Knowl Inf Syst (2014) 39:153–173
DOI 10.1007/s10115-013-0703-1

REGULAR PAPER

Modeling and exploiting tag relevance for Web service mining

**Liang Chen · Jian Wu · Zibin Zheng ·
Michael R. Lyu · Zhaohui Wu**

Received: 29 October 2012 / Revised: 1 August 2013 / Accepted: 15 October 2013 /
Published online: 12 November 2013

Abstract Web service tags, i.e., terms annotated by users to describe the functionality or other aspects of Web services, are being treated as collective user knowledge for Web service mining. Since user tagging is inherently uncontrolled, ambiguous, and overly personalized, a critical and fundamental problem is how to measure the relevance of a user-contributed tag with respect to the functionality of the annotated Web service. In this paper, we propose a hybrid mechanism by using Web Service Description Language documents and service-tag network information to compute the relevance scores of tags by employing semantic computation and Hyperlink-Induced Topic Search model, respectively. Further, we introduce tag relevance measurement mechanism into three applications of Web service mining: (1) Web service clustering; (2) Web service tag recommendation; and (3) tag-based Web service retrieval. To evaluate the accuracy of tag relevance measurement and its impact to Web service mining, experiments are implemented based on *Titan* which is a Web service search engine constructed based on 15,968 real Web services. Comprehensive experiments demonstrate the effectiveness of the proposed tag relevance measurement mechanism and its active promotion to the usage of tagging data in Web service mining.

Keywords Web service · Tag · Relevance · Service clustering · HITS

L. Chen · J. Wu (✉) · Z. Wu
College of Computer Science, Zhejiang University, Hangzhou, China
e-mail: wujian2000@zju.edu.cn

L. Chen
e-mail: cliang@zju.edu.cn

Z. Wu
e-mail: wzh@zju.edu.cn

Z. Zheng · M. R. Lyu
Department of Computer Science and Engineering, The Chinese University of Hong Kong,
Hong Kong, China
e-mail: zbzheng@cse.cuhk.edu.hk

M. R. Lyu
e-mail: lyu@cse.cuhk.edu.hk

1 Introduction

Web service[1] has become an important paradigm for developing Web applications. In particular, the emergence of cloud infrastructure offers a powerful and economical platform to greatly facilitate the development and deployment of a large number of Web services. Based on the most recent statistics,[2] there are 28,593 Web services being provided by 7,728 distinct providers over the world and these numbers keep increasing in a fast rate.

Web Service Description Language (WSDL) documents and extra descriptions given by service providers are two major kinds of data to be utilized for Web services mining [1]. Despite the abundance of extra service description for most current Web services, limited information can be obtained from the XML-based description document, i.e., WSDL document. The fast growing number of Web services but the limited information can be obtained pose significant challenges to Web service mining, e.g., Web service clustering and Web service searching.

In recent years, tagging, the act of adding keywords (tags) to objects, has become a popular mean to annotate various Web resources, e.g., Web page bookmarks, online documents, and multimedia objects. Tags provide meaningful descriptions of objects and allow users to organize and index their contents. Tagging data were proved to be very useful in many domains such as multimedia, information retrieval, and data mining [2,3]. In Web service domain, some Web service search engines, such as *SeekDa!*, also allow users to annotate tags to Web services. Recently, Web service tags attract much attention and are being treated as collective user knowledge to fill the gap between fast growing Web services and limited information about them. Some studies have been conducted to employ tagging data for Web service clustering [4,5], Web service discovery [6,7], Web service composition [8], etc.

However, existing studies reveal that many tags provided by Social Network System (SNS) users are imprecise and there are only around 50 % tags actually related to the target object [9]. This is not surprising because of the uncontrolled nature of social tagging and the diversity of knowledge and cultural backgrounds of the users. Apart from the fact that tags can be subjective, irrelevant or even malicious, they may also be annotated to Web services by attackers, which seriously limits the effectiveness of tagging data in Web service mining. Hence, a critical and fundamental problem for utilizing tagging data in Web service mining is how to accurately measure the relevance of a tag with respect to the annotated Web service.

Generally, the relevance levels of tags cannot be distinguished from the order of current tag list, where tags are basically listed in a random order or chronological order without considering the relevance information. Figure 1 shows two exemplary Web services[3] from *SeekDa!* and their tags annotated by users. Take the *USWeather* Web service as an example, its most relevant tag, i.e., "weather", cannot be discovered from the order of tag list directly. Similarly, the most relevant tag to *XigniteQuotes* Web service is "stock quote", while its position in the tag list is the 7th. Furthermore, there are some imprecise tags annotated to Web services, such as "unknown", and "format".

To further investigate the position distribution of the most relevant tags in the tag list, we select 180 real Web services (each Web service has 5 or more tags) from *SeekDa!* search

[1] In this paper, we focus on non-semantic Web services. Non-semantic Web services are described by WSDL documents while semantic Web services use Web ontology languages (OWL-S) or Web Service Modeling Ontology (WSMO) as a description language. Non-semantic Web services are widely supported by both the industry and development tools.

[2] Statistics obtained from *SeekDa!* (a Web service search engine), http://webservices.seekda.com.

[3] ***USWeather***'s WSDL Address: http://webservices.seekda.com/providers/webservicex.net/USWeather
XigniteQuotes's WSDL Address: http://webservices.seekda.com/providers/xignite.com/XigniteQuotes.

Fig. 1 Two exemplary Web services from *SeekDa!*

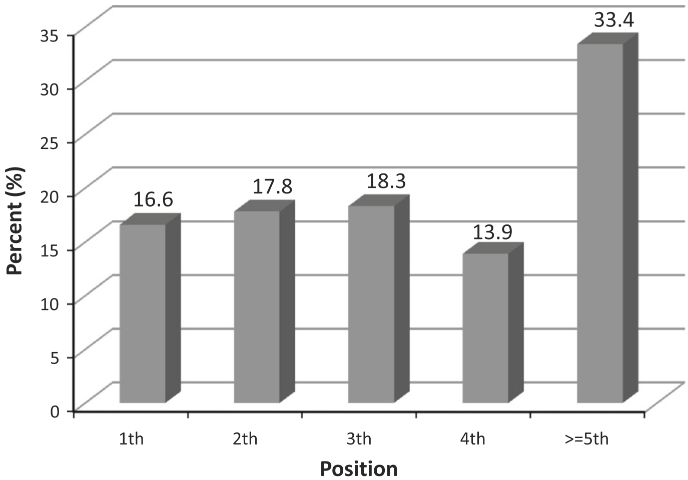


Fig. 2 Position distribution of the most relevant tag in the tag list

engine, and the most relevant tag identification is completed by volunteers. From Fig. 2, it can be observed that only 16.6 % Web services have their most relevant tags at the first position of the annotated tag list, while more than 33 % Web services have their most relevant tags at the fifth position or even behind. The position distribution in Fig. 2 indicates that the tags are basically in a random order in terms of relevance to the associated services.

Although there have been some works about utilizing tags in Web service mining [4,6–8,10], the problem of measuring relevance of a user-contributed tag with respect to the corresponding Web service has not been considered carefully. Instead, there are some works about utilizing tagging data in multimedia domain providing valuable methods for reference. Li et al. [11] proposed a neighbor-voting-based algorithm to predict the tag relevance. The basic idea of this algorithm is that if different persons label similar objects using the same tags, then these tags are likely to reflect objective aspects of these objects. Wu et al. [12] and Sigurbjrnsson and van Zwol [3] proposed to utilized tag co-occurrence to evaluate the tag relevance. Actually, both neighbor-voting and tag co-occurrence are the kind of methods which propose to measure tag relevance by exploring the relationships in Object-Tag Network.

