Service Recommendation Based on Topics and Trend Prediction

Lei Yu[1,2(✉)], Zhang Junxing[1], and Philip S. Yu[3]

[1] Inner Mongolia University, Hohhot, China
yuleiimu@sohu.com
[2] State Key Laboratory of Networking and Switching Technology, BUPT, Beijing, China
[3] University of Illinois at Chicago, Chicago, USA

Abstract. Web service recommendation is a challenging task when the number of services and service consumers are growing rapidly on the Internet. Previous research used information retrieve methods, such as keyword search and semantic matching, to speculate the intent of service consumers. The intent is matched with contents or topics of existing data. These methods help service consumers to select appropriate services according to their needs. However, service evolution over time and topic correlation has not been given sufficient attention. Thus we propose a service recommendation approach that is able to extract service evolution patterns from history statistic data and correlated topics from semantic service descriptions. To this end, time series prediction is used to obtain evolution patterns; Latent Dirichlet Allocation (LDA) is used to model the extracted topics. Experiments results show that our approach has higher precision than existing methods.

Keywords: Service recommendation · Trend Prediction · Latent Dirichlet Allocation

1 Introduction

An important task is service discovery in an automated style, because service composition rely on the precision of automated service searching. Several web sites are collecting web services and mashups, such as ProgrammableWeb and myExperiment [1] in the recent years. myExperiment is used for sharing a variety of scientific workflows, such as Taverna and RapidMiner. Although this kind of web sites provide an easy way for web service consumers, searching desired and suitable services in large databases of services is still a time-consuming and tedious job for the service consumers. Service recommendation methods can facilitate consumers discover the suitable services.

Most of service recommendation methods search the information by keywords or semantic. Keyword-based search is inefficient, and semantic-based search needs much time to construct semantic information. A search method based on Latent Dirichlet Allocation (LDA) [2] was proposed for the challenge. In the method, a collections of

S. Wang and A. Zhou (Eds.): CollaborateCom 2016, LNICST 201, pp. 343–352, 2017.
DOI: 10.1007/978-3-319-59288-6_31

words are extracted from WSDL, and assigned to several topics. Some other researchers [3] applied social network analysis for service recommendation.

In addition, temporal information has been ignored. The fact is that service topics change over time. Recommending services with popular topics may be reasonable for users, but how to decide which service is popular in the future is a problem. To deal with the problem, we collect the past usage of services at intervals, and predict the popularity in the future. Besides popularity, topics may be related each other. Using correlated topics can provide more related recommendation results in a recommendation system. It is especially useful when a search for one topic returns few recommendation results. In addition, we assume that services with similar functions aim to solve similar problems, thus these services are in the same topic.

We summarized contributions as follows: First, we extract a sequence of topic popularity from service usage history. Based on the sequence and Latent Dirichlet Allocation (LDA), we predict topic evolution and service popularity in the future. Second, based on the topic evolution model, we propose a method for service recommendation, called SRTT.

2 Related Work

Wang et al. [4] proposed an efficient QoS management approach for QoS-aware web service composition, and they classified web services according to similarity and then design a QoS tree to manage the QoS the classified web services. Chen et al. [5] proposed a collaborative filtering-based Web service recommender system to help users select services with optimal Quality-of-Service (QoS) performance. Their recommender system employed the location information and QoS values to cluster users and services, and made personalized service recommendation for users based on the clustering results. Lee et al. [6] developed a recommendation mechanism to predict user intention and activate the appropriate services. They chose to employ the event-condition-action model together with a rule induction algorithm to discover smartphone users' behavior patterns. Huang [7] proposed a three-phase network prediction approach (NPA) for evolution-aware recommendation. They introduced a network series model to formalize the evolution of the service ecosystem and developed a network analysis method to study the usage pattern with a special focus on its temporal evolution. In addition, a service network prediction method based on rank aggregation was proposed to predict the evolution of the network. Sun et al. [8] presented a new similarity measure for web service similarity computation and proposed a novel collaborative filtering approach, called normal recovery collaborative filtering, for personalized web service recommendation.

