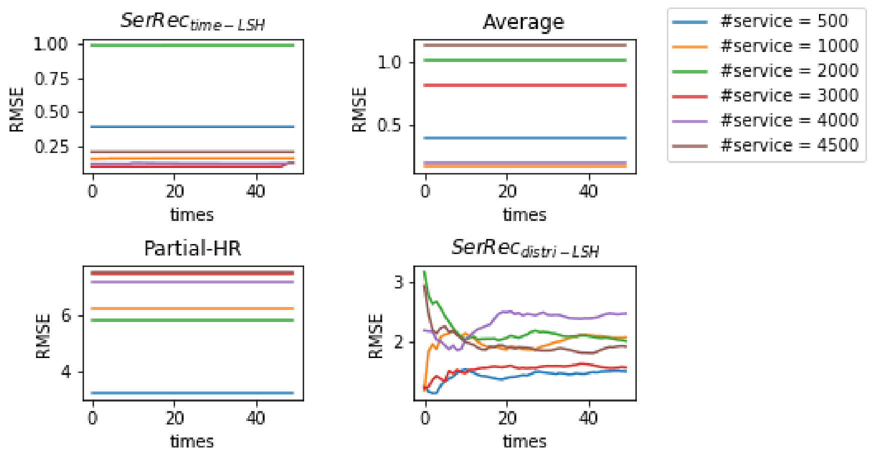

Fig.9 RMSE convergence w.r.t. experiment times

6 Conclusions

In the distributed environment, integrating the QoS data distributed across multiple platforms while protecting the private information of users is an indispensable condition for the success of service recommendation. Besides, the QoS data of services are often not static but varied with time. While existing service recommendation approaches seldom consider the time-aware QoS and the capability of privacy-preservation simultaneously. In view of this challenge, we proposed a novel time-aware and privacy-preserving service recommendation approach, i.e., *SerRec*$_{time\text{-}LSH}$. Firstly, we utilize the dynamic QoS data of services to build a time-aware user index table which is often less sensitive; Secondly, we use the less sensitive user indices to determine the similar time slots of the target user; Finally, we make service quality prediction and service recommendations to the target user based on the QoS data of services observed by the target user at the similar time slots. Through a wide range of experiments deployed on a real-world time-aware QoS dataset named *WS-DREAM*, we validate the effectiveness and efficiency of our proposal while guaranteeing privacy-preservation.

However, there exists several shortcomings in our approach. First, we assume that the recommendation decisions only depend on a single quality criterion, without considering the criterion diversity [26] as well as their weight values [33]. Therefore, in the future, we will investigate this multi-dimensional recommendation scenario to widen the applicability of our proposal. Second, the quality of services often depends on many context factors [31]. Thus, in the future, we will further refine our recommendation approach by considering more context factors besides time such as user location, service location, the distance between services and users.

Acknowledgements

This paper is partially supported by the Natural Science Foundation of China (No. 61872219, No. 61672276, No. 61702299, No. 61273232), National Key R&D Program of China (No. 2017YFB1001800), New Zealand Marsden Fund.

References

1. Z. Cai, H. Yan, P. Li, Z. Huang, et al: Towards secure and flexible EHR sharing in mobile health cloud under static assumptions, Cluster Comput. 20(3) (2017) 2415-2422.
2. X. Chen, J. Li, X. Huang, J. Ma, et al, New publicly verifiable databases with efficient updates, IEEE T. Depend. Secure 12(5) (2015) 546-556.
3. X. Chen, J. Li, J. Ma, Q. Tang, et al, New algorithms for secure outsourcing of modular exponentiations. IEEE T. Parall. Distr. 25(9) (2014) 2386-2396.
4. W. Dou, X. Zhang, J. Liu, J. Chen, HireSome-II: towards privacy-aware cross-cloud service composition for big data applications, IEEE T. Big Data 26(2) (2015) 455-466.
5. A. Gionis, P. Indyk, R. Motwani, Similarity search in high dimensions via hashing, VLDB 99(6) (1999) 518-529.
6. W. Gong, L. Qi, Y. Xu, Privacy-aware multi-dimensional mobile service quality prediction and recommendation in distributed fog environment, Wirel. Commun. Mob. Com. (2018).
7. C. Hu, X. Cheng, X. Liao, Secure and efficient data communication protocol for wireless body area networks, IEEE Transactions on Multi-Scale Computing Systems 2(2) (2016) 94-107.
8. C. Hu, W. Li, X. Cheng, J. Yu, et al, A secure and verifiable access control scheme for big data storage in clouds, IEEE T. Big Data (2016). DOI: 10.1109/TBDATA.2016.2621106.
9. C. Hu, N. Zhang, H. Li, X. Cheng, et al, Body area network security: a fuzzy attribute-based signcryption scheme, IEEE J. Sel. Area. Comm. 31(9) (2013) 37-46.
10. Y. Hu, Q. Peng, X. Hu, R. Yang, Time aware and data sparsity tolerant web service recommendation based on improved collaborative filtering, IEEE T. Serv. Comput. 8(5) (2015) 782-794.
11. Y. Ioannidis, et al, Data mining and query log analysis for scalable temporal and continuous query answering, http://www.optique-project.eu/, 2015.
12. R. Jhaveri, N. Patel, Y. Zhong, A. Sangaiah, Sensitivity analysis of an attack-pattern discovery based trusted routing scheme for mobile ad-hoc networks in industrial IoT, IEEE ACCESS (2018). DOI: 10.1109/ACCESS.2018.2822945.
13. D. Li, C. Chen, Q. Lv, L. Shang, et al, An algorithm for efficient privacy-preserving item-based collaborative filtering, Future Gener. Comp. Sy. 55 (2016) 311-320.

