Web Service Clustering and Discovery Mechanism Based on Topic Models (Postprint)

Authors: Li Hui, Hu Yunfeng

Date: 2017-10-11T00:00:00+00:00

Abstract

Objective: To address the massive number of Web services on the Internet, we propose an effective Web service clustering and discovery method. Method: Using BTM to learn the latent topics of the entire Web service description document collection, we infer the topic distribution of each document and perform clustering. Based on this, we create a fast Web service discovery mechanism. Results: Through comparative experiments with methods using LDA and external corpora, the precision and Normalized Discounted Cumulative Gain (NDCG) results of our method show that it can more accurately discover services that meet user requirements.

Full Text

Web Service Clustering and Discovery Mechanism Based on Topic Model

Li Hui, Hu Yunfeng
School of Economics and Management, Xidian University, Xi' an 710071

Abstract

[**Objective**] To address the massive number of Web services on the network, we propose an effective method for Web service clustering and discovery. [**Methods**] We utilize BTM (Biterm Topic Model) to learn the latent topics of the entire Web service description document collection, infer the topic distribution of each document, and perform clustering. On this basis, we create a rapid Web service discovery mechanism. [**Results**] Comparative experiments with methods using LDA and external corpora show that the proposed method achieves improvements in both precision rate and normalized discounted cumulative gain. [**Limitations**] The method only considers functional information of services and

does not incorporate quality information into the algorithm. [**Conclusion**] Experimental results demonstrate that this method can more accurately discover services that meet user requirements.

Keywords: Web service, Topic model, Clustering, Discovery

We are currently entering an era of service orientation, where SOA (Service Oriented Architecture) is widely applied, and Web services have gradually become the mainstream technology for implementing SOA. The SOA architecture follows a service pattern of discovery, binding, and execution. Web services are published by providers on private or public Internet platforms, and users discover Web services that meet their requirements from the massive number of available services, bind and invoke them to achieve their goals. In this process, users do not need to understand the implementation details of services; they only need services that can provide satisfactory execution results. As the number of services published on Internet platforms continues to grow, how to discover user-satisfactory services from the massive pool of Web services—that is, to find services that meet user expectations from published Web service descriptions—has become a critical link in realizing service-oriented architecture.

When searching for services, the lack of an effective classification mechanism makes it difficult to quickly and efficiently discover Web services. To address these problems, this paper proposes a Web clustering and discovery method utilizing BTM (Biterm Topic Model) [5]. BTM models the generation process of word pairs across the entire corpus, thereby learning the topic distribution and topic-word distribution of the whole corpus. Combined with vector space calculation of TF-IDF values for words, we can infer the topic distribution of each Web service description and subsequently cluster them. The Web service discovery process works as follows: obtain the category of the requested service; filter services within that category based on topic similarity, significantly narrowing the search scope; calculate the word vector similarity between the requested service and Web services; and combine topic similarity and word vector similarity to find a set of services that satisfy user requirements.

Web service description texts are short, feature-sparse, and contain limited information, making it infeasible to measure similarity based on word co-occurrence. Discovering Web services based on keywords, which completely relies on word co-occurrence, is highly inaccurate. To enrich Web service description texts, some semantic Web methods have been used for service discovery, such as approaches based on semantics or ontology [1-3]. However, building and maintaining ontologies is extremely difficult and requires substantial manual intervention [4]. Furthermore, in the face of massive Web services, a large amount of research work has been invested in ontology-based and dictionary-based discovery methods [1-2, 6-8].

Reference [2] proposes a Web service discovery method using domain ontology, which calculates semantic similarity between service requests and published ser-

vices through concept distances in the ontology. Reference [9] performs semantic annotation of Web services to facilitate discovery. However, such methods require extensive manual intervention, depend on the quality and maintenance of ontologies, and suffer from insufficient vocabulary coverage in certain domains and slow updates that may also lead to inaccurate discovery results. Moreover, these methods require service providers or requesters to supply relevant domain ontologies. However, service requesters are typically non-professional users who cannot provide professional ontologies, thus limiting the efficiency and generality of such methods. Additionally, the aforementioned Web service discovery methods cannot achieve real-time matching when facing massive numbers of Web services due to the lack of an effective classification mechanism.