Cao et al. [9] designed a cube model to explicitly describe the relationship among providers, consumers and Web services. they presented a Standard Deviation based Hybrid Collaborative Filtering (SD-HCF) for Web Service Recommendation (WSRec) and an Inverse consumer Frequency based User Collaborative Filtering (IF-UCF) for Potential Consumers Recommendation (PCRec). Finally, the decision-making process

of bidirectional recommendation was provided for both providers and consumers. Sets of experiments were conducted on real-world data provided by Planet-Lab.

Wu et al. [10] presented a neighborhood-based collaborative filtering approach to predict such unknown values for QoS-based selection. In addition, a two-phase neighbor selection strategy was proposed to improve its scalability. Zheng et al. [11] proposed a collaborative quality-of-service (QoS) prediction approach for web services by taking advantages of the past web service usage experiences of service users. They applied the concept of user-collaboration for the web service QoS information sharing. Based on the collected QoS data, a neighborhood-integrated approach was designed for personalized web service QoS value prediction.

Chen et al. [12] proposed a collaborative filtering algorithm designed for large-scale web service recommendation. The approach employed the characteristic of QoS and achieves considerable improvement on the recommendation accuracy. To avoid the time-consuming and expensive real-world service invocations, Zheng et al. [13] proposed a QoS ranking prediction framework for cloud services by taking advantage of the past service usage experiences of other consumers. The proposed framework requires no additional invocations of cloud services when making QoS ranking prediction. Two personalized QoS ranking prediction approaches were proposed to predict the QoS rankings directly. Zheng et al. [14] proposed two personalized reliability prediction approaches of Web services, that is, neighborhood-based approach and model-based approach. The neighborhood-based approach employed past failure data of similar neighbors (either service users or Web services) to predict the Web service reliability. On the other hand, the model-based approach fits a factor model based on the available Web service failure data and use this factor model to make further reliability prediction. Yu et al. [15] proposed a clustering method and a recommendation method for Web services. The clustering method combines TF-IDF (Term Frequency-Inverse Document Frequency) and ontology to compute the similarity of Web services, and it uses Ward's Distance to identify irregular shapes. The recommendation method uses matrix factorization to recommend proper services.

Related works mentioned above have some deficiencies. They do not consider the changing of service popularity, which may make high ranked services degradation. As a result, some usable services will not be recommended to the user. Third, the precision of recommendation results and speed of previous methods still have space to improve.

3 Service Recommendation Method

The recommendation will provide services according to a user query, in which the user query is matched with service description or mashup description.

After preprocessing on service descriptions, a collection of separated words w_1, w_2, …, w_n can describe and represent the functions of a service. Likewise, a collection of separated words w_1, w_2, …, w_n can describe and represent the functions of a mashup. A user search contains several words that indicate the intent of the user. These words

can be represented by $Q = \{q_1, q_2, \ldots, q_n\}$, and will be matched with the first two collections. In the next step, a list of ranked services $R(m)$ will be generated. Higher $R(m)$ is more likely to be recommended to the user for creating a new mashup.

Considering service history information and content of services, we propose a service recommendation method. Thus, the components of our approach are TP (Trend Prediction) and CM (Content Matching) respectively.

(1) TP predict service activity according to service usage history. Regardless of functional requirements of a mashup, TP provides popularity scores of services in the near future.TP can offer a list of hot services invoked by a large amount of users. Hot services may be the result from low fee, high efficiency or lovely appearance viewing by majority of users, but may not be the required service by the current user. For complementation, CM and TC give higher score to the functional relevant services.
(2) By calculating semantic similarity between the requirements and the descriptions of services, CM selects services with similar functional requirements from existing mashups.

3.1 Trend Prediction

Popular services cater for users in many different ways, which is a common experience of users in every consumer market. That is the reason why new users like to consume prevalent services at current time. The prevalent services must have interesting topics. Analyzing the service usage history to recommend popular services is a reasonable idea to topic modeling. Topic modeling assumes that each service has several topics, or called latent topics which is needed to be calculated. LDA [16], a method for topic mining, can be used to describe the service generative process.

