EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Improved LSH for privacy-aware and robust recommender system with sparse data in edge environment

Xuening Chen[1], Hanwen Liu[2] and Dan Yang[3*]

Abstract

Recommender systems have become a popular and effective way to quickly discover new service items that are probably preferred by prospective users. Through analyzing the historical service usage data produced in the past, a recommender system can infer the potential user preferences and make accurate recommendations accordingly. However, in the edge environment, the service usage data stored in each edge server are often very sparse, which may result in expected cold-start problems. Besides, in the edge environment, the data required to make an optimal service recommendation decision are often stored in different edge clients or servers, which require additional privacy-preservation strategies to secure the sensitive data involved. Considering the above two drawbacks, traditional locality-sensitive hashing (LSH) is improved to be multi-probing LSH and then introduced to aid the recommendation process so as to guarantee the security and robustness of recommender systems. Experiments conducted on well-known dataset prove the effectiveness and efficiency of the work.

Keywords: Recommender system, Privacy, Robustness, LSH, Sparse data, Edge

1 Introduction

With the popularization of service-oriented architecture in web of things, a large number of services flooded in people's daily life. On the one hand, so many candidate services mean that for a prospective user, the cost and time for finding a preferred service is often high; on the other hand, the users' expectations of high timeliness and high accuracy for service-oriented applications require that a recommender system can provide real-time and accurate recommended lists for users [1–4]. In view of this, personalized and light-weight recommendation technology emerges to achieve the above goals. By analyzing user preferences, recommending systems can provide users with rapid and accurate personalized recommended lists, which greatly reduce the users' selection cost and alleviate their decision-making burden [5, 6].

Due to the unique characteristics of domain independence and easy interpretation of recommended results, collaborative filtering (CF) has become one of the most popular techniques in various recommendation systems. Typically, by analyzing a great deal of historical QoS (quality of service) data, similar neighbors of a target user could be discovered; then, the probably preferred services of the target user are recommended to him or her with limited time cost. This way, the target users' decision-making burden can be reduced to some extent.

Nevertheless, traditional collaborative filtering recommendation methods mostly suppose the decision-making data used for recommendation are centralized without considering the fragmented data distributed in different parties (such as Google and Microsoft). For example, part of QoS data may be stored in the Google platform, while other data are owned by Microsoft. From the angle of providing users with more accurate and comprehensive recommended results, it is obliged to fuse the QoS data stored in Google and Microsoft. However, as two parties with conflicts of interest, Google and Microsoft are often not willing to reveal their own data to each other due to the necessity to protect users' private information, which block the cross-platform data integration and multi-source recommendation.

* Correspondence: asyangdan@163.com
[3]School of Software, University of Science and Technology Liaoning, Anshan, China
Full list of author information is available at the end of the article

LSH (locality-sensitive hashing) [7] has already been applied to service recommendation to realize the balance between user privacy protection and accurate data integration. Concretely, a data platform (e.g., Google or Microsoft) first transforms its original data into corresponding hash values with less privacy and then open these hash values to external platforms for collaboration purpose. Finally, according to the hash values of data from different platforms, a recommender system can produce accurate service recommended list. This way, user privacy can be protected; additionally, the recommendation efficiency can be enhanced significantly as hash tables can be built offline.

However, in the edge environment, the computation and storage capabilities of each edge server or client are often limited; therefore, the historical QoS data kept by each edge client or server for recommendations are often not dense enough, but sparse enough. Therefore, it is of high probability that a target user cannot find similar neighbors or target services. In this situation, data sparsity problems occur while traditional LSH method cannot deal with this problem effectively, which brings a great challenge for more comprehensive and accurate service recommendations.

Considering the abovementioned data sparsity and privacy-preservation challenges, we introduce the multi-probing LSH technique and put forward a robust and privacy-preserving method to improve the recommendation robustness in the sparse-data environment. Briefly, the major contributions of our work are mainly the following three aspects.

(1) We recognize the drawback of traditional LSH technique in handling the distributed recommendation applications with sparse data in the edge environment.
(2) We improve the traditional LSH technique to be multi-probing one and recruit it to deal with the sparse data problems to pursue robust and privacy-aware recommendations.
(3) A series of experiments are deployed on a freely available dataset, i.e., *Movielens* [8]. The results of experiments validate that our method achieves higher robustness and acceptable accuracy compared to other competitive methods.

The structure of the paper is shown as follows. Related work about service recommendation is presented in Section 2. We formulate the data sparsity problems of recommendation in the edge environment and afterwards motivate the method by a vivid example in Section 3. In Section 4, we detail the proposed service recommendation method concretely. Experiments are implemented and results are analyzed in Section 5. In Section 6, we summarize this paper and point out the possible improvement directions in the future research.

2 Related work

In this section, we briefly present an overview of existing works from the following two aspects: data sparsity problem in service recommendation and recommendation with privacy preservation.

2.1 Data sparsity problem in service recommendation

When more and more web services are appearing in the network, the possibility that a web service would be selected is becoming smaller and smaller. Therefore, users often do not rate enough web services, and hence, historical QoS data are often not sufficient for determining a target user's similar neighbors, i.e., data sparsity problems occur [9]. Many studies have tried to alleviate the data sparsity problems. Jing [10] proposed a sparse probabilistic matrix factorization method (SPMF) by employing Laplacian distribution to determine the service/user latent feature; Laplacian distribution has the ability to generate sparse coding and help to recommend the tail services. Hu [11] proposed an enhanced memory-based collaborative filtering method to discover potential similarity relationships between users or items by using limited user-item interactions; thus, they incorporate user similarity reinforcement and item similarity reinforcement into a comprehensive framework and make more reliable rating predictions. Qi [12] put forward a structural balance theory (SBT)-based recommendation method for the sparsity of big rating data in E-commerce. According to the rule of structural balance theory, indirect friends of a target user are discovered and "possibly similar product items" are recommended to him or her.

In order to produce more reasonable and comprehensive recommended results under sparse data conditions, auxiliary information is incorporated to the recommendation methods. Wang [13] integrates time information into collaborative filtering method, and a hybrid personalized random walk algorithm for searching indirect similar user/service is proposed to alleviate the data sparsity problem. Hu [14] proposed a hierarchical Bayesian model called collaborative deep learning (CDL), which performs deep representation learning for the content information and collaborative filtering for the ratings' matrix. By CDL, deep feature representation extracted from content information can capture similarities and discover implicit relationships between users and items. However, one drawback of these methods based on auxiliary resource is high dependency on auxiliary information that increases the storage cost and raises privacy or security concerns.

