2017

A Method for Predicting Service Deprecation in Service Systems

Bofei Xia
Department of Automation, Tsinghua University, Beijing 100084, China.

Yushun Fan
Department of Automation, Tsinghua University, Beijing 100084, China.

Cheng Wu
Department of Automation, Tsinghua University, Beijing 100084, China.

Bing Bai
Department of Automation, Tsinghua University, Beijing 100084, China.

Junqi Zhang
Department of Automation, Tsinghua University, Beijing 100084, China.

Follow this and additional works at: https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology

Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation
Bofei Xia, Yushun Fan, Cheng Wu et al. A Method for Predicting Service Deprecation in Service Systems. Tsinghua Science and Technology 2017, 22(1): 52-61.
A Method for Predicting Service Deprecation in Service Systems

Bofei Xia, Yushun Fan*, Cheng Wu, Bing Bai, and Junqi Zhang

Abstract: An increasing number of web services are being invoked by users to create user applications (e.g., mashups). However, over time, a few good services in service systems have become deprecated, i.e., the service is initially available and is invoked by service users, but later becomes unavailable. Therefore, the prediction of service deprecation has become a key issue in creating reliable long-term user applications. Most existing research has overlooked service deprecation in service systems and has failed to consider long-term service reliability when making service recommendations. In this paper, we propose a method for predicting service deprecation, which comprises two components: Service Comprehensive Feature Modeling (SCFM) for extracting service features relevant to service deprecation andDeprecated Service Prediction (DSP) for building a service deprecation prediction model. Our experimental results on a real-world dataset demonstrate that our method yields an improved Area-Under-the-Curve (AUC) value over existing methods and thus has better accuracy in service deprecation prediction.

Key words: web service; service deprecation predict; Latent Dirichlet Allocation (LDA); extreme learning machine

1 Introduction

Service-Oriented Computing (SOC) technique has led to a new day in software engineering, by changing the way we design, develop, deliver, and consume software applications[1, 2]. Recently, a number of online service repositories have been established, such as ProgrammableWeb, myExperiment, and Biocatalogue. In these repositories, a large number of published services offer Application Programming Interfaces (APIs) for external invocation, whereby users can invoke various services with different functionalities to create their own user applications (mashups) to meet specific requirements. An increasing number

- Bofei Xia, Yushun Fan, Cheng Wu, Bing Bai, and Junqi Zhang are with Department of Automation, Tsinghua University, Beijing 100084, China. E-mail: xbf9xbf@qq.com; fanyus@tsinghua.edu.cn; wuc@tsinghua.edu.cn; bb13@mails.tsinghua.edu.cn; jq-zhang15@mails.tsinghua.edu.cn.
- To whom correspondence should be addressed.

Manuscript received: 2015-12-13; revised: 2016-02-05; accepted: 2016-07-01

of services are being published by various service providers and invoked by a wide range of service users. These services, service providers, and service users form service systems[3, 4].

However, over time, a number of services in the service systems become deprecated, i.e., the service is initially available for use and is invoked by service users, but later it becomes unavailable. As such, many service users who are developing applications (mashups) are invoking services that will soon become deprecated, leading to the inevitable failure of the developed applications. For example, in the service system ProgrammableWeb, more than 1000 of the 10,000 published services have become deprecated, which has caused failure in more than 2000 user-developed mashups. At this point, it is necessary to predict service deprecation to ensure the long-term reliability of user-developed applications. However, most existing research has neglected the service deprecation problem.

In this paper, we propose a method for predicting service deprecation in service systems, which has two main components. The first component is a Service
Comprehensive Feature Modeling (SCFM) model for extracting service features, including a service textual description file and service provider information. SCFM integrates the services comprehensive features into a fixed-length vector. The second component is a Deprecated Service Prediction (DSP) model for learning the mapping pattern between the service comprehensive feature vector and whether that service was deprecated, based on historical records. DSP is actually a binary classification model which ultimately predicts whether a given service will become deprecated in the near future. We evaluate our method using a dataset from ProgrammableWeb, and the results of evaluation experiment indicate that our proposed method better predicts service deprecation than the existing method.

The remainder of the paper is organized as follows. In Section 2, we describe our motivation for this work, using a real-world example. In Section 3, we define the problem and overall framework of our method. In Section 4, we present the two main components of our method. In Section 5, we present our experiments and discuss the results with respect to a real-world dataset. In Section 6, we discuss related work and we draw our conclusions in Section 7.