Suppose k_i^{t+1} is topic popularity for topic i at time t + 1. For every topic i, $(k_i^0, k_i^1, k_i^2, \ldots, k_i^{t+1})$ become a time series. The future popularity of topic i can be forecasted by Eq. 1. Several methods are able to solve the above problem, e.g. auto regression and linear weighted moving average [17]. We use the linear weighted moving average as follows.

$$k_i^{t+1} = \sum_{j=0}^{n} w_j \times k_i^{t-j} \tag{1}$$

$\boldsymbol{w}$ are weightings, which are positive real numbers, and satisfy the constraint $\sum w_i = 1$. Giving higher weightings to recent k will amplify the popularity.

According to LDA, a matrix can be represented by two matrices. For extension, we assume matrix C is represented by matrix Θ, matrix E(k) and matrix Φ. Matrix Θ and Φ are similar to that of LDA, but matrix E(k) is the extension. Θ is a word-topic matrix, as shown below. Its elements indicate the probability that a word belongs to a topic.

$$\Theta = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \quad \Phi = \begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{m1} & \cdots & b_{mn} \end{pmatrix}$$

Φ is a topic-service matrix, as shown below. Its elements indicate the probability of a topic over a service. E(k) is a coefficient matrix, as shown below. It is also a diagonal matrix that do not change the number of rows and columns of the matrix it is multiplied. Its elements, which are calculated from Eq. 1, indicate the popularity of each topic.

$$E(k_i^{t+1}) = \begin{pmatrix} k_1^{t+1} & & 0 \\ & \ddots & \\ 0 & & k_n^{t+1} \end{pmatrix}$$

C is a word-service matrix, indicating that each service is described by several words. Considering popularity of services, C should equal to the multiplication of matrix Θ, E(k) and Φ.

$$\mathrm{C} = \Theta \times E(k_i^{t+1}) \times \Phi$$

When a service consumer submits a query including several words (Q) for a new service (s), Eq. 2 shows the topic similarity between Q and service descriptions. Services with higher $P(Q|s)$ will be provided to the consumers.

$$P_{tp}(Q|s) = \prod_{q_n \in Q} \sum_{k=1}^{T} P(q_n|k_i^{t+1}) E(k_i^{t+1}) P(k_i^{t+1}|s) \tag{2}$$

3.2 Content Matching

Collaborative filtering is used broadly for recommendation [11] in many areas. It assumes that similar users possibly need similar items. Likewise, we calculate the mashups similarities by their descriptions, and then provide the component services from similar mashups for a new mashup based on historical information. For example, if the descriptions of a new mashup m_q is similar to an existing mashup m_e, and m_e is composed by services A and B, then m_q probably invokes A and B.

According to mashup descriptions, LDA is used to calculate the similarity among mashups. A collection of words, which describe functions of mashups, is the input of LDA. LDA uses Gibbs Sampling to obtain two posterior distributions, which are mashups-topics $p(k|m)$ and topics-words $p(w|k)$. When a user queries, by submitting with words $Q = \{q_1, q_2, \ldots, q_n\}$ for a new mashup m_q, the similarity between m_q and an existing mashup m_e can be computed as follows:

$$sim(m_q, m_e) = \prod_{q \in Q} \sum_{k=1}^{T} p(q|k) p(k|m_e) \tag{3}$$

After computing two mashups similarities, mashup m_q should use a component service s form similar mashups according to the following equation:

$$P_{cm}(Q|s) = \sum_{m_e \in U(k)} sim(m_q, m_e) d(m_e, s) \tag{4}$$

$U(k)$ contains the Top K mashups that are similar to m_q. $d(m_e, s)$ indicates that mashup m_e contains the service s. Comparing to Eq. 2, this method does not consider trend information.

4 Service Recommendation Framework

TP and CM have been introduced in the previous section. They will be integrated for service recommendation in this section. TP is used to obtain tendency. CM is used to obtain semantic similarity. The probability that a service s is invoked by a new mashup m_q is:

$$P(Q|s) = w_1 P_{tp}(Q|s) + w_2 P_{cm}(Q|s) \tag{5}$$

Where w_i are weightings. The ranked services for mashup m_q will provide to users in a descending order of $P_{tc}(Q|s)$. Alternatively, a technique from paper [18] can be used to aggregate each ranking produced by one component of our method in the last section to generate a final ranking.