Through the above methods we mentioned, the remarkable role of data sparsity in service recommendation has been considered; however, they still have several drawbacks. First, they are vulnerable to privacy leakage when cross-platform QoS data are demanded to be fused to achieve better recommendation performances. Second, similarity calculation or model training of large-scale data repeatedly may block high efficiency and scalability when QoS data increase and update constantly.

2.2 Recommendation with privacy preservation

Data fusion in distributed environment often leads to privacy issues. Anonymity technique is a common way to protect private information of the user. Zhang [15] adopted anonymity technique by local recoding and combines the t-ancestors (similar to K-means) with the aggregation clustering algorithm for solving the problem of privacy information disclosure to third party. Casino [16] implements K-anonymity through microaggregation technology to protect user privacy during service recommendation. Memon [17] generalizes and blurs the unique location information of users by applied K-anonymity, so as to protect the location information of users while recommending services. However, the recommendation accuracy of the above three methods decreases accordingly while certain data are distorted simultaneously. In terms of protecting privacy information in QoS data, Dou [18] focuses on finding the "representative" QoS values in users' historical records, which can be implemented and shared by distributed platforms, while the user privacy information contained in certain QoS values may still be partially disclosed to the public. Zheng [19] takes "the amount of data that a user needs to be open" as an adjustable parameter and model multiobjective optimization problem as NP-hard with high time complexity, thus obtaining a good compromise between the availability and privacy of the data; however, a small amount of QoS data opened may also be leaked to some extent. Zhu [20] obscures real QoS data by adding random value to each service-specific QoS data; then, user similarity and service recommendation are calculated by obfuscated data to protect privacy values, but QoS values of the user itself in this way will still be partially leaked.

Due to the unique characteristics of LSH (i.e., neighboring points are still close after hashing with large probability), LSH has already been utilized to secure users' privacy in service recommendation for privacy-preserving and efficient outcome [21–23]. However, LSH-based recommendation methods often face the below difficulty: it does not take into account the sparsity data problem, which decreases the recommendation robustness severely.

Based on the above analysis, existing service recommendation methods can seldom cope with the sparse data issue and privacy-preservation requirements simultaneously. Considering this challenge, we bring forth a novel recommendation method that is not only robust but also privacy preserving in the following sections.

3 Formulation and motivation

3.1 Problem formulation

To simplify the following discussion, the symbols to be recruited in this paper can be described as below. For facilitation, the recommendation problem can be formalized by a four-tuple (M, U, E, $m_{i,j}.q$) as below:

1. $M = \{m_1, \ldots, m_n\}$: candidate services
2. $U = \{u_1, \ldots, u_m\}$: users who ever invoked services in set M
3. $E = \{e_1, \ldots, e_s\}$: distributed edge platforms where e_k ($1 \le e \le s$) possess the kth part of records of QoS data in set M
4. $m_{i,j}.q$: QoS values on dimension q for service m_j ($1 \le j \le n$) rated by user u_i ($1 \le i \le m$). Particularly, if u_i did not rate m_j, then $m_{i,j}.q = 0$ holds

In view of the above formal symbol representation, the privacy-preserving recommendation problem with sparse data can be described as follows: a recommender system integrates the distributed $m_{i,j}.q$ data to discover appropriate services for a target user; during the above process, real QoS values cannot be leaked to the outside. To tackle this issue, we introduce a novel method in Section 4.

3.2 Motivation

To clarify the idea of this paper, an intuitive example is demonstrated in Fig. 1 to introduce our motivation in detail.

In Fig. 1, we can see that historical QoS data are distributed in two edge platforms: Google and Microsoft. Suppose that there are a total of n candidate services $\{m_1, \ldots, m_n\}$ and four users $\{u_1, u_2, u_3, u_4\}$; several QoS data records $m_{i,j}.q$ in each distributed edge platform are also provided where "$m_{i,j}.q = 0$" means that u_i has never rated m_j before. In this situation, the size of service intersection between users is fairly small, which seriously hinders a target user from finding his/her similar neighbors and interested services. Therefore, the recommender system needs to integrate the QoS values from both Google and Microsoft so as to make a more

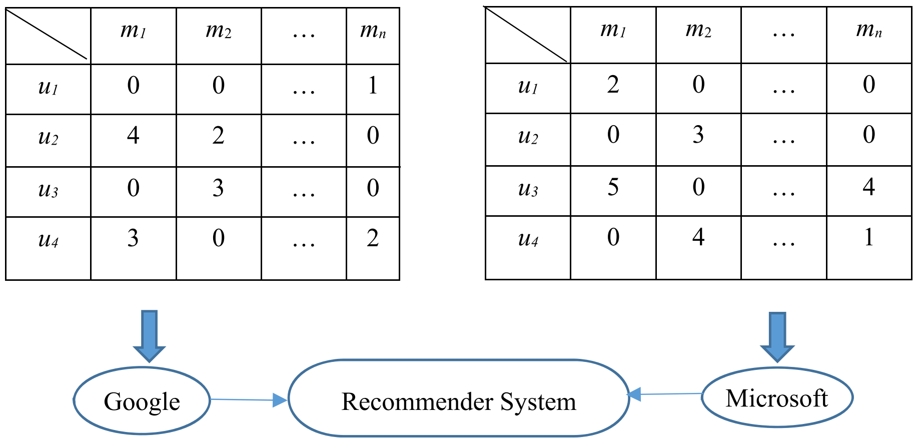

Fig. 1 Cross-edge recommendation with sparse data and privacy concerns

comprehensive decision; afterwards, similarity between different users, e.g., u_1 and u_2, can be obtained for seeking similar neighbors in recommender system, and then, more accurate and comprehensive recommended results can be achieved in a privacy-preserving way.

In view of this, a novel service recommendation method adapted to data sparsity and privacy preservation is presented in the next section.

4 Our solution: *SerRec$_{sparsity\text{-}LSH}$*

In this section, we bring forth a novel recommendation method under sparse data environment based on multi-probing LSH, named *SerRec$_{sparsity\text{-}LSH}$*. In Section 4.1, basic LSH technique is briefly introduced; then, the details of our proposed method are presented in Section 4.2.