In this paper, we propose a hybrid Web Service Tag Relevance Measurement mechanism (named *WS-TRM*), to measure the relevance of the user-contributed tag with respect to its corresponding Web service. In *WS-TRM*, we not only consider the semantic relevance of the tag to the Web service, but also take the relationships in Service-Tag Network (*STNet*) into consideration. Specifically, we extract a content vector from the WSDL document of the annotated Web service and compare it with the tag to obtain the semantic relevance. Hyperlink-Induced Topic Search (HITS) [13] model is employed to explore the relationships in *STNet* by evaluating the authority of the tag in *STNet* which is conducted based on the whole collection of Web services and annotated tags.

To demonstrate the effectiveness of tag relevance for Web service mining, we apply *WS-TRM* into three applications:

(1) *Web services clustering* Web services clustering groups the Web services with the same or similar functionality. As tags can partially reflect the functionality of Web services, tags are recently employed for Web services clustering [4]. If the relevance of tags can be measured accurately, a better clustering result can be obtained.
(2) *Web service tag recommendation* Web service tag recommendation suggests tags to the Web services with few or even no tags. Both tag co-occurrence and tag relevance are considered for tag recommendation in this paper.
(3) *Tag-based Web service retrieval* Recently, tag-based object retrieval is quite popular, due to the organizational capability of tagging data and the development of social network. In this paper, we realize the functionality of tag-based Web service retrieval and demonstrate that its performance could be improved if the tag relevance is measured accurately.

In particular, the main contributions of this paper can be summarized as follows:

(1) It identifies the critical problem of tag relevance measurement for Web service mining and proposes a hybrid approach named *WS-TRM* in which both semantic relevance and relationships in *STNet* are considered.
(2) Extensive real-world experiments are conducted to study the performance of *WS-TRM*. Further, we apply it into three Web service mining applications and evaluate its impact on each application.
(3) We publicly release our Web service tag dataset to promote future research, which includes 15,968 real-world Web services and their tags.[4] The released dataset makes our experiment reproducible.

The rest of this paper is organized as follows: Sect. 2 gives a survey of related work in Web service mining by utilizing tag information. Section 3 gives an overview of our proposed *WS-TRM* approach. The detailed tag relevance measurement process is introduced in Sect. 4. Section 4.3 reports the performance of *WS-TRM* based on real Web services, while Sect. 5 evaluates the impact of *WS-TRM* on three Web service mining applications. Finally, Sect. 6 concludes this paper.

[4] Dataset can be downloaded from http://www.zjujason.com.

2 Related work

Web service mining [1], which combines traditional service-oriented computing (SOC) and state-of-the-art data mining techniques, is becoming a hot research direction. WSDL documents are the main information source for the research of Web service mining. Recently, tagging data, which are annotated by users and provide meaningful descriptions, are utilized as another information source for Web service mining. In the following, we introduce the existing research of tagging data for handling different problems in Web service mining.

2.1 Tab-based Web service clustering

Web service clustering, a popular research issue in Web service mining, attracts the attention of many researchers and is presented as a novel solution to handle the low recall of Web service search engine. In traditional work, vectors extracted from WSDL documents are leveraged to determine the similarities between Web services [14–16]. Chukmol et al. [17] propose a folksonomic annotation model allowing users to express their perception on service functionality (after testing or using them) for the purpose of facilitating Web service clustering. In our prior work [4], we first propose to utilize both WSDL documents and tags to cluster Web services by combining users' knowledge and service providers' knowledge. In particular, we treat tagging data an a vector and compute the similarities between Web services according to the tag vector and another 5-dimensional vector extracted from WSDL document.

2.2 Web service tag recommendation

To handle the problem of limited tags, Azmeh et al. [18] propose to employ machine learning technology and WordNet synsets to automatically annotate tags to Web services. Chen et al. [4] propose to recommend tags to the Web services with few tags according to the tag co-occurrence. Fang et al. [19] propose two tagging strategies, tag enriching and tag extraction. In the first strategy, Web services are clustered according to WSDL documents, and the enriched tags for a service are the tags of other Web services in the same cluster. In the second strategy, recommended tags are extracted from WSDL documents and related descriptions. To make services easily accessible and attractive to end users, Katakis et al. [20] propose to automate tagging services by modeling this problem as a multi-label classification problem.

2.3 Tag-based Web service discovery

With the growth of Web services, traditional web service discovery mechanisms have become inefficient because of their low precision, due to the simplicity of information source for service discovery. Ding et al. [21] propose to improve the performance of Web service discovery by introducing tagging data. In particular, the service-tag relationships are considered in the process of discovery. Fernandez et al. [22] propose a mixed service discovery model based on two main ideas. Firstly, users are encouraged to provide tags to each Web service to form tag cloud, which could be matched using standard similarity measures against user requests. Then existing service tag clouds are hierarchically clustered to achieve lightweight, browsable service ontologies, represented by discriminating tags per cluster.

2.4 Discussion

Above works promote the usage of tagging data in Web service mining by utilizing tagging data to handle kinds of problems in Web service mining. A common premise of above works is that the annotated tags are highly relevant to the corresponding Web services. However, many imprecise, irrelevant, or even malicious tags may also be annotated to Web services, which limits the effectiveness of tagging data in Web service mining and were considered carefully in the previous work. In this paper, we propose to handle this fundamental problem, i.e., tag relevance measurement, to facilitate the usage of tagging data in Web service mining.

3 Web service tag relevance measurement

In this section, we first give an overview of the proposed *WS-TRM* approach in Sect. 3.1 and then introduce the computation of semantic tag relevance and HITS based tag authority in Sects. 3.2 and 3.3, respectively. Finally, the computation of final tag relevance by integrating semantic tag relevance and tag authority is introduced in Sect. 3.4.

3.1 Overview of WS-TRM

Figure 3 presents an overview of our proposed *WS-TRM* mechanism, which mainly consists of two parts: (1) semantic relevance computation and (2) tag authority computation by using the HITS model. In particular, the example of Web service 1 in Fig. 3 is the weather report Web service mentioned in Fig. 1. Given a tag list associated with one Web service, we first compute the relevance score of each tag by evaluating the sematic relevance between each tag and the WSDL document of the corresponding service. In particular, we extract a content vector (i.e., a set of keywords) from WSDL document for semantic relevance computation between a tag and service. Although the relevance scores obtained in this way reflect the semantic relevance between tags and services, the relationships in *STNet* have not been considered. In the second part, the HITS model is employed to explore the relationships in *STNet* to compute the authorities of tags, which reflect the meaningfulness of tags. In particular, *STNet* is constructed by utilizing the association relationship between tags and Web services. Finally, the relevance score of a tag is generated by integrating semantic relevance and tag authority.

