Intelligent Automation And Soft Computing, 2019
Copyright © 2019, TSI® Press
Vol. 25, no. 3, 513–525
https://doi.org/10.31209/2019.100000108

A Novel Service Recommendation Approach in Mashup Creation

Yanmei Zhang[1], Xiao Geng[2], and Shuiguang Deng[3]

[1]Information School, Central University of Finance and Economics Beijing 100081, China
[2]School of Computer Science, Wuhan University, Wuhan 430072 China
[3]College of Computer Science and Technology Zhejiang University, Hangzhou 310027, China

ABSTRACT
With the development of service computing technologies, the online services are massive and disordered now. How to find appropriate services quickly and build a more powerful composed service according to user interests has been a research focus in recent years. Current service recommendation algorithms often directly follow the traditional recommendation framework of ecommerce, which cannot effectively assist users to complete dynamic online business construction. Therefore, a novel service recommendation approach named UISCS (User-Interest- initial Services-Correlation-successor Services) is proposed, which is designed for interactive scenario of service composition, and it mines the user implicit interests and the service correlations for service recommendation. A series of experiments are conducted on a real-world dataset crawled from the ProgrammableWeb, and the results show that as a step-by-step service recommendation approach, the UISCS approach has obviously improved the performance of some mainstream recommendation algorithms, such as LDA, ICF , SVD and graph-based TSR.

KEY WORDS: service recommendation, collaborative filtering, mass diffusion algorithm, semi-automatic service composition

1 INTRODUCTION

WITH the development of service computing technologies, a variety of services such as web APIs and mashups have appeared on the Internet. The service composition technologies make it possible to build a new and more powerful composed service (i.e. mashup) based on existing atomic services (i.e. APIs). That is an important technique for service reuse and Internet resources collaboration. Under the real service composition environment, manual service composition is laborious, and automatic service composition cannot meet comprehensive user demands completely. The semi-automatic service composition not only can help users create business processes with existing knowledges and experiences, but also can take user dynamically changing requirements into consideration. Therefore, the semi-automatic service composition is more practical than manual or automatic options. Since user knowledge about the immense and disordered Internet services is very limited, service recommendation plays a very important role during the process of semi-automatic service composition. The performance of service composition depends on the performance of service recommendation. However, there are still some problems if current service recommendation approaches are used in semi-automatic service composition directly:

(1) The limitation of existing recommendation approach. Users may configure and implement services simultaneously under the semi-automatic service composition environment. So both the effectiveness and the efficiency of service recommendation have an influence on the quality of the whole composed service. However, most service recommendation approaches follow the three-layer framework 'user-interest-item' in the electronic commerce or information retrieval area. So they are usually suitable for recommending initial services, but not suitable for successor services. Actually, there are just weak relations among items in the e-commerce area. By comparison, there exist strong relations (such as interface matching) between the successor and preceding services in the service computing area, and the strong relations among services are usually expressed as co-occurrence relations in

CONTACT Yanmei Zhang jlzym0309@sina.com

the real-world application. However, the strong relations among services have not been used completely as far as we know.

(2) The insufficiency of existing metrics. The mashup data from the real service composition platform named ProgrammableWeb is sparse and unbalanced. Sparsity means most of the APIs are never or seldom invoked. Unbalance means a few services are invoked frequently, while most of the other services are rarely invoked. These barely-known services jointly occupy a segment market as compared to the most popular services. This phenomenon is called the long tail. Therefore, the long tail phenomenon of services should be considered. Diversity and novelty are suitable metrics to measure how much an algorithm could solve the long tail problem, but they are neglected in most existing works.

To meet the above challenges and aid users in completing a semi-automatic service composition more effectively and efficiently, a novel service recommendation approach named UISCS (User-Interest-initial Services-Correlation- successor Services) is proposed. UISCS has the following features:

Firstly, as the semi-automatic service composition is a dynamic process of business construction, that the service recommendation can be divided into two phases. Initial services based on user interest are first recommended, and then successor services based on both interests and correlations of services are recommended. The successor services recommendation process is iterative until the user completes the business construction.

Secondly, considering the long tail problems of both the users and the services, the performance of service recommendation algorithms will be evaluated in multi-dimension metrics, such as accuracy, diversity, novelty and timeliness.

Thirdly, the applicability of UISCS approach is good, and it can improve the performance of mainstream recommendation algorithms, such as LDA, ICF, SVD and graph-based TSR.

The remainder of this paper is organized as follows. Section 2 introduces and analyzes the related work. Section 3 proposes the UISCS approach and designs the corresponding service recommendation algorithms based on UISCS. Section 4 presents the experimental simulation evaluation and analysis based on a real dataset crawled from ProgrammableWeb. Section 5 presents the paper's conclusions and explores possible future work.

2 RELATED WORK

THE two key problems of UISCS are user modeling and service correlation mining. We will elaborate the researches related to them in this section.

1.1 User modelling

User modeling is the core technology of service recommendation and has a very important effect on it.

Most of current researches concerned trust relationships between users and services. For example, Abdurrahman et al. (2011) and Deng et al. (2014) made recommendations based on the trust relationship between users and services, which are mined from users' evaluations of services. They also established user networks and service networks by using co-occurrence relations (two users who evaluated the same service, or two services evaluated by the same user).Tang et al. (2014) put the trust relationship into two categories, including a global trust relationship between users and a local trust relationship between friends. Ye et al. (2016) presented a novel trust-aware worker recommendation method called CrowdRec, which used a novel trust sub-network extraction approach to tackling dishonest behaviors and data sparsity problems. Yin et al. (2016) proposed a hybrid recommendation algorithm combining the similarity in the collaborative filtering and trust in sociology. Specially, trust path model and loop trust model were merged with the linear weighting to get user trust in their paper. Zheng et al. (2017) proposed a novel hybrid recommender system called Hybrid Matrix Factorization (HMF) model which used hypergraph topology to describe and analyze the interior relation of social network in the system and fused important contextual information to improve its accuracy.