4.1 Basic locality-sensitive hashing

Locality-sensitive hashing (LSH) has been put forward by Aristides Gionis in 1999 [7] and has been proved to be appropriate for fast approximate querying millions of data; the accuracy of query results is same as that of “brute force” query basically and widely used in many fields, such as videos and image queries. The main idea of LSH theory can be shown as in Fig. 2. First, original data space is converted into a LSH data space by hyperplanes in Fig. 2a. Afterwards, (1) two neighboring points like A and B in the original data space are still neighbors with high probability (i.e., A and B are on the same side of the hyperplane l_1) in LSH data space; (2) two dissimilar points like A and C (or B and C) are still not neighbors with high probability (i.e., the two points are on the different sides of the hyperplane l_2) in LSH data space. Furthermore, the distance from the point to the hyperplane can be calculated to find the nearest neighbors or the most similar neighbors.

We utilize the example in Fig. 2b to illustrate the distance calculation process. Assume that two points a and b are on the same side of the hyperplane l_1 in LSH data space, $\overrightarrow{op}$ ($\overrightarrow{op} \neq \mathbf{0}$) is a normal vector of hyperplane l_1 while vectors $\overrightarrow{oa}$ and $\overrightarrow{ob}$ are used to compute the distance between points a and b to the hyperplane, respectively. We assume O_1 and O_2 are projection points of points a and b on normal vector $\overrightarrow{op}$, OO_1 and OO_2 are corresponding to the projection distance (i.e., the distance from the point to hyperplane) of vectors $\overrightarrow{oa}$ and $\overrightarrow{ob}$ to the normal vector $\overrightarrow{op}$. Then, according to dot product principle, the distance OO_1 (or OO_2) from the point a (or b) to hyperplane l_1 can be obtained by formula (1). Here, $\|\overrightarrow{op}\|$ represents the modulus of normal vector $\overrightarrow{op}$ and symbol “ $\circ$ ” represents the dot product between two vectors. Thus, the distance from each point to the hyperplane can be gained in LSH data space, which makes it easier to find out the nearest neighbors of fixed points.

$$proj(OO1) = \frac{\overrightarrow{oa} \circ \overrightarrow{op}}{\|\overrightarrow{op}\|} \tag{1}$$

In the above example, we have observed the basic theory and three advantages of LSH technique. First, transforming the original data into LSH data space greatly improves the efficiency and scalability of the process for similar neighbors finding. Second, the original data in LSH data space is transparent to users, through which user privacy can be protected significantly. Third, distributed data could be integrated into a unified LSH data space, which improves the accuracy of uniform calculation considerably. Therefore, the LSH technique is extended in this paper to

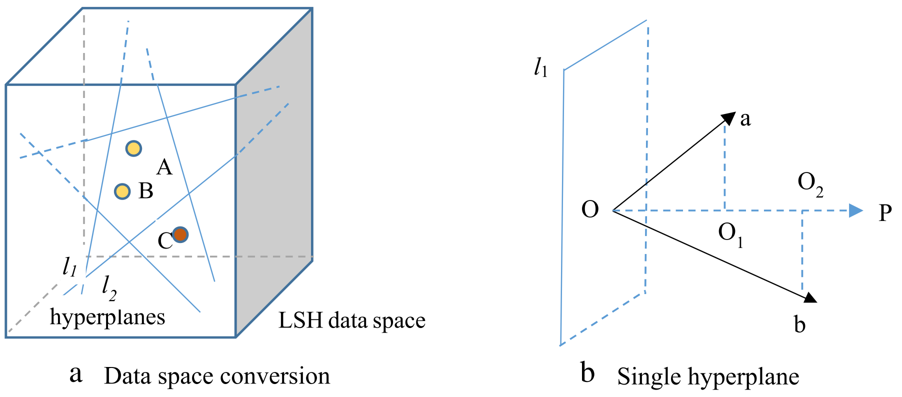

Fig. 2 a, b Rationale of locality-sensitive hashing

achieve the goal of privacy-preserving service recommendation with sparse data in edge environment.

4.2 *SerRec$_{sparsity\text{-}LSH}$*: recommendation based on multi-probing LSH

Concretely, our method mainly includes three steps as elaborated in the reminder of this subsection.

4.2.1 Step 1: Generate index tables

We utilize user-service QoS matrix to generate a user index table on the basis of LSH. Here, Pearson correlation coefficient (PCC) is introduced for similarity computation between two web users or web services in the collaborative filtering applications [21–23]. In order to preserve the property of "similarity keeping" of LSH, we employ LSH functions for PCC distance to achieve the conversion from original data to user indices, which can simplify the similarity calculation process greatly. Specifically, for each $u_i \in U$, his/her obtained QoS values over n services $\{m_1, \ldots, m_n\}$ in set M is converted into m-dimensional vector $\overrightarrow{u(i)} = (m_{i.1}.q, m_{i.2}.q, \ldots, m_{i.n}.q)$ where q is a quality dimension of each service and $m_{i.j}.q = 0$ if user u_i does not rate m_j before. Then, based on [6], the hash value of vector $\overrightarrow{u(i)}$, denoted by $h(u_{(i)})$, could be derived by the function adopted in (2). In the function, $\overrightarrow{v} = (v_1, \ldots, v_n)$ is a vector whose dimensional value v_j $(1 \le j \le n)$ is random data belonging to $[-1,1]$.

$$h(u(i)) = \begin{cases} 1 & \text{if } \overrightarrow{u(i)} \circ \overrightarrow{v} > 0 \\ 0 & \text{if } \overrightarrow{u(i)} \circ \overrightarrow{v} \le 0 \end{cases} \tag{2}$$

Afterwards, repeat the above conversion process R times through different vectors $\overrightarrow{v}$ (i.e., the number of vectors $\overrightarrow{v}$ is denoted by R), and then, the index for user u_i can be built as $H(u_{(i)}) = (h_1(u_{(i)}), h_2(u_{(i)}), \ldots, h_R(u_{(i)}))$. Continually, the conversion between original data space and LSH data space through hash mapping "$u_i \rightarrow H(u_i)$" are preserved in a hash table, represented by *user_table* offline. Thus, if an edge platform intends to share its data but fear to disclose user privacy to other parties, it can expose less-sensitive user indices to other parties. The integrated user index tables of multiple edge platforms can not only protect sensitive user data, but also provide more comprehensive recommendation bases for users.