3.2 Semantic relevance computation

Web Service Description Language (WSDL) document, which describes the functionality of a Web service, is actually an XML style document. Therefore, we can use some IR

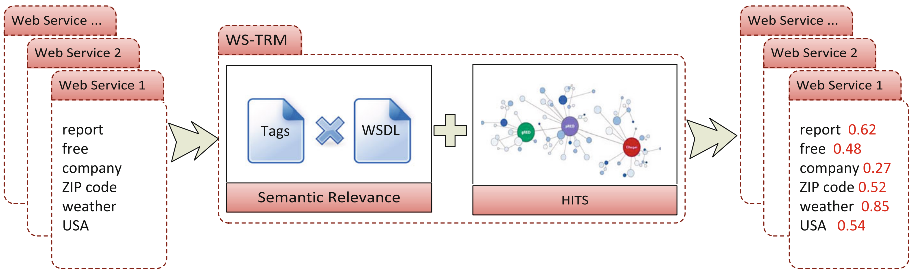


Fig. 3 An overview of Web service tag relevance measurement (WS-TRM) mechanism

approaches to extract a vector of meaningful content words which can be used as a feature for semantic relevance computation. It has been demonstrated to be effective in some previous works [4,16,23]. In this paper, we build the content vector in four steps:

(1) *Building original vector* In this step, we split the WSDL content according to the white space to produce the original content vector. For the term such as "WeatherReport," we split it into two single words "Weather" and "Report."
(2) *Suffix Stripping* Words with a common stem will usually have the same meaning, for example, *connect*, *connected*, *connecting*, *connection*, and *connections* all have the same stem *connect* [15]. For the purpose of convenient statistics, we strip the suffix of all these words that have the same stem by using a Porter stemmer [24].
(3) *Pruning* In this step, we propose to remove two kinds of words from the content vector. The first kind of words to be removed are XML tags, e.g., *s:element* and *s:complexType*, which are not meaningful for the semantic relevance computation. The second kind of words to be removed are function words which have little or no contribution to the meanings of texts. Poisson distribution is employed to model word occurrence in documents for the purpose of distinguishing function words [25]. Typically, a way to decide whether a word w in the content vector is a function word is computing the degree of overestimation of the observed document frequency of the word w, denoted by n_w using Poisson distribution. The overestimation factor can be calculated as follows.

$$\Lambda_w = \frac{\hat{n_w}}{n_w}, \tag{1}$$

where $\hat{n_w}$ is the estimated document frequency of the word w. Specifically, the word with higher value of Λ_w has higher possibility to be a content word. In this paper, we set a threshold Λ_T for Λ_w and take the words which have Λ_w higher than threshold as content words. The value of threshold Λ_T is as follows.

$$\Lambda_T = \begin{cases} avg[\Lambda] & \text{if } (avg[\Lambda] > 1) \\ 1 & \text{otherwise,} \end{cases} \tag{2}$$

where $avg[\Lambda]$ is the average value of the observed document frequency of all words considered. After the process of pruning, we can obtain a new content vector, in which both XML tags and function words are removed.
(4) *Refining* Words with very high occurrence frequency are likely to be too general to discriminate between Web services. After the step of pruning, we implement a step of refining, in which words with too general meanings are removed. Clustering-based approaches were adopted to handle this problem in some related work [15,16]. In this paper, we choose a simple approach by computing the frequencies of words in all WSDL documents and setting a threshold to decide whether a word has to be removed.

After the above 4 steps, we can obtain the final content vector. Through our observation, the dimension of the content vector of most Web services for experiments (i.e., 15,968 real Web service) is in the range of 10–30.

As mentioned above, WSDL is an XML structure document. Thus, the position of a content word takes in the XML structure should be considered in the process of semantic relevance computation. That is, the importance of content words in different positions of the structure should be discriminated. In this paper, we classify the positions of content words in an XML structure into 4 categories:

(1) *Name property* In the definition of elements or other objects (e.g., message, type, operation) in a WSDL document, there is always a name property. Take this record <s:element name="GetWeatherResponse"> as a example, the positions of "Get," "Weather," and "Response" are all *name property*.
(2) *Value property* Similar to *name property, value property* is another kind of property for an element or other objects in a WSDL document.
(3) *Text* There is always some text description for the operation in WSDL. We call this kind of position as *text*.
(4) *Annotation* At the beginning of WSDL document, there may be some annotation given by a service provider. In *annotation*, some information about the service provider or the functionality of service is presented.

In this paper, we use c_1, c_2, c_3, and c_4 to represent *name property*, *value property*, *text*, and *annotation*, respectively. And f_1, f_2, f_3, and f_4 are their corresponding weights for different position categories, $f_1 + f_2 + f_3 + f_4 = 1$. Given a content vector *content* (consists of a set of words, $w_1, \ldots, w_n$) and a tag t, the semantic relevance between t and *content* is computed as follows:

$$SR(t, content) = \frac{\sum_{i=1}^{n} Sim(t, w_i) \sum_{j=1}^{4} f_j \times Occur_{ij}}{\sum_{i=1}^{n} \sum_{j=1}^{4} f_j \times Occur_{ij}}, \tag{3}$$

where $Occur_{ij}$ means the occurrence number of word w_i in position c_j, and $Sim(t, w_i)$ means the semantic similarity between t and w_i. In this paper, Normalized Google Distance (NGD) [26] is employed to compute the semantic similarity between two words:

$$\begin{aligned} sim(t, w_i) &= 1 - NGD\,(t, w_i) \\ NGD(t, w_i) &= \frac{\max\{\log f(t), \log f(w_i)\} - \log f(t, w_i)}{\log N - \min\{\log f(t), \log f(w_i)\}}, \end{aligned} \tag{4}$$

where $f(w_i)$ denotes the number of pages containing w_i and $f(t, w_i)$ denotes the number of pages containing both t and w_i, as reported by Google. N is the total number of Web pages searched by Google.

By employing Eqs. (3) and (4), we can obtain the semantic relevance between tag t and the content vector extracted from the WSDL document of service s, and we set $SR(t, s) = SR(t, content)$ as the semantic relevance of t to s. As the number of words left in the content vector is limited after above 4 steps, the time cost for semantic relevance computation can be accepted.

3.3 STNet-adapted HITS

Hyperlink-Induced Topic Search (HITS, also known as hubs and authorities) is a link analysis algorithm that rates Web pages, developed by Kleinberg [27]. It is a precursor to PageRank. The idea behind HITS stemmed from a particular insight into the creation of Web pages when the Internet was originally forming. Compared with PageRank, the authority value computed by HITS algorithm is more appropriate to reflect the importance of tag, while the meaning of the value computed by PageRank is more general. Thus, we propose to obtain the authority of a tag based on the *STNet*, which could reflect the importance of a tag. In the following, we first introduce how to build *STNet* and then present a *STNet*-adapted HITS algorithm for tag authority computation.

3.3.1 STNet building

Service-Tag Network (*STNet*) can be modeled as a weighted directed graph G, where node s_i means a service and node t_i means a tag. For each node in G, it has two values, i.e., hub and authority. There are three kinds of directed edges in G:

(1) Edge from a service node to tag node. Given a service s_1 annotated with three tags t_1, t_2, and t_3, then there is a directed edge from s_1 to t_1, t_2, and t_3, respectively. In particular, the weight of this kind of edge is 1.
(2) Edge from a service node to service node. Given two services s_1 and s_2, if there is one or more than one common tags annotated to these two services, we create one directed edge from s_1 to s_2 and one directed edge from s_2 to s_1. These two edges have the same weight, which depends on the common tags, i.e., $w(s_1, s_2) = w(s_2, s_1) = \frac{|t_{s_1} \bigcap t_{s_2}|}{|t_{s_1} \bigcup t_{s_2}|}$, where t_{s_1} and t_{s_2} mean the set of tags annotated to s_1 and s_2, respectively.
(3) Edge from a tag node to tag node. Given two tags t_1 and t_2, these two tags are annotated to one or more than one services. Similarly, we create one directed edge from t_1 to t_2 and one directed edge from t_2 to t_1. The weight of edge also depends on the common services, i.e., $w(t_1, t_2) = w(t_2, t_1) = \frac{|s_{t_1} \bigcap s_{t_2}|}{|s_{t_1} \bigcup s_{t_2}|}$, where s_{t_1} and s_{t_2} mean the set of services containing t_1 and t_2, respectively.

