Received August 28, 2018, accepted September 24, 2018, date of publication October 5, 2018, date of current version October 29, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2873708

Interest-Aware Service Association Rule Creation for Service Recommendation and Linking Mode Recommendation in User-Generated Service

TIELIANG GAO[1], **BO CHENG**[1], **JUNLIANG CHEN**[1], **HUAJIAN XUE**[2], **LI DUAN**[3], **AND SHOULU HOU**[1]

[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Key Laboratory of IoT Perception and Control, Xinjiang Technical Institute of Physics and Chemistry, Chinese Academy of Sciences, Ürümqi 830011, China
[3]School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China

Corresponding author: Bo Cheng (chengbo@bupt.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1003800, in part by the Doctor Project for CAS through the Light of West China Program under Grant Y42S481, in part by the Outstanding Talent Program of Science and Technology Innovation in Henan Province under Grant 174200510008, and in part by the China Postdoctoral Science Foundation under Grant 2017M620617.

ABSTRACT In the area of user-generated service, service recommendation and service-linking mode recommendation can help end users select suitable service components for the creation of a new service application. However, existing service recommendation and service-linking mode recommendation methods typically focus on analyzing user-service dualistic relationship and predicting the quality of service values for service composition; the information about the latent interests of the user and the service is largely underexplored, resulting in large computational costs and weak recommendation explanations. To address these issues, we propose an interest-aware service association rule creation (SARC-IA) approach for recommending suitable service components and service linking modes to end users. The proposed solution includes two phases: 1) extracting the latent interests of the user and the service by utilizing the latent Dirichlet allocation model to train the invocation records of users; 2) forming interest subsets by collecting the users with the same interest; and 3) SARC algorithm is employed to generate the service-linking modes in the interest subsets. The performance of the proposed SARC-IA is evaluated via extensive experiments with the ProgrammableWeb data set. The experimental results show that the SARC-IA method outperforms the reference schemes in terms of its support, confidence, recall, precision, and F1-measure results.

INDEX TERMS Service recommendation, LDA, UGS, association rule, ProgrammableWeb.

I. INTRODUCTION

The increasing adoption of the service-oriented architecture (SOA) paradigm enables service composition as a means of building distributed applications [1]. In recent years, with the improvement of web 2.0 and the success of user-generated content (UGC), a new idea in which end users are enabled to create their own applications has emerged, termed user-generated service (UGS) [2]. UGS is a user-oriented service mashup or service process that allows users to create a new service mashup with large-granularity and added value via the aggregation of multiple service components [3]. To facilitate the creation of the service mashup for non-experienced end users, a number of development tools and frameworks have emerged in both academia and industry (e.g., Yahoo!Pipes, IFTTT, EZWEB, OMELETTE, LSMP etc.) [4]–[8].

These mashup tools assist in reducing the complexity of UGS by encapsulating multisource heterogeneous services as reusable service components, which greatly facilitates the improvement and development of UGS. Although these service mashup tools are gaining acceptance, creating the service mashup to address situational requirements is still challenging for end users with little or no programming skills [9]. To improve the quality and efficiency of UGS, service recommendation and service-linking mode recommendation are essential for end users who intend to create the service mashup for their individual requirements.

Existing service recommendation approaches and linking mode generation methods are mainly based on quality of service (QoS) measurement [10], [11], service network analyses (SNA) [15], [19], [20] and service association rule creation (SARC) [25], [26]. QoS-based recommendation methods apply QoS to evaluate the non-functional property of services, and aim to select candidate service components to create a service mashup with an optimal composite QoS [28], [29]. Algorithms for QoS measurement include probability matrix factorization (PMF) [11], regularized singular value decomposition (RSVD) [14] and the heuristic optimization algorithm [12]. SNA-based solutions regard services and relations among services as vertices and edges of a graph, respectively; the solutions of this category mainly include some linking algorithms (e.g., PageRank [32], Voronoi graph [38] etc.), which are used to find similar service nodes for the invocation service. SARC-based solutions mainly utilize FP-Growth [33] and Apriori [24], [34] to recommend suitable service components that can be connected to the given invocation service.

These algorithms improve the success rate of service mashups creation. However, few of these algorithms have considered the plight faced by the end users. The UGS domain has the following characteristics [38]. 1) Compared with the traditional movie domain and news domain, the data density of service invocation records for the UGS domain is sparser, which leads to the problem of ''data sparsity''; in addition, many users are invoking services for the first time, which is regarded as the ''cold start'' problem. Under data sparsity and cold start conditions, the actual similarities between services/users cannot be captured by analyzing user-service dualistic relationship. Therefore, an interest-aware (IA) model should be considered to extract the interests of the user by identifying the particular category of the current invocation service. 2) The aim of users' service invocation is to create a service mashup; however, non-experienced end users seldom understand current service technologies, such as web service definition language (WSDL), simple object access protocol (SOAP) and representational state transfer (REST) [39], and they cannot clearly analyze the correlations among service components. In other words, when an end user invokes a service, finding the services that match it is difficult. To improve the efficiency and correctness of the UGS, whenever the user invokes an existing service, a set of syntactically or semantically related services connected to the existing service should be suggested to the end user because these service linking modes can reduce confusion of the user.

The above analysis indicates that users' interests and service-linking modes must be considered synthetically. The interests of users are extracted via the users' service invocation, and the association among services is mined via the service composition records. QoS-based methods focus on the interests of users for non-functional attributes, but neglect function attributes of services and the relation among services in the service ecosystem. Huang *et al.* [18] pointed out that the QoS value of the service changes over time, so the service composition records and the service invocation history can more accurately represent the users' preferences and interests. Therefore, the end users in the service ecosystem are more concerned about the function attributes and the correlations among services relative to the non-functional properties of services. SNA-based methods mainly focus on the similarity of the services and the users, which neglect the the linking relations among the services. SARC-based methods mainly focus on the linking modes of services, which neglect the service invocation behaviors of users that depend on their latent interests. Previous studies [41], [42] have modeled the users' interests via term frequency/inverse document frequency (TF/IDF) and memory-based collaborative filtering (CF). However, without solving the problem of polysemy, the users' interests extracted via TF/IDF cannot meet the requirements of users, and since traditional CF algorithms lack a statistical basis, the user interests extracted from these models cannot be interpreted and the system is unable to recommend meaningful services.

To address the aforementioned challenges, we propose an interest-aware service association rule creation (SARC-IA) scheme with the following solutions: 1) we exploit the topic model latent Dirichlet allocation (LDA) [35], [36] as the IA model, which represents the relations of users and services via a three-layer scheme (user-interest-service). Each interest is a probability distribution over services, and each user is a probability distribution over interest. Therefore, the interest can be regarded as the feature factor of the service. When an end user invokes a certain service, we can extract the interest of the user via the features of the invocation service and recommend the services with the same features to the user; 2) we utilize the SARC algorithm to mine the service-linking modes based on the IA model and adopt a lift measurement to eliminate the negative rules. In contrast to previously presented association rules algorithms [34], [47], we adopt the SARC algorithm to create the association linking rules in the interest subsets of the dataset, thus enabling the algorithm to perform classification computation instead of global computation. In particular, the contributions of this paper can be summarized as following.

- The effective IA solution uses the LDA model to extract the latent interest of the user by analyzing current invocation service. The services that fit the user's interest are recommended to the user based on the optimal recommendation parameters and recommendation strategy.
- The interest subsets are formed based on the IA solution, the SARC algorithm is performed in the interest subsets to generate the service-linking modes, which can be recommended to the user for creating service mashups.
- Extensive experiments on the real-world Programmable Web service dataset show the following : 1) the number of interests in SARC-IA is 70; the optimal values of the hyperparameters α and β are 0.017 and 0.014, respectively; the recommendation parameters for the length of the recommendation list L and the return services r of each interest category are 40 and 15, respectively; the recommendation strategy considers the interests of both the current invocation service and the previous invocation service; 2) SARC-IA increases the precision by up to 8.1%, 25.5% and 73.6% compared with probabilistic

latent semantic analysis (PLSA) [37], latent semantic indexing (LSI) [50] and HITS [51], respectively, when the density of training set is 90%; 3) SARC-IA increases the recall by up to 26.6%, 42.9% and 83.2% compared with the PLSA, LSI and HITS, respectively, when the density of the training set is 90%; and 4) SARC-IA reduces the average time consumption by up to 71.3% compared with SARC [52] when generating the service-linking modes.

The remainder of this paper is organized as follows. Section II introduces the motivating example and overview of the SARC-IA scheme. Section III presents the details of the SARC-IA method for effective service recommendation and generation of positive service linking modes. Performance evaluation is provided in Section IV. We review and discuss the related works in Section V. Finally, conclusion remarks are given in Section VI.

II. MOTIVATING EXAMPLE AND OVERVIEW OF MODEL

In this section, we introduce a specific real-life example of UGS to illustrate the necessity of our scheme, then, we describe the offline training and online recommendation of the SARC-IA scheme. The basic notations used throughout this paper are listed in Table 1.

TABLE 1. Basic notations used throughout this paper.

Symbol	Definition
K	Number of interests (topics)
M	Number of users
V	Number of services
Θ	Multinomial distribution of interest to the user, θ_{ij} is equals to $p(z_j\|u_i)$
Φ	Multinomial distribution of services to the topic, ϕ_{ij} is equals to $p(s_j\|z_i)$
$\vec{S}$	Invocation record of a user
$\vec{\alpha}$	Dirichlet priors to the distribution Θ
$\vec{\beta}$	Dirichlet priors to the distribution Φ
z_{ij}	Interest associate with the i-th service token in service record of user j
s_{ij}	The i-th service token in service record of user j
$\triangle(\vec{\alpha})$	The normalization factor of Dirichlet distribution
$n_k^{(t)}$	The number of times service t that belongs to interest category k is invoked
$n_m^{(k)}$	The number of invocation services in k-th interest of the m-th user
$n_{k,\neg i}^{(t)}$	The number of services that are assigned to interest k except the service t
$n_{m,\neg i}^{(k)}$	The number of services about interest k that are invoked by user u_m except the service t
S_X, S_Y	The service set of interest subsets

A. MOTIVATING EXAMPLE

End users may not always have a clear idea of service composition goals in the process of creating a service mashup, more often they are interested in one or several multisource heterogeneous services (e.g., web service, Restful API, XML and RSS) and want to create an interest service application on the basis of these service components. An end user that invokes the Facebook service suffers from two pain-point problems as shown in Fig. 1:

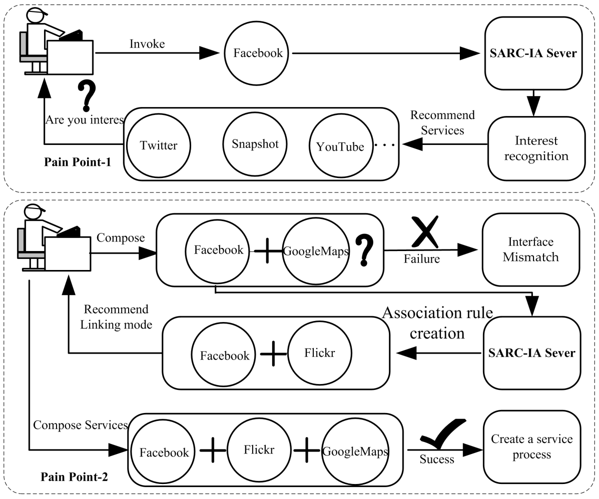


FIGURE 1. Pain point problems of a user in the UGS.