2 Motivation

In this section, we use a real-world example to illustrate the need for our work when users want to create long-term reliable mashup applications.

The ProgrammableWeb, by far the largest online Web Open API (service) repository, contains more than 10,000 APIs with various functionalities. Among these, for example, are Google Maps, Yahoo Maps, and Bing Maps in the mapping category; Amazon and eBay in the e-commerce category; and Facebook and Twitter in the social media category. These reusable and programmable APIs enable users to select specific applications to create their own new value-added user applications (mashups). For example, let’s imagine that a user wants to create a mashup application that could easily search for and find local auction information. The recommendation engine may return two lists of currently available services to the user: Google Maps, Bing Maps, and Yahoo Maps as possible mapping functions and Amazon Product Advertising, eBay, and Shopping.com for the auction searching function. In our example, the user selects the Yahoo Maps API and the eBay API to build the mashup application, as shown in Fig. 1.

However, in time, Yahoo Maps becomes no longer available and is deprecated by Yahoo. Therefore, although Yahoo Maps was available when the user was creating the mashup application, its later demise caused this mashup application to become unusable. Furthermore, according to ProgrammableWeb’s mashup records, 137 mashup applications invoking the Yahoo Maps API died along with Yahoo Maps.

The main flaw in the mashup creation process is that when a user selects a service from a good set of services with similar functionalities, there is no way of knowing whether a currently available service will become deprecated at some future time. If the user is given information regarding the future deprecation of Yahoo Maps and the relative long-term reliability of Google Maps and Bing Maps, then this more comprehensive overview helps the user to avoid future mashup application failure due to deprecating services. To date, most existing research has concentrated on fulfilling the functional requirements of users and has overlooked the long-term reliability of services.

The objective of this paper is to develop a method for predicting whether a service will become deprecated

![Fig. 1](image-url)
in order to identify reliable long-term services and help users create reliable long-term mashup applications.

3 Problem Definition

Our objective is to predict whether a web service in a service system will become deprecated in the near future. To clarify the study problem, we introduce two definitions.

Definition 1 (Service Contents) Each service \( S \) comprises a collection of words \( W \) that describe its functions. Each service also corresponds to a service provider \( P \) and one provider may provide more than one service. Formally, the service content can be defined as \( S \sim \{P, W\} \).

Definition 2 (Service Indicator) We define an indicator \( I(S) \) to indicate whether or not the service \( S \) will be deprecated, and \( I(S) \) is represented as a binary variable: “1” means service \( S \) will become deprecated and “0” means services will not become deprecated for a long time.

Based on the above two definitions, we formulated the problem of predicting service deprecation in service systems as follows.

Problem (Predicting Service Deprecation) Given a service and its content \( S \sim \{P, W\} \), we predict the value of \( I(S) \) in order to tell the service user whether or not the service will become deprecated in the near future.

Predicting service deprecation has thus become a classification problem, which we can address in two steps (as shown in Fig. 2). These steps comprise the two main components of our work:

1. SCFM: SCFM takes the service content, which includes service description words and service provider information, as input and transforms it into a feature vector. The feature vector quantifies the service features that relate to its long-term reliability.
2. DSP: By exploiting the historical records of service deprecation records and the service feature vectors produced by SCFM, DSP learns the mapping pattern between the service feature vectors and whether a given service will become deprecated.

When a new service \( S' \) is introduced, we can use SCFM to extract the features of this new service, then use DSP to calculate its predicted \( I(S') \) value. In this way, the service user is informed about whether the service \( S \) will be deprecated in the near future. In the following section, we discuss the details and rationale of SCFM and DSP.

4 Model and Algorithms

In this section, we describe the construction of the two components in our approach: SCFM and DSP.

4.1 SCFM

The objective of SCFM is to extract features from service content that are related to the long-term reliability of services. In our opinion, as the service system has evolved, two main service features have emerged that affect whether a service will be deprecated in the near future: the service functionality and the provider. As illustrated in previous work, services with different functions may differ in their popularity [4]. In addition, services offered by different providers may provide similar functions but they also differ in quality [3]. Therefore, we model both a service function feature and a service provider feature and then merge them into a comprehensive service feature vector.

4.1.1 Service Functional Feature Modeling (SFFM)

Service description words contain an abundance of information about the service function. To exploit these words, we apply a concept similar to topic modeling and probabilistically analyze the service textual description. Specifically, we view services as articles and service functions as latent topics. Thus, we can employ the concept of latent Dirichlet allocation [5] to quantitatively model the service functionality. We called this model service functional feature modeling.