Clustering is an effective method for processing large amounts of data. It reorganizes data into different clusters based on certain similarity criteria, enabling rapid information retrieval. Abramowicz et al. [9] propose a method for Web service filtering and clustering, but the filtering mechanism is based on Web services described in OWL-S (Web Ontology Language for Service), which still suffers from the drawback of ontology dependency. Most available Web services are described using WSDL (Web Service Description Language), and many approaches utilize WSDL for Web service classification. Nayak et al. [10] transform Web service descriptions into a multi-dimensional word vector space and calculate the distance between two services using the cosine of the angle between two vectors for clustering. This is a classification method based on mathematical statistics that can normalize large-scale text sets, but it ignores semantic relationships between words in descriptive texts, and its runtime and storage consumption increase with the scale of the text collection.

Cassar et al. [11] investigate the use of PLSA (Probabilistic Latent Semantic Analysis) and LDA (Latent Dirichlet Allocation) to mine topics from Web service descriptions for clustering. Experimental results show that the LDA model performs well in facilitating automatic service discovery in large-scale service collections. LDA evolved from PLSA. Blei et al. [12] introduced Dirichlet prior distributions to extend the PLSA model, proposing the LDA model, which can process large amounts of text data and classify them by discovering latent topics.

Aznag et al. [13] preprocess Web service description documents (feature extraction, tokenization, stopword removal, stemming, etc.) and then use CTM (Correlated Topic Model) to learn latent topics of Web services and service requests for classification, matching the similarity between offered services and requested services to obtain a final candidate Web service set. This method has several shortcomings: Web service description texts are typically short, similar to short texts, but this method does not expand the service texts, lacking sufficient word frequency co-occurrence; it uses only the Web service repository as a training set, which is small in scale and makes it difficult to obtain a high-quality topic model. These two points make it challenging to learn the true latent topics of Web services. Additionally, simply assigning Web services to the topic with the largest distribution in their topic mixture is not precise enough for classification.

The CTM topic model used by Aznag et al. [13] was proposed by Blei et al. [14]. This model introduces a log-normal distribution to replace the Dirichlet distribution in LDA. The prior parameters of the CTM model include a covariance matrix that describes correlations between each pair of topics, and the number of parameters in the covariance matrix is proportional to the square of the number of topics. Wei Qiang et al. [15] use Word2vec and Relatedness text expansion methods to extract features from Wikipedia to enrich Web service description texts, using English Wikipedia as a training set and employing the HDP non-parametric topic model for topic modeling. They propose a Signature method for service matching, which improves service discovery effectiveness to some extent. However, during the service matching phase, when calculating input-output similarity, they adopt five types: exact match, plug-in match, subsume match, intersection match, and fail match, which oversimplifies the distinction of concept similarity and is not suitable for situations with large numbers of services.

Using external knowledge bases to expand features for short texts is a relatively common method, but finding appropriate and suitable external corpora is not easy. Web services involve many aspects, making it difficult to find a properly suited external corpus. Reference [15] uses Wikipedia as an external corpus and topic model training set. However, Wikipedia, as a comprehensive text repository, is not accurate when used for feature supplementation and topic model training. Web services do not cover all industries; some domains have frequent usage and large numbers of Web services, while others have few or no Web services. If Web services are non-existent or scarce in a domain, a comprehensive external corpus contains too much irrelevant information, which instead decreases precision. Reference [16] found that using an external knowledge base reduces modeling accuracy in short text classification.

This paper proposes using BTM to model Web service description texts. BTM models the generation process of unordered word pairs across the entire corpus, utilizing all word frequency co-occurrence information in the whole repository to learn latent topics, which solves the problem of inaccurate topic learning caused by sparsity in short texts. Since BTM learns the topic distribution and topic-word distribution of the entire Web service repository, we propose an inference method to calculate the topic distribution of each Web service description document.

3 Web Service Clustering and Discovery Based on BTM

The framework for Web service clustering and discovery based on topic models is shown in Figure 1 [Figure 1: see original paper]. The preprocessed Web service repository serves as input for BTM modeling. The modeling output consists of the topic distribution and topic-word distribution of the entire repository. Combined with the VSM (Vector Space Model) obtained from preprocessing, we infer and calculate the topic distribution of each document as input for clustering. When a service request arrives, preprocessing yields the word vector

of request r. At this point, the previously calculated topic distribution and word distribution of the entire repository remain unchanged and do not require recalculation. We directly use them to infer and calculate the topic distribution of request r and perform classification. After obtaining the category of r, we calculate topic similarity and word vector similarity to find Web services that satisfy users.