In recent years, the LDA (Latent Dirichlet Allocation) probabilistic topic model has been introduced to the user modeling field. The main idea was to extract implicit interests of users based on the LDA model. Each user was represented as a probability distribution of interests, and each interest was represented as a probability distribution of items. Researches on user modeling based on LDA, like the iExpand proposed by Liu et al. (2012), AToT proposed by Xu et al. (2014), and RIM frameworks proposed by Liu et al. (2013), were used mainly in the areas of e-commerce and information retrieval. According to our literature search, the use of the LDA model for service recommendation has just been proposed in recent years. For example, Li et al (2013) proposed a method of service discovery based on probability. Firstly, they used the LDA model to extract potential topics between services and user queries and then used those potential topics for service matching. Zhong et al. (2014) used the LDA model to extract evolutionary patterns of services, and proposed a time-aware service recommendation by combining the evolution of services with collaborative filtering and a matching method for content. Huang et al. (2014) proposed a prediction model of network evolution based on the LDA model, and applied it to service recommendation and service chain recommendation. Bai et al. (2015) combined the collaborative filtering algorithms with the LDA model of/for service recommendation, and relieved the cold start problems of mashups in this way. Liu and Fulia (2015) proposed a collaborative topic regression model, which combined the methods of probability matrix decomposition with a probabilistic topic model, thereby generating user-related, service-related, topic-related, and other potential factor models to predict user interest. Hao et al. (2017) characterized the latent topic features of reconstructed service descriptions and user queries by LDA, and then calculate the similarity between them and generate the ranked list of services based on the similarity scores. Zhang et al. (2018) adopt the LDA model to extract the user implicit demand factors, and then utilize the bipartite graph modeling and random walk algorithm to extend implicit demand factors to predict short-term changes and diversity of user demand.

In a word, there are numerous of studies about user model for service recommendation, however, those studies are still based on the e-commerce recommendation framework of 'user-interest-item'. The recommendations for successor services are neglected.

2.1 Service correlation mining

Service correlation mining has been used as the basis of recommendations for successor services. Co-occurrence and predecessor-successor relation are commonly used in related work.

Many studies made service recommendation based on the co-occurrence relation of services. Yao et al. (2015) thought that the service co-occurrence relation in a mashup was determined by the explicit text similarity and implicit service correlation, so they proposed a probability matrix decomposition method to reveal the potential relation between services through analyzing the co-occurrence pattern of services. Gao et al. (2016) proposed a SeCo-LDA approach to mine latent topic models over service co-occurrence patterns, and the key idea was to treat each service as a document, and to treat its bag of co-occurring services as a bag of words in that document.

Co-occurrence frequency represents the combined strength of service composition, but this co-occurrence relation cannot express true predecessor-successor relations (such as interface matching), and therefore it cannot accurately express a potential composite relation among services. Taking a different approach from those above, Wang et al. (2014) took predecessor-successor relations into account and then recommended a successor service. The method first used input-output interface matching information to build a service network model, and then used the context tags of user selections to recommend a service based on that model. Wang et al. (2014) used the example of geospatial information systems to study the problems of recommendation for service chains. A service network model was built by using service description information and the semantic information of input-output interfaces. Users simply needed to give the start and end information, and the system could search for chains matched in the service network model, and then rank and recommend them. Ma et al. (2017) proposed a novel approach, ServRel, to recommends three kinds of relevant services: competitive, pre-cooperative, and post-cooperative services for the target services. And the two cooperative relations were calculated based on the semantic distance between the input and output parameters of the target service and the candidates by the Hungarian Algorithm.

Some works considered the dynamic features of service correlation, and proposed evolving service network. Zhong et al. (2014) considered services and their mashup (service composition) evolved over time (such as publishing, demise, and update interface). And they thought that the quality of service recommendation could be improved by a service ecosystem network. Huang et al. (2014) introduced a network series model representing an evolved service system, and they recommended possible top-k services and service chains according to a network evolution forecast. Naim et al. (2016) integrated web service discovery and composition. They took the diversity of discovered results into account in a unified way, chosen a set of selected web services based on relevancy, service diversity and service density, and finally proposed a new method to generate a service dependency network using the FCA (Formal Concept Analysis) framework.

In addition, there are some papers mining other relations between services for service recommendation. Discovering that mining the potential similarity relations among services in Cyber-physical systems (CPS) is really helpful to improve the prediction accuracy, Yin et.al. (2017) proposed a novel service recommendation method for CPS, which utilized network location as context information and made full use of the similarity relations.

All the above studies did service correlation mining and organized services based on the service correlations, which enhanced the effectiveness of service recommendation to some degree. However, most of the related work evaluated the accuracy of the algorithm, but miss other useful metrics, such as diversity and novelty.

3 UISCS RECOMMENDATION APPROACH

A novel service recommendation approach named UISCS (User-Interest-initial Services-Correlation-successor Services) is proposed, and the algorithm framework is shown as Figure 1. The service recommendation mechanisms are designed as follows: the system recommends top-k initial services (or service chains) to a user based on user interests, as soon as the user select an initial service, the system will recommend top-k successor services (or service chains) according to both the user interest to services and the service correlations. The successor service recommendation process is iterative until the user completes the business construction. To realize the UISCS algorithm, two key problems must be solved. Firstly, how to mine user interests for services? Secondly, how to mine the correlations among services? For the first question, the LDA probabilistic topic model is used to model user interest. For the second question, mass diffusion algorithm is used to mine the correlations among services.

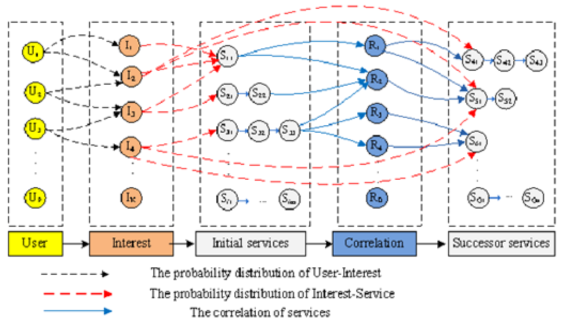

User
Interest
Initial services
Correlation
Successor services
The probability distribution of User-Interest
The probability distribution of Interest-Service
The correlation of services

Figure 1. The framework of UISCS algorithm

To describe the problem clearly, the relevant notations used in the UISCS algorithm are defined in Table 1.

Table 1. Notations and definitions

Notations	Definitions
U={U$_1$, U$_1$...U$_P$}	User set
S={S$_1$, S$_2$...S$_Q$}	Service set
I ={I$_1$, I$_2$...I$_K$}	Interest set
R ={R$_1$,R$_2$...R$_D$}	Service correlation set
P	Number of users
Q	Number of services
K	Number of interests
D	Number of service correlations
θ	Probability distribution matrix of ‘user-interest’
φ	Probability distribution matrix of interest service
α	Parameter alpha in LDA
β	Parameter beta in LDA
W(*i, j*)	Correlation degree between service Si and service Sj

3.1 User Interest Mining Based on the LDA Model

The LDA model in the machine learning has been widely used to extract document topics, and it is adopted for mining user interest topics in a data set including users set P and services set Q. The probabilistic graphical model for mining of user interest topics can be represented as in Figure 2.