4.2.2 Step 2: Neighbor search

According to step 1, we can search for neighbors of u_{target} from the user index table *user_table*, i.e., the hash values of neighbors are equal to the hash value of u_{target}. However, it is a very strict condition for discovering similar neighbors in a sparse data environment, as there is a high probability that no similar neighbors can be found for u_{target}. In this situation, we relax the condition for neighbor search so as to improve the robustness of the recommender systems. Furthermore, the found neighbors of u_{target} are put in set *Nei_Set*. Thus, although we can obtain a set of neighbors of u_{target}, we do not know the proximity of the neighbors in *Nei_Set* to the target user in LSH data space. Motivated by this drawback, we put forward a "distance" formula as in (3) (the physical meaning of the distance is illustrated in Section 4.1) to measure the "distance" between similar neighbors and target user so as to find the "most similar" neighbors. Concretely, each $u_i \in U$ has obtained a unique index table *user_table* (i.e., R bit of hash values) in step 1 where each hash value corresponds to a hyperplane or a normal vector. The distance calculation for a hash value is exampled in (3), where the normal vector $\overrightarrow{vr}$ is corresponding to the rth hyperplane and $\|\overrightarrow{vr}\|$ represents the modulus of normal vector $\overrightarrow{vr}$.

$$Dr(u(i)) = \frac{\overrightarrow{u(i)} \circ \overrightarrow{vr}}{\|\overrightarrow{vr}\|} \tag{3}$$

Then, the distance corresponding to each hash value can be calculated and the pair ($h_r(u_{(i)})$, $D_r(u_{(i)})$) for each user can be obtained in a user index table *user_table*. For example, if $r = 4$, then the hash value string "1101" for the distance to each hyperplane are 6, 10, − 5, 8, respectively; symbol "−" represents the other side of the hyperplane. Here, the users in set *Nei_Set* are considered as u_{target}'s similar neighbors based on step 1. In order to find the most similar neighbor set, we need to calculate the proximity between the users in *Nei_Set* and the target user in the LSH data space. If $u_i \in$ *Nei_Set*, then the proximity (corresponding to $R_{(1 \le r \le R)}$ hash values in an index table) between u_i and u_{target} could be measured in *user_table* by (4); symbol "$|a|$" represents the absolute value of a. Then, we sort the *Pro* values of candidate users u_i in ascending order, and the smaller the *Pro* value is, the closer the neighbor u_i is to u_{target}. Finally, the neighbors corresponding to the *top-k Pro* values are regarded as "most similar" neighbors of u_{target} and placed in set *Sim_Set.*

$$Pro(u\text{target}, ui)user\_table = \sum_{1 \le r \le R}^{R} \mid Dr(u(\text{target})) – Dr(u(i)) \mid \tag{4}$$

While LSH is a probability-based neighbor-fast-search technique, so one index table for user u_i cannot retain the "similarity keeping" property of LSH. In view of the shortcoming, multiple user index tables, i.e., $user\_table_1$, $user\_table_2$,..., $user\_table_T$ are implemented through the same hash mapping process; thus, we adopt the "OR" operation to relax the conditions for neighbor searching. Namely, if u_i is one of the "most similar" neighbors of u_{target} in any hash table, then u_i is deemed to be a "most similar" neighbor of u_{target}.

4.2.3 Step 3: Optimal service selection

If service m_j was not rated by u_{target} before, then m_j's quality over dimension q by u_{target}, i.e., $m_{\text{target},j}.q$ could be predicted through (5). Therein, |*Sim_Set*| denotes the size of *Sim_Set* and $m_{i,j}.q$ denotes the QoS values of m_j observed by u_i. Finally, we rank all the candidate services m_j by their predicted values in (5) and the service with the highest predicted value $m_{\text{target},j}.q$ will be selected and put in set *M_Set*. At last, the services in *M_Set* are returned to u_{target}.

$$m_{\text{target},j}.q = \frac{\sum\limits_{u_i \in Sim\_Set} m_{i,j}.q}{\mid Sim\_Set \mid} \tag{5}$$

Through the abovementioned three steps of the $SerRec_{sparsity\text{-}LSH}$ method, a recommender system can make privacy-preserving and robust recommendation decisions when the data are sparse in the edge environment. Moreover, the pseudo code of our proposal can be divided into three algorithms and exhibited as follows.

Algorithm 1: **Generate index tables**

```
Inputs: $U$, $M$, $E$, $m_{i,j}.q$
Output: user index table $User\_table$
    for $e$ = 1 to $s$ do
  for $r$ = 1 to $R$ do
    for $j$ = 1 to $n$ do
          $v_j$ = random[-1,1]
      end for
      $\vec{v_r}$ = ($v_1$,$v_2$,…,$v_n$)
      for i = 1 to $m$ do
      $\overrightarrow{u_{(i)}}$ = ($m_{i,1}$ .$q$, $m_{i,2}$.$q$, …, $m_{i,n}$ .$q$)
        if $\overrightarrow{u_{(i)}}$ * $\vec{v_r}$ > 0
          then $h(u_{(i)})$ =1
          else $h(u_{(i)})$ = 0
        end if
       end for
      end for
end for
for each $u_i \in U$ do
  $H(u_{(i)})$ = ($h_1(u_{(i)})$, $h_2(u_{(i)})$,…, $h_R(u_{(i)})$)
put $H(u_{(i)})$ into $User\_table$
return $User\_table$
```

Algorithm 2: **Neighbor search**

```
Input: $User\_table$, $u_{target}$, $v$, $Nei\_Set$, $Sim\_Set$
Output: $Sim\_set$
$Nei\_Set$ = null; $Sim\_Set$ = null
for $t$ = 1 to $T$ do
  iterate algorithm 1 to create $User\_table$
end for
search for neighbors whose single hash value is different from the ones in $User\_table(u_{target})_t$
put neighbors into $Nei\_Set$
If $u_i \in Nei\_Set$
  then $Pro(u_{target},u_i)$ = 0
  for $t$ = 1 to $T$ do
    for $r$ = 1 to $R$ do
          $D_r(u_{(i)})$ = $\overrightarrow{u(i)}$ ∘ $\vec{v_r}$/ || $\vec{v_r}$ ||
          $Pro(u_{target},u_i)_{User\_table\ t}$ = | $Dr(u_{(target)})$ - $Dr(u_{(target)})$ |
          $Pro(u_{target},u_i)$ = $Pro(u_{target},u_i)$ + $Pro(u_{target},u_i)_{User\_table\ t}$
        end for
      end for
   end if
   rank $Pro(u_{target},u_i)$ values in ascending order
   neighbors corresponding to the $top$-$k$ $Pro(u_{target},u_i)$ values are put into $Sim\_set$
```