3.1 Preprocessing

Every Web service has a document described using WSDL, which is an XML-based Web service description framework standardized by W3C, with versions 1.1 and 2.0. Version 2.0 is simpler and more practical. Currently, version 1.1 is more widely used, but version 2.0 may gradually replace it in the future. The preprocessing of service text data described using WSDL (versions 1.1 and 2.0) mainly includes the following steps:

(1) **Feature extraction**. Extract all features describing Web services from WSDL, including service name, service functional text description, operation name, input/output parameter names, parameter types, etc.

(2) **Tokenization**. Some terms in the extracted text consist of multiple words, called compound words, which need to be split. For example, "BusinessDataCatalog" is split into "Business", "Data", and "Catalog".

(3) **Tag and stopword removal**. Remove tags and stopwords from Web service description texts to avoid affecting accuracy during modeling. Use the Stanford POS Tagger to remove all tags and stopwords, keeping only words with noun, verb, or adjective parts of speech. Removed stopwords include "a", "is", "that", etc., and removed tags include WSDL tags such as "soap", "type", "binding", etc.

(4) **Stemming**. Words with the same stem usually have similar meanings. For example, "recommended" is simply the past tense of "recommend". Using Porter Stemmer for word stemming and representing Web services with word vectors in their root form can more effectively discover services.

(5) **Biterm extraction**. Unlike LDA, which directly models word co-occurrence frequencies in documents, BTM models word pair co-occurrence rates across the entire corpus. For the initial short text "BusinessData search for users", after extraction, tokenization, tag/stopword removal, and stemming, the extracted word pairs are {(business, data), (business, search), (business, user), (data, search), ···}. All word pairs from the entire repository serve as input for BTM model training.

(6) **Service matrix**. After extracting all useful words, calculate TF-IDF values for words and represent all Web services as a vector space using VSM (Vector Space Model), with each Web service represented as a word vector $\{w_1, w_2, \cdots, w\}$. The weight value of each word is calculated by

TF-IDF, where tf is the frequency of word j in document i, and idf is the inverse document frequency, obtained by taking the logarithm of the total number of documents divided by the number of documents containing word j. A word that appears frequently in a document but rarely in other documents has higher importance and a larger weight value.

3.2 BTM Model

BTM learns latent topics by modeling word co-occurrence across the entire corpus. Unlike LDA, which models the word generation process within individual documents, single short texts lack sufficient word frequency co-occurrence, making LDA modeling results unstable. BTM models the generation process of word pairs across the entire corpus; the frequency of word pairs in the entire corpus is more stable and better reveals relationships between words, enabling the learning of latent topics for the whole repository.

Figure 2 [Figure 2: see original paper] Probabilistic Models

The document generation process modeled by LDA is shown in Figure 2(a): for each document, a topic distribution d is randomly generated, the topic z for the i-th word is sampled from d, and the word w is then generated step by step from z. It can be seen that each document has a topic distribution, and the topic estimation of words depends on other words in the same document. In the unigram mixture model, all words in a document share a single topic z, which is generated from a global topic distribution . This assumes the entire corpus is viewed as a mixture of topics, using statistical information from the whole corpus to avoid the problem of information sparsity in short texts. However, the unigram mixture model assumes each document has only one topic, which does not match reality and results in poor topic learning. BTM can be seen as a combination of the unigram model and LDA. As shown in Figure 2(b), BTM assumes a global topic distribution , but it divides each document into word pairs, with each pair belonging to a topic. BTM allows a document to have multiple topics, thus avoiding the limitations of the unigram mixture model while solving the problem that LDA cannot achieve good results in short text modeling.

3.3 Web Service Clustering

BTM modeling yields two main parameters: is the topic distribution of the Web service set, a K-dimensional vector where K is the number of topics in the entire service set; Φ is the topic-word distribution matrix with K rows and N columns, where each row represents the generation probability of different words under a topic Z. Since BTM does not model the generation process of each document, we cannot directly obtain each document' s topic distribution. However, it can be inferred through and Φ. The calculation formula is proposed as follows:

P(z|d) Σ P(z|w)P(w |d)

where P(w |d) is the TF-IDF value calculated in step (6) of Section 3.1. P(z|w) can be calculated using Bayes' formula:

P(z|w) = P(z)P(w |z) / Σ P(z)P(w |z)

where P(w |z) is the probability of the i-th word under topic j in the topic-word distribution Φ.