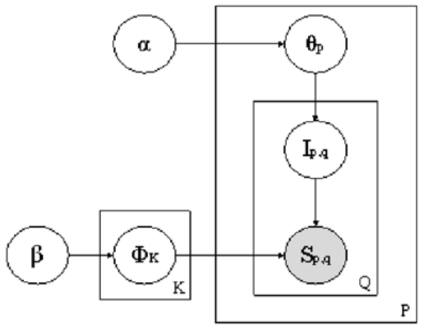


Figure 2. LDA Probabilistic Graphical Model For Mining Interest

The Dirichlet priori parameter α determines the multiple distribution θ_p of the user p on the interests, and the priori parameter β of Dirichlet determines the multiple distribution φ_k of the interest k to the services. When the user p invokes the q_{th} service, he firstly determines the interest $I_{p,q}$ that the service belonging to based on θ_p, and then selects the service $S_{p,q}$ according to the multiple distribution φ of the interest $I_{p,q}$. Every user generates all their services in this way, and then the generating process of the model is completed.

In the inference phase, we need to estimate all the hidden variables according to the observable variable $S_{p,q}$, which belongs to a NP problem by direct calculation. So we use Gibbs algorithm to solve this problem. The efficient algorithm eventually assigns a certain interest to each service in the user-service matrix by posterior probability computation and sampling techniques and then estimates the hidden variables based on the matrix by the following two formulas, where $n^{(k)}$ represents the absolute frequency of the services invoked by user p belonging to the interest k; and $n^{(q)}$ denotes the total times of service q invoked by all users assigned to the interest k:

$$\theta_{p,k} = P\left(I_k \middle| U_p\right) = \frac{n_p^{(k)}+\alpha}{\sum_{k=1}^{K} n_p^{(k)}+\alpha} \tag{1}$$

$$\varphi_{k,q} = P\left(S_q \middle| I_k\right) = \frac{n_k^{(q)}+\beta}{\sum_{n=1}^{N} n_k^{(q)}+\beta} \tag{2}$$

After above process, the distribution of user on interest and that of interest conditioned on the service are obtained. This means that the ‘user-service’ relation matrix is factorized into the ‘user-interest’ relation matrix θ and ‘interest-service’ relation matrix φ. Next, we calculate a score ranking of users on services based on above results and formula (3), and then recommend the services with the highest scores to the users.

$$P\left(S_q\middle|U_p\right) = \sum_{k=1}^{K} P(S_q\,|I_k)P\left(I_k\middle|U_p\right) = \sum_{k}^{K} \theta_{p,k}\emptyset_{k,q} \tag{3}$$

The 'user-service' scoring matrix we obtained is the main basis of the initial services recommendation, according to the score ranking. The recommendations of successor services are based on both the score matrix and the service correlation described in 3.2.

3.2 Service Correlation Mining Based on the Mass Diffusion Theory

The relation between users and services is represented as a bipartite graph and expressed as *US = {U, S, R}*, Where *U = {U₁, U₂ ...U_P}* denotes the user set, and P is the number of users; $S = \{S_1, S_2 \ldots S_Q\}$ denotes service set, and Q is the number of services; R is a collection of undirected edges are the invoking relation between the user U and the service S.

According to the theory of mass diffusion, the service itself owns resources, and resources are reallocated after the user invokes a service. If the user U_i had ever used the service s_j, then the value of $US_{(i,j)}$ should be set to 1, and otherwise to 0. $f(U_i)$ and $f(S_j)$ are resources carried by user U_i and service s_j respectively. As shown in Figure 3, the mining algorithms of service correlation based on the theory of mass diffusion mainly include two steps:

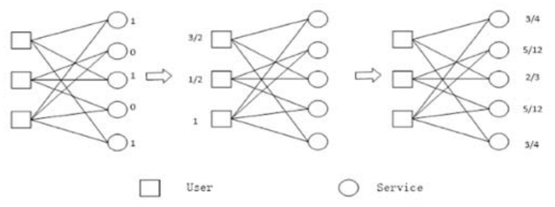


Figure 3. The Process Of Mass Diffusion Algorithm

(1) Diffusion of resources of services: Services allocate their initial resources averagely to the user nodes connected to them. So the mass resources are delivered to the users, and the resource of the user node l can be expressed as $f(U_l)$:

$$f(U_l) = \sum_{i=1}^{Q} \frac{US_{il}f(S_i)}{k(S_i)} \tag{4}$$

In formula (4), $k(S_i)$ is the number of users who had invoked service i.

(2) Rediffusion of resources of users: Users allocate their resources averagely to the service nodes connected to them. The resources are passed to the services, and then the resources of service node j can be expressed as $f(S_i)$:

$$f\left(S_j\right) = \sum_{l=1}^{P} \frac{US_{lj}f(U_l)}{k(U_l)} \tag{5}$$

The formula (6) is obtained according to formula (4) and (5):

$$f\left(S_j\right) = \sum_{l=1}^{P} \frac{US_{lj}}{k(U_l)} \sum_{i=1}^{Q} \frac{US_{li}f(S_i)}{k(S_i)} \tag{6}$$

$W_{i,j}$ in Formula (7) denotes services correlation between services S_i and S_j. Formula (8) obtained by further derivation of (6) and (7) calculates the value of the 'service-service' correlation matrix:

$$f\left(S_j\right) = \sum_{i=1}^{Q} W_{ij} f(S_i) \tag{7}$$

$$W_{ij} = \frac{1}{k(S_i)} \sum_{l=1}^{P} \frac{US_{li}US_{lj}}{k(U_l)} \tag{8}$$

3.3 Service Recommendation Algorithm based on UISCS Approach

User interest score is calculated in Section 3.1 and service correlation score is obtained in Section 3.2. Now it's time to introduce the service recommendation algorithm of UISCS approach. When making initial services recommendation, the user interest score is used. When recommending successor services, both the user interest score and the service correlation score are taken into consideration.

On one hand, the user interest score obtained by LDA is likely to increase the possibility of recommendation of the services that suits the user interest or the service which is in the tail of the long-tail. The interest score of user U_p on candidate service S_q, denoted by *IS(p,q),*can be calculated by formula(3). That is $IS(p,q)=P(S_q, U_p)$.