Table 1 summarizes the parameters that we used in SFFM.

4.1.2 Generation process of SFFM

Figure 3 shows a probabilistic graphical model of SFFM. The generative process of the service functions and service description words are described by
Bofei Xia et al.: A Method for Predicting Service Deprecation in Service Systems

Table 1 Parameters used in SFFM.

| Parameter | Explanation |
|-----------|-------------|
| M         | Total number of services |
| K         | Total number of functions (topics) evolved in all services |
| V         | Total number of unique words |
| \( N_m \) | Number of word tokens in service \( m \)’s description file |
| \( \theta_m \) | Multinomial distribution of functions specific to service \( m \), in the proportion of one for each service, \( \theta = \{ \theta_m \}_{m=1}^{M} (M \times K \text{ matrix}) \) |
| \( \phi_k \) | Multinomial distribution of words specific to function (topics) \( k \), in the proportion of one for each function, \( \phi = \{ \phi_k \}_{k=1}^{K} (K \times V \text{ matrix}) \) |
| \( f_{m,n} \) | Function associated with the \( n \)-th word token in service \( m \)’s description file |
| \( w_{m,n} \) | The \( n \)-th word token in service \( m \)’s description file |
| \( \alpha \) | Dirichlet priors to the multinomial distribution \( \theta \) |
| \( \beta \) | Dirichlet priors to the multinomial distribution \( \phi \) |

Algorithm 1

According to the generative process shown in the probabilistic graphical representation (Fig. 3), by placing Dirichlet priors, we derive the joint probability of all the service description words \( W \) and all their corresponding word functions \( F \) as follows:

\[
P(W, F | \alpha, \beta) = \int P(W, F | \Theta, \Phi) P(\Theta | \alpha) P(\Phi | \beta) d\Theta d\Phi =
\]

\[
\prod_{k=1}^{K} \frac{\Gamma(\sum_{v=1}^{V} \beta_v)}{\prod_{v=1}^{V} \Gamma(\beta_v)} \prod_{m=1}^{M} \frac{\Gamma(\sum_{k=1}^{K} \alpha_k)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \cdot \prod_{m=1}^{M} \prod_{k=1}^{K} \Gamma(\sum_{v=1}^{V} (n_{kv} + \beta_v)) \cdot \prod_{m=1}^{M} \prod_{k=1}^{K} \Gamma(\sum_{k=1}^{K} (n_{mk} + \alpha_k))
\]

where \( n_{kv} \) is the number of times that word token \( v \) is generated by function \( k \) and \( n_{mk} \) is the number of times that function \( k \) is associated with service \( m \).

4.1.3 Parameter learning of SFFM

A variety of algorithms have been developed to estimate the parameters of topic models. Here, we adopt the Gibbs sampling algorithm to infer the unknown parameters \{\( \Theta, \Phi \)\}. Specifically, we begin with the posterior probability of \( f_{m,n} \), as follows:

\[
P(f_{m,n} | W, F_{-i(m,n)}, \alpha, \beta) \propto \frac{n_{f_{m,n}w_{m,n}} + \beta w_{m,n} - 1}{\sum_{v=1}^{V} (n_{f_{m,n}v} + \beta_v) - 1} \times (n_{mf_{m,n}} + \alpha f_{m,n} - 1)
\]

where “\( -i \)” indicates a quantity excluding the current instance. \( n_{f_{m,n}w_{m,n}} \) is the number of times that word token \( w_{m,n} \) is generated by function \( f_{m,n} \) in all service description files. \( n_{mf_{m,n}} \) is the number of times that word token \( v \) is generated by function \( f_{m,n} \). \( n_{mf_{m,n}} \) is the number of times that function \( f_{m,n} \) is associated with service \( m \)’s description file.

After a sufficient number of sampling turns, we estimate the parameters based on the sampling results. Through a similar deduction using latent Dirichlet allocation, we can also estimate the parameters as follows:

\[
\phi_{k,v} = \frac{n_{k,v} + \beta_v}{\sum_{v=1}^{V} (n_{k,v} + \beta_v)},
\]

\[
\theta_{m,k} = \frac{n_{m,k} + \alpha_k}{\sum_{k=1}^{K} (n_{m,k} + \alpha_k)}
\]

where \( \phi_{k,v} \) is the occurrence probability of word \( v \) given function \( k \), \( \theta_{m,k} \) is the relevance probability of service \( m \) to function \( k \). The Gibbs sampling algorithm is detailed in Algorithm 2.

