Gaussian LDA and Word Embedding for Semantic Sparse Web Service Discovery

Gang Tian[1,2], Jian Wang[1(✉)], Ziqi Zhao[2], and Junju Liu[3]

[1] State Key Lab of Software Engineering, Wuhan University, Wuhan, China
{tiangang,jianwang}@whu.edu.cn
[2] College of Information Science and Engineering,
Shandong University of Science and Technology, Qingdao, China
1362190630@qq.com
[3] Zhixing College, Hubei University, Wuhan, China
orange_wh@qq.com

Abstract. In recent years, more and more Web services are published in API marketplaces founded by cloud service providers or third party registries. In this situation, users rely heavily on the search engine model to retrieve their expected Web services. However, due to the fact that Web services registered in API marketplaces are described in short texts, the search engine based discovery method suffers from the semantic sparsity problem, which in turn leads to a poor recall during service discovery. To address this issue, in this paper, we propose a novel Web service discovery approach that uses Gaussian Latent Dirichlet Allocation (Gaussian LDA) and word embedding. More specifically, instead of clustering Web services like most existing service discovery approaches, we use word embedding to map the words as continuous word embeddings to extend and enrich the semantics of service descriptions. We also leverage the Gaussian LDA in service discovery, which takes continuous word distribution as the input and interprets the Web service description as a hierarchical model by its two distributions. Based on the Gaussian LDA and word embedding, we propose a Web service query and ranking approach. Experiments conducted on a real-world Web service dataset demonstrate the effectiveness of the proposed approach.

Keywords: Word embedding · Gaussian LDA · Semantic sparsity · Web service discovery

1 Introduction

Benefited by well-constructed Internet infrastructure and the advantages of service-oriented computing, more and more enterprises are driven to develop or transform their business applications into distributed Web services. These Web services that are usually deployed on the providers' servers are scattered over the Internet. Many enterprises tend to build their own Web service marketplaces on which the registered Web services are mainly built by using the platform owner's

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2017
S. Wang and A. Zhou (Eds.): CollaborateCom 2016, LNICST 201, pp. 48–59, 2017.
DOI: 10.1007/978-3-319-59288-6_5

cloud services or closely related to the platform owner's business. Google Apps Marketplace, Amazon Web service marketplace and Microsoft Azure Web service marketplace are the well known representatives. In this trend, the search engine based approach is widely used to help users in discovering Web services. However, Web service search engines that mainly focus on keyword-based matching may result in the poor recall problem due to the scattered registration, the lack of keywords in Web service descriptions, the use of synonyms, or variations of keywords [5]. To enhance the capability of search engines by clustering services into functionally similar groups is a useful method to handle this problem. However, some new issues are emerging in recent years. One typical problem is the semantic sparsity, which results from short text descriptions of Web services provided by most marketplaces. That is, there is no sufficient information to express the full semantics of Web services. Towards this issue, many works on how to transfer external information to enrich the semantic representation of short text documents have been proposed. For example, Jin et al. [7] present a transfer learning approach, which can facilitate short texts clustering by using auxiliary long texts. Hu et al. [6] introduce a short text clustering method by using world knowledge. These works generally make an implicit assumption that the auxiliary information is semantically related to the short texts. However, it is not trivial to find such kind of auxiliary information, which makes the assumption not always true in real data.

To address this issue, we introduce the word embeddings which have been shown to capture lexico-semantic regularities in language. In the embedding space, words with similar syntactic and semantic properties are found to be close to each other [11]. Thus, this feature is particularly suitable to solve the problems of using synonyms/variations of keywords in the query. Furthermore, the context information such as the co-occurrence information in the word embeddings can be effectively used to enrich the semantics of a document. Inspired by this, we leverage word embeddings to handle the semantic sparsity problem in Web service discovery. Our main contributions are listed as follows:

- We leverage pre-trained word embeddings to enrich the semantics of Web service descriptions.
- We use Gaussian LDA to model the user query and Web service descriptions based on the continuous word embeddings, aiming to improve the quality of Web service discovery.
- We conduct experiments to illustrate the feasibility of the proposed approach.

The rest of the paper is organized as follows. Section 2 discusses related works in the area of Web service discovery. Section 3 introduces the proposed model in detail. Section 4 reports our empirical experiments. Section 5 concludes the paper.

2 Related Work

Web service discovery is viewed as a significant part of service computing and cloud computing, and many progresses have been made in this area. Because

Web services are often described in different description languages, the non-semantic based discovery approaches are fairly different. For example, Liu et al. [10] extract four features including service content, context, host name, and service name from WSDL documents to cluster Web services by text mining techniques. Similarly, Elgazzar et al. [5] also extract content, types, messages, ports, and service name from WSDL documents as features of their information retrieval model to cluster Web services. In these works, the features in the information retrieval models come from the content extracted WSDL documents. If Web services are described in other description languages, these WSDL based methods may not be efficient enough or even fail to work. In this paper, we focus on the discovery of Web services which contain even less features. Therefore, the above methods may fail to work since they lack ways to handle the semantic sparsity problem.