In this way, we obtain *STNet* by building a weighted directed graph. It should be noted that the reputation of taggers and Web services will be helpful to make the weights of edges more accurate. However, these kinds of data cannot be crawled so far.

3.3.2 Tag authority computation

Hyperlink-Induced Topic Search (HITS) is a kind of iterative algorithm. We consider two types of updates as follows:

- *Authority Update* For each node p (could be service node or tag node) in G, we update the authority of node p to be:

$$Auth(p) = \sum_{i=1}^{n} Hub(p_i) \times w(p_i, p), \tag{5}$$

where $p_i (i = 1, \ldots, n)$ means the node that points to p, and $w(p_i, p)$ is the weight of edge from p_i to p. That is, the authority of node p is the sum of all the weighted hub values of nodes that point to p.
- *Hub Update* For each node p in G, we update the hub value of p to be:

$$Hub(p) = \sum_{i=1}^{n} Auth(p_i) \times w(p, p_i), \tag{6}$$

where $p_i (i = 1, \ldots, n)$ means the node that p points to, and $w(p, p_i)$ means the weight of edge from p to p_i.

Algorithm 1 shows the detailed *STNet*-adapted HITS computation process. As the initialization, we set the authority value and hub value of each node in G as 1 (lines 1–3). K in line 4 means the number of iterations. Empirically, we set $K = 50$ in the experiments. The parameter $norm$ is used for normalization and is initialized as 0 (line 5). According to the Authority Update rule, we compute the authorities of all nodes in G and then normalize them by using parameter *norm* (lines 6–16). Similarly, hub values of nodes can be computed by employing Hub Authority rule (lines 18–29). After K iterations, we return the authorities of all tag nodes (lines 30–32).

Algorithm 1 *STNet*-Adapted HITS Algorithm

```
Input: G: STNet; K: number of iterations
Output: Auth(t): authority of tag node
1: for all node p in G do
2:   Auth(p)=1,Hub(p)=1
3: end for
4: for iteration from 1 to K do do
5:   norm=0
6:   for all node p in G do
7:     Auth(p)=0
8:     for all node p_i which points to p do
9:       Auth(p)+=Hub(p_i) ×weight(p_i, p)
10:    end for
11:    norm+=square(Auth(p))
12:  end for
13:  norm=sqrt(norm)
14:  for all node p in G do
15:    Auth(p)=Auth(p)/norm
16:  end for
17:  norm=0
18:  for all node p in G do
19:    Hub(p)=0
20:    for all node p_i that p points to do
21:      Hub(p)+=Auth(p_i) ×weight(p, p_i)
22:    end for
23:    norm+=square(Hub(p))
24:  end for
25:  norm=sqrt(norm)
26:  for all node p in G do
27:    Hub(p)=Hub(p)/norm
28:  end for
29: end for
30: for all tag node t in G do
31:   return Auth(t)
32: end for
```

3.4 Relevance integration

Semantic Relevance score $SR(t, s)$ obtained in Sect. 3.2 reflects the semantic relevance between tag t and service s, while the authority of tag $Auth(t)$ obtained in Sect. 3.3 reflects the meaningfulness of tag t in the whole *STNet*. In this paper, we integrate semantic relevance and tag authority to be the final relevance of user-contributed tag t with respect to service s.

Given a service s with a set of tag T annotated to it, the relevance score of each tag $t \in T$ is computed as follows:

$$Score(t, s) = (1 - \lambda)SR(t, s) + \lambda Auth(t), \tag{7}$$

where λ is the weight of tag authority. The range of λ is [0,1]. Specifically, *WS-TRM* only considers the semantic relevance of t to s when $\lambda = 0$, while *WS-TRM* ranks tags only according to the tag authority in *STNet* when $\lambda = 1$.

4 Experiment

In this section, we compare the performance of different tag ranking approaches and evaluate the impact of λ on the performance of *WS-TRM*.

4.1 Dataset description & experiment setup

To evaluate the performance of *WS-TRM*, we crawl 15,968 real Web services from the Web service search engine Seekda!. For each Web service, we crawl the information of its service name, WSDL document, tags, availability, and the name of its provider. We have published this dataset, downloadable from http://www.zjujason.com

For each service, each of its tags is labeled as one of the five levels: Most Relevant (score 5), Relevant (score 4), Partially Relevant (score 3), Weakly Relevant (score 2), and Irrelevant (score 1). As the manual creation of ground truth costs much work, we select 240 Web services from the dataset and distinguish them into the following categories: "Email", "Stock", "Tourism", "Weather", "Communication", and "Finance". Specifically, there are 31 Web services in "Email"category, 39 Web services in "Stock"category, 39 Web services in "Tourism"category, 42 Web services in "Weather"category, 37 Web services in "Communication"category, and 52 Web services in "Finance"category. Due to the space limitation, we do not show their detailed information.

All experiments are implemented with JDK 1.6.0-21, Eclipse 3.6.0 and conducted on a Dell Inspire R13 machine with an *2.27GHZ Intel Core I5 CPU* and *6GB RAM*, running *Windows7 OS*.

4.2 Evaluation metric

To evaluate the performance of Web service tag relevance measurement, we treat it as a ranking problem and evaluate the performance of *WS-TRM* in an indirect manner: First, given a Web service s_1, its associated tags are ranked according to the relevance score computed by *WS-TRM*; second, comparing the tag list ranked by *WS-TRM* with the tag list ranked by the manually labeled ground truth, we employ the Normalized Discounted Cumulative Gain (NDCG) [28] metric, which is widely accepted as the metric for ranking evaluation in information retrieval. Given the ideal tag ranking of target service (used as ground truth) and a predicted tag ranking, the NDCG value of the Top-k ranked tag can be calculated by:

$$NDCG@k = \frac{DCG@k}{IDCG@k}, \tag{8}$$

where $DCG@k$ and $IDCG@k$ are the discounted cumulative gain (DCG) values of the Top-K tags of the predicted ranking and ideal ranking, respectively. The value of $DCG@k$ can be calculated by:

$$DCG@k = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{\log_2 i}, \tag{9}$$

where rel_i is the graded relevance score of the tag at position i of the ranking. The DCG value is accumulated from the top of the ranking to the bottom with the gain of each result discounted at lower ranks. The ideal rank achieves the highest gain among different rankings. The $NDCG@k$ value is in the interval of 0–1, where a larger value stands for better ranking accuracy, indicating that the predicted ranking is closer to the ideal ranking. The value of k is in the interval of 1 to n, where n is the total number of tags.