1) When the end user invokes the Facebook service, it indicates that the user is interested in social service or entertainment service (the Facebook service belongs to the social and entertainment category). Since the user's invocation behavior depends on the latent interest, therefore services with similar functions (e.g., Twitter, Weibo, YouTube and Snapshot, etc.) are also likely to be target services of interest for the user. When a large number of services fit the latent interest of a user, ranking services and recommending services to the user is challenging. In the traditional information domain, we can utilize the users' explicit rating values and preference keywords to recommend potential items of interest to the user; however, in the UGS domain, the only information that we can utilize is the sparse service invocation records of users and a large number of services. To address the "data sparsity" and "service overload" problem faced by the user in the service network, it is necessary to extract the latent interests of the user by identifying the interest categories of the current invocation service.

2) Without sufficient professional domain knowledge, the user cannot accurately invoke the appropriate services that can be linked by Facebook. If the user aims to compose the Facebook and Google Maps to show some photos of Facebook on Google Maps, the result of the service composition will fail because of the interface mismatch between Facebook and Google Maps. Fig. 1 shows that when linking mode *Facebook* $\rightarrow$ *Flickr* is recommended to the user, the user successfully creates the service mashup *Facebook* $\rightarrow$ *Flickr* $\rightarrow$ *Google Maps* to complete the request of the user. Therefore, discovering and recommending the appropriate association linking service is a key point for the user to create a successful service mashup.

If recommendation system cannot remedy these pain-point problems and makes recommendations only via the traditional dualistic relationship-based algorithms, then the end users will not be able to efficiently create successful service mashups.

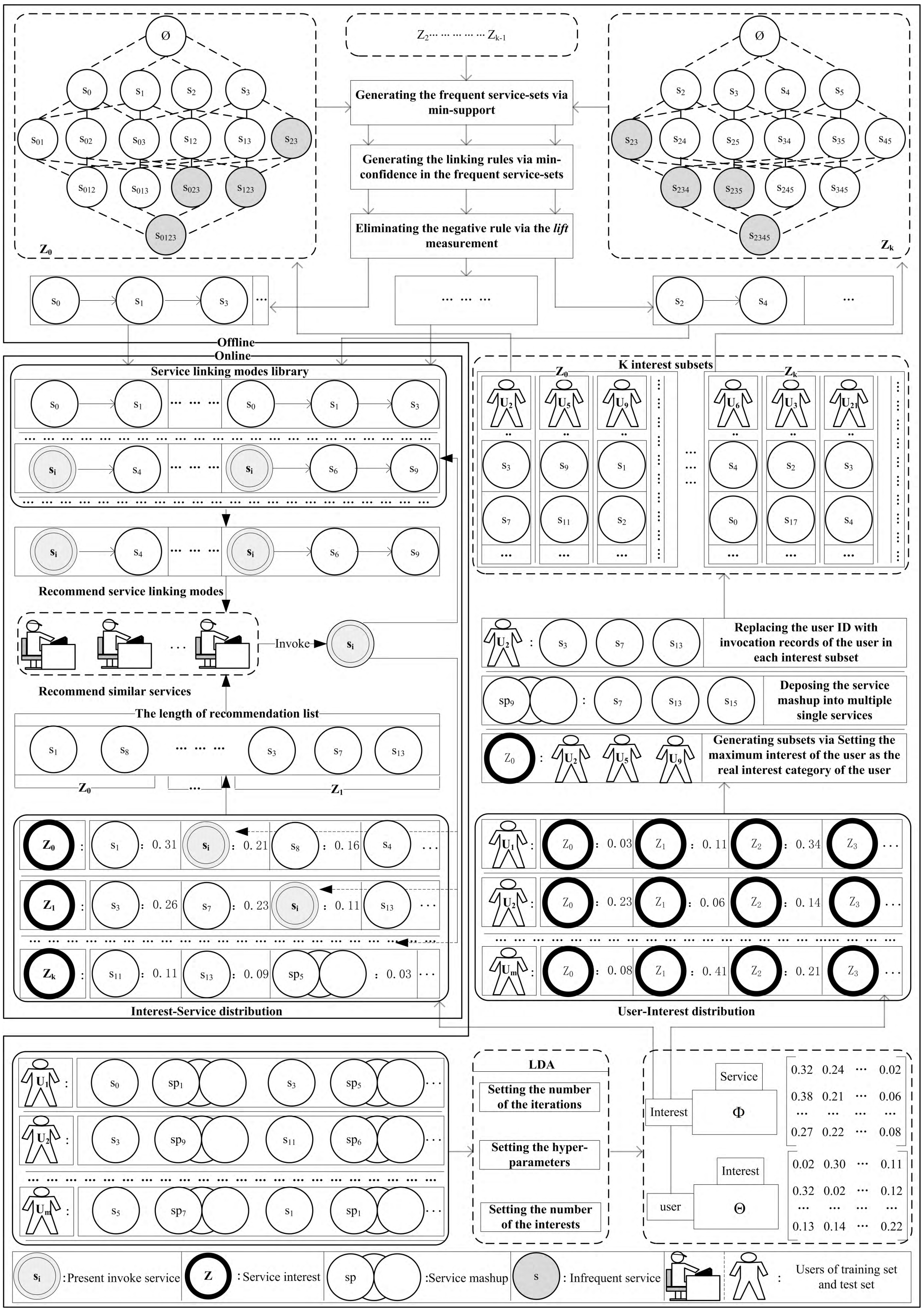

Generating the frequent service-sets via min-support
Generating the linking rules via min-confidence in the frequent service-sets
Eliminating the negative rule via the lift measurement
Offline
Online
Service linking modes library
Recommend service linking modes
Invoke
Recommend similar services
The length of recommendation list
Interest-Service distribution
K interest subsets
Replacing the user ID with invocation records of the user in each interest subset
Deposing the service mashup into multiple single services
Generating subsets via Setting the maximum interest of the user as the real interest category of the user
User-Interest distribution
LDA
Setting the number of the iterations
Setting the hyper-parameters
Setting the number of the interests
Service
Interest
Φ
user
Θ
: Present invoke service
: Service interest
: Service mashup
: Infrequent service
Users of training set and test set

FIGURE 2. Overview of the SARC-IA scheme.

B. OVERVIEW OF THE MODEL FRAMEWORK

As shown in Fig. 2, the SARC-IA framework includes two fundamental dynamic parts: offline training and online recommendation.

In the offline training stage, the aim of the SARC-IA framework is to generate the service interest categories and linking modes library. From Fig. 2, we can observe that the interest probability distribution and the user probability distribution are generated by utilizing the LDA model to train the invocation records of users. The interest probability distribution indicates that similar services are clustered to the same interest category, and each interest probability distribution can be regarded as a service interest category that can be used to recommend similar services. The user probability distribution indicates the probability of each user on each interest and can be used to generate the interest subsets of users. The service-linking modes can be generated in the user-interest subsets by utilizing the SARC algorithm. As illustrated in Fig. 2, we set the maximum interest of the user-interest distribution as the real interest category of the user, similar users are clustered to the same interest category, and different interest subsets are formed by replacing the user ID with the service invocation records of the user. In each interest subset, the service-linking mode generation process is as follows: first, the frequent service sets are discovered by setting a min-support value, and the service-linking modes are generated via setting a suitable min-confidence value; then negative linking modes are eliminated via the lift measurement; finally a service-linking mode library is formed by aggregating the service-linking modes of each interest subset.

In the online recommendation stage, the aim of the SARC-IA is to recommend suitable similar services and service-linking modes. When a user invokes a certain service, the latent interest of the user is extracted by identifying the interest category that this service belongs to, meanwhile, the service-linking modes related to this service are looked up in the service-linking mode library. Therefore, two recommendation lists are generated for the user: one recommends similar services to the user, and the other recommends the linking modes about the invoked service. In general, a service belongs to more than one interest category, the similar services list is composed of return services of these interest categories. In our SARC-IA scheme, for the service recommendation, we focus on the recommendation parameters (i.e., the length of recommendation list and the number of return services for each interest category) and recommendation strategy (i.e., how to deal with the relationship between the current invocation service and the previous invocation service) to improve the recommendation efficiency, for the service-linking mode recommendation, since the linking relationship among services is relatively fixed, we focus on the generation efficiency of service-linking modes (i.e., suitable confidence values and support values for the dataset, the number of rules to be deleted via lift measurement, time consumption of linking modes generation).

III. PROPOSED SARC-IA METHOD

In this section, we show how to extract the latent interests of users and services via the LDA model and create the service-linking modes via SARC algorithm in the interest subsets.

A. EXTRACTING INTERESTS VIA THE LDA MODEL

The LDA model is a probabilistic generative model that can be used to estimate the properties of multinomial observations

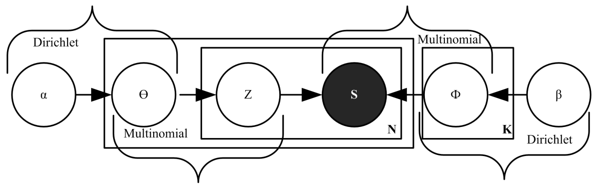


FIGURE 3. LDA probabilistic graphical model.

by unsupervised learning as illustrated in Fig. 3, in which shaded and unshaded variables indicate observed and latent variables, respectively; boxes represent sampling with repetition; and arrows express dependent relationships among variables. In this paper, the LDA model is utilized as the IA model of the SARC-IA scheme, the observed variables are the service invocation records of users, and the services invocations of each user are represented as a "bag of service", which is defined as $\vec{s}$. Each invocation service is considered as an observed variable, and the latent variable z represents the interests of users. As a type of generative probabilistic model, the LDA model can be decomposed into two main physical processes: $\vec{\alpha} \to \vec{\theta}_m \to \vec{z}_{ij}$ and $\vec{\beta} \to \vec{\phi}_k \to \vec{s}_{ij}$. These two processes are the Dirichlet-Multinomial conjugated structure. As shown in Table 1, Θ is a matrix with the dimensions $M \times K$, Φ is a matrix with the dimensions $K \times V$, $\vec{\theta}_m$ represents the interest distribution of the m-th user, and $\vec{\phi}_k$ represents the service distribution of the k-th interest. $\vec{\theta}_m \to \vec{z}_{ij}$ indicates that the distribution of a user's interest corresponds to the Multinomial distribution, and $\vec{\alpha} \to \vec{\theta}_m$ indicates that the priori distribution of the interest distribution is the Dirichlet distribution $Dir(\vec{\theta}|\vec{\alpha})$, whose parameter is α. $\vec{\alpha} \to \vec{\theta}_m \to \vec{z}_{ij}$ indicates that the behavior of the user's service invocation depends on the corresponding latent interest. Similarly, $\vec{\beta} \to \vec{\phi}_k$ corresponds to the Dirichlet distribution, $\vec{\phi}_k \to \vec{s}_{ij}$ corresponds to the Multinomial distribution, $\vec{\beta} \to \vec{\phi}_k \to \vec{s}_{ij}$ indicates that the service distribution of the interest category is the Multinomial distribution, and the prior distribution of this distribution is the Dirichlet distribution $Dir(\vec{\phi}|\vec{\beta})$, whose parameter is β. The probability density function of the Dirichlet distribution can be identified by the following equation:

$$Dir(\vec{p}|\vec{\alpha}) = \underbrace{\frac{\Gamma(\sum_{m=1}^{M} \alpha_i)}{\prod_{m=1}^{M} \Gamma(\alpha_i)}}_{1/\triangle(\vec{\alpha})} \prod_{m=1}^{M} p_m^{\alpha_m - 1} \tag{1}$$

where $\triangle(\vec{\alpha})$ (or$\triangle(\vec{\beta})$) is the normalization factor of the Dirichlet distribution. The default value of α in the LDA model is generally less than 1 since the interests of users are always focused on some certain categories in a real-life scenario. To extract the latent interest of users and services, the joint distribution of latent variables is defined as follows:

$$p(\vec{s}, \vec{z}|\vec{\alpha}, \vec{\beta}) = p(\vec{s}|\vec{z}, \vec{\beta})p(\vec{z}|\vec{\alpha}) \tag{2}$$