Several studies have revealed that it is helpful to integrate external information to improve the performance of Web service discovery. Chen et al. [3] propose an augment LDA model to integrate WSDL and tags, which can improve the performance of Web service clustering. It has been shown that leveraging external information can enhance the discovery method to a certain extent. In the research field of Information Retrieval (IR), there are also some works to handle the semantic sparsity problem. For example, Hu et al. [6] aim to improve the performance of clustering short text by using world knowledge. Jin et al. [7] introduce a transfer learning based approach which clusters short texts by using auxiliary long texts. These methods can partially handle the semantic sparsity problem. However, they also have some limitations. For example, the work in [6] make the implicit assumption that the auxiliary data are semantically related to the short texts, which may have difficulties in practice. Similarly, the work [7] assumes that the topical structures of two domains are completely identical, which would be also unreasonable in the practice. Faced with these problems, we integrate external context information using word embeddings which can boost the performance in information retrieval tasks such as computing the short text similarity [8].

Probabilistic models such as Probabilistic Latent Semantic Analysis (PLSA), LDA and extensions of these models have been identified as efficient methods for boosting the performance of Web services discovery [2,3]. However, the basic assumption of these probabilistic models is that the words are discrete multinomial distribution, these models can not benefit from the word embeddings whose vectors are continuous. In contrast, we use Gaussian LDA [4] to leverage the advantages of both word embeddings and probabilistic models.

To the best of our knowledge, there is still no reported approach on leveraging word embedding techniques combined with Gaussian LDA for the semantic sparsity Web service discovery.

3 The Proposed Approach

We first describe the discovery process of the proposed approach in Sect. 3.1. In Sect. 3.2, we illustrate the service modelling method based on Gaussian LDA.

Finally, we introduce the query modelling method that integrates the word embedding in Sect. 3.3.

3.1 The Discovery Process of the Proposed Approach

Figure 1 illustrates the discovery process of the proposed approach, which consists of three major parts: service modeling, query modeling and service ranking.

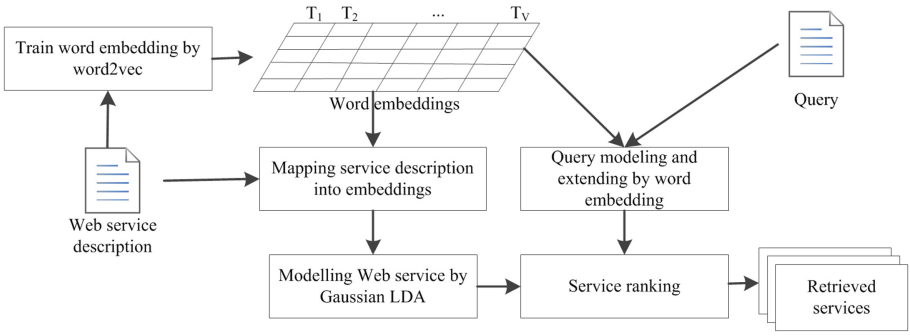

Fig. 1. Overall discovery process.

As shown in Fig. 1, service descriptions distributed on the Internet are firstly crawled and preprocessed. Then these service descriptions are taken as the input of the word2vec model[1] to create word embeddings. In this step, we can also introduce the pre-trained word embeddings directly, such as the word embeddings trained by using Wikipedia. After getting the word embeddings, we map the words in service descriptions into word embeddings to generate the input of Gaussian LDA. The Gaussian LDA plays the role of modeling each Web service as hierarchies of latent factors.

In the query modeling phase, the words in a given query are firstly mapped into embeddings by using the word embedding set produced in the service modeling step. Afterwards, in order to integrate more contextual information to enrich the semantics of the query, we use a query extension algorithm which finds similar neighbors of each word in a query by using the word embedding set to extend the query. After that, an extended query is obtained.

In the service ranking step, based on the hierarchies and the extended query, a probabilistic service ranking model is proposed to retrieve relevant services for the user.

Please note that the steps of training word embeddings and service modeling are conducted offline, only the query modeling and service ranking steps are performed online. Hence, the focus of the paper is placed more on the accuracy of discovery, not the efficiency.

[1] https://code.google.com/archive/p/word2vec/.

3.2 Web Service Modelling Using Gaussian LDA

In Gaussian LDA [4], the word embedding of each term w or v_{di} in a document d at position i is written as $v(w) \in R^M$, and M is the length of the word embedding. Thus, the words in a document are no longer discrete values but are mapped into continuous vectors in an M dimensional space. As a result, each topic k is characterized as a multivariate Gaussian distribution with mean μ_k and covariance Σ_k. The parameterization for the Gaussian is justified by both analytic convenience and the semantic similarity of embeddings [4]. To govern the mean and variance of each Gaussian, the Gaussian distribution centered at zero and an inverse Wishart distribution for the covariance are placed as the conjugate priors.