Algorithm 3: **Optimal service selection**

Input: *Sim_Set*, *M*, $m_{i.j}.q$

Output: *M_Set*

```
1  the results set $M\_Set = \varnothing$; $m_{target.j}.q = 0$; $number = 0$
2  for service $m$ in the edge platform $e$ do
3    for $u_i \in Sim\_Set$ do
4      number++
5      if $m_{i.j}.q \neq 0$
6        then $m_{target.j}.q = m_{target.j}.q + m_{i.j}.q$
7      end if
8    end for
9  end for
10 put the neighbors with the highest $m_{target.j}.q$ into $M\_Set$
11 return $M\_Set$ to $u_{target}$
```

5 Experiments

In Section 5.1, we introduce the experimental setup; in Section 5.2, we describe and analyze the experimental results of the proposed method.

5.1 Experimental setup

Experiments are implemented on *Movielens* dataset, which contains ratings of 3900 movies from 6040 users. To validate the prediction performance, we compare $SerRec_{sparsity\text{-}LSH}$ with three state-of-the-art methods: *Random*, *WSRec* [24], and *SBT-Rec* [12]. *Random* is a benchmark that chooses services randomly; *WSRec* predicts the missing QoS data by considering the average QoS of the target service invoked by all users and the average QoS of all services invoked by the target user; *SBT-Rec* first searches for the indirect neighbors of the target user through social balance theory and collaborative filtering, and then recommends optimal services based on the derived neighbors. The experiments were running on a laptop with an i7-6500U CPU (2.50 GHz) and 16.0 GB RAM, Windows 7 operation system and Python 3.6. The experiments were executed 50 times, and average values are applied ultimately.

5.2 Experimental results

Four groups of experiments are performed and evaluated. Four parameters are displayed in the experiments: *m* and *n* mean the number of users and services, respectively; *L* and *R* mean the number of hash tables and hash functions, respectively.

5.2.1 Evaluation item 1: accuracy of four methods

Users always expect to receive accurate recommended results from the recommender systems. So we test the accuracy values of the mentioned four competitive methods via MAE. The value of *n* is equal to 3900; the value of *m* changes from 1000 to 6000; $L = 10$; $R = 10$. Comparison results are reported in Fig. 3. As can be observed that $SerRec_{sparsity\text{-}LSH}$ has achieved a smaller MAE (i.e., a higher accuracy) than other three methods as LSH can promise to find out those real neighbors of a target user.

5.2.2 Evaluation item 2: robustness of four methods

We utilize the successful rate (%) to depict the robustness of the four methods (Fig. 4). Experiment parameters are the same as that in evaluation item 1, and results are shown in Fig. 3. The Random and *WSRec* methods have the highest values of successful rate (100%) as they can randomly assign a value for any missing QoS data.

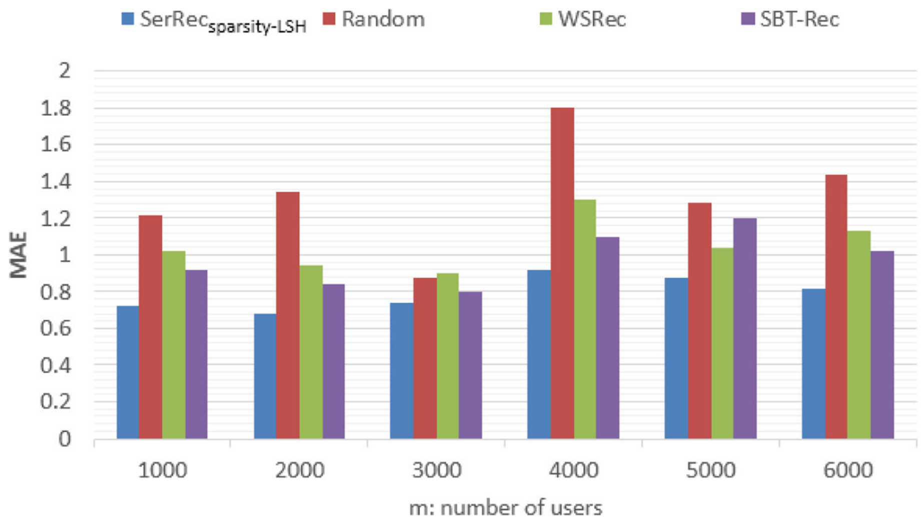

Fig. 3 Accuracy of the different methods

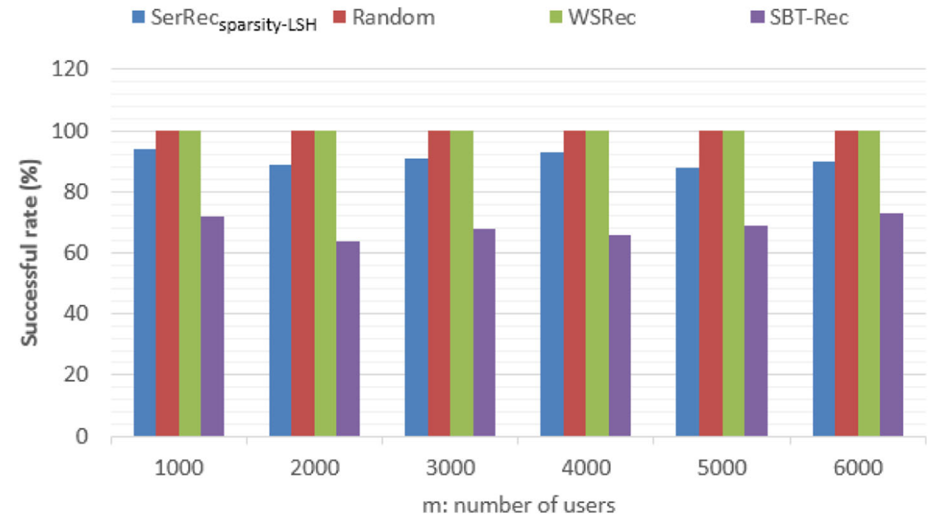

Fig. 4 Robustness of the different methods

However, these two methods did not take any strategy to secure the sensitive data of users, while the *SerRec*$_{sparsity\text{-}LSH}$ method approaches the above two methods in terms of successful rate and outperforms that of the *SBT-Rec* method.