4.1.4 Service functional vector derivation

After learning the parameter \( \Theta_{M \times K} \) of the SFFM model, each service’s function (topic) distribution will correspond to a \( \Theta_{M \times K} \) row. More specifically, the \( m \)-th row of the \( \Theta_{M \times K} \) matrix, \( \theta_m = \{ \theta_{m1}, \theta_{m2}, \ldots, \theta_{mK} \} \).
Algorithm 2  Gibbs sampling process

Input:
1) Hyper-parameters $\alpha$ and $\beta$
2) All the service description words $W$
3) Preset iteration number $T$

Output:
1) Parameters estimates $\{\Theta, \Phi\}$

Procedure:
01. Randomly initialize $F$
02. For $\text{iter} = 1:T$
03. For each word token $w_{m,n}$
04. Sample $f_{m,n}$ according to Eq. (2)
05. End
06. End
07. Read out $\{\Theta, \Phi\}$ according to Eq. (3)

represents the function distribution of service $m$ and $\theta_{mk}$ denotes the probability of service $m$ being related to the function $k$.

At this point, the SFFM takes a good set of service description words as input and outputs each service’s function features as a fixed-length vector.

### 4.1.5 Service Provider Information (SPF)

We define a $\Psi_{M \times P}$ matrix to formulate an overview of the services and their providers. $M$ is the total number of services and $P$ is the total number of service providers. More specifically, the $m$-th row of the $\Psi_{M \times P}$ matrix, vector $\psi_m = \{\psi_{m1}, \psi_{m2}, \ldots, \psi_{mp}\}$ represents the provider information of service $m$ and $\psi_{mp}$ is a binary variable for which “1” means service $m$ is provided by provider $p$, and “0” means service $m$ is not provided by provider $p$. The SPF function also organizes each service provider’s information into a fixed-length vector.

### 4.1.6 Service comprehensive feature derivation

To combine the matrices of $\Theta_{M \times K}$ and $\Psi_{M \times P}$, we define an $I_{M \times (K+P)}$ matrix to integrate the service function feature with the service provider information. More specifically, the $m$-th row of the $I_{M \times (K+P)}$ matrix represents the comprehensive features of service $m$.

Lastly, the SCFM takes both the service description words and service provider information as input and quantifies the comprehensive feature of a service into a fixed-length vector. In the next section, we use $I_{M \times (K+P)}$ to develop our service deprecation prediction method.

For new services, we can input the service description words and service provider information into our SCFM model and transform the new service into a service feature vector.

### 4.2 DSP

In this section, we introduce the DSP model and how it predicts services that will be deprecated.

#### 4.2.1 Model description

In addition to the SCFM, we obtain each service’s comprehensive feature vector. Then, from historical data, we can know which services have become deprecated over time and which services are always reliable. Hence, we adopt a machine learning method to learn the mapping pattern between service comprehensive feature vectors and whether a service will become deprecated from history records. In this way, we can predict if a new service will be deprecated in the future.

As shown in Fig. 4, the DSP designed is similar to that of the extreme learning machine [7] binary classification model. DSP comprises three components: input matrix $IM_{(K+P) \times H}$, hidden layer bias vector $BV_{1 \times H}$, and output matrix $OM_{H \times 1}$, where $K$ is the number of functions involved in all services, $P$ is the number of service providers, and $H$ is the number of hidden layer units. The DSP input is a $K + P$ dimension vector $i_m$, as derived in Section 4.1.6. Then, DSP calculates the output result $p$ using the following formula:

$$p = (i_m \cdot IM + BV) \cdot OM$$ (4)

If $p$ is larger than the threshold value, the DSP outputs 1 and predicts that service $m$ will be deprecated in the near future; otherwise, the DSP outputs 0 and predicts that service $m$ will not be deprecated in the near future.