We use approximate inference (e.g. Gibbs Sampling) to obtain the latent variables in LDA. Gibbs Sampling is formally a kind of Markov-Chain Monte Carlo.

Algorithm 1. Service Recommendation

Input:

T: The number of topics in LDA

SD: Service description texts

α: the parameter of the Dirichlet prior distributions for topics

β: the parameter of the Dirichlet prior distributions for words

$Q = \{q_1, q_2, \ldots, q_n\}$: a user query containing n words

Output:

Recommended services

Procedure:

1. Use Gibbs Sampling (α, β, SD) to obtain Matrix Θ and Φ
2. Compute tendency by Equ.1 and obtain E(k)
3. Obtain a user query
4. Calculate similarity by Equ.2 (TP) using parameters θ_1, ϕ_1
5. Calculate similarity by Equ.4 (CM) using parameters θ_1, ϕ_1
6. Calculate similarity between service descriptions and the user query by Equ.5
7. Recommend the high-ranked services

5 Experiments

Our experimental platform is: Operation system: Microsoft Windows 7. CPU: Intel Core i7, 2.5 GHz. RAM: 4 GB. Programming language: Java. Two data sources were used in our experiments. They are ProgrammableWeb.com and SAWSDL-TC [19]. The former data source is mainly used to evaluate the precision of each component of our method, and the latter data source is mainly used for comparing with other existing methods in the regard of advanced precision and response speed.

For more precise evaluation, we evaluate our algorithm by calculating the Precision@n (proportion of the top-n relevant services) and the Normalized Discounted Cumulative Gain (NDCG@n).

Precision@n computes the precision of the service discovery for the first n retrieved services. Precision@n indicates the relevance between the user query and retrieved n services. The Precision@n is given by:

$$Precision@n = \frac{r}{n}$$

n indicates the number of retrieved services. r indicates the number of relevant services from the set of retrieved services. DCG is defined as follows.

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

Where p is a particular rank position, and rel_i is the graded relevance at position i. The normalized discounted cumulative gain is computed as:

$$NDCG_p = \frac{DCG_p}{IDCG_p}$$

We evaluated the effectiveness of our method based on CT by computing the Precision@n and NDCG@n. all of 42 queries are encoded by SAWSDL which contain semantic requirements. We compare our method with syntax-based methods, Apache Lucene [20] and SAWSDL-MX2 [21]. The queries are represented in a collection of words in our approach. The queries are submitted in the form of strings of text descriptions in the Apache Lucene. At last, the most relevant services will be selected for the user.

Comparing the averaged Precision@n and the NDCG@n, we evaluated different levels of concept lattices. The Precision@20 and NDCG@20 values are obtained in the three methods. The results are shown in Fig. 1.