On the other hand, we user *CS(p,q)* to denote the correlation score of a candidate service S_q to the other services that the user U_p has chosen. Suppose that a user has selected K services step by step and constructed a service chain. Owing to the user identifies his or her demand gradually, the sooner a service is selected on the chain, the greater impact it has on the following candidate services. So the correlation score *CS(p,q)* of candidate service S_q can be calculated by formula (9) :

$$CS(q) = P\left(S_q\middle|U_p\right) = \sum_{i=1}^{K} \sigma_i W_{i,q} = \sum_{i=1}^{K} \frac{W_{i,q}}{i} \tag{9}$$

In the formula (9), i is the distance that from a chosen service in the chain to the candidate service. For example, the weight of the last selected service in the chain is the biggest with value 1, and the weight of the first one is the smallest, just 1/n, where n is the length of the chain. And $W_{(i,q)}$ is the correlation between service S_q and the i_{th} service in the chain.

In the end, the linear weighting method is utilized to combine the two scores above. Denoting the weight of interest score as μ, the weight of service correlation score is $(1-\mu)$, and the composite score of U_p on S_q can be computed by formula (10):

$$\mathrm{FS(q)} = \alpha IS(p,q) + (1-\alpha)CS(q) \tag{10}$$

After calculating composite score of all candidate services using formula (10), we can select top-K services of the highest scores and complete this step of recommendation of the successor services.

3.4 Analysis on UISCS Approach

Algorithm based on the UISCS approach is as Table 2 shows.

User's implicit interest is calculated first (lines 1-7). Specifically, after assigning an implicit interest randomly to each service on the user-service matrix to complete initialization, the algorithm iterates T times. During each iteration, for each service on the matrix, the algorithm calculates the posterior probability that the service belongs to each interest and assigns it a new interest accordingly.

After T iterations, user-interest matrix θ and interest-service Matrix φ can be estimated based on the interest distribution of each service at this time. Then the users' score ratings for each service are calculated based on these two matrices and the formula (1), and an initial recommendation is made (lines 8-10). The time complexity of the above part of is $O(PQT)$, as the number of iterations T is constant, the time complexity is actually $O(PQ)$.

Table 2. Algorithm for UISCS approach

```
Algorithm for UISCS approach

Input: US: the matrix of User-Service; K: the number of Interest;
       α; β: the hyperparameters of dirichlet distribution in LDA;
       N: Steps that users want to be recommended.
Output: AL: Final recommendation list.

1   Set: AL=null, candidate list (CL)=all services;
2   Assign a random interest to each service invoked by someone (item) in the US and complete the initialization for Gibbs;
3   For iteration=1 to T:
4       For each item in the US:
5           Compute its probability distribution of being assigned to each interest according to the interest of other services;
6           Sample a new interest for the service based on the above distribution;
7   Compute the matrix φ and θ based on the interest of each item in the US, formula (1) and (2);
8   For each service in CL:
9       Compute its interest score (IS) by formula (3) and matrix φ and θ;
10  Recommend the top-K service with the highest IS to the user;
11  Interact with the user, and then remove the service the user select from CL and add it to AL;
12  For service i in CL:
13      For service j in CL:
14          Compute the correlation between service i and j (W(i;j)) using formula (8);
15  For step=1 to N:
16      For each service in CL:
17          Compute its correlation score CS according to the formula (9)and CL;
18          Add up IS and CS score and get its final score (FS);
19      Recommend the top-K service with the highest FS to the user;
20      Add the service the user selects into AL and remove it from CL;
21  Return AL to the user as the final recommendation.
```

At the same time, the correlation matrix of candidate services is calculated based on mass diffusion algorithm (lines 12-14), and the time complexity is $O(Q^2)$. Then a N-step successor recommendation is made (lines 15-20). The recommendation takes full account of the correlation between the candidate service and each chosen service, as well as the interest score of the candidate service. The time complexity of this part is $O(N^2Q)$. Since N is generally a small real value, the actual time complexity is $O(Q)$.

Finally, we output the recommended list in line 21. The total time complexity of the algorithm is $O(QT+Q^2+Q) = O(Q(Q+P))$, which is quadratic and related to the number of users and that of services.

4 EXPERIMENTS

IN this section, we conduct several experiments on real data sets to tune parameters and optimize our algorithm, and to compare the recommendation performances of our approach with other state-of-the-art recommendation methods.

Our experiments aim at addressing the following questions:

Q1: How to set the parameters of LDA model to make the UIS and UISCS algorithm have the best performance?

Q2: What is the impact of different strategies by which users select successor services on recommendation accuracy?

Q3: The user interest and the degree of correlation among services are the important basis for successor service recommendation, so how does the weight allocation between them affect the recommendation performance?

Q4: How does UISCS perform compared with UIS which is based on the LDA? And how do UISCS and UIS perform with different sparsity of data sets?

Q5: Does the UISCS approach have a wider range of applicability except LDA? Can it improve the recommendation performance of other state-of-the-art algorithms, such as ICF, SVD and TSR?

The running environment of this experiment is under Windows 8 operating system, 2.60GHz CPU, 8GB memory, and the code is implemented in Java programming environment.

4.1 Evaluation Metrics

To evaluate the algorithm comprehensively, six metrics were chosen to test accuracy, diversity, novelty, and timeliness. Precision, recall, and F1 score are three metrics widely used for accuracy. Diversity and novelty are suitable metrics to measure how much an algorithm could solve the long tail problem.

1) Precision

Precision measures the probability that a user is interested in a service recommended to him. It is defined as the ratio of the user's interested services or correct services to all services in the recommended list, which can be represented as formula (11), where N_{rs} represents the number of correct services in the recommended list, Ns represents the total number of the services recommended by the algorithm.

$$P = \frac{N_{rs}}{N_s} \tag{11}$$

The Recall indicates the probability of the user's interested service being recommended to him: It is defined as the ratio of the correct services in the recommended list to the ones in the system. We can calculate it by the following formula:

$$R = \frac{N_{rs}}{N_r} \tag{12}$$

And N_r represents the number of services that users are interested in.

The accuracy and recall rates need to be used together to fully evaluate the algorithm's quality. The F1 is widely used because it unifies the accuracy and recall. It is expressed as the harmonic mean of the two indicators:

$$\mathrm{F1} = \frac{2PR}{P+R} \tag{13}$$

2) Hamming

Average Hamming distance is used to measure the diversity of recommended lists for different users in the recommended results. The Hamming distance of two user u_i, u_j can be expressed as:

$$Hamming_{ij} = 1 - \frac{Q_{ij}}{L} \tag{14}$$

In the formula (14), L is the length of recommendation list, $Q_{i,j}$ is the number of the same services in the recommendation list for user i and user j. The average Hamming distance of the recommended system is defined as the average of all Hamming distances between every pair of users. Its maximum value is 1, meaning that the recommended list for all users is completely different; the minimum value is 0, that is, the recommended list is identical.