After obtaining the document' s topic distribution, we can transform the document' s word vector representation, which lacks semantic information, into a topic vector representation containing semantic information: {P(z_1|d), P(z_2|d), ···, P(z |d)}. Each item in the vector is a topic probability. Therefore, text similarity can be calculated using KL divergence (Kullback-Leibler Divergence) [17]:

D_{KL}(p||q) = Σ p(i) log(p(i)/q(i))

However, since KL distance is asymmetric, i.e., D_{KL}(s_1||s_2) ≠ D_{KL}(s_2||s_1), we use the improved symmetric version of KL distance—JS distance (Jensen-Shannon Divergence) [18]:

Sim_{JS}(s_1,s_2) = 1 - D_{JS}(s_1,s_2)

The calculated similarity of Web service description documents serves as input to the clustering algorithm.

3.4 Web Service Discovery

Service matching refers to quickly and accurately finding candidate services that meet user requirements from a large number of services. When a service request is input, we perform data preprocessing on it, calculate the topic distribution of the requested service through the topic distribution and topic-word distribution matrix, compute the JS distance between the requested service and the center services of each cluster, and assign it to the nearest cluster, thereby locking onto the service subset of the same category as the service request.

Due to the huge number of services, to save matching time, after obtaining the service subset of the same category, we filter the Web services in the subset based on topics. We use JS distance to calculate the topic similarity Sim_T(r, s) between the service request' s topics and the topics of services in the subset, setting a threshold. When the similarity is greater than the threshold, the service is added to the candidate service set. We then calculate the word vector similarity between Web services in the candidate set and the service request.

The word vector representation of Web services is described in Section 3.1. The similarity calculation between Web service and request word vectors uses cosine distance, which is the most commonly used method for calculating vector space similarity. It calculates the angle between two vectors in vector space; the smaller the angle, the greater the similarity. The cosine distance calculation formula is:

Sim_{cos}(r,s) = (r · s) / (||r|| × ||s||)

Calculating topic similarity can yield the semantic dimension similarity between services, while word vector similarity reflects to some extent the statistical-level similarity between services. Therefore, we combine the calculated topic similarity and word vector similarity to obtain the total similarity:

Sim(r,s) = α × Sim_{cos}(r,s) + (1-α) × Sim_T(r,s)

where α is the weight of Web service word vector similarity, with $0 < \alpha < 1$.

4 Experimental Results and Analysis

To validate the method proposed in this paper, our experiments adopt the dataset provided by WS-Dream [19], which contains 3,378 WSDL files with 15,811 operations from 69 countries. This paper uses the KNN algorithm [20] in Weka for Web service clustering. Precision and normalized discounted cumulative gain are used for effectiveness evaluation.

Precision is a measure of the ratio of useful results among all retrieved results, i.e., the ratio of the number of relevant Web services retrieved to the total number of Web services retrieved:

Precision = |{relevant services} {retrieved services}| / |{retrieved services}|

Precision does not consider the position information of discovery results and can only indicate the overall quality of the results. Discounted Cumulative Gain (DCG) is a statistical method that ranks the relevance level of each retrieved result, where higher relevance results are better when ranked higher, and high-relevance results contribute much more than low-relevance results. The formula is:

DCG $= \Sigma_{1}^{n}$ (2^{rel } - 1) / $\log_2(1 + i)$

where rel is the relevance level of the result at position i in the discovery results.

Different discovery methods produce different result content and quantities. To enable comparison between different discovery methods, we can use Normalized Discounted Cumulative Gain (NDCG):

NDCG = DCG / IDCG

where IDCG is the DCG calculated when the discovery results are in optimal ranking order.

In the experiments, we compare the effectiveness of three topic learning methods for service discovery: BTM, LDA, and a method using LDA with Wikipedia as an external knowledge base. In a Java environment, using tools such as JDK, Eclipse, and JGibbLDA, and considering the scale of our dataset, we set the number of topics from 20 to 100, continuously adjusting the topic count through iterations. We calculate the F-measure of clustering results corresponding to different numbers of topics. The results show that in our dataset, BTM, LDA,

and LDA+Wiki achieve optimal clustering effects with 48, 55, and 71 topics, respectively. Table 1 lists some topics from PAM clustering and the top-ranked keywords for each topic.

Table 1 Partial Topic-Word Distribution

Topic	Keywords and Probabilities
1	cinema, route, music, country, price
2	service, location, album, british, version
3	budget, parameter, result, weather, release
4	welfare, retrieve, tourist, culture, airplane
5	party, ···

The word distribution of topics can well represent the semantic information of each topic. Topics 1, 2, 3, 4, and 5 correspond to camera, time, tourism, music, and country information, respectively. Using these topics, we can reduce the dimensionality of multi-dimensional word vectors to lower-dimensional topic vectors with semantic information and representativeness, capable of representing document features.