$p(\vec{s}|\vec{z},\vec{\beta})$ and $p(\vec{z}|\vec{\alpha})$ can be dealt with separately, $p(\vec{s}|\vec{z},\vec{\beta})$ denotes the likelihood function of the service distribution under the given interests of users, and $n_k^{(t)}$ is introduced to represent the number of times that service t, which belongs to interest category k, is invoked. Then, $p(\vec{s}|\vec{z},\vec{\beta})$ can be represented as follows:

$$\begin{aligned} p(\vec{s}|\vec{z},\vec{\beta}) &= \int \underbrace{p(\vec{s}|\vec{z},\phi)}_{\prod_{k=1}^{K}\prod_{t=1}^{V}\phi_{k,t}^{n_k^{(t)}}} \quad \underbrace{p(\phi|\vec{\beta})}_{\prod_{k=1}^{K}\frac{1}{\Delta(\vec{\beta})}\prod_{t=1}^{V}\phi_{k,t}^{\beta_t-1}} \quad d\phi \\ &= \int \prod_{k=1}^{K}\frac{1}{\Delta(\vec{\beta})}\prod_{t=1}^{V}\phi_{k,t}^{n_k^{(t)}+\beta_t-1}d\vec{\phi}_k \\ &= \prod_{k=1}^{K}\frac{\Delta(\vec{n}_k+\vec{\beta})}{\Delta(\vec{\beta})}, \quad \vec{n}_k=\{n_k^{(t)}\}_{t=1}^{V} \end{aligned} \tag{3}$$

Equation (3) is a Dirichlet-Multinomial conjugated structure, where $\vec{n}_k$ represents the distribution of services for the k-th interest category. Similar to $p(\vec{s}|\vec{z},\vec{\beta})$, $p(\vec{z}|\vec{\alpha})$ can be represented as follows:

$$\begin{aligned} p(\vec{z}|\vec{\alpha}) &= \int p(\vec{z}|\theta)p(\theta|\vec{\alpha})d\theta \\ &= \int \prod_{m=1}^{M}\frac{1}{\Delta(\vec{\alpha})}\prod_{k=1}^{K}\theta_{m,k}^{n_m^{(k)}+\alpha_k-1}d\vec{\theta}_m \\ &= \prod_{m=1}^{M}\frac{\Delta(\vec{n}_m+\vec{\alpha})}{\Delta(\vec{\alpha})}, \quad \vec{n}_m=\{n_m^{(k)}\}_{k=1}^{K} \end{aligned} \tag{4}$$

where $\vec{n}_m$ represents the distribution of interests for the n-th users, and $n_m^{(k)}$ denotes the number of invocation services in the k-th interest of the m-th user (i.e., the count of interest k assigned to the services). The joint distribution of latent variables can be expressed as follows by employing (3) and (4):

$$p(\vec{s},\vec{z}|\vec{\alpha},\vec{\beta}) = \prod_{k=1}^{K}\frac{\Delta(\vec{n}_k+\vec{\beta})}{\Delta(\vec{\beta})}\cdot\prod_{m=1}^{M}\frac{\Delta(\vec{n}_m+\vec{\alpha})}{\Delta(\vec{\alpha})} \tag{5}$$

In the service recommendation process, the interests of a user are extracted by identifying the interest category of the current invocation service of the user (i.e., $p(\vec{z}|\vec{s})$), in this paper, we select the Collapsed Gibbs sampling algorithm [44] to calculate $p(\vec{z}|\vec{s})$. The Gibbs sampling algorithm is one of Markov chain Monte Carlo (MCMC) algorithms that constructs random samples of a multivariate probability distribution by utilizing the convergence principle of the Markov chain, $(\vec{s},\vec{z})$ is regarded as the random variable, and (5) is the probability distribution of the random variable. The main idea of Gibbs sampling is to estimate the interest transition probability of service by sampling the probability distribution (5). During the initial steps, each service token is assigned to a random interest, and then in the process of sampling, the interest transition probability of each service can be obtained by employing the interest distribution of other services, in other words, we utilize the $p(z_i|\vec{z}_{\neg i},\vec{s})$ to fit $p(\vec{z}|\vec{s})$. Suppose an observed variable $s_i = t$, where $i = (m, n)$ is the subscript that refers to the n-th invocation service of user u_m's invocation records; then the sampling expression (i.e., full conditional probability of services) can be obtained conveniently via the Bayesian rule:

$$\begin{aligned} p(z_i=k|\vec{z}_{\neg i},\vec{s}) &= \frac{p(\vec{s},\vec{z})}{p(\vec{s},\vec{z}_{\neg i})} = \frac{p(\vec{s}|\vec{z})}{p(\vec{s}_{\neg i}|\vec{z}_{\neg i})p(s_i)}\cdot\frac{p(\vec{z})}{p(\vec{z}_{\neg i})} \\ &\propto \frac{\Delta(\vec{n}_z+\vec{\beta})}{\Delta(\vec{n}_{z,\neg i}+\vec{\beta})}\cdot\frac{\Delta(\vec{n}_m+\vec{\alpha})}{\Delta(\vec{n}_{m,\neg i}+\vec{\alpha})} \\ &= \frac{n_{k,\neg i}^{(t)}+\beta t}{\sum_{t=1}^{V}(n_{k,\neg i}^{(t)}+\beta_t)}\cdot\frac{n_{m,\neg i}^{(k)}+\alpha_k}{[\sum_{k=1}^{K}(n_m^{(k)}+\alpha_k)]-1} \end{aligned} \tag{6}$$

where $\vec{z}_{\neg i}$ denotes all the current interest assignments of services except for the service s_i, $n_{k,\neg i}^{(t)}$ refers to the number of services that are assigned to interest k except the service t (i.e., s_i), and $n_{m,\neg i}^{(k)}$ represents the number of services about interest k that are invoked by user u_m except for the service s_i.

The conditional probability of the service over each interest can be obtained by (6), where each service is assigned a new interest by employing the roulette algorithm in each iteration, When the number of iterations is sufficient, the interest assignment for each service token will converge and each service token is assigned to a "stable" interest. The distribution and prior distribution of $\vec{\theta}_m$ and $\vec{\phi}_k$ are the Multinomial distribution and the Dirichlet distribution, respectively. According to the property of the Dirichlet-Multinomial conjugate distribution, the posterior probabilities of $\vec{\theta}_m$ and $\vec{\phi}_k$ submit to the Dirichlet distribution, which are $Dir(\vec{\theta}_m|\vec{n}_{m,\neg i}+\vec{\alpha})$ and $Dir(\vec{\theta}_m|\vec{n}_{k,\neg i}+\vec{\beta})$, respectively. The distribution of interest over service ϕ_{kt} and the distribution of user over interest θ_{mk} can be obtained by calculating the expectation of $Dir(\vec{\theta}_m|\vec{n}_{m,\neg i}+\vec{\alpha})$ and $Dir(\vec{\theta}_m|\vec{n}_{k,\neg i}+\vec{\beta})$, and the expression is shown below:

$$\begin{aligned} \theta_{mk} &= p(z_k|u_m) = \frac{n_{m,\neg i}^{(k)}+\alpha_k}{\sum_{k=1}^{K}(n_{m,\neg i}^{(k)}+\alpha_k)} \\ \phi_{kt} &= p(s_t|z_k) = \frac{n_{k,\neg i}^{(t)}+\beta_t}{\sum_{t=1}^{V}(n_{k,\neg i}^{(t)}+\beta_t)} \end{aligned} \tag{7}$$

The user-service dataset is trained via (6) and (7) to generate the two data matrices, which are the interest-service matrix and the user-interest matrix. The interest-service matrix is used for online recommendation. When a user invokes a certain service s_i, this service is retrieved to determine which interest categories it belongs to, and the preceding r items of interest categories are inserted into the recommendation list after the interest categories are identified. The services of the recommendation list are similar services of the current invocation service. These recommendation services are identified by the following equation:

$$Re(s_i) = \{s_{z_k}^r|N(\sum_{r=1}^{R}\sum_{k=1}^{K}s_{z_k}^r) = L, (s_i, s_{z_k}^r) \in \vec{z}_k\} \tag{8}$$

where $Re(s_i)$ represents the recommendation services of the invocation service s_i, $s^r_{z_k}$ denotes the r-th return service of interest category z_k, L indicates the length of recommendation list for s_i, and the range of r and L are [5 − 20] and [5 − 50], respectively. When a certain service of the recommendation list is invoked, the process described above is repeated.

B. DISCOVERING THE SERVICE LINKING MODE VIA THE SARC ALGORITHM

An association rule is an implication of the form $A \rightarrow B$, where A and B are frequent item-sets in a database and $A \cap B = \varnothing$. A is called the antecedent of the rule, and B is called the consequent of the rule. In practical UGS applications, the rule $s_x \rightarrow s_y$ can be used to perform the following prediction: "if a service s_x is invoked in creating a service mashup by a user, then service s_y will likely also occur in the same service mashup". Therefore, the SARC (service association rule creation) algorithm can be used to recommend suitable service-linking modes for the active user during service mashup creation. Since users with different interests are focused on different service mashups, the implementation of the SARC algorithm in the whole dataset will lead to low efficiency and high time consumption. Our SARC-IA scheme performs the classification calculation based on the interests of users (i.e., the SARC algorithm is performed in the interest subsets), and the generated service-linking rules are collected to form the linking modes library for linking mode recommendation. The generation of interest subsets is shown in Fig. 4. First, the user-service frequent