3) Novelty

A good recommendation system should not only have a higher accuracy, but also recommend a little-known product to the user. Novelty measures the ability of a recommender system to recommend new information to users. The smaller the value is, the higher the novelty is. It can express as formula (15), in which n is the number of the users, $k(I_r)$ is the degree of a service or the times the services is called in this experiment.

$$\text{Novelty} = \frac{1}{nL}\sum_{i=1}^{n}\sum_{I_r \in I_R^i} k(I_r) \tag{15}$$

4) Timeliness

The running time of recommendation system can be divided into two parts: algorithm modeling time and system response time, in which the response time is directly related to user experience. In general, the response time is the time a system spends in giving a user all recommendations at one time; for UISCS, it should be the time spent on each step of the recommendations, so here we regard it as the average of the multi-step recommendations, and the calculation formula is:

$$M_{UISCS} = \frac{\sum_{i=1}^{n} M_{API}(i)}{n} \tag{16}$$

In the formula (16), $M_{API(i)}$ is the time of the i_{th} step recommendation, n is the number of steps and M_{UISCS} is response time of UISCS.

4.2 Baseline Algorithms

Four baseline recommendation approaches are chosen as comparisons: UIS, a collaborative filtering recommendation algorithm based on LDA; ICF, a service-oriented and neighborhood-based collaborative filtering recommendation algorithm; SVD, an algorithm based on singular value decomposition; TSR, an algorithm based on trust relation on the graph models.

(1) *UIS*. The algorithm extracts the user-interest matrix and the interest-service matrix from the user-service invocation matrix using the LDA model and then calculates the user's score on a service based on these two matrices and the formula (3).

(2) *ICF*. This algorithm is a reasonably popular collaborative filtering algorithm proposed by Sarwar et al. (2001) and Deshpande et al. (2004). After calculating the similarity between the services based on the user-service invocation matrix, the algorithm calculated the score of a candidate service i according to the scores that the user p give on each invoked service j and the similarity between the candidate and each invoked service. It could expressed as formula (17): where s_u is the set of all the services that the user had scored, $R_{(u,j)}$ is the score given by user u

on the service j , and $sim_{(i,j)}$ is the similarity between service i and j.

$$R_{u,i} = \frac{\sum_{j \in S_u} Sim_{i,j} R_{u,j}}{\sum_{j \in S_u} Sim_{i,j}} \quad (17)$$

(3) *SVD*. West et al. (2015) used a singular value decomposition technique in linear algebra to complete a decomposition of user-item invoke matrix. After calculating the similarity between the services based on the user-service invocation matrix, the algorithm calculated the score of a candidate service i according to the scores that the user p give on each invoked service j and the similarity between the candidate and each invoked service. It could expressed as formula (17): where s_u is the set of all the services that the user had scored, $R_{(u,j)}$ is the score given by user u on the service j , and $sim_{(i,j)}$ is the similarity between service i and j.

$$R_{u,i} = \frac{\sum_{j \in S_u} Sim_{i,j} R_{u,j}}{\sum_{j \in S_u} Sim_{i,j}} \quad (18)$$

(4) *TSR*. As a trust-aware algorithm, Deng et al. (2015) assumed that apart from their own interests, users were also influenced by the impact of their friends when making choices. So TSR first quantified the trust value among friends based on social networks and then, in the final recommendation, combined the influence of friends' interest weighted by above trust values and of personal interest.

4.3 Tuning Parameters for LDA

This experiment aims at answering Q1. In order to improve the performance of UISCS approach, the parameters of LDA model are tuned firstly. We gradually increase the number of hidden topics from 25 to 200 by step 25. The hyper parameters , α and β, are both initialized to 0.5 and then processed by Fixed-Point Iteration proposed by Minka et al. (2000) and Steyvers et al. (2007), and they are approximately convergent after 15 iterations.

At this point, the optimal models with different number of hidden topics are obtained, and then the similarities of all topics (topicSim) of each model are calculated by the methods of Cao et al. (2009). The topic number, the convergent value of hyper parameters and the similarity of these topics of each model are shown in Table 3 and Figure 4.

Table 3. Optimal parameters of LDA with different topics

topicNum	α	β	topicSim
25	0.039575	0.106627	0.028321
50	0.021002	0.104067	0.01362
75	0.01496	0.104617	0.009354
100	0.018813	0.097738	0.007449
125	0.092607	0.093596	0.009092
150	0.526684	0.060865	0.024607
175	0.524797	0.080585	0.025821
200	0.516553	0.095917	0.026546

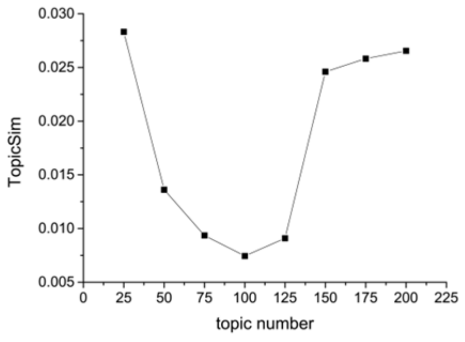


Figure 4. Similarity of all topics vs. topic number

4.4 The impact of service selection strategies on recommendation performance

This experiment aims at answering Q2. When users participate in the construction of a composite service online, different service selection strategies they adopt have a certain impact on the service recommendation. Three possible service selection strategies are considered on the basis of fully analysis of user demands:

(1) *Random*. Suppose that users do not distinguish between candidate services, they always choose a service randomly from the recommendation list.

(2) *BestScore*. Assuming that the service with the highest comprehensive score is most likely to meet the user's needs, users would always choose such service from the recommendation list.

(3) *Intelligence*. Assuming that the mashup in the experimental data is the optimal combination of APIs with users having sufficient knowledge, the service is obviously better than others in the recommendation list if it exists in the test set. Therefore, users always give priority to these services.

We test the UISCS algorithm based on the above three service selection strategies respectively, and their F1 varies with different size of recommendation list, as is shown in Figure 5.