14. Q. Lin, H. Yan, Z. Huang, W. Chen, et al, An id-based linearly homomorphic signature scheme and its application in blockchain, IEEE Access 6(1) (2018) 20632-20640.
15. T. Ma, Y. Zhang, J. Cao, J. Shen, et al, KDVEM: a k-degree anonymity with vertex and edge modification algorithm, Computing 70(6) (2015) 1336-1344.
16. X. Ma, F. Zhang, X. Chen, J. Shen, Privacy preserving multi-party computation delegation for deep learning in cloud computing, Inform. Sciences 459 (2018) 103-116.
17. W. Meng, E. Tischhauser, Q. Wang, Y. Wang, et al, When intrusion detection meets blockchain technology: a review, IEEE Access, DOI: 10.1109/ACCESS.2018.2799854.
18. L. Qi, W. Dou, C. Hu, Y. Zhou, et al, A context-aware service evaluation approach over big data for cloud applications, IEEE T. Cloud Comput. (2015). DOI: 10.1109/TCC.2015.2511764.
19. L. Qi, X. Xu, W. Dou, J. Yu, et al, Time-aware IoE service recommendation on sparse data, Mob. Inf. Syst. (2016). DOI: 10.1155/2016/4397061.
20. L. Qi, X. Zhang, W. Dou, C. Hu, et al, A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment. Future Gener. Comp. Sy., 2018. DOI: 10.1016/j.future.2018.02.050.
21. L. Qi, X. Zhang, W. Dou, Q. Ni, A distributed locality-sensitive hashing based approach for cloud service recommendation from multi-source data, IEEE J. Sel. Area. Comm. 35(11) (2017) 2616-2624.
22. L. R. Joseph, W. Alan Nicewander, Thirteen ways to look at the correlation coefficient, Am. Stat. 42(1) (1988) 59-66.
23. J. Shen, Z. Gui, S. Ji, J. Shen, et al, Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks, J. Netw. Comput. Appl. 106 (2018) 117-123.
24. J. Shen, C. Wang, T. Li, X. Chen, et al, Secure data uploading scheme for a smart home system, Inform. Sciences (2018). DOI: 10.1016/j.ins.2018.04.048.
25. C. Wang, J. Shen, Q. Liu, Y. Ren, et al, A novel security scheme based on instant encrypted transmission for Internet-of-Things, Secur. Commun. Netw. (2018). DOI: 10.1155/2018/3680851.
26. P. Wang, L. Zhao, Some geometrical properties of convex level sets of minimal graph on 2-dimensional riemannian manifolds, Nonlinear Anal.-Theor. 130 (2016) 1-17.
27. X. Wang, J. Zhu, Z. Zheng, W. Song et al, A spatial-temporal qos prediction approach for time-aware web service recommendation, ACM T. Web 10(1) Article 7 (2016) 1-25.
28. K. Xing, C. Hu, J. Yu, X. Cheng, et al, Mutual privacy preserving k-means clustering in social participatory sensing, IEEE T. Ind. Inform. 13(4) (2017) 2066-2076.
29. J. Xu, L. Wei, Y. Zhang, A. Wang, et al, Dynamic fully homomorphic encryption-based Merkle Tree for lightweight streaming authenticated data structures, J. Netw. Comput. Appl. 107 (2018) 113-124.
30. Y. Xu, L. Qi, W. Dou, J. Yu, Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment, Complexity, 2017 Article ID 3437854 (2017) 1-9.
31. X. Xue, S. Wang, B. Gui, Z. Hou, A computational experiment-based evaluation method for context-aware services in complicated environment, Inform. Sciences 373 (2016) 269-286.
32. L. Yang, Z. Han, Z. Huang, J. Ma, A remotely keyed file encryption scheme under mobile cloud computing, J. Netw. Comput. Appl. 106 (2018) 90-99.
33. S. Yang, X. Kong, C. Tang, A construction of linear codes and their complete weight enumerators, Finite Fields Th. App. 48 (2017) 196-226.
34. C. Yu, L. Huang, A web service qos prediction approach based on time- and location-aware collaborative filtering, Service Oriented Computing and Applications, 10(2) (2016) 135-149.
35. H. Yuan, X. Chen, T. Jiang, X. Zhang, et al, DedupDUM: secure and scalable data deduplication with dynamic user management, Inform. Sciences 456 (2018) 159-173.

36. X. Zhang, T. Jiang, K. C. Li, A. Castiglione, et al, New publicly verifiable computation for batch matrix multiplication, Inform. Sciences (2017). DOI: /10.1016/j.ins.2017.11.063.
37. Y. Zhang, Z. Zheng, M. R. Lyu, WSPred: a time-aware personalized qos prediction framework for web services. In: 22nd IEEE International Symposium on Software Reliability Engineering. pp. 210-219, IEEE, New York (2011).
38. Z. Zheng, Y. Zhang, M. R. Lyu, Investigating qos of real world web services, IEEE T. Serv. Comput. 7(1) (2014) 32-39.
39. Y. Zhong, Y. Fan, K. Huang, W. Tan, et al, Time-aware service recommendation for mashup creation, IEEE T. Serv. Comput. 8(3) (2015) 356-368.
40. J. Zhu, P. He, Z. Zheng, M. R. Lyu, A privacy-preserving qos prediction framework for web service recommendation, In: 22nd International Conference on Web Services, pp. 241-248, IEEE, New York (2015).