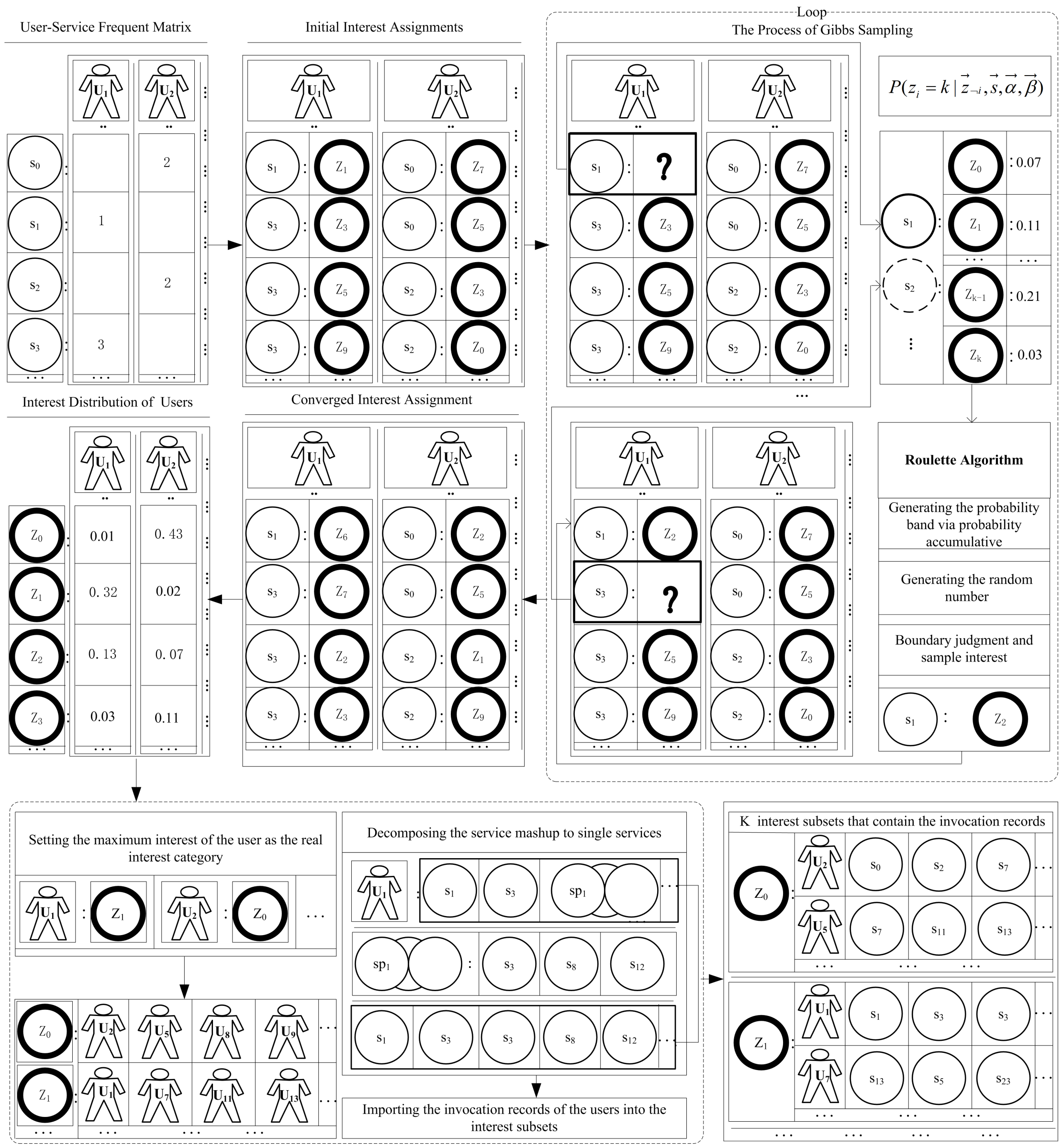

User-Service Frequent Matrix
Initial Interest Assignments
Loop
The Process of Gibbs Sampling
$P(z_i = k \mid \vec{z}_{\neg i}, \vec{s}, \vec{\alpha}, \vec{\beta})$
Interest Distribution of Users
Converged Interest Assignment
Roulette Algorithm
Generating the probability band via probability accumulative
Generating the random number
Boundary judgment and sample interest
Setting the maximum interest of the user as the real interest category
Decomposing the service mashup to single services
Importing the invocation records of the users into the interest subsets
K interest subsets that contain the invocation records

FIGURE 4. Generation of the interest subsets.

matrix is converted into the interest distribution matrix via the Gibbs sampling algorithm (the details of Gibbs sampling have been described in subsection A), and users with different preferences have different distributions of interest. Then, for each user, we set the maximum interest ratio of the user as the interest category of the user, and interest-user subsets are generated by collecting users with the same interest category. Finally, by decomposing the service mashup into the single services and importing the service invocation records of users, K interest subsets that contain the invocation records are formed.

A set of services (such as the antecedent or the consequent of a rule) is called a service-set. The quality of a service linking rule is defined by its *support* and *confidence*. The *support* of a rule is an estimate of the probability of the service-set, and the *confidence* of a rule is the conditional probability of service-set S_Y given service-set S_X. The support and confidence can be calculated via the following equation:

$$\begin{aligned} Support(S_X \rightarrow S_Y) &= p(S_X.S_Y) \\ Confidence(S_X \rightarrow S_Y) &= p(S_Y|S_X) = \frac{p(S_X.S_Y)}{p(S_X)} \end{aligned} \quad (9)$$

The idea of the SARC algorithm is to explore $(q+1)$-service sets by utilizing q-service sets. By scanning the services invocation records of an interest subset, the frequent 1-service sets that meet the minimum support are found, the resulting service sets are denoted L_1, and then L_1 are explored to search the set of frequent 2-service sets. Similarly, 2-service sets are denoted L_2 and can be used to search L_3. In the UGS domain, most service mashups consist of no more than four service components, and 3-service sets are considered sufficient for mining the service-linking modes. The frequent service sets are generated by scanning the whole dataset, which is unnecessary and very time-consuming, The SARC algorithm employs a special strategy that is termed the Apriori property to reduce the search space. In general, the SARC algorithm consists of the join step and the prune step.

The join step: To find L_q, a set of candidate q-service sets is generated by joining L_{q-1} with itself. The candidate service set is termed C_q. Suppose l_1 and l_2 are service sets of L_{q-1}, the condition that l_1 and l_2 can perform the join operation is as follows:

$$(l_1[1] = l_2[1]) \wedge (l_1[1] = l_2[1]) \wedge \ldots (l_1[k-1] < l_2[k-1]) \quad (10)$$

The prune step: C_q generated in the join step is huge and includes a number of non-frequent service sets. The aim of prune step is to improve the efficiency of the level-wise search and reduce the computational complexity by utilizing the Apriori property and minimum support count. By scanning the service invocation records of the similar user set, the frequent service-sets can be found by comparing the count of candidate services to the minimum support count. If any $(q-1)$-subset of the candidate q-service set is not in the collection (i.e., L_{q-1}), they are considered the non-frequent service sets and can be removed from C_q. A set of service linking rules are created by setting the minimum confidence after the frequent service sets are generated.

However, relying only on support and confidence of the SARC to find the service-linking rules is insufficient, we present an example to describe a misleading notion in the support-confidence framework of SARC. Suppose that the users are interested in analyzing services linking relations at services invoke records with respect to Google Maps and Last.fm. Of the 10,000 users' service invocation records, the data show that 6,500 of the service invocation records include Last.fm, 7,000 include Google Maps, and 4,000 include both Last.fm and Google Maps. The following service-linking rule is discovered by using a minimum support of 0.2 and a minimum confidence of 0.6:

$$\begin{aligned} &Invoke(X,' Last.fm') \rightarrow Invoke(X,' GoogleMaps') \\ &\qquad [support = 0.4, confidence = 0.61] \end{aligned} \quad (11)$$

where X is a variable representing users who invoke services when generating service mashups. (11) is a strong service-linking association rule since its support value and confidence value satisfy the minimum support and minimum confidence thresholds, respectively. However, (11) is misleading because the probability of invoking Google Maps is 0.70, which is greater than 0.61. In fact, Last.fm and Google Maps have negative linking association because the invocation of one service actually decreases the likelihood of invoking the other service.

The above example illustrates that the confidence of the rule $S_X \rightarrow S_Y$ can be deceiving because it is only an estimate of the conditional probability of service set S_Y given service set S_X and does not measure the real strength (or lack of strength) of the correlation and implication between S_X and S_Y. Hence, optimizing the support-confidence framework of SARC algorithm is necessary to mine service linking rules. To tackle this weakness of the support-confidence framework, a correlation measure can be utilized to augment the support-confidence framework for linking association rules. The form of the correlation rules is as follows:

$$S_X \rightarrow S_Y[support, confidence, correlation] \quad (12)$$

The correlation is measured not only by its support and confidence but also by the correlation between service-set S_X and service-set S_Y. Lift is a correlation measure that is given as follows. The occurrence of service-set S_X is independent of the occurrence of service-set S_Y if $p(S_X \cup S_Y) = p(S_X)p(S_Y)$; otherwise, service-set S_X and service-set S_Y are dependent and correlated as events. The *lift* measurement can be represented by the following equation:

$$lift(S_X, S_Y) = \frac{p(S_X \cup S_Y)}{p(S_X)pS_Y} = \frac{confidence(S_X \rightarrow S_Y)}{support(S_Y)} \quad (13)$$

If the result of (13) is less than 1, then the occurrence of S_X is negatively correlated with the occurrence of S_Y. We can calculate the lift value of the previous example via (13), and the lift value is $p(Last.fm, GoogleMaps)/\{p(Last.fm).p(GoogleMaps)\} = 0.4/(0.65 \times 0.7) = 0.88$. Since the lift value is less than 1, a negative correlation is observed between the occurrence of Last.fm and Google Maps, and such a negative correlation cannot be identified by the support-confidence framework. Therefore, the generation of a strong service-linking rule needs to content not only the requirement of min-support and min-confidence, but also the lift value of the linking rule greater than 1. After the strong service-linking rules of each interest subset are generated, the service-linking modes library is formed by collecting the strong service-linking rules of all interest subsets. When a user invokes a certain service, the SARC-IA method retrieves the linking modes that are related to this service in the service-linking modes library, and these linking modes are recommended to the user.

IV. EXPERIMENTS

In this section, we conduct experiments to validate our proposed method and compare the performance of the method with other methods. Our experiments are intended to address the following questions: 1) How are the optimal parameters determined in the process of extracting interest? 2) How do the recommendation parameters and recommendation strategy affect the recommendation performance in the recommendation process? 3) How does our approach compare with the baseline algorithms regarding the service recommendation performance? 4) How does our approach compare with the baseline algorithm regarding the performance of creating service-linking modes?

A. EXPERIMENTAL SETUP

We illustrate the experimental dataset, evaluation metrics and baseline algorithms in the experimental setup section.

1) DATASET DESCRIPTION

Experiments are conducted by utilizing the three datasets of the ProgrammableWeb website, which are the service invocation records of users, the service mashups dataset and the service components dataset. As shown in Table 2, the Service mashups dataset includes 13082 mashups, the service components dataset includes 10648 services, and the service invocation records of users include invocation records of 20035 users. In a real-life scenario, some users invoke a small number of services, to improve the accuracy of the experiments, we eliminate the data records of inactive users (in this paper, we regard users who invoke the service less than 3 as inactive users, therefore, the service invocation records of 11730 active users are used to perform the experiments).

TABLE 2. Statistics of the ProgrammableWeb data-sets.

Statistics	Values
Number of users	20035
Number of services component	10648
Number of of mashup	13082
Number of services component category	67
Number of mashup category	24

2) EVALUATION METRICS

To achieve the optimal performance of our method, perplexity is used to discover the optimal parameters of the LDA model. To perform validation and performance analyses of proposed method, we utilize the recall, precision and F1-measure to evaluate the recommendation quality of our method in comparison with other baseline algorithms. Perplexity indicates the average probability value of the service. Precision indicates the ratio of invocation services in the recommendation list to the total L recommendation services. Recall indicates the ratio of invocation services in the recommendation list to the total invocation services of users. These metric formulas are defined as follows:

$$\begin{aligned} Perplexity &= exp \wedge \{-(\sum_{t=1}^{V} log(p(s_t)))/(V)\} \\ where\,[p(s_t) &= \sum_{k=0}^{K}\sum_{m=0}^{M} p(s_t|z_k).p(z_k|u_m)] \\ Precision &= \frac{N(Re(s) \cap U_{re,\neg s})}{L} \\ Recall &= \frac{N(Re(s) \cap U_{re,\neg s})}{N(U_{re,\neg s})} \\ F1-measure &= \frac{2.Precision.Recall}{Precision + Recall} \end{aligned} \tag{14}$$

where V indicates the number of services tokens, $Re(s)$ represents the recommendation services for the current invocation service s, $U_{re,\neg s}$ indicates the invocation records of a user except the current invocation service s, and L represents the length of the recommendation list.