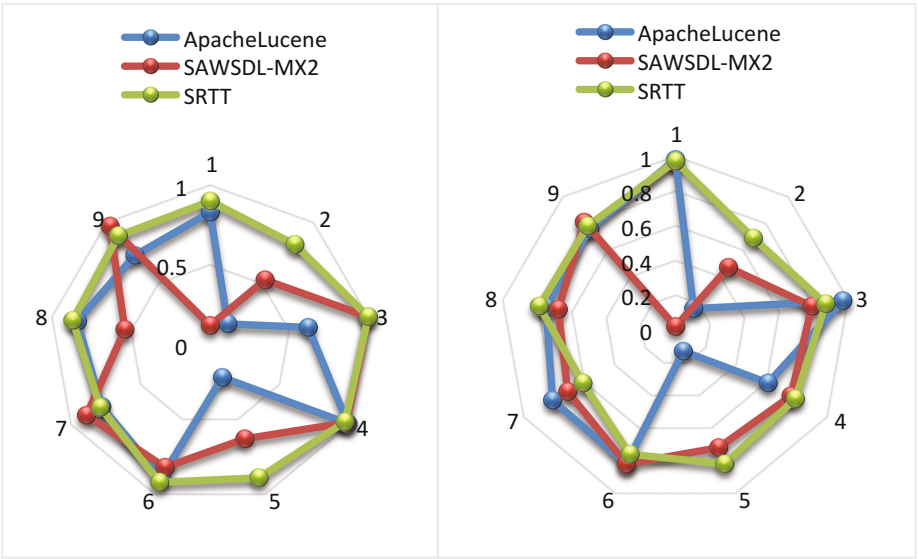

Fig. 1. Precision@20 and NDCG@20 for nine queries

In both cases, our SRTT method obtains higher Precision@20 and NDCG@20 for 9 queries. The Apache Lucene and SAWSDL-MX2 do not find the relevant services for three queries. Figures 2 and 3 show the average Precision@n and NDCG@n values of 42 queries. SRTT obtains the highest precision values than others.

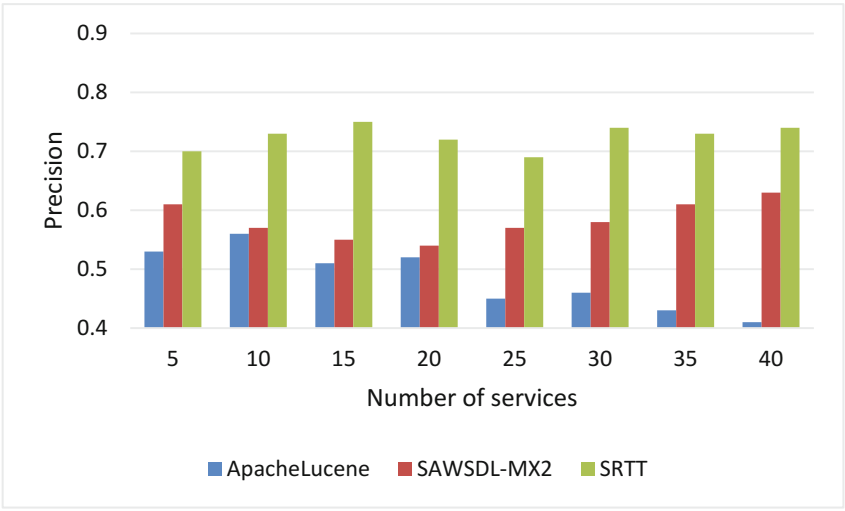

Fig. 2. Precision in each number of services for 42 queries

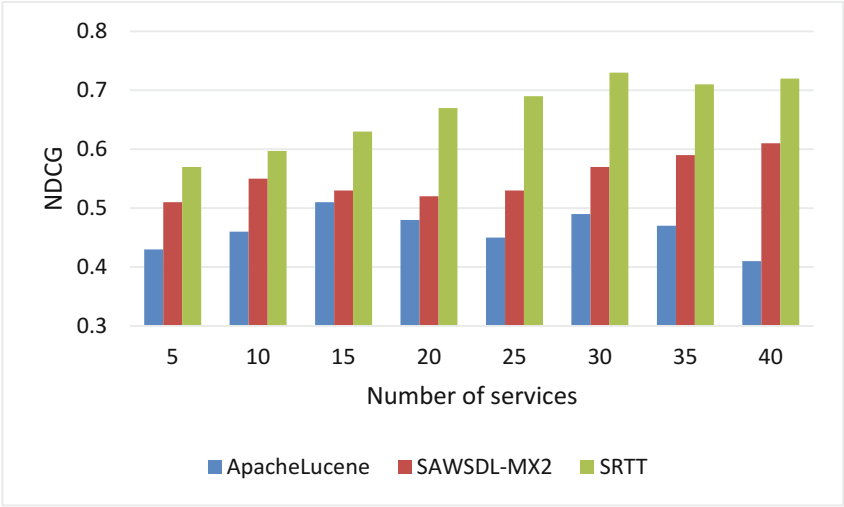

Fig. 3. NDCG in each number of services for 42 queries

In the future, we will improve SRTT method to adapt to big data environment [22]. We will propose a new service recommendation method that can be executed in parallel, with the help of Spark framework.