#### 4.2.2 Parameter learning

The $IM$ matrix, $BV$ vector, and $OM$ matrix are the three parameters required in order to learn from the historical data. We put the training data into two matrices. The $I_{M \times (K+P)}$ matrix derived in Section 4.1.6 serves as the input feature data. For the label data, we define

![Fig. 4  Workflow of DSP model.](image-url)
another matrix \( D_{M \times 1} = \{d_1, d_2, \ldots, d_M\} \) to denote whether the service was deprecated, according to the historical data. \( d_m \) equals 0 denotes that service \( m \) was not deprecated and \( d_m \) equals 1 denotes that service \( m \) was deprecated. The details of the parameter learning algorithm, which are similar to those of the training algorithm in Ref. [7], are provided in Algorithm 3. Furthermore, a different number of hidden-layer units, \( H \), will affect the DSP prediction accuracy. We will discuss this point in Section 5.

### 4.2.3 Prediction algorithm

In addition to the parameters learned from historical data, we use DSP to predict if a new service will be deprecated in the near future. The details of this prediction algorithm are given in Algorithm 4.

### 5 Experiment

We conducted a series of experiments on the ProgrammableWeb dataset, to compare the prediction accuracy of our proposed DSP method with other state-of-the-art methods.

#### 5.1 Data set description

We crawled the metadata of a service’s description file and provider information from ProgrammableWeb.com, ranging from June 2005 to June 2013. In this data set, the textual information of each service consists of its description file, service tags, and summary. Most services have information about the publishing company and assign the company who published the service as its provider. Deprecated services are tagged as “deprecated”, otherwise, the service is not deprecated.

After removing meaningless or vacant services, we collect a total of 6700 services. Details of this data-set are listed in Table 2.

#### 5.2 Evaluation metrics

We use the Area-Under-the-Curve (AUC) value\(^8\) to measure the performance of the DSP method, and derive this value as follows: Since deprecating service prediction problem is a binary classification problem, the prediction result falls into one of four categories in a confusion matrix, as shown in Table 3. We can obtain the True Positive Rate (TPR) and False Positive Rate (FPR)\(^9\) from the confusion matrix. Specifically, TPR = TP/(TP + FP) and FPR = FP/(FP + TN).

Different DSP thresholds in Algorithm 4 yield different TPR and FPR value pairs. With FPR as the x-axis and TPR as the y-axis, we can obtain Receiver

---

**Table 2** Data set from ProgrammableWeb.com.

| Statistical Value                  |
|------------------------------------|
| Total number of services           | 6700 |
| The number of deprecated services  | 818  |
| Total number of terms in services textual corpus | 370 000 |
Table 3  Confusion matrix of deprecated service prediction problem.

| Actual deprecated service | Predicted as deprecated service | Predicted as non-deprecated service |
|---------------------------|---------------------------------|-------------------------------------|
| True Positive (TP)        | False Negative (FN)             |                                     |
| Actual not deprecated service | False Positive (FP)            | True Negative (TN)                  |

Operating Characteristic (ROC) curve of the algorithm and calculate the area between the curve and x-axis, which is the AUC value. The AUC value ranges between 0 and 1. The larger is the AUC value, the better is the performance of this method.

5.3  Experimental results

In this section, first we compare the performance of the proposed method, DSP method, with the classical Naive Bayes binary classification method[10]. Next, we discuss how the number of DSP hidden-layer units affects prediction accuracy.

5.3.1  Performance comparison

We conducted a 10-fold cross validation[11] of the overall dataset to obtain the ROC curve in order to compare the AUC values of our proposed DSP and the Naive Bayes. With respect to the Naive Bayes method, we used the service description words, service providers, and other textual information as input and then calculated the probability of a service becoming deprecated. With respect to the proposed DSP method, we had to preset some method parameters: we set the number of functions $K$ (in Table 1) to 40, the number of iterations of Gibbs sampling $T$ (in Algorithm 2) to 2000, and the number of hidden-layer unit $H$ to 50 (In Section 4.2.1). Figure 5 shows the ROC curve comparison.

As shown in the figure, the ROC curve of the DSP is higher than that of Naive Bayes method, which intuitively indicates that DSP performs better than Naive Bayes. More specially, the AUC of the DSP (0.858) is larger than that of the Naive Bayes (0.805). As stated above, a larger AUC means better prediction accuracy.

5.3.2  Impact of parameter $H$

In this section, we discuss how parameter $H$ impacts the performance of DSP. As stated in Section 4.2.2, parameter $H$ denotes the number of hidden-layer units in the DSP model. The larger is the $H$ value, the more powerful is the DSP learning ability. But if $H$ is too large, the DSP will be subject to the learning over-fitting problem, which means that the DSP model fits only the historical instance and lacks generalization ability to predict new instances. In this context, we preset different values of $H$ and obtained AUC values for the corresponding DSP for both the training and testing dataset.