3) BASELINE ALGORITHMS

To show the performance of our SARC-IA scheme, we compare our approach with the following methods:

1) LSI [50]. This approach is proposed by Scott Deerwester et al. and utilizes the technique of Singular Value Decomposition (SVD) to reduce the data dimensions and discover similar elements (e.g., similar semantic or interest).
2) PLSA [37]. This method is a topic model and employs the EM (expectation-maximization) algorithm to infer the hidden variable of the model (e.g., document topic or interest).
3) HITS [51]. This method is a node-analysis model and it defines the two concepts of authority and hub, certain high-quality services or items can be discovered via the mutual reinforcement of the authority and hub.
4) SARC [52]. This method is used to create service association rules and employs the layer-by-layer iterative search strategy to discover the frequent composition modes; association rules can be inferred via the frequent composition patterns.

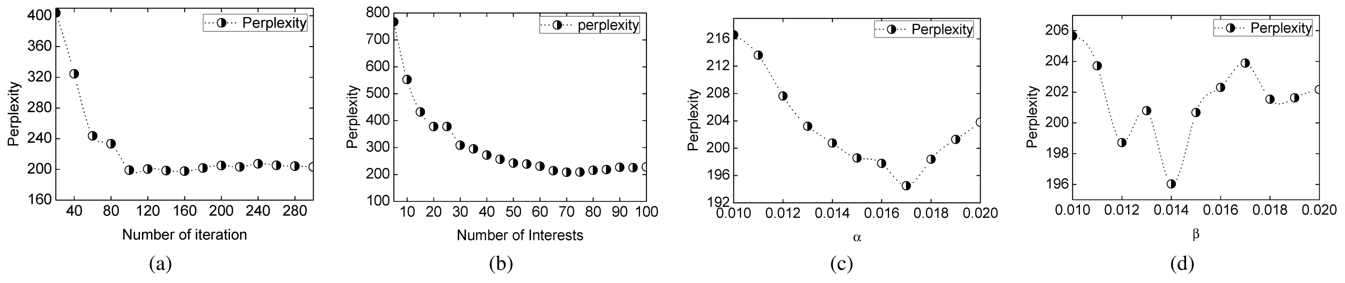


FIGURE 5. LDA parameters.

To demonstrate the recommendation efficiency of SARC-IA, we compare it with LSI, PLSA and HITS. To demonstrate the service-linking modes generation efficiency of SARC-IA, we compare it with SARC.

B. DISCOVERING THE OPTIMAL PARAMETERS OF LDA

The optimal parameters of the proposed scheme should be discovered before we compare these algorithms. For the IA (interest aware) algorithm of SARC-IA (i.e., LDA algorithm), the parameters are K (the number of interests), α (hyperparameter of the users-interest distribution) and β (hyperparameter of the interest-service distribution).

The service invocation records of users include single services and service mashups, with 67 categories of single services and 24 categories of service mashups. Therefore, we first assume that number of interests is 60 (i.e., $K = 60$) based on the existing service categories. The hyper parameters of the LDA model are set as the default value (i.e., $\alpha = 1/K$ and $\beta = 1/K$). Then, a set of perplexity values of the model are calculated by changing the number of iterations from 20 to 300, we can obtain the optimal iterations number for model convergence by comparing the perplexity values obtained above. Finally, the hyperparameters can be calculated by the fixed iterative method (i.e., the optimal value of a hyperparameter is calculated by fixing the other hyperparameter and the number of users' interests). The results of the training parameters are shown in Fig. 5.

Fig. 5a shows that the LDA model is convergent when the number of iterations is 160: at this point, the perplexity value of the model is 207.71, which is the minimum among 15 iterative processes. In Fig. 5b, a series of values of perplexity are shown by changing the number of interests from 5 to 100. When the number of interests is 70, the perplexity of the model is at the minimum of 205.27 (i.e., the performance of the model is best when the number of interests is 70). To discover the optimal value of α, we set the number of interests is 70 and the value of β is 1/70, we then obtain a series of perplexity values by changing the value of α from 0.010 to 0.020. Fig. 5c shows that the optimal value of α is 0.017. The parameter value of β can be calculated by setting the optimal value of the other parameters (i.e., $K = 70$, $\alpha = 0.017$), from Fig. 5d we observe that the optimal value of β is 0.014.

C. PARAMETERS AND STRATEGIES IN THE RECOMMENDATION PROCESS

In the LDA model, we set the optimal value of parameters as follows: $K = 70, \alpha = 0.017, \beta = 0.014$. The service invocation records of the users are trained by the LDA model to obtain two data distribution: the interest-service clustering set and the user-interest distribution set. When a certain user invokes a service, the service is first determined to belong to particular interest categories, and then the preceding r items of the interest categories are inserted into the recommendation list for the user. In the process of recommending services to users, three aspects of the problem should be considered. 1) The first problem is the length of the recommendation list. 2) The second problem is that when a service invoked by the user belongs to more than one interest category, the number of services that each interest category returns should be considered. 3) The third problem is that when a user invokes the second service, the strategy of addressing the relationship between the interest of the first service and the interest of the second service should be considered. For the first of these three problems, a series of values of three metrics can be obtained by changing the length of the recommendation list L, and the optimal length of the recommendation list that corresponds to the optimal values of the metrics can be discovered. For the second of the three problems, when a service invoked by a user belongs to multiple interest categories, the preceding r terms of each interest category are inserted into the recommendation list. A series of metric values are obtained by changing the value of r, and the optimal value of the r can be discovered via the optimal metric values. Three strategies are used to address the last problem: 1) only services similar to the second services are recommended, 2) the services of common interests between the first and the second service are recommended, and 3) the interests of the first and second service are both considered. We set 90% of the dataset as the training set and the remaining 10% of the dataset as the test set. These three problems can be regarded as solving the optimal values of the two parameters (i.e., the length of the recommendation list L and the number of return services r for each interest category) under the above three strategies. The precision, recall and F1-measure of the services recommendation change as the parameters and strategies change, and the experimental data are shown in Fig. 6 to Fig. 8.

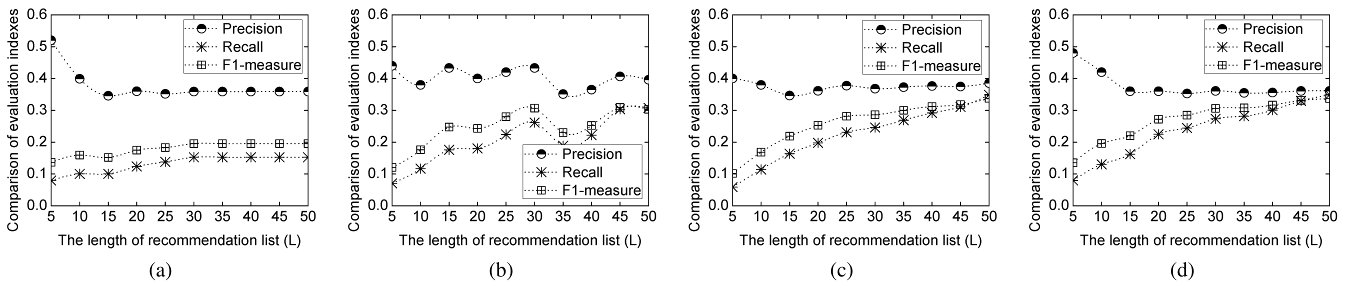


FIGURE 6. Impact of parameters and strategy 1 (a: r=5, b:r=10, c:r=15, d: r=20).

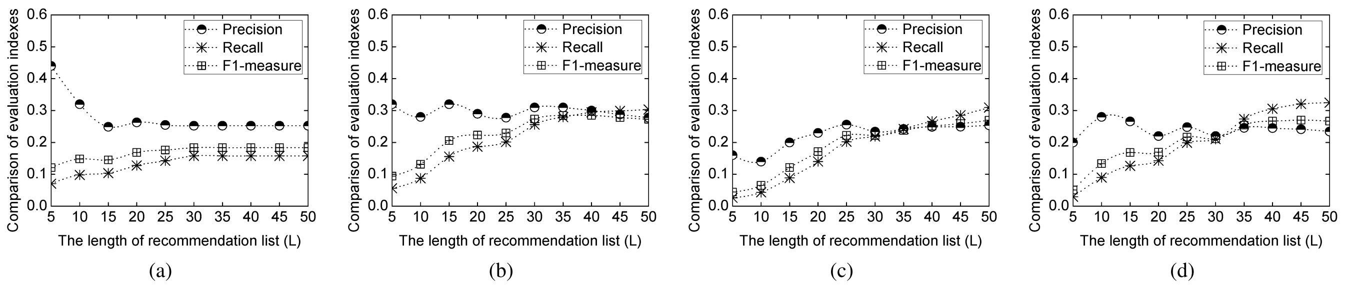


FIGURE 7. Impact of parameters and strategy 2 (a: r=5, b:r=10, c:r=15, d: r=20).

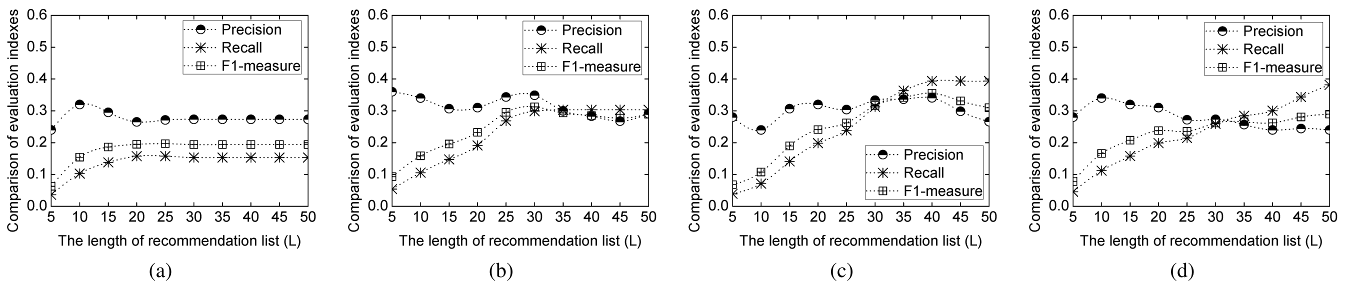


FIGURE 8. Impact of parameters and strategy 3 (a: r=5, b:r=10, c:r=15, d: r=20).

Fig. 6 illustrates the changes in recommendation parameters only when the current invocation service is considered. The precision value is highest when the number of return services is 5; however, the F1-measure value and the recall value are the lowest at this time. The precision value is lowest when the number of return services is 15. When the number of return services is 10, the F1-measure value is highest. Fig. 6a to Fig. 6d show that the value of three metric are optimal when the length of the recommendation list is 35.