Table 1 NDCG@K performance of Web service tag relevance measuring approaches

NDCG@K	Method	Tourism	Weather	Commu	Finance	Stock	Email	Average
$K=3$	Baseline	0.723	0.831	0.741	0.637	0.781	0.839	0.759
	Semantic	0.793	0.878	0.912	0.869	0.921	0.934	0.884
	HITS	0.823	0.931	0.852	0.791	0.913	0.893	0.867
	WS-TRM	0.863	0.956	0.962	0.893	0.941	0.952	0.928
$K=5$	Baseline	0.705	0.863	0.725	0.747	0.841	0.852	0.789
	Semantic	0.805	0.912	0.905	0.913	0.947	0.936	0.903
	HITS	0.794	0.908	0.913	0.828	0.913	0.894	0.875
	WS-TRM	0.841	0.965	0.958	0.931	0.967	0.959	0.937

4.3 Performance evaluation of WS-TRM

To study the performance of Web service tag relevance measurement, we first compute the NDCG value of *Baseline* (i.e., original tag lists) and then compare the performance of the following three approaches:

- *Semantic* In this approach, semantic relevance between tag and service is employed to rank tags, i.e., only Eqs. (3) and (4) are employed.
- *HITS* In this approach, linking relationship in *STNet* is employed to rank tags. In this experiment, we choose the HITS model to represent the linking relationship.
- *WS-TRM* Both semantic relevance and linking relationship are employed in *WS-TRM*, while λ is used to balance the importance of two components.

Table 1 shows the ranking performance of the above 4 approaches, respectively employing NDCG@3 and NDCG@5 as the evaluation metric. NDCG@k indicates that only the ranking accuracy of the top-k tags is investigated. Given one category of Web services, we compute the NDCG@k value of each Web service and set the average value as the NDCG@k value of this category. Empirically, we set $f_1 = f_2 = 0.3$, $f_3 = f_4 = 0.2$, and $\lambda = 0.1$ in *WS-TRM* in this experiment. The impact of λ on the performance of *WS-TRM* will be evaluated in Sect. 4.4. For each column in Table 1, we have highlighted the best performer among all approaches.

From Table 1, it can be observed that all three tag ranking approaches largely improve the accuracy of tag ranking, that is, the tag relevance computed by these three approaches is more accurate than the tag relevance obtained from the order of the original tag list. Compared with the *Baseline*, the improvement brought by *WS-TRM* achieves 40.2 % at the highest point and achieves 11.8 % in the worst case.

Among these three approaches, the performance of *WS-TRM* is the best, while the performance of *HITS* is the worst in most cases. This is because it utilizes only the linking relationship in *STNet*, which reflects the authority of tags but can only partially represent the relevance between a tag and service.

4.4 Impact of λ

As λ is used to balance the importance of semantic relevance and linking relationship, the choice of λ value greatly influences the performance of *WS-TRM*. In this section, we try to find the optimal value of λ by evaluating the impact of λ on *WS-TRM*.

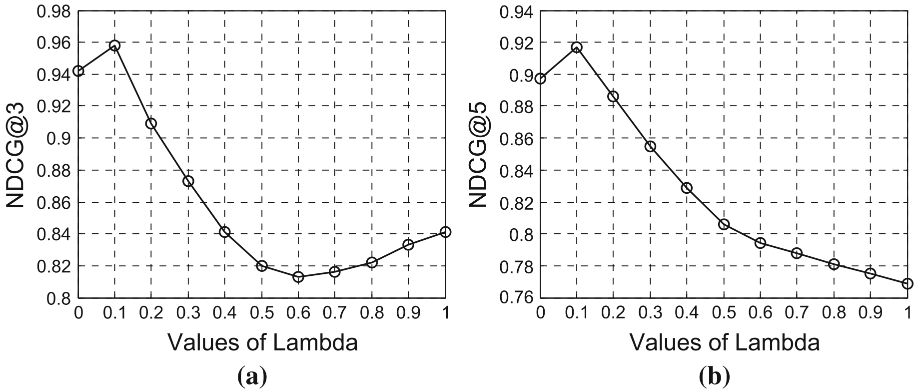


Fig. 4 Impact of λ to the performance of WS-TRM. **a** $k = 3$, **b** $k = 5$

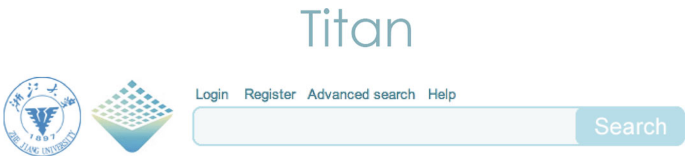


Fig. 5 Site home page of Titan search engine

Figure 4a, b show the impact of λ on *WS-TRM* with the metric NDCG@3 and NDCG@5, respectively. Specifically, NDCG@k in these two figures is the average one in all categories. From Fig. 4, we can find that the performance of the *Semantic* approach is better than *HITS*, as the value of NDCG@k ($\lambda = 0$) is higher than NDCG@k ($\lambda = 1$). By observing the trend of curves, we can also find the performance of *WS-TRM* in both two figures first increases and then decreases with the increase of λ, and it achieves the highest point when $\lambda = 0.1$. Therefore, the optimal value of λ is 0.1.

5 Application in Web service mining

In this section, we apply *WS-TRM* into three real applications of Web service mining (i.e., Web service clustering, Web service tag recommendation, and tag-based Web service retrieval) to evaluate the impact of *WS-TRM* in Sects. 5.1, 5.2, and 5.3, respectively. In particular, all these evaluations are implemented based on *Titan search engine*[5] [29], and above three applications have been realized and embedded into *Titan* (Fig. 5).

5.1 Web service clustering

Recently, Web service clustering is employed to handle the low recall of Web service search engine, which is caused by the keyword matching [15, 16]. In their opinion, if Web services with similar functionality are placed into the same cluster, more relevant Web services could be retrieved in the search result. In our prior work [4], a hybrid approach by utilizing both

[5] Titan is constructed based on 15,968 real Web services, and it has been accepted by WWW 2012 Demo Track. Link to Titan: http://ccnt.zju.edu.cn:8080.

WSDL documents and tags to cluster Web services is proposed and outperforms the previous clustering approaches, in which only WSDL documents are utilized. Specifically, given two Web services s_1 and s_2, not only the similarity between the WSDL of s_1 and the one of s_2 [i.e., $Sim_{wsdl}(s_1, s_2)$] is considered, but also the similarity between the tags of s_1 and the ones of s_2 [i.e., $Sim_{tag}(s_1, s_2)$] is considered. The detailed process of Web service clustering can be found in [4].

However, the relevance of user-contributed tags with respect to the Web services has not been considered in [4], that is, the tags associated with Web services were all treated as totally relevant, which may limit or even bring negative effect on the performance of tagging data in Web service clustering. In this paper, we propose to employ *WS-TRM* to obtain tag relevance scores and weight $Sim_{tag}(s_1, s_2)$ by the relevance of corresponding tags. That is, the similarity between s_1 and s_2 is generated by integrating $Sim_{wsdl}(s_1, s_2)$ and weighted $Sim_{tag}(s_1, s_2)$.

To evaluate the impact of *WS-TRM* on Web service clustering, we implement two versions of clustering, one version employs *WS-TRM*, while the other one does not employ *WS-TRM*. In this experiment, we employ the six categories of Web services mentioned in Sect. 4.1 (i.e., *Weather*, *Email*, *Stock*, *Tourism*, *Finance*, and *Communication*) to do Web service clustering. To evaluate the performance of Web service clustering, we introduce two metrics (precision and recall), which are widely adopted in the Information Retrieval domain.

$$Precision_{c_i} = \frac{succ(c_i)}{succ(c_i) + mispl(c_i)}, \tag{10}$$

$$Recall_{c_i} = \frac{succ(c_i)}{succ(c_i) + missed(c_i)} \tag{11}$$

where $succ(c_i)$ is the number of services successfully placed into cluster c_i, $mispl(c_i)$ is the number of services that are incorrectly placed into cluster c_i, and $missed(c_i)$ is the number of services that should be placed into c_i but are placed into another cluster.