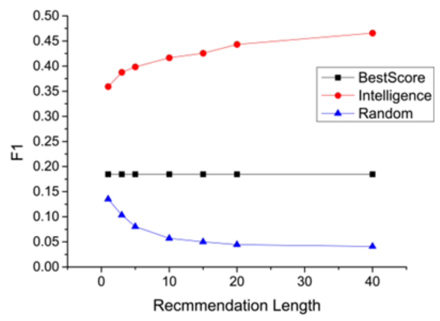


Figure 5. Recommendation accuracy under different service selection strategies

Regardless of the length of the recommendation list, the F1 value of Intelligence strategy is always much higher than that of Random and BestScore. The reason is that intelligence strategy makes full use of the user's feedback information and excavates their true preference, thereby enhancing the accuracy.

BestScore, however, always chooses the service with the highest FS value regardless of the size of the recommendation list, so its accuracy does not change with the increase in the number of service candidates. Owing to the difference between FS and the user's true preference, it's difficult for this mechanical method to accurately grasp the user's personalized interest, resulting in low accuracy. Random has neither considered system scoring nor user feedback, and its accuracy is very poor.

For Intelligence, accuracy is improved when the length of recommended list per step is gradually increased. This is because the set of candidate services becomes larger and the possibility of a user finding a satisfying service increases. However, it should also be noted that when the recommendation list is greater than 10, accuracy is increased slowly, while the difficulty of selection is significantly increased and the user experience is poor. So we make a tradeoff between accuracy and user experience, setting the length of recommended list to 10 in subsequent experiments.

In a word, we find that when selecting successor services, intelligence strategy that makes full use of the user's feedback information has the best accuracy, solving our second problem.

4.5 The impact of weight distribution between interest and correlation on recommendation

This experiment aims at answering Q3. The user interest score (IS) calculated by LDA model and the service correlation score (CS) calculated by mass diffusion are the important basis for successor service recommendation. In order to visually observe the impact of the weight distribution on the final recommendation, we increase the weight of CS from 0 to 100% by step 10%. Accordingly, μ, the weight of IS reduced from 100% to 0 gradually. The recommendation performance under different weight distribution is shown in Table 4.

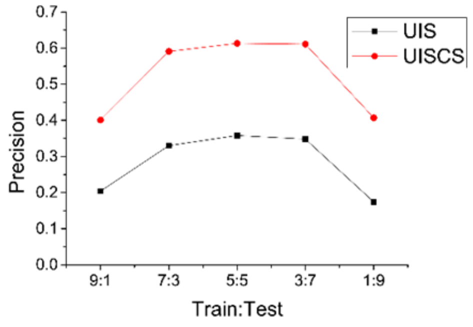
Table 4. Recommendation performance of different weight distribution

1-μ:μ	F1	Hamming	Novelty	Time(ms)
0:100%	0.39554	0.95096	51.28994	25.6
10%:90%	0.39495	0.95123	50.99053	30.0
20%:80%	0.39584	0.94945	51.34398	30.2
30%:70%	0.39535	0.95144	50.97475	26.2
40%:60%	0.39832	0.94874	52.19842	26.8
50%:50%	0.40017	0.94783	52.74596	29.4
60%:40%	0.40185	0.94741	52.80158	27.4
70%:30%	0.40569	0.94511	54.11479	25.8
80%:20%	0.40961	0.94425	54.73136	28.2
90%:10%	0.40767	0.93910	56.19329	29.4

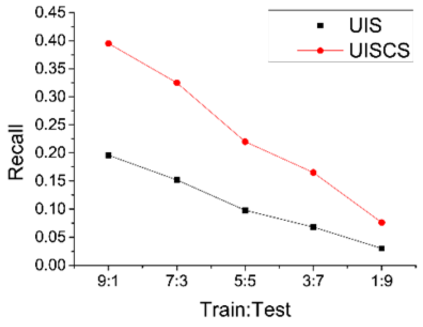
As is shown above, with the increase of CS weight, the recommendation accuracy (including precision, recall, and F1) tends to increase. This is because, with the mass diffusion model integrated, UISCS can analyze the service the user selects in real time, thereby dynamically updating his real needs and finally giving a more accurate recommendation result. When the weight ratio of CS and IS is 80%: 20%, the Recommendation accuracy is the highest and the diversity and novelty have reached the approximate optimality, so this weight setting is the answer to question (3) and is adopted in subsequent experiments.

4.6 Performance experiment of UISCS

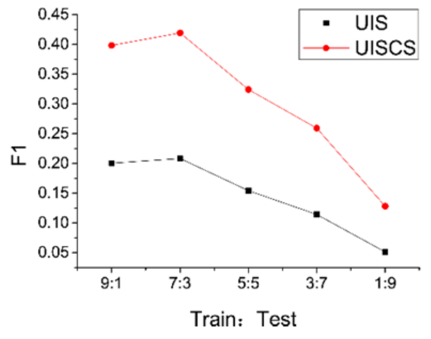
This experiment aims at answering Q4. In order to verify the performance of UISCS, we implement UISCS based on LDA algorithm and compare it with UIS on the data sets of different sparsity. Since Mashups on the Programmableweb generally consist of less than 3 APIs, the successor recommendation of UISCS is set to 3 steps. According to the experiment 4.4 and 4.5, the length of recommendation list for both UIS and UISCS are set to 10. The performance of the two approaches is shown in the following Figure 6(a) – Figure 6(f).



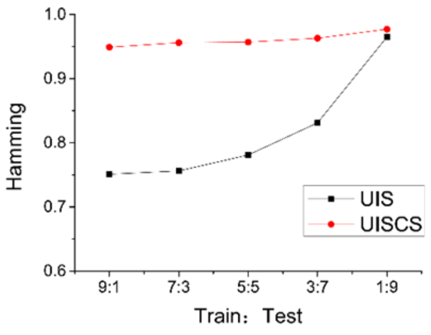
(a) Precision



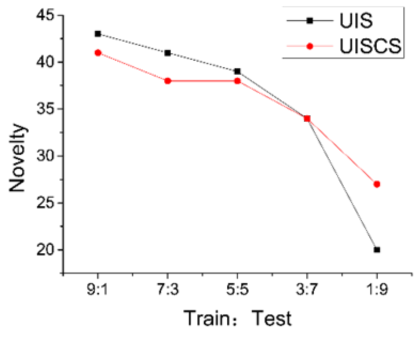
(b) Recall



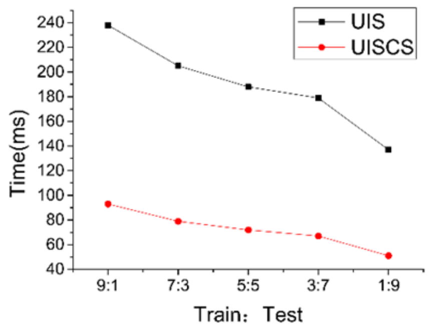
(c) F1



(d) Hamming



(e) Novelty



(f) Time

Figure 6. **The performance of UISCS based on LDA**

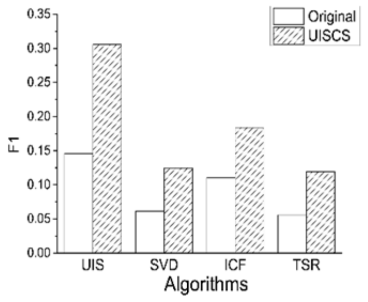
From the experimental results, we can find the answer to Q4: the performance of UISCS approach is better than UIS in most cases.