Fig. 7 shows the changes in recommendation parameters when the common interests among the previous invocation service and current invocation service are considered. The four sub-figures illustrate that the performance of Fig. 7b is the best (i.e., when the number of return services is 10, the precision and F1-measure values are both the highest). A comparison with Fig. 6 shows that the performance obviously declines in Fig. 7. When the number of return services is 15, the performance of the model is the lowest, which indicates that considering the common interest of services can weaken the expansibility of interests.

Fig. 8 shows the changes in the recommendation parameter when both the interest of the previous service and the interest of the current service are considered. The recommendation list is composed of the return services of the two services' interests. In our experiments, the proportions of the recommendation list are as follows: the return services of the present invocation service account for 75%, and the return services of the previous invocation service account for 25%. The four sub-figures of Fig. 8 show that the performance of the recommendation is the best when the length of the recommendation list is 40 and the number of return services is 15.

The optimal values of the parameters (i.e., L and r) and the optimal strategy can be obtained by comparing the results in the three figures.

1) Length of the recommendation list L: In most cases, the performance of the recommendation decreases when L is greater than 40. Although the performance of Fig. 8d increases when L is 45, a recommendation list with a length exceeding 40 will wreck the user experience, and thus optimal value of L is 40.

2) Number of services returned by each interest r: As shown in the above figures, when r is 5, at most 30 services can be discovered. Even if L is 40, the number of recommendation services is only 30. The precision value is high in this case, but the recall value is low. When r is 15, the precision values in Fig. 6c, Fig. 7c and Fig. 8c are 0.3765, 0.2488 and 0.3407, respectively, which are the highest precision values under the same strategy, and thus optimal value of r is 15.
3) Strategy for addressing the relationship between the current invocation service and the previous invocation service: Compared with Fig. 6 and Fig. 8, Fig. 7 shows the worst performance because considering the common interests of services will limit the diversity of interests. Fig. 6 illustrates high precision compared with Fig. 7 and Fig. 8, and Fig. 6c has the highest precision value when the length is 40 and the number of return services is 15. The precision in Fig. 6c is 10.6% higher than that in Fig. 8c. However, the recall of Fig. 6c is 34.5% lower than that in Fig. 8c, and the F1-measure of Fig. 8c is 13.9% higher than that in Fig. 6c. Therefore, the optimal strategy is to consider both the interest of the current invocation service and the interest of the previous invocation service by comprehensive comparison based on the three metrics. Such a strategy can guarantee the precision of the model and increase the diversity of interests.

D. PERFORMANCE COMPARISON FOR SERVICE RECOMMENDATION

To compare the results of these algorithms, we select different densities (i.e., 30%, 50%, 70% and 90%) of the service invocation dataset to compare their performance. Data density of 30% means that we randomly select 30% of the data as the training data to predict the remaining 70% of the data. The parameters settings and strategy selection are as follow: $K = 70, \alpha = 0.017, \beta = 0.014, L = 40, r = 15$, and the recommendation strategy comprehensively considers the interest of the current service and previous service. The comparison results of the experiments are shown in Table 3.

Table 3 shows that our SARC-IA method obtains larger values for the three evaluation metrics in different matrix densities, which indicates that SARC-IA has a better recommendation performance than do the other baseline algorithms. With the increase of matrix density, the metric values of the four algorithms also increase. Among the four algorithms, the topic models, including the SARC-IA, PLSA and LSI, perform far better than the HITS model. When the matrix density is 90%, the performance of SARC-IA is almost 1.8 times higher than that of HITS because the HITS model focuses on only the user-service dualistic relationship; it recommends services by calculating hub and authority without considering the topic level, which results in the low recommendation performance of HITS. When the matrix density is 90%, the performance of PLSA and LSI are close to that of SARC-IA; this finding is consistent with our previous work [31], which validates that PLSA and LSI perform better in large datasets (e.g., MovieLens) than in small datasets (e.g., ProgrammableWeb). When the matrix density is 30%, the precision values of PLSA and LSI are 0.0943 and 0.0874, respectively. This finding shows that PLSA and LSI suffer from the over-fitting problem due to the sparsity of the training set, which directly reduces the efficiency of recommendations. The precision of SARC-IA is 0.1432, and this performance is 1.5 times higher than that of PLSA, which indicates that our SARC-IA method can effectively alleviate the over-fitting problem by setting suitable hyperparameters. These results indicate that SARC-IA can solve the cold-start problem of data sparsity.

E. PERFORMANCE COMPARISON OF THE GENERATION OF SERVICE-LINKING MODES

Since the service-linking modes are relatively fixed, we aim to rank the linking modes of dataset via suitable support values and confidence values. Therefore, for the service-linking mode recommendation, we focus on the generation efficiency of linking modes, which include time consumption of algorithm, the number of negative linking modes, suitable support values and confidence values. The SARC-IA method utilizes the SARC algorithm to generate the service-linking modes in each interest category. To illustrate the necessity of classification calculations, we conduct experiments to compare the generation efficiency of SARC-IA and SARC.

TABLE 3. Performance comparison.

Evaluation Metrics	Methods	Matrix Density=30%	Matrix Density=50%	Matrix Density=70%	Matrix Density=90%
precision	**SARC-IA**	**0.1432**	**0.1812**	**0.2571**	**0.3407**
	PLSA	0.0943	0.1434	0.2325	0.3150
	LSI	0.0874	0.1386	0.2211	0.2714
	HITS	0.0429	0.0921	0.1467	0.1962
Recall	**SARC-IA**	**0.1815**	**0.2137**	**0.2845**	**0.3936**
	PLSA	0.1132	0.1824	0.2621	0.3108
	LSI	0.0986	0.1677	0.2418	0.2753
	HITS	0.0511	0.1023	0.1523	0.1846
F1-measure	**SARC-IA**	**0.1599**	**0.1961**	**0.2655**	**0.3552**
	PLSA	0.1028	0.1666	0.2464	0.3177
	LSI	0.0926	0.1517	0.2309	0.2755
	HITS	0.0466	0.0969	0.1494	0.1817

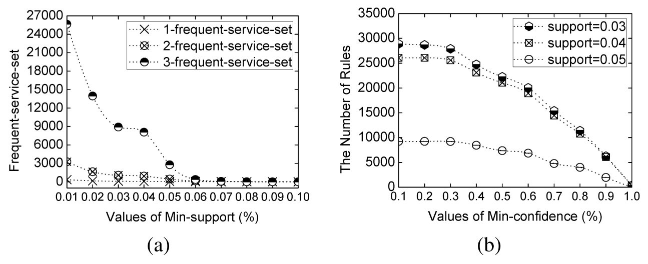


FIGURE 9. Generation of the Service-linking modes (global computation).

The user-service dataset has a large number of inactive users, and their number of invocation services is no more than 3. These users are regarded as noise in the dataset, which greatly affects the support and confidence of the algorithm. Therefore, the original dataset is turned into an active-user dataset by eliminating the inactive user. The active-user dataset is implemented in the experiment by utilizing the SARC algorithm, and the experimental results are as follows.

Fig. 9a shows that the maximum support value of the 2-frequent service sets and 3-frequent service sets is not more than 10% in the service invocation records of active users, and the frequent service sets are insufficient to generate the rules that meet the requirement of users when the support value is from 7% to 10%. When the support value rises from 4% to 5%, the generated frequent service sets have a large fluctuation, with the number of the 2-frequent service sets reduced from 927 to 445, and the number of the 3-frequent service sets reduced from 8075 to 2781. When the support value rises from 5% to 6%, the number of the 2-frequent service sets is reduced from 445 to 148, and the number of the 3-frequent service-set is reduced from 2781 to 392. We focus on the frequent service sets whose support value is from 3% to 5%; within this range, the confidence level varies from to 10% to 100%, and the generation of linking modes is shown in Fig. 9b. The number of rules generated is 22257 when the confidence is 50% and support is 3%; and the number of linking modes is 21051 when the confidence is 50% and support is 4%.

From the above data illustration, we can conclude that the SARC model presents three shortcomings: 1) with the change in support, large fluctuations are observed in the generation of frequent sets, which increases the difficulty of finding a suitable support value; 2) The gap between the support value and confidence value is too large to allow the lift to work; 3) Although we focus on only the generation of the 2-frequent service sets and 3-frequent service sets, the performance of the SARC algorithm still suffered from the unsatisfied time consumption (the time consumption of SARC is as shown in Fig. 10l).

The SARC-IA method implements the association rules algorithm in the interest subsets, and it can overcome

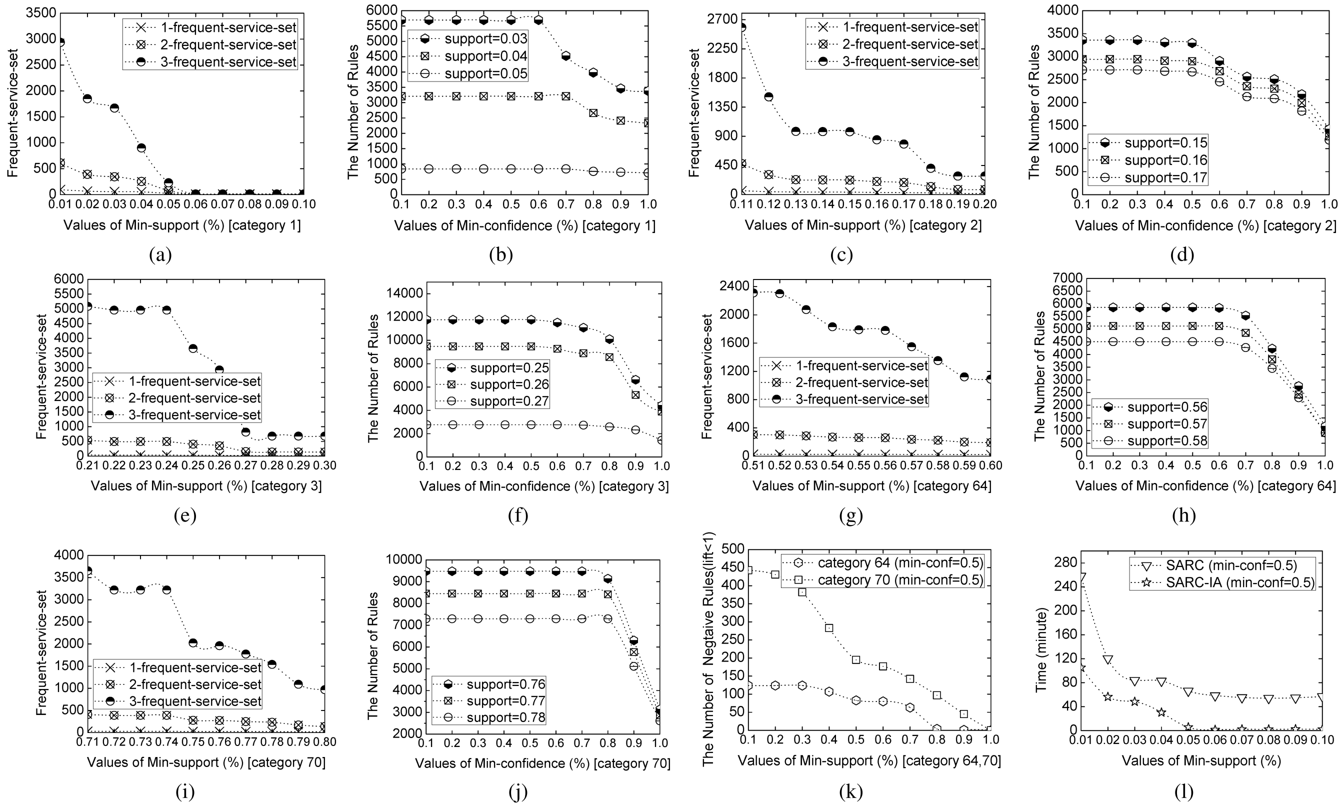


FIGURE 10. Performance comparison of SARC-IA and SARC (Fig. 10a to Fig. 10j indicate the generation of service linking modes (classification calculation), Fig. 10k indicates the impact of the lift measurement for linking modes, Fig. 10l indicates the comparison result of execution time for SARC-IA and SARC).

the shortcomings of SARC via classification computations. We select the experimental results for five interest subsets (i.e., category 1, category 2, category 3, category 64 and category 70) as samples to describe the necessity of classification computations, and the experimental results are illustrated in Fig. 10.