Figure 6 shows the performance comparison of above two versions of Web service clustering. From Fig. 6, we can observe that *Clustering with WS-TRM* outperforms *Clustering without WS-TRM* in both precision and recall. Specifically, the average improvement caused by the employment of *WS-TRM* achieves 16 % in terms of precision, and 10 % in terms of recall. As we discussed above, the neglect of tag relevance limits or even brings negative

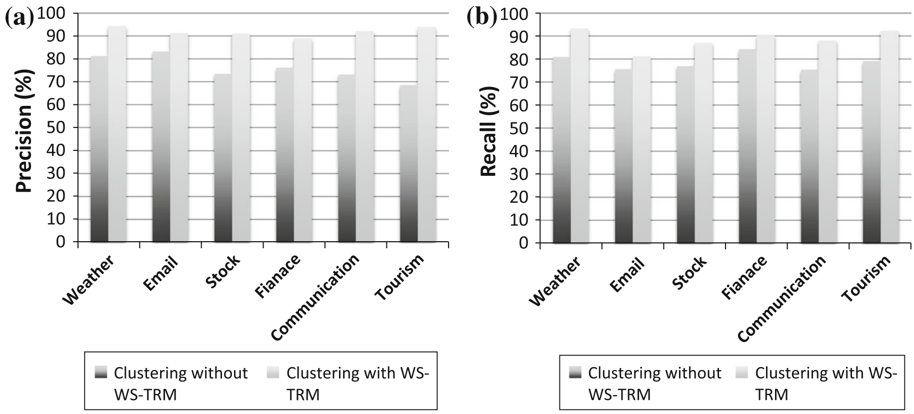


Fig. 6 Impact of WS-TRM to the performance of Web service clustering

effect on the performance of tagging data. Results in Fig. 6 demonstrate that the employment of tag relevance facilitates Web service clustering.

5.2 Web service tag recommendation

Similar to the multimedia tagging and document tagging, some inherent properties in Web service tagging, e.g., uneven tag distribution, influence the effectiveness of tagging data in Web service mining. This property is easy to be understood because tagging is a kind of user behavior. Hot Web services are usually annotated with lots of tags, while less popular Web services may be annotated with few or even no tags.

Tag recommendation technique is a widely accepted approach to handle this problem. *Vote* and *Sum* are two classical tag recommendation approaches, in which tag co-occurrence is utilized to compute a score for each candidate tag and the top-*K* tags with the highest scores are selected as the recommended tags. Details about *Vote* and *Sum* can be found in [3]. In this paper, we utilize the proposed *WS-TRM* to improve their performance by considering both tag relevance and co-occurrence in the process of Web service tag recommendation. In particular, for a candidate tag t, the weighted average value of the normalized tag relevance $TR(t)$ and the normalized tag co-occurrence score $TC(t)$ are utilized for tag recommendation. To evaluate the impact of tag relevance, the following approaches are implemented:

- *Sum* In this approach, tag co-occurrence score $TC(t)$, which is computed by using the *Sum* strategy, is utilized as the metric for tag recommendation.
- *Vote* In this approach, $TC(t)$ is also employed as the metric for tag recommendation, while it is computed by using the $Vote$ strategy.
- *Sum*$^+$ In this approach, the tag relevance value $TR(t)$ is introduced to improve the performance of *Sum*.
- *Vote*$^+$ In this approach, the tag relevance value $TR(t)$ is employed to improve the performance of *Vote*.

Before evaluating the performance of tag recommendation, we select 1,800 web services which contain 1,254 unique tags as the dataset for evaluation. The ground truth is manually created through a blind review pooling method, where for each of the 1,800 web services, the top 10 recommendations from each of the two strategies are taken to construct the pool. The volunteers are then asked to evaluate the descriptiveness of each of the recommended tags in context of the web services. We provide the WSDL documents and web service descriptions to volunteers to help them. The volunteers are asked to judge the descriptiveness on a three-point scale: *very good*, *good, not good*. The distinction between *very good* and *good* is defined to make the assessment task conceptually easier for the user. Finally, we receive 212 *very good* judgements (16.9 %), 298 *good* judgements (23.7 %), and 744 *not good* judgements (59.4 %).

To evaluate the performance of Web service tag recommendation, we adopt two metrics which capture the performance at different aspects:

- *Success at rank K (S@K)* The success at rank *K* is defined as the percentage of *good* or *very good* tags take in the top *K* recommended tags, averaged over all judged web services.
- *Precision at rank K (P@K)* Precision at rank *K* is defined as the proportion of retrieved tags that is relevant, averaged over all judged web services.

Table 2 shows the S@K comparison of four tag recommendation strategies, where the *Given Tag* means the number of tags that the target web service has. Take the *Sum* strategy as

Table 2 S@K comparison of four tag recommendation strategies

Given tag	Method	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$
1–2	Sum	0.8132	0.7081	0.6738	0.7087	0.7181
	Sum^+	0.8331	0.7129	0.7033	0.7221	0.7318
	$Vote$	0.6392	0.5949	0.6737	0.7005	0.6972
	$Vote^+$	0.6875	0.6112	0.6745	0.7143	0.7384
3–5	Sum	0.7534	0.7143	0.7380	0.6852	0.6720
	Sum^+	0.7745	0.7322	0.7449	0.7208	0.6775
	Vote	0.7867	0.6646	0.7042	0.7022	0.7103
	$Vote^+$	0.7958	0.7436	0.7323	0.7128	0.7219
>5	Sum	0.7632	0.7211	0.6944	0.6975	0.6647
	Sum^+	0.7822	0.7318	0.7098	0.7145	0.6897
	Vote	0.8136	0.7769	0.7749	0.7262	0.6973
	$Vote^+$	0.8364	0.8012	0.7943	0.7438	0.7012

Table 3 P@K comparison of four tag recommendation strategies

Given tag	Method	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$
1–2	Sum	0.6933	0.5083	0.4277	0.3788	0.3562
	Sum^+	0.7612	0.5329	0.4879	0.4374	0.4038
	Vote	0.7879	0.5495	0.4503	0.3947	0.3689
	$Vote^+$	0.7945	0.5983	0.4832	0.4329	0.3925
3–5	Sum	0.6512	0.4857	0.4171	0.3654	0.3345
	Sum^+	0.6856	0.5134	0.4658	0.3765	0.3564
	Vote	0.7415	0.5414	0.4496	0.3925	0.3494
	$Vote^+$	0.7667	0.5934	0.5092	0.4333	0.3764
>5	Sum	0.5894	0.4656	0.4365	0.3451	0.3508
	Sum^+	0.6219	0.5043	0.4754	0.3922	0.3657
	Vote	0.7148	0.5478	0.4105	0.4026	0.3658
	$Vote^+$	0.7443	0.5874	0.4459	0.4322	0.3745

example. When *Given Tag* varies from 1 to 2, the average value of S@K is over 0.7, which means that more than 70 % recommended tags have *good* or *very good* descriptiveness. From Table 2, it can be observed that the introduction of tag relevance largely improves the performance of traditional tag recommendation strategies, as the S@K values of both Sum^+ and $Vote^+$ are larger than the S@K values of the original strategies. A trend can be identified that the S@K values of all four strategies decrease with the increase of K in most cases. This is because the most relevant tags have a high probability to be included in the tag recommendation list when K is small, and some irrelevant tags may also be included in the top-k recommendation list when K is large.