5.2.3 Evaluation item 3: efficiency of four methods

Experiment parameters are the same as that in evaluation item 1. We measure the consumed time of four methods, and comparisons are depicted in Fig. 5. From the comparison results, we can see that the *Random* method achieves the optimal performance as it does not need to consider the concrete QoS data of every service. While *SerRec*$_{sparsity\text{-}LSH}$ method approaches the *Random* method in terms of time cost and performs better than those of *WSRec* and *SBT-Rec* methods as the employed LSH technique are very efficient in neighbor search as the search bases (i.e., index tables) can be achieved offline before recommendation process begins.

5.2.4 Evaluation item 4: accuracy of SerRec$_{sparsity\text{-}LSH}$ with L-R pairs

The LSH parameters L and R can narrow or relax the search condition for neighbors of target users in *SerRec*$_{sparsity\text{-}LSH}$ method and further influence the performance of *SerRec*$_{sparsity\text{-}LSH}$ method. Here, we study the inner correlations between accuracy of the *SerRec*$_{sparsity\text{-}LSH}$ method with L-R pairs. L and R are both varied from 2 to 10. Comparison results are shown in Fig. 6. We can observe a drop of MAE when R grows as more LSH functions indicate a narrower search condition for neighbors in *SerRec*$_{sparsity\text{-}LSH}$. In addition, we have also

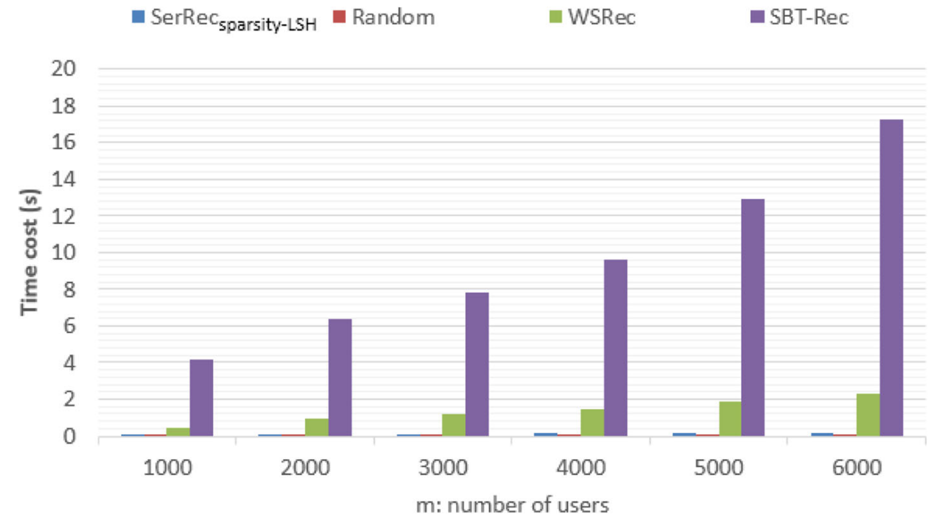

Fig. 5 Efficiency of the different methods

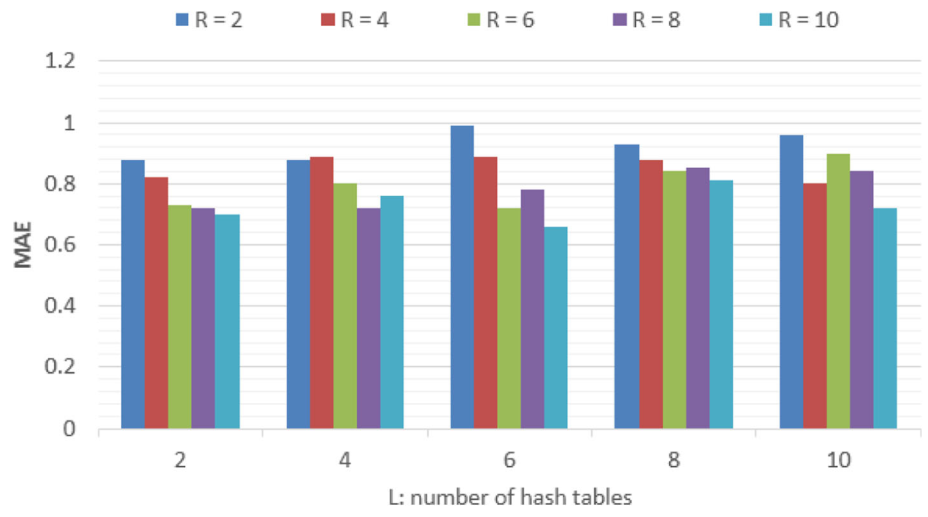

Fig. 6 MAE w.r.t. *L-R* pairs

noticed a rise of MAE when L grows as more LSH tables indicate a more relaxed search condition for neighbors in the suggested $SerRec_{sparsity\text{-}LSH}$ method.

5.3 Discussions

LSH is a kind of hash technique, which means that although LSH can protect the sensitive information of users when performing service recommendation, the effect of privacy preservation of LSH cannot be measured easily. For simple experiments, we only test the discrete data in Movielens dataset without testing other data types popular in the big data environment, e.g., continuous data type [25–29], Boolean data type [30], and fuzzy data type [31–33]. In addition, we only test a single criterion (user rating) without discussing the popular cases with multiple criteria [34–44] and their inner linear correlations [45–48], nonlinear correlations [49–66], and weights [67–73]. Therefore, we hope to refine our method by addressing these more complicated situations in the future research.

6 Conclusion

Through analyzing historical service usage data, a recommender system can infer the potential user preferences and make corresponding recommendations. However, in the edge environment, the service usage data stored in each edge server are often very sparse and sensitive, which may result in expected cold-start problems and privacy leakage risks. Considering these drawbacks, traditional LSH technique is improved to be multi-probing LSH and then introduced to aid the recommendation process so as to guarantee the security and robustness of recommender systems. Experiments conducted on well-known dataset prove the effectiveness and efficiency of the work. In the upcoming research work, we will extend our work further by incorporating more complex multi-criterion and multi-data-type cases. Besides, measuring the capability of privacy preservation of our recommendation approach is another research direction in the future work.