As shown in Fig. 10a to Fig. 10j, the support of each subset varies: the support of category-1 is from 1% to 10%, which is similar to the support of the whole dataset; the support of category-2 is from 10% to 20%; the support of category-3 is from 20% to 30%, and the support of category-64 and category-70 are more than 50%. The gap of the support interval shows that the interest of the user-service invocation is uneven and indicates that if the classification calculation is not performed for the service set, then the accuracy of the data analysis will be weakened. The confidence of the subset is greater than 50%, which proves that the linking modes among services are relatively fixed. Due to the high support of certain categories (e.g., category-64 and category-70), lift measurement can be used to discover the negative service-linking rules as shown in Fig. 10k. When the confidence of category-64 and category-70 is 0.5, the number of negative service-linking rules for these two categories is 83 and 195, respectively. Fig. 10l shows the execution time of SARC and SARC-IA for the support intervals from 0.01 to 0.1. The time consumption of the two algorithms is gradually reduced as the support increases, and the execution time of SARC is far greater than that of SARC-IA. When the support of the two methods is 0.01, the execution time of SARC and SARC-IA is 258.26 and 105.21, respectively. Thus, the time consumption of SARC is 2 times greater than that of SARC-IA, and when the support of the two methods is 0.05, the time consumption of SARC is 13 times greater than that of SARC-IA. Considering the sparse users of the experimental dataset (i.e., only 11730 active users in the experimental dataset) and the particulars of UGS (i.e., considering only the 2-frequent service set and 3-frequent service set), the SARC-IA method will present a performance advantage for large datasets.

V. RELATED WORK AND DISCUSSION

In recent years, UGS has attracted intensive attention from both industry and academia. The related work on UGS (or service mashup) can be generally classified into two categories: 1) UGS development environment; 2) service recommendation and service-linking modes creation for UGS.

A. UGS INTEGRATION FRAMEWORK

UGS development environment mainly focuses on encapsulating the multisource heterogeneous services (e.g., web service, Restful API, XML, RSS, etc.) into reusable service components for users to easily mash up these services. Yahoo!Pipes [4] is a web-based UGS tool provided by Yahoo. A pipe of Yahoo!Pipes is composed of one or more modules, which can sort and filter data services. End users can utilize the modules to combine various websites or blogs into an aggregated result (i.e., a pipe) in a single page. IFTTT [5] is an acronym for ‘‘If This, Then That’’. It can automatically carry out an action when a specific action occurs. Various services (e.g., Facebook, Twitter, YouTube, Flickr, E-mail, etc.) are packaged as channels in IFTTT, and different channels can be combined into recipes to satisfy users’ individual requirements. Google Mashup Editor (GME) [45] is a set of UGS development tools for mashup applications. End users can use extended XML tags provided by Google and some standardized AJAX services to create a novel application. OMELETTE [7] is a set of UGS development tools, which include three aspects of My-Cocktail, RAVE and Wookie. My-Cocktail makes it easy for end users to compose services into widgets, which can be stored in the Wookie; RAVE provides a live environment, in which end users can compose these widgets to generate service processes. EZWEB [6] is an enterprise compositional platform, which can bridge users and services as an integral composition solution. In the platform of EZWEB, the underlying resources are used as core building blocks to compose personal applications on top of existing services. Liang *et al.* [47] advance an end users-oriented programming environment called Mashroom. It takes the nested table as the data structure and formally defines a set of visual mashup operators to support a spreadsheet programming. Cheng *et al.* [8] propose a light service composition platform named LSMP, which defines four frameworks to encapsulate atom services into graphical components. End users with little programming skills can create new personalized service mashups by combining these graphical service components on a web browser.

End users can utilize the UGS development tools to compose service mashups via simple operations (e.g., drag and drop operation). However, the efficiency of these tools is not high in the absence of the suggestion mechanism. To improve the composition quality and the success rate of service mashups, the service component recommendation and the linking mode recommendation are necessary functionality for these tools.

B. SERVICES RECOMMENDATION AND SERVICE LINKING MODES CREATION FOR THE UGS

The existing studies in this domain can be divided into: 1) QoS-based service recommendation, e.g., [11], [13], and [48]; 2) SNA-based service recommendation, e.g., [18], [38], [40], [49], and [50]; 3) SARC-based linking rules creation, e.g., [21], [24], [33], [34], [47], and [51].

QoS-based approaches predict the QoS values of services or recommend services with high QoS values for the service composition. Wang *et al.* [48] propose a cloud model-based skyline approach, which aim to recommend optimistic QoS candidate services to complete service compositions. Ma *et al.* [13] in literature point out QoS data is part of object data by investigating the difference between QoS data and MovieLens rating data, the traditional similarity methods for subjective data are not suitable for QoS prediction. A highly accurate QoS prediction (HAQP) method is presented to predict unknown service QoS values by integrating the wave theory and the regression algorithm. Zheng *et al.* [11] advance a neighbor integrated matrix factorization (NIMF) approach, which takes advantage of the past web service usage experience of service users to make service QoS values prediction for current users.

SNA-based approaches recommend services based on network analyze and graph theory. Chen *et al.* [38] propose a trace-based Voronoi graph scheme to recommend suitable services to users. The aim of the approach is to search the services that associated with the interests of the active users. To improve the efficiency of searching, the service nodes are constructed into a Voronoi graph, where K nearest neighbors (KNN) algorithm is utilized to calculate the similarity between the candidate service nodes and the current invocation service. Cao *et al.* [40] present a service mashups recommendation scheme by combining the interests of users and social network. Firstly, interests of users are extracted via TF/IDF in service ecosystem, then a social network is constructed via service process records of users, service component and service tags, finally services required by end users are recommended by integrating users' interests and social relation network. Maaradji et al. [49] present a social composer (SoCo) scheme, which construct a social network by transforming user-service relation into user-user relation. Huang *et al.* [18] propose three kinds of recommendations that include potential composition recommendation, top service recommendation and service chain by analyzing service network. Zhou *et al.* [51] exploit the HITS algorithm to calculate two weights (hub and authority) of the service nodes in service network, the resource importance (RI) of service nodes can be calculated via hub and authority, when a user invoke a given service, some service nodes with high RI are recommended to the users.

SARC-based approaches mainly focus on correlations or linking modes among services. Ni *et al.* [21] propose a negative-connection-aware service recommendation (NCSR) scheme to recommend tag-based service rules in the sparse service network. Service association rules are mined to supply the service composition trend by integrating positive composition patterns and negative composition patterns. Liang *et al.* [47] define three different levels of service usage data, which are user request level, template level and instance level. The service invocation patterns are mined by adopting the association rule algorithm on each level. Shafiqi *et al.* [33] present a semantic-based FP-tree algorithm to rank services and service-linking modes. The extended association rule method is used to reduce the search space during the process of service creation. Dam [34] propose a method that utilize association rule algorithm to mine composition modes of services. The composition modes is used to predict changes of the service ecosystem. Rong [24] propose a service matching scheme to recommend services by integrating Pearson Correlation Coefficient and associate rule algorithm. In the scheme, the user subsets are formed by adopting Pearson Correlation Coefficient to calculate the similarity of users; associate rule algorithm is used to generate linking modes in each user subset. when a user queries a certain service, the services that can match this service are recommended to users. Tapia [52] advance a global co-utilization API ranking (GCAR) approach to discover candidate APIs for service mashup generation by utilizing Apriori algorithm. The cold start problem for APIs invocation can be alleviated by utilizing MashupRECO (i.e., a mashup recommendation tool) and GCAR.

In conclusion, the existing service recommendation solutions mainly focus on QoS measurement and dualistic relationship analysis in the user-service and the service-service network. Therefore, there are some differences between these studies and our work. 1) Target users of QoS-based methods are the enterprise developers, who consider that a service component with a certain function has many candidates with different QoS values; therefore, the developers try to create the service mashups by composing service components with optimal QoS (i.e., the enterprise developers pay more attention to Non-functional attributes). Target users of our work are non-experienced end users, and the end users suffer from interests extraction and service linking modes in the process of creating service mashups (i.e., the end users pay more attention to functional attributes). 2) SNA-based methods mainly focus on the dualistic relationship between services and users. Graph-theory and traditional CF algorithms are adopted to recommend suitable services for users. These methods neglect the important of semantic level in service recommendation. Our SARC-IA scheme extends the dualistic relationship (i.e., user-service) to the ternary relationship (i.e., user-interest-service), which can explain the recommendation reasons from the semantic level. 3) SARC-based methods generally adopt Apriori or FP-Growth to generate positive linking rules in the whole dataset. Different from the above scheme, our SARC-IA method utilizes the SARC algorithm to generate positive rules in the interest subsets of the dataset. The experiments prove that classification calculation is necessary due to the different supports of the subsets. Although the algorithms [24] exploits the CF method to generate subgroups, this method cannot determine the number of subsets, which results in the failure to utilize lift metric to eliminate the negative linking rules.

VI. CONCLUSION AND FUTURE WORK

The purpose of the UGS is to create service mashups that satisfy end users' requirements by composing existing service components. Without sufficient domain knowledge and affirmatory interest, the service mashups created by end users are always incomplete and low quality. To improve the efficiency and success rate of UGS, providing the service recommendation and generating service-linking modes are critical steps toward creating service mashups with appropriate service components for end users.

This paper proposes the SARC-IA scheme, which assists the end users in the creation of high-quality personalized service mashups. Two recommendation lists are generated by the SARC-IA method, with one service recommendation list focused on similar services, and the other association rule list focused on the service-linking mode. Experiments and analyses showed that our solution outperforms the reference algorithms in terms of the recommendation efficiency of similar services and generation efficiency of service linking modes. In future work, we will study the problem of mashup recommendation by integrating the word2vec model and long short-term memory (LSTM) in the UGS.