Table 3 shows the comparison of four tag recommendation strategies in terms of P@K. Similarly, it can be found that the introduction of *WS-TRM* improves the performance of tag recommendation in terms of P@K. From Table 3, one trend can be identified that the P@K values of all four strategies decrease when *Given Tag* increases. This is

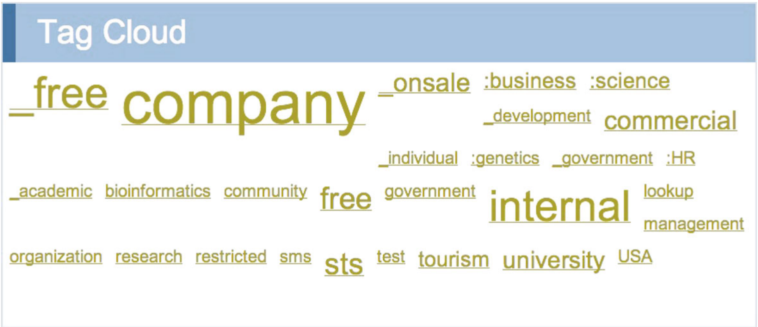


Fig. 7 Tag Cloud of Titan Web Service Search Engine

because the number of relevant tags to one certain Web service is limited. When *Given Tag* increases, the number of left relevant tags decreases, which leads to the decrease of P@K. In addition, P@K achieves its largest value when $K=1$, and decreases when K increases.

5.3 Tag-based Web service retrieval

Tagging data were recently employed to improve the performance of Web object retrieval due to the rich semantic information included in the user-contributed tags, especially in the domain of multimedia. The performance of Web service retrieval is also unsatisfied since the simplicity of information source can be utilized for service retrieval, i.e., WSDL. Intuitively, tagging data associated with Web services could be employed to improve the performance of Web service retrieval. In our prior work [4], a brief introduction to tagging data in Web service retrieval is proposed. Figure 7 shows the tag cloud of *Titan* Web service search engine, in which the most frequently annotated tags are listed and the tags with higher frequency have larger fonts.

However, if the Web service tag relevance is ignored, the employment of Web service tags may provide limited contribution or even bring negative effect on the performance of Web service retrieval. To evaluate the impact of tag relevance on the performance of Web service retrieval, we implement two versions of service retrieval, one version does not employ *WS-TRM* and treats the relevance of every tag as 1 (called as *Baseline*), while the other one employs *WS-TRM* and considers the relevance of tags in the process of service retrieval. Due to the limitation of space, we do not introduce the detailed process of service retrieval here. As for the evaluation metric, we choose *Precision at k* (P@k), which means the proportion of relevance instances in the top k retrieved results, as defined in Sect. 6.2.

Table 4 shows the results of evaluation implemented based on *Titan* Web service search engine. In Table 4, for each query, we compare the performance of *Baseline* and *WS-TRM* in terms of P@5 and P@20. From Table 4, it can be discovered that *WS-TRM* largely outperforms *Baseline* in most cases, in terms of P@5 and P@20. This is because some user-contributed tags are imprecise, ambiguous, or even irrelevant. In *Baseline*, all associated tags are treated as totally relevant, which limits the performance of tagging data in Web service retrieval. On the other hand, by employing *WS-TRM*, the effect of these imprecise, ambiguous, irrelevant tags are weakened in the process of tag-based Web service retrieval.

Table 4 P@K Performance of Web service retrieval

Query	Precision at 5		Precision at 20	
	Baseline	WS-TRM	Baseline	WS-TRM
Weather	0.800	**1.000**	0.650	**0.900**
Sms	0.800	**1.000**	0.700	**1.000**
Tourism	0.400	**0.800**	0.500	**0.650**
Stock	**0.800**	**0.800**	0.750	**0.900**
ZIP	0.600	**1.000**	0.800	**1.000**
Location	0.400	**0.800**	0.550	**0.750**
Commercial	**0.800**	**0.800**	0.650	**0.850**
Bioinformatics	0.400	**0.600**	0.500	**0.750**
University	0.600	**1.000**	0.650	**0.900**
Average	0.640	**0.840**	0.645	**0.845**

The better performance are in bold

6 Conclusion and future work

In this paper, we propose to handle a fundamental problem, i.e., tag-service relevance measurement, for the purpose of promoting the usage of tagging data in Web services mining. In our proposed *WS-TRM* approach, we not only consider the semantic relevance between WSDL documents and tags, but also take the linking relationships in service-tag network into account. In particular, content feature is extracted from WSDL documents for semantic relevance computation, while HITS is employed to compute the authorities of tags in a service-tag network. The experimental results based on real Web services demonstrate the advantage of *WS-TRM*.

To demonstrate the effectiveness of tag relevance measurement, we employ *WS-TRM* tagging data into three real applications of Web service mining, i.e., clustering, tag recommendation, and tab-based Web service retrieval. Evaluations are implemented based on *Titan* search engine, and above three applications have been all realized and embedded into *Titan*. Experimental results show that the use of *WS-TRM* really improves the effectiveness of tagging data in Web service mining.

So far, the scale of Web service tag dataset is still small, which limits the tag-related research in Web service mining. In our future work, we plan to expand the scale of tag dataset by inviting volunteers and employing automated tagging approaches, for the purpose of promoting the usage of tagging data in Web service mining. With the development of the proposed Titan search engine, more tagging data and user feedback will be collected for further experimental evaluation. Further, PageRank algorithm will be employed in WS-TRM to compare with HITS algorithm.

Social information (e.g., User's social relationship) and location information could be utilized to improve the performance of personalized recommendation, which has been demonstrated in some other domains. In our future work, we will try to utilize social information and location information to facilitate personalized Web service recommendation.

Acknowledgments This research was partially supported by the National Technology Support Program under Grant No. 2011BAH16B04, the National Natural Science Foundation of China under Grant No. 61173176, National High-Tech Research and Development Plan of China under Grant No. 2013AA01A604, the Shenzhen Basic Research Program (Project No. JCYJ20120619153834216, JC201104220300A), National

Key Science and Technology Research Program of China (2009ZX01043-003- 003), and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 415311 of General Research Fund).