First of all, the accuracy and diversity of UISCS are higher than that of UIS to varying degrees regardless of the sparsity of the data set. This is because UISCS fully integrates the user's prior knowledge, models the user more precisely and grasps the user's dynamic needs in real time. Therefore, UISCS can provide more accurate and personalized recommendation for users with diverse needs. It is worth mentioning that the accuracy of the UISCS is still significantly better than UIS when the sparseness of the data set is high, indicating that it can reduce the interference of the cold start problem to a certain extent.

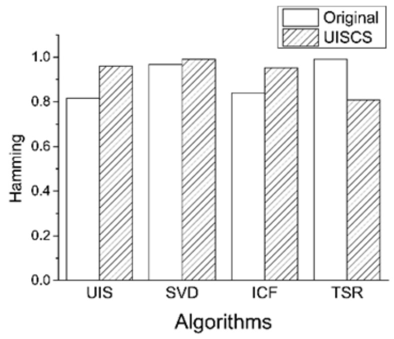
Secondly, the degree of data sparseness has a great influence on the novelty of UISCS approach. The UISCS is superior to the UIS when the proportion of test set is greater than 10%, but the result is opposite at 10%. This is because the LDA model tends to reduce the novelty by relatively increasing the scoring factors of some new but high quality services, whereas the mass diffusion model is the opposite. When the training set is intensive, the trained LDA model is of better quality, having a greater effect on reducing the novelty than the effect of mass diffusion model on increasing the novelty. Therefore, the novelty of the UISCS approach is lower than that of the UIS. And the situation is exactly the opposite when the data is sparse.

4.7 *Applicability test of the UISCS approach*

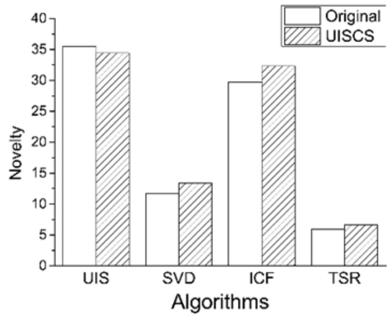
This experiment aims at answering Q5. In order to test whether the framework of UISCS can improve the performance of service recommendation generally or not, we improve some mainstream service recommendation algorithms, such as ICF, SVD and TSR, by integrating mass diffusion theory for successor service recommendation. After making the initial services recommendation using these algorithms, we recommend the successor services based on the mass diffusion model, and then compare the performance of each algorithm before and after improvement. The experiments are conducted on five different datasets with different sparsity. And we use the average value of each metric and each algorithm on each dataset as the final performance of the algorithm. The specific result of the recommendation is shown in the following Figures from Figure 7(a) to Figure 7(c).



(a) F1



(b) Hamming



(c) Novelty

Figure 7. The performance of UISCS based on mainstream algorithms

In Figure 7, the word "Original" represent the original algorithms, including UIS(i.e. LDA), ICF, SVD and TSR; and "UISCS" represent the improved algorithms of LDA, ICF, SVD and TSR, which are optimized by mass diffusion algorithm.

Firstly, regardless of the data set sparsity, the precision, recall, F1, and Hamming of ICF and SVD algorithms are improved to some extent when optimized by mass diffusion algorithm. The reason is UISCS approach can gradually recognize the real demand of users, and then give a more accurate and personalized recommendations.

Secondly, since the smaller the value is, the better the novelty is, the novelty of ICF and SVD algorithms are poorer when optimized by mass diffusion algorithm. This is because the ICF is based on the relationship among services, which is similar to the mass diffusion algorithm, and tends to give higher scores to those frequent services and thus increase the novelty value. Although the SVD has positive impact on infrequent service, the decrease in novelty value of SVD can't offset the increase of mass diffusion, and resulting in higher novelty value.

Thirdly, since the TSR algorithm requires the input of user's accurate ranking scores on the services, the recommendation performance of the TSR is likely to be affected and not good as expected in above experiment for its input is 0-1 matrix. However, the accuracy of the UISCS is still significantly improved compared with original TSR with the same input.

In short, as a step-by-step service recommendation approach, the UISCS approach has obvious improved the performance of some mainstream recommendation algorithms, such as LDA, SVD, ICF and graph-based TSR. And our last question is validated.

5 CONCLUSIONS AND FUTURE WORK

CURRENT systems of service recommendations cannot effectively assist the user to complete a service composition, so a novel service composition model is proposed. The model subdivides the process of service recommendation into an initial service and successor service recommendation. The model works by discovering user interests and mining correlations among services, recommending an initial service based on user interest, and then recommending successor services based on both the user interests and the correlations among services.

However, the algorithm can be improved in the following two aspects: First, The services can be clustered into different granularities of the service chains based on the different thresholds of correlation between the services, so multi-granularity of services or the service chains can be supplied as required, this means the recommendation includes not only the single services, but also service chains in a different granularity and some mashups. Second, the time information is not considered when mining user interest. If we mined user current interest using timing information and predicted further interests by a sequence revolution, the performance of UISCS approach would be better.

6 ACKNOWLEDGEMENT

THIS work was partially supported by the National Natural Science Foundation of China (Nos.61602536, 61309029, and 61273293), Key Research and Development Project of Zhejiang Province (No. 2015C01027), National Science Foundation of China (No. 61772461), Natural Science Foundation of Zhejiang Province (No. LR18F020003).

7 REFERENCES

Agarwal, S. (2007). A Goal Specification Language for Automated Discovery and Composition of Web Services. *Web Intelligence, IEEE/WIC/ACM International Conference on* (pp.528-534). IEEE.

Bai, B., Fan, Y., Huang, K., Tan, W., Xia, B., & Chen, S. (2015). Service Recommendation for Mashup Creation Based on Time-Aware Collaborative Domain Regression. *IEEE International Conference on Web Services* (pp.209-216). IEEE.

Cao, J., Xia, T., Li, J., Zhang, Y., & Tang, S. (2009). A density-based method for adaptive lda model selection. *Neurocomputing*, 72(7-9), 1775-1781.