Abbreviations

CDL: Collaborative deep learning; CF: Collaborative filtering; LSH: Locality-sensitive hashing; QoS: Quality of service

Acknowledgements

Not applicable.

Authors' contributions

XC designed the algorithm and finished the English writing. HL completed the experiments. DY introduced the idea of this work. All authors read and approved the final manuscript.

Funding

Not applicable.

Availability of data and materials

The recruited experiment dataset *Movielens* is available at https://grouplens.org/datasets/movielens/.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Student Affairs Office, Qufu Normal University, Rizhao, China. [2]School of Information Science and Engineering, Qufu Normal University, Rizhao, China. [3]School of Software, University of Science and Technology Liaoning, Anshan, China.

Received: 3 April 2019 Accepted: 29 May 2019
Published online: 26 June 2019

References

1. L. Qi et al., Weighted principal component analysis-based service selection method for multimedia services in cloud. Computing **98**(1), 195–214 (2016)
2. Y. Xu et al., Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment. Complexity **2017**, Article ID 3437854, 9 (2017)

3. C. Yan et al., Privacy-aware data publishing and integration for collaborative service recommendation. IEEE ACCESS **6**, 43021–43028 (2018)
4. L. Qi et al., "Time-location-frequency"-aware internet of things service selection based on historical records. Int. J. Distrib. Sens. Netw. **13**(1), 1–9 (2017)
5. W. Gong et al., Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment. Wirel. Commun. Mob. Comput. **2018**, 8 (2018, Article ID 3075849)
6. L. Qi et al., A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data. IEEE J. Sel. Areas Commun. **35**(11), 2616–2624 (2017)
7. A. Gionis et al., *Similarity Search in High Dimensions Via Hashing. International Conference on Very Large Databases* (1999), pp. 518–529
8. https://grouplens.org/datasets/movielens/. Accessed 12 Feb 2019
9. L. Qi et al., Data-sparsity tolerant web service recommendation approach based on improved collaborative filtering. IEICE Trans. Inf. Syst. **E100D**(9), 2092–2099 (2017)
10. L. Jing et al., *Sparse Probabilistic Matrix Factorization by Laplace Distribution for Collaborative Filtering*, International Joint Conference on Artificial Intelligence (2015), pp. 1771–1777
11. Y. Hu et al., Mitigating data sparsity using similarity reinforcement-enhanced collaborative filtering. ACM Trans. Internet Technol. **17**(3), 31 (2017)
12. L. Qi et al., Structural balance theory-based E-commerce recommendation over big rating data. IEEE Transactions Big Data **4**(3), 301–312 (2018)
13. H. Wang et al., *Collaborative Deep Learning for Recommender Systems*, ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2015), pp. 1235–1244
14. Y. Hu et al., *A Time-Aware and Data Sparsity Tolerant Approach for Web Service Recommendation*, IEEE International Conference on Web Services (2014), pp. 33–40
15. X. Zhang et al., Proximity-aware local-recoding anonymization with Mapreduce for scalable big data privacy preservation in cloud. IEEE Trans. Comput. **64**(8), 2293–2307 (2015)
16. F. Casino et al., A K-anonymous approach to privacy preserving collaborative filtering. J. Comput. Syst. Sci. **81**(6), 1000–1011 (2015)
17. I. Memon, Authentication user's privacy: an integrating location privacy protection algorithm for secure moving objects in location based services. Wirel. Pers. Commun. **82**(3), 1585–1600 (2015)
18. W. Dou et al., HireSome-II: towards privacy-aware cross-cloud service composition for big data applications. IEEE Trans. Parallel Distrib. Syst. **26**(2), 455–466 (2015)
19. X. Zheng et al., *Location Privacy-Aware Review Publication Mechanism for Local Business Service Systems*, IEEE Conference on Computer Communications (2017), pp. 1–9
20. J. Zhu et al., *A Privacy-Preserving QoS Prediction Framework for Web Service Recommendation*, IEEE International Conference on Web Services (2015), pp. 241–248
21. L. Qi et al., Time-aware distributed service recommendation with privacy-preservation. Inf. Sci. **480**, 354–364 (2019)
22. L. Qi et al., A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment. Futur. Gener. Comput. Syst. **88**, 636–643 (2018)
23. L. Qi et al., An exception handling approach for privacy-preserving service recommendation failure in a cloud environment. Sensors **18**(7), 1–11 (2018)
24. Z. Zheng et al., QoS-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. **4**(2), 140–152 (2011)
25. C. Hou et al., Continuity of (α,β)-derivations of operator algebras. J. Korean Math. Soc **48**(4), 823–835 (2011)
26. C. Ma et al., On formability of linear continuous multi-agent systems. J. Syst. Sci. Complex. **25**(1), 13–29 (2012)
27. H. Wu et al., Continuous dependence property of BSDE with constraints. Appl. Math. Lett. **45**, 41–46 (2015)
28. L.L. Liu, Continued fractions and the derangement polynomials of types A and B. Ars Combinatoria **125**, 321–330 (2016)
29. H. Feng, The modulus of continuity theorem for G-Brownian motion. Commun Stat Theory Methods **46**(7), 3586–3598 (2017)
30. B. Zhang, Remarks on the maximum gap in binary cyclotomic polynomials. Bulletin Mathematique De La Societe Des Sciences Mathematiques De Roumanie **59**(1), 109–115 (2016)
31. L. Wang, The fixed point method for intuitionistic fuzzy stability of a quadratic functional equation. J. fixed point theory appl. **2010**, 1–7 (2010, Article ID 107182)
32. D. Xinsheng, Z. Zhao, On fixed point theorems of mixed monotone operators. Fixed Point Theory Appl. **2011**, 1–8 (2011, ArticleID 563136)
33. B. Zhu, L. Liu, Y. Wu, Local and global existence of mild solutions for a class of nonlinear fractional reaction-diffusion equation with delay. Appl. Math. Lett. **61**, 73–79 (2016)
34. M. Wang et al., Robust group non-convex estimations for high-dimensional partially linear models. J. Nonparametric Stat. **28**(1), 49–67 (2016)
35. X. Wang et al., Variable selection for high-dimensional generalized linear models with the weighted elastic-net procedure. J. Appl. Stat. **43**(5), 796–809 (2016)
36. P. Wang et al., Some geometrical properties of convex level sets of minimal graph on 2-dimensional riemannian manifolds. Nonlinear Anal. **130**, 1–17 (2016)
37. X. Wang et al., Adaptive group bridge estimation for high-dimensional partially linear models. J Inequalities Appl. **2017**(158), 1–18 (2017)
38. X. Wang et al., Restricted profile estimation for partially linear models with large-dimensional covariates. Statist. Probab. Lett. **128**, 71–76 (2017)
39. H. Tian et al., Bifurcation of periodic orbits by perturbing high-dimensional piecewise smooth integrable systems. J. Differ. Equ. **263**, 7448–7474 (2017)
40. P. Wang et al., The geometric properties of harmonic function on 2-dimensional Riemannian manifolds. Nonlinear Anal. **103**, 2–8 (2014)
41. M. Wang et al., Adaptive lasso estimators for ultrahigh dimensional generalized linear models. Statist. Probab. Lett. **89**, 41–50 (2014)
42. M. Wang et al., A note on the one-step estimator for ultrahigh dimensionality. J. Comput. Appl. Math. **260**, 91–98 (2014)
43. G. Tian et al., Variable selection in the high-dimensional continuous generalized linear model with current status data. J. Appl. Stat. **41**, 467–483 (2014)
44. X. Zhou et al., Academic influence aware and multidimensional network analysis for research collaboration navigation based on scholarly big data. IEEE Trans. Emerg. Top. Comput. (2018). https://doi.org/10.1109/TETC.2018.2860051
45. G. Guo et al., Parallel tempering for dynamic generalized linear models. Commun. Stat. Theory Methods **45**(21), 6299–6310 (2016)
46. L. Liu et al., Recurrence relations for linear transformations preserving the strong q-log-convexity. Electron. J. Comb. **23**(3), 1–11 (2016)
47. H. Li et al., Partial condition number for the equality constrained linear least squares problem. Calcolo **54**(4), 1121–1146 (2017)
48. Z. Zhao et al., Existence and uniqueness of positive solutions for some singular boundary value problems with linear functional boundary conditions. Acta Math. Sin. (Engl. Ser.) **27**(10), 2073–2084 (2011)
49. H. Liu et al., Some new nonlinear integral inequalities with weakly singular kernel and their applications to FDEs. J. Inequalities. Appl. **2015**(209), 1–17 (2015)
50. X. Zhang et al., Entire large solutions for a Schrödinger systems with a nonlinear random operator. J. Math. Anal. Appl. **423**(2), 1650–1659 (2015)
51. Z. Zong et al., On Jensen's inequality, Holder's inequality and Minkowski's inequality for dynamically consistent nonlinear evaluations. J. Inequal. Appl. **2015**(152), 1–18 (2015)
52. X. Hao et al., Positive solutions for nonlinear fractional semipositone differential equation with nonlocal boundary conditions. J. Nonlinear Sci. Appl. **9**(6), 3992–4002 (2016)
53. X. Hao et al., Iterative solution for nonlinear impulsive advection-reaction-diffusion equations. J. Nonlinear Sci. Appl. **9**(6), 4070–4077 (2016)
54. J. Shao et al., Oscillation theorems for second order forced neutral nonlinear differential equations with delayed argument. Int. J. Differ. Equ. **2010**, 1–15 (2010, article ID 181784)
55. Y. Bai et al., On a class of Volterra nonlinear equations of parabolic type. Appl. Math. Comput. **2010**(216), 236–240 (2010)
56. Y. Bai, Backward solutions to nonlinear integro-differential systems. Central Eur. J. Math. **8**(4), 807–815 (2010)
57. F. Li et al., Uniform energy decay rates for nonlinear viscoelastic wave equation with nonlocal boundary damping. Nonlinear Anal. **74**, 3468–3477 (2011)
58. F. Li et al., Global existence uniqueness and decay estimates for nonlinear viscoelastic wave equation with boundary dissipation. Nonlinear Anal. **12**, 1770–1784 (2011)
59. A. Qian, Sing-changing solutions for some nonlinear problems with strong resonance. Boundary Value Problems **18**, 1–9 (2011)
60. Y. Wang et al., Positive solutions for a class of fractional boundary value problem with changing sign nonlinearity. Nonlinear Anal. **74**(17), 6434–6441 (2011)
61. W. Fan et al., Nontrivial solutions of singular fourth-order Sturm-Liouville boundary value problems with a sign-changing nonlinear term. Appl. Math. Comput. **217**(15), 6700–6708 (2011)