In this experiment, we randomly selected 12 query conditions. Each query condition' s relevant service set consists of a group of related services, with each service having a relevance level i {1, 2, 3}, where 3 indicates high relevance and 1 indicates low relevance. By comparing discovery results with the relevant service set, we can obtain the number of discovered relevant services and the relevance level of each service, thereby calculating precision and NDCG. We compare our method with two approaches: (1) directly using LDA modeling on preprocessed data to learn latent topics and cluster Web services, and (2) using Wikipedia-based expansion followed by LDA modeling. The results are shown in Figure 3 [Figure 3: see original paper] and Figure 4 [Figure 4: see original paper].

Our method is denoted as BTM, the LDA-only method as LDA, and the expansion-based method as LDA+Wiki.

Precision Comparison

As shown in Figure 3 [Figure 3: see original paper], the precision of LDA and LDA+Wiki cross at approximately 10 and 25 retrieved services, but overall LDA performs slightly better than LDA+Wiki. BTM achieves higher precision than the other two methods overall, especially when the number of retrieved services is larger. When the number reaches 50 (the maximum in the figure), the difference between BTM and LDA/LDA+Wiki reaches its maximum, with BTM' s precision being 0.7% higher than LDA' s.

NDCG reflects the ability to discover relevant results. As shown in Figure 4 [Figure 4: see original paper], BTM outperforms LDA and LDA+Wiki by

approximately 0.1%–0.8%. It is 0.1% higher when the number of services is minimal (5), and 0.8% higher than LDA when the number of services reaches 30. This indicates that LDA and LDA+Wiki miss some highly relevant Web services due to their lower precision. Overall, the discovery method proposed in this paper better aligns with the characteristics of Web service description documents and can help discover Web services more effectively, demonstrating superior performance in both precision and normalized discounted cumulative gain.

BTM performs global modeling based on the entire Web service description repository, fully utilizing the semantic information of the whole repository to learn latent topics, which compensates for the shortcomings of short Web description texts, lack of word frequency co-occurrence, and semantic sparsity. In contrast, LDA models at the document level and is easily affected by document length, resulting in inaccurate learned topics that cannot well express semantic information, leading to inferior Web service discovery performance. For short texts, using Wikipedia for expansion and then applying LDA modeling on the expanded text vectors even performs worse than directly using LDA modeling. This is because, on one hand, external corpora do not fully contain the latent topics of the Web service description document collection; on the other hand, comprehensive external corpora contain too many relationships, including industry-specific terms and overly rich vocabulary, which reduces discovery precision.

The number of available Web services is increasing daily. To quickly and effectively discover services that meet user requirements from massive Web services, it is necessary to cluster Web services with similar functions. However, Web service description documents are short, and after removing tags and stopwords, few feature words remain. Using LDA to learn latent topics for clustering is not ideal because LDA models the document generation process and heavily depends on text length, resulting in poor clustering effects for Web services that do not facilitate service discovery. To address this problem, we use BTM to learn the latent topics of the Web service document collection, infer the topic distribution of each document, calculate the similarity between documents using JS distance, and use it as input to the KNN algorithm for clustering Web services. During the Web service discovery phase, we combine topic similarity and word similarity to discover Web services. The proposed discovery method fully utilizes the semantic resources of the entire Web service repository to learn more accurate latent topics without relying on external knowledge bases, reducing noise information caused by too many relationships in external knowledge bases. By inferring the topics of each description document and performing clustering, we identify the category of service requests, greatly narrowing the search scope and improving query efficiency. Experiments demonstrate that the proposed Web service discovery method has certain superiority in accuracy.

However, this method only considers the functionality of Web services and does not incorporate service quality into the calculation, so the actual execution

rate of Web services cannot be guaranteed. Future work will mainly focus on calculating service quality, combining service price, reliability, and response time with service discovery to provide users with more reliable services. Additionally, how to label each Web service with tags so that users can easily understand and select services that meet their requirements for careful examination is also a future direction.

References

[1] Farrag T A, Saleh A I, Ali H A. Semantic Web Services Matchmaking: Semantic Distance-based Approach [J]. Computer and Electrical Engineering, 2013, 39(2): 497-511.

[2] Lu G, Wang T, Zhang G, et al. Semantic Web Services Discovery Based on Domain Ontology [C]. In: Proceedings of the 2012 World Automation Congress (WAC). 2012: 1-4.