Deng, S., Huang, L., Yin, Y., & Tang, W. (2015). Trust-based service recommendation in social network. *Applied Mathematics & Information Sciences,* 9(3), 1567.

Deshpande, M., & Karypis, G. (2004). Item-based top-n recommendation algorithms. *Acm Trans.inf.syst*, 22(1), 143-177.

Gao, Z., Fan, Y., Wu, C., Tan, W., Zhang, J., & Ni, Y., et al. (2016). SeCo-LDA: Mining Service Co-occurrence Topics for Recommendation. *IEEE International Conference on Web Services* (pp.25-32). IEEE.

Hao, Y., Fan, Y., Tan, W., & Zhang, J. (2017). Service Recommendation Based on Targeted Reconstruction of Service Descriptions. *IEEE International Conference on Web Services* (pp.285-292). IEEE.

Huang, K., Fan, Y., & Tan, W. (2014). Recommendation in an evolving service ecosystem based on network prediction. *IEEE Transactions on Automation Science & Engineering*, 11(3), 906-920.

Li, C., Zhang, R., Huai, J., Guo, X., & Sun, H. (2013). A Probabilistic Approach for Web Service Discovery.

Liu, Q., Chen, E., Xiong, H., Ding, C. H. Q., & Chen, J. (2012). Enhancing collaborative filtering by user interest expansion via personalized ranking. *IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society,* 42(1), 218-233.

Liu, X., & Fulia, I. (2015). Incorporating User, Topic, and Service Related Latent Factors into Web Service Recommendation. *IEEE International Conference on Web Services* (pp.185-192). IEEE.

Liu, X., & Turtle, H. (2013). Real - time user interest modeling for real - time ranking. *Journal of the Association for Information Science & Technology*, 64(8), 1557 - 1576.

Ma, S. P., Lin, H. J., Yu, C. A., & Lee, C. Y. (2017). Web API recommendation based on service cooperative network. *International Conference on Applied System Innovation* (pp.1922-1925). IEEE.

Minka, T. (2013). Estimating a dirichlet distribution. , 39(3273), 115.

Naim, H., Aznag, M., Quafafou, M., & Durand, N. (2016). Probabilistic Approach for Diversifying Web Services Discovery and Composition. *IEEE International Conference on Web Services* (pp.73-80). IEEE.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. International *Conference on World Wide Web* (Vol.4, pp.285-295). ACM.

David, M. (2010). Probabilistic topic models. *IEEE Signal Processing Magazine*, 27(6), 55-65.

Tang, M., Xu, Y., Liu, J., Zheng, Z., & Liu, X. (2014). Combining Global and Local Trust for Service Recommendation. *IEEE International Conference on Web Services* (pp.305-312).

Wang, G., Zhang, S., Liu, C., & Han, Y. (2014). A Dataflow-Pattern-Based Recommendation Approach for Data Service Mashups. *IEEE International Conference on Services Computing* (Vol.8, pp.163-170). IEEE.

Wang, X., Cheng, Z. H., Zhou, Z., & Ning, K. (2014). Geospatial Web Service Sub-chain Ranking and Recommendation. *IEEE International Conference on Services Computing* (pp.91-98). IEEE Computer Society.

West, R., Paranjape, A., & Leskovec, J. (2015). Mining Missing Hyperlinks from Human Navigation Traces: A Case Study of Wikipedia. (Vol.2015, pp.1242). *NIH Public Access*.

Xu, S., Shi, Q., Qiao, X., Zhu, L., Jung, H., & Lee, S., et al. (2014). Author-Topic over Time (AToT): A Dynamic Users' Interest Model. Mobile, Ubiquitous, and Intelligent Computing. *Springer Berlin Heidelberg.*

Xu, Y., Yin, J., Huang, J., & Yin, Y. (2018). Hierarchical topic modeling with automatic knowledge mining. *Expert Systems with Applications*, 103, 106-117.

Yao, L., Wang, X., Sheng, Q. Z., Ruan, W., & Zhang, W. (2015). Service Recommendation for Mashup Composition with Implicit Correlation Regularization. *IEEE International Conference on Web Services* (Vol.8, pp.217-224). IEEE Computer Society.

Ye, B., & Wang, Y. (2016). CrowdRec: Trust-Aware Worker Recommendation in Crowdsourcing Environments. *IEEE International Conference on Web Services* (pp.1-8). IEEE Computer Society.

Yin, C., Wang, J., & Park, J. H. (2017). An improved recommendation algorithm for big data cloud service based on the trust in sociology. *Neurocomputing*.

Yin, Y., Yu, F., Xu, Y., Yu, L., & Mu, J. (2017). Network location-aware service recommendation with random walk in cyber-physical systems. *Sensors*, 17(9), 2059.

Zhang, Y., Lei, T., & Qin, Z. (2018). A service recommendation algorithm based on modeling of dynamic and diverse demands. *International Journal of Web Services Research*, 15(1), 47-70.

Zheng, X., Luo, Y., Sun, L., Ding, X., & Zhang, J. (2017). A novel social network hybrid recommender system based on hypergraph topologic structure. *World Wide Web-internet & Web Information Systems*(3), 1-29.

Zhong, Y., Fan, Y., Huang, K., Tan, W., & Zhang, J. (2014). Time-Aware Service Recommendation for Mashup Creation in an Evolving Service Ecosystem. *IEEE International Conference on Web Services* (Vol.8, pp.25-32). IEEE.

8 CONTRIBUTORS

Yanmei Zhang, received her Ph.D. degree in Computer Department from China University of Mining and Technology (CUMT), Beijing, China, in 2010. She was a Visiting Scholar with Northwestern University, Evanston, IL, USA, in 2007 and 2017, respectively. She is currently an Associate Professor of the Department of Computer Science and Technology, Central University of Finance and Economics (CUFE), Beijing, China. Her current research interest includes service computing, personalized recommendation systems, and web information processing.

Xiao Geng is currently a postgraduate at the school of computer science of Wuhan University. He is a leader of a national innovation training program which was selected as an outstanding one in acceptance stage. He won the National Endeavor Fellowship, academic scholarship of undergraduates. His interest of research areas including service recommendation, topic detection and deep learning.

Shuiguang Deng, received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 2002 and 2007, respectively. He was a Visiting Scholar with the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2014, and Stanford University, Stanford, CA, USA, in 2015. He is currently an Associate Professor with the College of Computer Science and Technology, Zhejiang University. His current research interests include service computing and business process management.