6 Conclusion

In this paper, we presented a method that combines service popularity and service content of services to recommend services to a mashup. We first extract topics from semantic service descriptions, and then analyze historic information to predict service popularity. In the experiment, the results of Precision@n and NDCG@n show that our method is better than the other two methods. We also conduct experiments on ProgrammableWeb.com, a real-world data set. The results show that our method is better than content matching and collaborative filtering, with respect to mean average precision for service recommendation.

Acknowledgment. This work was supported by Scientific projects of higher school of Inner Mongolia [NJZY009], Open Foundation of State Key Laboratory of Networking and Switching Technology (SKLNST-2016-1-01), Programs of Higher-level talents of Inner Mongolia University [215005145143], Natural Science Foundation of Inner Mongolia Autonomous Region [2015BS0603].

References

1. De Roure, D., Goble, C., Stevens, R.: The design and realization of the myExperiment virtual research environment for social sharing of workflows. Future Gener. Comput. Syst. **25**, 561–567 (2009)
2. Li, C., Zhang, R., Huai, J., Guo, X., Sun, H.: A probabilistic approach for web service discovery. In: Proceedings of the IEEE International Conference on Services Computing, pp. 49–56 (2013)
3. Cao, J., Xu, W., Hu, L., Wang, J., Li, M.: A social-aware service recommendation approach for mashup creation. Int. J. Web Serv. Res. **10**, 53–72 (2013)
4. Wang, S.G., Zhu, X.L., Yang, F.C.: Efficient QoS management for QoS-aware web service composition. Int. J. Web Grid Serv. **10**(1), 1–23 (2014)
5. Chen, X., et al.: Web service recommendation via exploiting location and QoS information. IEEE Trans. Parallel Distrib. Syst. **25**(7), 1913–1924 (2014)
6. Lee, W., Lee, K.: Making smartphone service recommendations by predicting users' intentions: a context-aware approach. Inf. Sci. **277**, 21–35 (2014)
7. Huang, K., Fan, Y., Tan, W.: Recommendation in an evolving service ecosystem based on network prediction. IEEE Trans. Autom. Sci. Eng. **11**(3), 906–920 (2014)
8. Sun, H.F., et al.: Personalized web service recommendation via normal recovery collaborative filtering. IEEE Trans. Serv. Comput. **6**(4), 573–579 (2013)
9. Cao, J., et al.: Hybrid Collaborative Filtering algorithm for bidirectional Web service recommendation. Knowl. Inf. Syst. **36**(3), 607–627 (2013)
10. Wu, J., et al.: Predicting quality of service for selection by neighborhood-based collaborative filtering. IEEE Trans. Syst. Man Cybern. Syst. **43**(2), 428–439 (2013)
11. Zibin, Z., et al.: Collaborative web service QoS prediction via neighborhood integrated matrix factorization. IEEE Trans. Serv. Comput. **6**(3), 289–299 (2013)
12. Chen, X., et al.: Personalized QoS-aware web service recommendation and visualization. IEEE Trans. Serv. Comput. **6**(1), 35–47 (2013)
13. Zheng, Z., et al.: QoS ranking prediction for cloud services. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1213–1222 (2013)
14. Zheng, Z., Lyu, M.R.: Personalized reliability prediction of web services. ACM Trans. Softw. Eng. Methodol. **22**(2), 1–25 (2013)
15. Yu, L., Wang, Z.-L., Meng, L.-M., et al.: Clustering and recommendation for semantic web service in time series. KSII Trans. Internet Inf. Syst. **8**(8), 2743–2762 (2014)
16. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
17. Matsubara, Y., Sakurai, Y., Faloutsos, C., Iwata, T., Yoshikawa, M.: Fast mining and forecasting of complex time-stamped events. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 271–279 (2012)
18. Sheng, X., et al.: SOR: an objective ranking system based on mobile phone sensing. In: IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014, June 30, Madrid (2014)
19. http://www.semwebcentral.org/projects/sawsdl-tc
20. http://lucene.apache.org/
21. http://projects.semwebcentral.org/projects/sawsdl-mx
22. Dobre, C., Xhafa, F.: Intelligent services for Big Data science. Future Gener. Comput. Syst. **37**, 267–281 (2014)