[3] Shi Min, Zhao Wendong, Zhang Lei. A Semantic Web Service Discovery Algorithm Based on Ontology Partition [J]. Computer Engineering, 2014, 40(2): 175-179.

[4] Atkinson C, Bostan P, Hummel O, et al. A Practical Approach to Web Service Discovery and Retrieval[C]. In: Proceedings of the 2007 IEEE International Conference on Web Service. 2007: 241-248.

[5] Yan X, Guo J, Lan Y, et al. A Biterm Topic Model for Short Texts [C]. In: Proceedings of the 22nd International World Wide Web Conferences. 2013: 1445-1456.

[6] Qu M, Liu S, Bao T. On the Trusted Ontology Model for Evaluating the Semantic Web Services[C]. In: Proceedings of the 14th International Conference on Computer Supported Cooperative Work in Design. 2010: 368-369.

[7] Kopecký J, Vitvar T, Bournez C, et al. Semantic Annotations for WSDL and XML Schema [J]. IEEE Internet Computing, 2007, 11(6): 60-67.

[8] Yang Huirong, Liu Shanshan, Yin Baocai, et al. Matching Algorithm of Services Based on Semantic Distance [J]. Journal of Beijing University of Technology, 2011, 37(4): 591-595.

[9] Abramowicz W, Haniewicz K, Kaczmarek M, et al. Architecture for Web Services Filtering and Clustering [C]. In: Proceedings of the 2nd International Conference on Internet and Web Applications and Services. 2007.

[10] Nayak R, Lee B. Web Service Discovery with Additional Semantics and Clustering [C]. In: Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence. 2007: 555-558.

[11] Cassar G, Barnaghi P, Moessner K. Probabilistic Methods for Service Clustering [J]. In: Proceeding of the 4th International Workshop on Service Matchmaking & Resource Retrieval.

[12] Blei D M, Ng A Y, Jordan M I. Latent Dirichlet Allocation[J]. Journal of Machine Learning Research, 2003, 3: 993-1022.

[13] Aznag M, Quafafou M, Rochd E M, et al. Probabilistic Topic Models for Web Services Clustering and Discovery[A]. // Service-Oriented and Cloud Computing[M]. Springer-Verlag Berlin Heidelberg, 2013.

[14] Blei D M, Lafferty J D. Correlated Topic Models[C]. In: Proceedings of the 23rd International Conference on Machine Learning. 2005.

[15] Wei Qiang, Jin Zhi, Xu Yan. Service Discovery for Internet of Things Based on Probabilistic Topic Model [J]. Journal of Software, 2014, 25(8): 1640-1658.

[16] Zhu Y, Li L, Luo L. Learning to Classify Short Text with Topic Model and External Knowledge[A]. //Knowledge Science, Engineering and Management[M]. Springer Berlin Heidelberg, 2013.

[17] Duda R O, Hart P E, Stork D G. Pattern Classification[M]. Translated by Li Hongdong, Yao Tianxiang, et al. The 2nd Edition. China Machine Press, 2003.

[18] Lin J. Divergence Measures Based on the Shannon Entropy [J]. IEEE Transactions on Information Theory, 1991, 37(1): 145-151.

[19] Zhang Y L, Zheng Z B, Lyu M R. A QoS-aware Search Engine for Web Services [C]. In: Proceedings of the 8th International Conference on Web Services. Miami, Florida, USA. 2010.

[20] Cover T M, Hart P E. Nearest Neighbor Pattern Classification[J]. IEEE Transactions on Information Theory, 1967, 13(1): 21-27.

Data Availability

[1] Li Hui, Hu Yunfeng. wsdldataset.rar. Web service data provided by WS-Dream.
[2] Li Hui, Hu Yunfeng. stopwords.dat. Original stopword list downloaded from Datatang, with WSDL tags and other stopwords added.
[3] Li Hui, Hu Yunfeng. pre-result.dat. Preprocessed Web services.
[4] Li Hui, Hu Yunfeng. topic.txt. Topic-word distribution obtained from modeling.
[5] Li Hui, Hu Yunfeng. simT.dat. Web service similarity matrix.

Received: 2015-12-22
Revised: 2016-02-03

Author Contributions: Li Hui: Conceived the research idea and experimental process, revised the paper; Hu Yunfeng: Proposed the specific solution, conducted the experiments, wrote the paper.

Conflict of Interest Statement: All authors declare no conflict of interest.

Supporting Data: Supporting data is self-archived by the authors, E-mail: 1540520650@qq.com.

Note: Figure translations are in progress. See original paper for figures.