As shown in Fig. 6, we calculated the AUC value of the DSP on the training dataset as follows: $H = 5$ (AUC $= 0.81$), $H = 50$ (AUC $= 0.87$), $H = 100$ (AUC $= 0.89$), and $H = 500$ (AUC $= 0.90$). Also shown in Fig 7, we calculate the AUC value of DSP on the testing dataset as follows: $H = 5$ (AUC $= 0.712$), $H = 50$ (AUC $= 0.858$), $H = 100$ (AUC $= 0.827$), and $H = 500$ (AUC$=0.731$). A higher AUC value in the training dataset indicates that the DSP better fit the historical data. A higher AUC value in the testing dataset indicates the method has better generalization ability and can better predict new instance. When $H$ is 5, the DSP algorithm is unable to learn the matching
Fig. 7 ROC curves of DSP model with different $H$ on the testing dataset.

pattern between the service comprehensive features and service deprecation from the historical data. Therefore, both the AUC value for the training dataset (0.87) and that for the test dataset (0.712) are lower than the others. When $H$ is 500, although DSP is powerful enough to fit the historical data (its AUC for the training dataset is 0.9), it has the over-fitting (as its AUC value on the test data set is only 0.73).

In conclusion, we must carefully increase the parameter $H$ to realize greater DSP prediction power but also consider the over-fitting problem associated with too large $H$ value.

6 Related Work

The deprecated service prediction problem is the key issue in recommending reliable long-term services to users. In recent years, a few good of studies have focused on service recommendation.

Semantic matching methods recommend services based on the semantic relevance between services and user queries. Li et al.\cite{12} used the probabilistic topic model, Latent Dirichlet Allocation (LDA), to extract the functional attributes of services from Web Services Description Language (WSDL) documents. Their recommendation method is based on the topic-level semantic matching probability. Meng et al.\cite{13} developed a keyword-aware service recommendation method KASR, in which key words are used to indicate user’s preferences, and a user-based collaborative filtering algorithm generates appropriate recommendations.

Many service recommendation studies have also taken Quality of Service (QoS) into consideration. Ahmed et al.\cite{14} proposed a Hidden Markov Model (HMM) method for QoS metrification, which measures and predicts the behavior of Web services in terms of response time. Zheng et al.\cite{15} used a matrix factorization technique to develop a collaborative QoS prediction approach for Web services that takes advantage of the past Web service usage experiences of service users. Tang et al.\cite{16} considered the locations of both users and services when predicting the QoS values of Web services and recommending service candidates. Xiang et al.\cite{17} developed a lightweight description method for predicting QoS for the large-scale service-oriented Internet of Things. Other studies\cite{18,19} have used the QoS index to efficiently create web service compositions.

Collaborative relations is another approach used to enhance the service recommendation process. Cao et al.\cite{20} extracted user interests from their mashup service usage history, and then built a social network based on social relationships information to support service recommendations for mashups. Xu et al.\cite{21} presented a socially aware service recommendation approach, which uses a coupled matrix model to describe multi-dimensional social relationships among potential users, topics, and mashups.

However, most research work has overlooked the issue of deprecated services in the service system and fail to take service long-term reliability into consideration when making service recommendations. Our work provides an effective quantitative way of predicting deprecated services and increases the long-term reliability of services being recommended.

7 Conclusion

In this paper, we focused on service deprecation in the service system and proposed a comprehensive method for predicting service deprecation in order to help users create long-term mashup application.

Our method mainly comprises two component models. SCFM model extracts relevant service reliability features from service description words and service provider information and models these features into a fixed-length vector. DSP model uses historical data to learn and then predict deprecated services. Our experimental results demonstrate that our proposed method acquires a higher AUC value and thus performs better than the existing method.

In future work, we plan to factor time into our model and identify ways to realize more effective service deprecation prediction.
References

[1] W. Yi and M. B. Blake, Service-oriented computing and cloud computing: Challenges and opportunities, *IEEE Internet Computing*, vol. 14, pp. 72–75, 2010.

[2] L.-J. Zhang, J. Zhang, and H. Cai, *Services Computing*. Springer, 2007.

[3] K. Huang, Y. Fan, and W. Tan, An empirical study of programmable web: A network analysis on a service-mashup system, in *Proceedings of IEEE 19th International Conference on Web Services (ICWS)*, 2012, pp. 552–559.