REFERENCES

[1] T. Erl, *SOA: Principles of Service Design*. Upper Saddle River, NJ, USA: Prentice-Hall, 2007.
[2] Z. Zhao, N. Laga, and N. Crespi, "A survey of user generated service," in *Proc. IEEE Int. Conf. Netw. Infrastruct. Digit. Content*, Nov. 2009, pp. 241–246.
[3] J. Yu, B. Benatallah, F. Casati, and F. Daniel, "Understanding mashup development," *IEEE Internet Comput.*, vol. 12, no. 5, pp. 44–52, Sep. 2008.
[4] K. T. Stolee, S. Elbaum, and A. Sarma, "Discovering how end-user programmers and their communities use public repositories: A study on Yahoo! Pipes," *Inf. Softw. Technol.*, vol. 55, no. 7, pp. 1289–1303, 2013.
[5] S. Ovadia, "Automate the Internet with 'if this then that' (IFTTT)," *Behav. Social Sci. Librarian*, vol. 33, no. 4, pp. 208–211, 2014.
[6] D. Lizcano, J. Soriano, M. Reyes, and J. J. Hierro, "EzWeb/FAST: Reporting on a successful mashup-based solution for developing and deploying composite applications in the upcoming 'ubiquitous SOA,'" in *Proc. 2nd Int. Conf. Mobile Ubiquitous Comput., Syst., Services Technol.*, Sep./Oct. 2008, pp. 488–495.
[7] O. Chudnovskyy *et al.*, "End-user-oriented telco mashups: The OMELETTE approach," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 235–238.
[8] B. Cheng, Z. Zhai, S. Zhao, and J. Chen, "LSMP: A lightweight service mashup platform for ordinary users," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 116–123, Apr. 2017.
[9] G. Wang, Y. Han, Z. Zhang, and S. Zhang, "A dataflow-pattern-based recommendation framework for data service mashup," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 889–902, Dec. 2015.
[10] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware Web service recommendation by collaborative filtering," *IEEE Trans. Services Comput.*, vol. 4, no. 2, pp. 140–152, Apr./Jun. 2011.
[11] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative Web service QoS prediction via neighborhood integrated matrix factorization," *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 289–299, Jul. 2012.
[12] S. Wang, X. Zhu, and F. Yang "Efficient QoS management for QoS-aware web service composition," *Int. J. Web Grid Services*, vol. 10, no. 1, pp. 1–23, 2014.
[13] Y. Ma, S. Wang, P. C. K. Hung, C. Hsu, Q. Sun, and F. Yang, "A highly accurate prediction algorithm for unknown Web service QoS values," *IEEE Trans. Services Comput.*, vol. 9, no. 4, pp. 511–523, Apr. 2017.
[14] S. Wang, Y. Ma, B. Cheng, F. Yang, and R. Chang, "Multi-dimensional QoS prediction for service recommendations," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2016.2584058.
[15] X. Luo, Z. Xu, J. Yu, and X. Chen, "Building association link network for semantic link on Web resources," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 3, pp. 482–494, Jul. 2011.
[16] S. Yu and C. J. Woodard, "Innovation in the programmable Web: Characterizing the mashup ecosystem," in *Proc. Int. Conf. Service-Oriented Comput.*, 2008, pp. 136–147.
[17] M. Weiss and G. R. Gangadharan, "Modeling the mashup ecosystem: Structure and growth," *R D Manage.*, vol. 40, no. 1, pp. 40–49, 2010.
[18] K. Huang, Y. Fan, and W. Tan, "Recommendation in an evolving service ecosystem based on network prediction," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 906–920, Jul. 2014.
[19] K. Huang, Y. Fan, W. Tan, and M. Qian, "BSNet: A network-based framework for service-oriented business ecosystem management," *Concurrency Comput. Pract. Exper.*, vol. 25, no. 13, pp. 1861–1878, 2013.
[20] L. Yi, Y. Fan, and K. Huang, "Web service ecosystem model and service recommendation strategy from overall perspective," *Comput. Integr. Manuf. Syst.*, vol. 22, pp. 133–143, 2016, doi: 10.13196/j.cims.2016.01.013.
[21] Y. Ni, Y. Fan, W. Tan, K. Huang, and J. Bi, "NCSR: Negative-connection-aware service recommendation for large sparse service network," *IEEE Trans. Automat. Sci. Eng.*, vol. 13, no. 2, pp. 579–590, Apr. 2016.
[22] Y. Han, S. Chen, and Z. Feng, *Mining Integration Patterns of Programmable Ecosystem With Social Tags*. New York, NY, USA: Springer-Verlag, 2014.
[23] C.-H. Lee, S.-Y. Hwang, and I.-L. Yen, "A service pattern model for flexible service composition," in *Proc. IEEE 19th Int. Conf. Web Services*, Jun. 2012, pp. 626–627.
[24] W. Rong, K. Liu, and L. Liang, "Personalized Web service ranking via user group combining association rule," in *Proc. IEEE Int. Conf. Web Services*, Jul. 2009, pp. 445–452.
[25] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 641–650.
[26] L. Cagliero and P. Garza, "Infrequent weighted itemset mining using frequent pattern growth," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 4, pp. 903–915, Apr. 2014.
[27] X. Chen, Z. Zheng, Q. Yu, and M. R. Lyu, "Web service recommendation via exploiting location and QoS information," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1913–1924, Jul. 2014.
[28] M. Alrifai, D. Skoutas, and T. Risse, "Selecting skyline services for QoS-based Web service composition," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 11–20.
[29] S. Deng, H. Wu, J. Taheri, A. Y. Zomaya, and Z. Wu, "Cost performance driven service mashup: A developer perspective," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 8, pp. 2234–2247, Aug. 2016.
[30] M. You, X. Xin, W. Shangguang, L. Jinglin, S. Qibo, and Y. Fangchun, "QoS evaluation for Web service recommendation," *China Commun.*, vol. 12, no. 4, pp. 151–160, Apr. 2015.
[31] T. Gao, B. Cheng, J. Chen, and M. Chen, "Enhancing collaborative filtering via topic model integrated uniform Euclidean distance," *China Commun.*, vol. 14, no. 11, pp. 48–58, Nov. 2017.
[32] W. Y. Zhang, S. Zhang, and S. S. Guo "A PageRank-based reputation model for personalised manufacturing service recommendation," *Enterprise Inf. Syst.*, vol. 11, no. 5, pp. 672–693, 2015.
[33] O. Shafiq, R. Alhajj, and J. G. Rokne, "Reducing search space for Web Service ranking using semantic logs and semantic FP-tree based association rule mining," in *Proc. IEEE 9th Int. Conf. Semantic Comput.*, Feb. 2015, pp. 1–8.
[34] H. K. Dam "Predicting change impact in Web service ecosystems," *Int. J. Web Inf. Syst.*, vol. 10, no. 3, pp. 275–290, 2014.
[35] Q. Liu, E. Chen, H. Xiong, C. H. Q. Ding, and J. Chen, "Enhancing collaborative filtering by user interest expansion via personalized ranking," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 218–233, Feb. 2012.
[36] X. Wei and W. B. Croft, "LDA-based document models for ad-hoc retrieval," in *Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2006, pp. 178–185.
[37] T. Hofmann "Probabilistic latent semantic analysis," *Proc. Uncertainty Artif. Intell.*, vol. 41, no. 6, pp. 289–296, 1999.
[38] H. Chen *et al.*, "Mashup by surfing a Web of data APIs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, Indianapolis, IN, USA, Jun. 2013, pp. 423–434.
[39] C. Pautasso, "RESTful Web service composition with BPEL for REST," *Data Knowl. Eng.*, vol. 68, no. 9, pp. 851–866, 2009.
[40] B. Cao, J. Liu, M. Tang, Z. Zheng, and G. Wang, "Mashup service recommendation based on user interest and social network," in *Proc. IEEE 20th Int. Conf. Web Services*, Jun./Jul. 2013, pp. 99–106.
[41] H. Deng *et al.*, "Service search and recommendation algorithm based on user's interest and service satisfaction," *Appl. Res. Comput.*, vol. 9, pp. 2613–2617, 2015.
[42] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res. Arch.*, vol. 3, pp. 993–1022, Jan. 2003.
[43] I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling, "Fast collapsed gibbs sampling for latent dirichlet allocation," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Las Vegas, NV, USA, 2008, pp. 569–577.
[44] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc. Ser. B (Methodol.)*, vol. 39, no. 1, pp. 1–38, 1977.
[45] G. Di Lorenzo, H. Hacid, H.-Y. Paik, and B. Benatallah, "Data integration in mashups," *ACM SIGMOD Rec.*, vol. 38, no. 1, pp. 59–66, 2009.

[46] G. Wang, S. Yang, and Y. Han, "Mashroom: End-user mashup programming using nested tables," in *Proc. 18th Int. Conf. World Wide Web*, 2009, pp. 861–870.

[47] Q. A. Liang, J. Chung, S. Miller, and Y. Ouyang,, "Service pattern discovery of Web service mining in Web service registry-repository," in *Proc. IEEE Int. Conf. E-Bus. Eng. (ICEBE)*, Oct. 20006, pp. 286–293.

[48] S. Wang, Q. Sun, H. Zou, and F. Yang, "Particle swarm optimization with skyline operator for fast cloud-based Web service composition," *Mobile Netw. Appl.*, vol. 18, no. 1, pp. 116–121, 2013.

[49] A. Maaradji *et al.*, "Social composer: A social-aware mashup creation environment," in *Proc. ACM CSCW*, 2010, pp. 549–550.

[50] S. Bansal, C. Gupta, and A. Arora, "User tweets based genre prediction and movie recommendation using LSI and SVD," in *Proc. IEEE 9th Int. Conf. Contemp. Comput. (IC3)*, Aug. 2016, pp. 1–6.

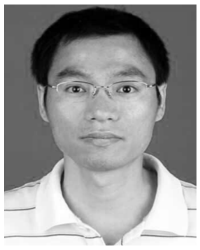
[51] C. Zhou, H. Chen, Z. Peng, Y. Ni, and G. Xie, "Ontology-driven mashup auto-completion on a data API network," *Tsinghua Sci. Technol.*, vol. 15, no. 6, pp. 657–667, 2010.

[52] B. Tapia, R. Torres, H. Astudillo, and P. Ortega, "Recommending APIs for mashup completion using association rules mined from real usage data," in *Proc. 30th Int. Conf. Chilean Comput. Sci. Soc.*, Nov. 2011, pp. 83–89.

TIELIANG GAO is currently pursuing the Ph.D. degree with the Beijing University of Posts and Telecommunications. His research interests are service computing and recommender systems.

BO CHENG received the Ph.D. degree in computer science from the University of Electronics Science and Technology of China in 2006. He is currently a Professor with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His research interests include Internet of Things, mobile Internet, and services computing.

JUNLIANG CHEN is currently a Professor with the Beijing University of Posts and Telecommunications. His research interests are in the areas of service creation technology. He was elected as a member of the Chinese Academy of Science in 1991 and the Chinese Academy of Engineering in 1994.



HUAJIAN XUE received the Ph.D. degree from the Chinese Academy of Sciences in 2012. He is currently an Associate Professor with The Xinjiang Technical Institute of Physics and Chemistry, Chinese Academy of Sciences. His research interests are in the Internet of Things and nature language processing.

LI DUAN received the Ph.D. degree from the Beijing University of Posts and Telecommunications in 2016. She is currently a Research Fellow with Nanyang Technological University and the University of Science and Technology Beijing. Her research interests are in services computing and Internet of Things, and network service security and privacy.

SHOULU HOU received the master's degree from the Shenyang University of Technology in 2014. She is currently pursuing the Ph.D. degree in computer science and technology with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. She is also pursuing the Joint-Training Ph.D. degree with Data61, CSIRO, Australia. Her research interests include service computing and performance evaluation.