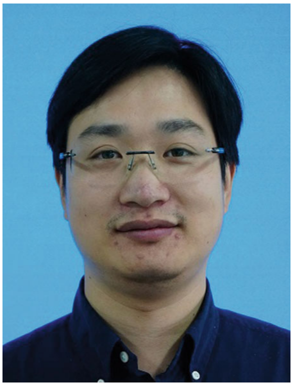
References

1. George Z, Athman B (2010) Web service mining. Springer, Berlin
2. Ames M, Naaman M (2007) Why we tag: motivations for annotation in mobile and online media. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI), pp 71–980
3. Sigurbjrnsson B, van Zwol R (2008) Flickr tag recommendation based on collective knowledge. In: Proceedings of the 17th international conference on World Wide Web (WWW), pp 327–336
4. Chen L, Hu L, Zheng Z, Wu J (2011) Wtcluster: utilizing tags for web services clustering. In: Proceedings of the 9th international conference on service oriented computing (ICSOC), pp 204–218
5. Wu J, Chen L, Zheng Z, Lyu MR, Wu Z (2013) Clustering web services to facilitate service discovery. Int J Knowl Inf Syst (KAIS), p. to appear
6. Averbakh A, Krause D, Skoutas D (2009) Exploiting user feedback to improve semantic web service discovery. In: International semantic web conference (ISWC), pp 33–48
7. Hou J, Zhang J, Nayak R, Bose A (2012) Semantics-based web service discovery using information retrieval techniques. In: Comparative evaluation of focused retrieval, pp 336–346
8. Bouillet E, Feblowitz M, Feng H, Liu Z, Ranganathan A, Riabov A (2008) A folksonomy-based model of web services for discovery and automatic composition. In: IEEE international conference on services computing, pp 389–396
9. Kennedy LS, Chang S-F, Kozintsev IV (2006), To search or to label?: predicting the performance of search-based automatic image classifiers. In: Proceedings of the 8th ACM international workshop on multimedia information retrieval, pp 249–258
10. Chen L, Zheng Z, Feng Y, Wu J, Lyu MR (2012), Wstrank: ranking tags to facilitate web service mining. In: Proceedings of the 10th international conference on service oriented computing, pp 574–581
11. Li X, Snoek CG, Worring M (2008), Learning tag relevance by neighbor voting for social image retrieval. In: Proceedings of the 1st ACM international conference on multimedia information retrieval, pp 180–187
12. Wu L, Yang L, Yu N (2009) Learning to tag. In: Proceedings of the 18st international conference companion on World Wide Web (WWW), pp 361–370
13. Li L, Shang Y, Zhang W (2002) Improvement of hits-based algorithms on web documents. In: Proceedings of the 11th international World Wide Web conference, pp 527–535
14. Dong X, Halevy A, Madhavan J, Nemes E, Zhang J (2004) Similarity search for web services. In: International conference on very large databases, pp 372–383
15. Liu W, Wong W (2009) Web service clustering using text mining techniques. Int J Agent-Oriented Softw Eng 3(1):6–26
16. Elgazzar K, Hassan AE, Martin P (2009) Clustering wsdl documents to bootstrap the discovery of web services. In: International conference on web services, pp 147–154
17. Chukmol U, Benharkat A-N, Amghar Y (2011) Bringing socialized semantics into web services based on user-centric collaborative tagging and usage experience. In: IEEE Asia-Pacific services computing conference, pp 450–455
18. Azmeh Z, My Falleri J-R, Huchard M, Tibermacine C (2011) Automatic web service tagging using machine learning and wordnet synsets. Lecture notes in business information processing 75(1):46–59
19. Fang L, Wang L, Li M, Zhao J, Zou Y, Shao L (2012) Towards automatic tagging for web services. In: 19th IEEE international conference on web services, pp 528–535
20. Katakis I, Pallis G, Dikaiakos MD, Onoufriou O (2012), Automated tagging for the retrieval of software resources in grid and cloud infrastructures In: 12th IEEE/ACM international symposium on cluster, cloud and grid computing, pp 628–635
21. Ding Z, Lei D, Yan J, Bin Z, Lun A (2010) A web service discovery method based on tag. In: International conference on complex, intelligent and software intensive systems, pp 404–408
22. Fernandez A, Hayes C, Loutas N, Peristeras V (2008) Closing the service discovery gap by collaborative tagging and clustering techniques. In: International workshop on service matchmaking and resource retrieval in the Semantic Web, pp 115–128
23. Nayak R (2008) Data mining in web service discovery and monitoring. Int J Web Serv Res 5(1):62–80
24. Porter MF (1980) An algorithm for suffix stripping. Program 14(3):130–137
25. Church K, Gale W (1995) Inverse document frequency (idf): a measure of deviations from poisson. In: Proceedings of the ACL 3rd workshop on very large corpora, pp 121–130
26. Cilibrasi RL, Vitnyi PMB (2007) The google similarity distance. IEEE Trans Knowl Data Eng 19(3):370–383

27. Kleinberg JM (1999) Hubs, authorities, and communities. ACM Comput Surv 31(4):1–3
28. Arvelin KJ, Kekalainen J (2002) Cumulated gain-based evaluation of IR techniques. ACM Trans Inf Syst 20(4):422–446
29. Wu J, Chen L, Xie Y, Zheng Z (2012) Titan: a system for effective web service discovery. In: Proceedings of the 21st international conference companion on World Wide Web (WWW), pp 441–444

Author Biographies

Liang Chen received the B.S. degree in computer science from Zhejiang University, Hangzhou, China, in 2009. He is currently working toward the Ph.D. degree in the College of Computer Science, Zhejiang University. His publications have appeared in some well-known conference proceedings and international journals. He also served as a Reviewer for some international conferences and journals (CIKM, TSMC, TSC, ICWS, etc). His research interests include service computing and data mining. He received the award of "Excellent Intern" from Microsoft Research Asia in 2010.



Jian Wu received his B.S. and Ph.D. Degrees in computer science from Zhejiang University, Hangzhou, China, in 1998 and 2004, respectively. He is currently an associate professor at the College of Computer Science, Zhejiang University, and visiting professor at University of Illinois at Urbana-Champaign. His research interests include service computing and data mining. He is the recipient of the second grade prize of the National Science Progress Award. He is currently leading some research projects supported by China National Natural Scientific Foundation and National High-tech R&D Program of China (863 Program).

Zibin Zheng is an associate research fellow at Shenzhen Research Institute, The Chinese University of Hong Kong. He received his Ph.D. degree from The Chinese University of Hong Kong in 2011. He received ACM SIGSOFT Distinguished Paper Award at ICSE'2010, Best Student Paper Award at ICWS'2010, First Runner-up Award at 2010 IEEE Hong Kong Postgraduate Research Paper Competition, and IBM Ph.D. Fellowship Award 2010-2011. He served as program committee member of IEEE CLOUD'2009, CLOUDCOMPUTING'2010, CLOUDCOMPUTING'2011, SCC'2011, etc. His research interests include service computing, cloud computing, and software reliability engineering.

Michael R. Lyu received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 1981; the M.S. degree in computer engineering from University of California, Santa Barbara, in 1985; and the Ph.D. degree in computer science from the University of California, Los Angeles, in 1988. He is currently a Professor in the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong, China. He is also Director of the Video over Internet and Wireless (VIEW) Technologies Laboratory. His research interests include software reliability engineering, distributed systems, fault-tolerant computing, mobile networks, Web technologies, multimedia information processing, and E-commerce systems. He has published over 400 refereed journal and conference papers in these areas. He has participated in more than 30 industrial projects and helped to develop many commercial systems and software tools. He was the editor of two book volumes: Software Fault Tolerance (New York: Wiley, 1995) and The Handbook of Software Reliability Engineering (New York: IEEE and New McGraw-Hill, 1996). Dr. Lyu received Best Paper Awards at ISSRE'98 and ISSRE'2003. Dr. Lyu initiated the First International Symposium on Software Reliability Engineering (ISSRE) in 1990. Dr. Lyu is an IEEE Fellow and an AAAS Fellow for his contributions to software reliability engineering and software fault tolerance. Dr. Lyu is also a Croucher Senior Research Fellow.

Zhaohui Wu received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 1993. From 1991 to 1993, he was a joint Ph.D. student in the area of knowledge representation and expert system with the German Research Center for Artificial Intelligence, Kaiserslautern, Germany. He is currently a Professor at the College of Computer Science and also the Director of the Institute of Computer System and Architecture, Zhejiang University. His research interests include intelligent transports systems, distributed artificial intelligence, semantic grid, and ubiquitous embedded systems. He is the author of four books and more than 100 referred papers. Dr. Wu is a Standing Council Member of China Computer Federation (CCF), Beijing. Since June 2005, he has been the Vice Chairman of the CCF Pervasive Computing Committee. He is also on the editorial boards of several journals and has served as a Program Chair for various international conferences.