[4] K. Huang, Y. Fan, and W. Tan, Recommendation in evolving service ecosystem based on network prediction, *IEEE Trans. Sci. Eng.*, vol. 11, no. 3, pp. 906–920, 2014.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan, Latent Dirichlet allocation, *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[6] 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*, 2008, pp. 569–577.

[7] G. Huang, Q. Zhu, and C. Siew, Extreme learning machine: Theory and applications, *Neurocomputing*, vol. 70, pp. 489–501, 2006.

[8] A. P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[9] J. Davis and M. Goadrich, The relationship between precision-recall and ROC curves, in *Proceedings of ACM 23rd International Conference on Machine learning (ICML)*, 2006, pp. 233–240.

[10] I. Rish, An empirical study of the naive Bayes classifier, in *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, 2001, pp. 41–46.

[11] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995, pp. 1137–1145.

[12] C. Li, R. Zhang, J. Huai, X. Guo, and H. Sun, A probabilistic approach for web service discovery, in *Proceedings of IEEE International Conference on Services Computing (SCC)*, Santa Clara, CA, USA, 2013, pp. 49–56.

[13] S. Meng, W. Dou, X. Zhang, and J. Chen, KASR: A keyword-aware service recommendation method on mapreduce for big data application, *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3221–3231, 2014.

[14] W. Ahmed, Y. Wu, and W. Zheng, Response time based optimal web service selection, *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 551–561, 2013.

[15] Z. Zheng, H. Ma, M. R. Lyu, and I. King, Collaborative web service QoS prediction via neighborhood integrated matrix factorization, *IEEE Transactions on Services Computing*, vol. 6, pp. 289–299, 2013.

[16] M. Tang, Y. Jiang, J. Liu, and X. Liu, Location-aware collaborative filtering for QoS-based service recommendation, in *Proceedings of IEEE 19th International Conference on Web Services (ICWS)*, Honolulu, HI, USA, 2012, pp. 202–209.

[17] C. Xiang, P. Yang, X. Wu, H. He, and S. Xiao, QoS-based service selection with lightweight description for large-scale service-oriented internet of things, *Tsinghua Science & Technology*, vol. 20, no. 4, pp. 336–347, 2015.

[18] B. Xu, S. Luo, and Y. Yan, Efficient composition of semantic web services with end-to-end QoS optimization, *Tsinghua Science & Technology*, vol. 15, no. 6, pp. 678–686, 2010.

[19] Y. Zhang, G. Cui, Y. Wang, X. Guo, and S. Zhao, An optimization algorithm for service composition based on an improved FOA, *Tsinghua Science & Technology*, vol. 20, no. 1, pp. 90–99, 2015.

[20] B. Cao, J. Liu, M. Tang, Z. Zheng, and G. Wang, Mashup service recommendation based on user interest and social network, in *Proceedings of IEEE 20th International Conference on Web Services (ICWS)*, Santa Clara, CA, USA, 2013, pp. 99–106.

[21] W. Xu, J. Cao, L. Hu, J. Wang, and M. Li, A social-aware service recommendation approach for mashup creation, in *Proceedings of IEEE 20th International Conference on Web Services (ICWS)*, Santa Clara, CA, USA, 2013, pp. 107–114.

Yushun Fan received the PhD degree in control theory and application from Tsinghua University, China, in 1990. He is currently a professor with Department of Automation. His research interests include enterprise modeling methods and optimization analysis, business process reengineering, workflow management, system integration, object-oriented technologies and flexible software systems, Petri nets modeling and analysis, and workshop management and control.

Cheng Wu received the BS and PhD degrees in control theory and application from Tsinghua University, China, in 1962 and 1966, respectively. He is currently a fellow of Chinese Academy of Engineering. His research interests include complex system modeling and optimization, and modeling and scheduling in supply chains.
Bofei Xia received the BS degree in 2011 from Dalian University of Technology. He is currently a PhD candidate at Department of Automation of Tsinghua University, China. His research interests include services computing, service recommendation, and big data.

Bing Bai received the BS degree in 2013 from Tsinghua University, China. He is currently working toward the PhD degree at Department of Automation, Tsinghua University. His research interests include services computing, service recommendation, and big data.

Junqi Zhang received the BS degree from Tsinghua University, in 2015. He is currently working towards the PhD degree at Department of Automation, Tsinghua University. His research interests include service recommendation.