62. L. Liu, Linear transformations preserving log-convexity. Ars Combinatoria **100**, 473–483 (2011)
63. J. Liu et al., Multiple positive solutions for second-order three-point boundary-value problems with sign changing nonlinearities. Electron. J. Differ. Equ. **2012**(152), 1–7 (2012)
64. S. Yang et al., The weight distributions of two classes of p-ary cyclic codes with few weights. Finite Fields Their Appl **44**, 76–91 (2017)
65. X. Zhou et al., Analysis of user network and correlation for community discovery based on topic-aware similarity and behavioral influence. IEEE Trans. Hum. Machine Syst. **48**(6), 559–571 (2018)
66. X. Wang et al., A cloud-edge computing framework for cyber-physical-social services. IEEE Commun. Mag. **55**(11), 80–85 (2017)
67. X. Wang et al., A tensor-based big data-driven routing recommendation approach for heterogeneous networks. IEEE Netw. Mag. **33**(1), 64–69 (2019)
68. Y. Wang et al., Uniform estimate for the tail probabilities of randomly weighted sums. Acta Math. Appl. Sin. Engl. Ser. **30**(4), 1063–1072 (2014)
69. S. Yang et al., A class of three-weight linear codes and their complete weight enumerators. Cryptogr. Commun. **9**, 133–149 (2017)
70. J. Cai, An implicit sigma (3) type condition for heavy cycles in weighted graphs. Ars Combinatoria **115**, 211–218 (2014)
71. S. Yang et al., Complete weight enumerators of a family of three-weight linear codes. Des. Codes Crypt. **82**, 663–674 (2017)
72. S. Yang et al., Complete weight enumerators of a class of linear codes. Discret. Math. **340**, 729–739 (2017)
73. S. Yang et al., A construction of linear codes and their complete weight enumerators. Finite Fields Their Appl. **48**, 196–226 (2017)

7 Publisher's Note