Using Gaussian LDA, each word w represented by its embedding e in a Web service description is associated with the latent variable topic z, and each topic z is associated with the service description d. With these two relations, a service can be viewed as two layers: the Service-Topic layer and the Topic-Embedding layer. First, each word w (e.g.,"API" and "access") is represented as a vector of a fixed length after running the word2vec model. For example, the word 'academic' is represented as [0.26 −0.30 −0.14 ... −0.05] with size of 50. Before using Gaussian LDA, each word in a Web service description is represented by the vector trained by word2vec. Thus, a Web service is mapped into a matrix with fixed columns of 50 in this paper. Secondly, the matrices representing the Web services are fed into the Gaussian LDA to produce that two layer represented by two distributions: Web service-topic distribution and topic-embedding distribution. The Web service-topic distribution is derived from the parameter θ ($\theta \in |services| \times |topics|$) as the traditional LDA model. In another side, the multivariate gaussian distributions are achieved by calculating the Cholesky triangular matrices. Finally, we established the hierarchies of Gaussian LDA by above mentioned two distributions.

To infer the posterior distribution of services over the topics and the topic assignments of individual words, a collapsed Gibbs sampler is derived to resample topic assignments to individual word vectors.

3.3 Retrieving Web Services

We define the process of retrieving Web services as three main steps: query modelling, query extension and service ranking.

Query Modelling. The first step is query modelling, which maps the user query into the word embedding feature space. A user query is defined as a set of words in the query: $Q = \{w_1, w_2, ..., w_{|Q|}\}$. We look up each word in the word embeddings.

Query Extension. Combining implicit or explicit semantics are helpful to enhance the performance of Web service matching and ranking by alleviating the problems due to service author/user heterogeneity [12]. Distributed word

Algorithm 1. Query Extension

Input: query Q, similarity weight τ, the embedding set E
Output: the extended query Q_e .
$Q_e \leftarrow \varnothing$;
for *word* $w \in Q$ **do**
  for *word* $e_w \in E.most\_similar(w)$ **do**
    $\Xi_w \leftarrow \varnothing$;
    if *similarity*(e_w,w) $\geq \tau$ **then**
      $\Xi_w \leftarrow \Xi_w \cup e_w$;
    end
    $Q_e \leftarrow Q_e \cup \Xi_w$;
  end
end
return Q_e

representations, such as word embeddings which map every word into a dimensional continuous space, are assumed to represent semantic and syntactic information of words [9]. In the continuous representation space, words with similar meanings have similar vectors, which means that synonym, near-synonym and semantic related words of a word have high probability to appear in the similar neighbors. For example, the top 3 most similar words to 'month' trained on our dataset by the word2vec model are 'minute', 'hour' and 'day'. According to this characteristic, we can simply extended the query by integrating the similar words based on the word embedding model to convey more context information since the query is often short and semantic sparse. As shown in Algorithm 1, we extend the query by using the close neighbors in the embedding space. In Algorithm 1, we take query Q, similarity weight τ and the embedding set E trained by word2vec as input and the extended query Q_e as output. According to the word2vec model, we add neighbors whose similarity weights are greater than parameter τ for each word in a query Q into the extended query Q_e.

Services Ranking. To rank the retrieved Web service for a query, we need to create the base criteria to calculate the similarity between the user queries and the retrieved Web services. Similar to the work in [1], we use the generated probabilities to calculate the similarity. To this end, we model the service retrieval as a probabilistic query to the topic model. We note this as $P(Q|s_i)$ where Q is the set of words contained in the query. s_i is the i-th Web service. Thus, using the assumptions of the Gaussian LDA, $P(Q|s_i)$ can be calculated by Eq. 1.

$$P(Q|s_i) = \prod_{e \in Q_e} P(e|s_i) = \prod_{e \in Q_e} \sum_{z=1}^{K} P(e|z)P(z|s_i) \tag{1}$$

Here, Q_e is obtained from Algorithm 1 which is the extended query of Q. $P(e|z)$ and $P(z|s)$ are the posterior probabilities which can be computed and the matrix θ, respectively.

The most relevant services are the ones that maximize the conditional probability of the query $P(Q|s_i)$. Consequently, relevant services are ranked according to their similarity scores to the query. In this way, we can obtain a ranking of the retrieved services towards a query.

4 Experiments

4.1 Experimental Preparation

To evaluate the performance of the proposed approach, we conducted several experiments on a real-world Web service discovery dataset called SAWSDL-TC3. We used a Python package named Gensim to train the word embeddings. As for Gaussian LDA, we directly used the Java implementation in Github[2].

Since different corpus conveys different context information for the word embeddings, we trained two word embedding sets using the corpus of Wikipedia and SAWSDL-TC3, respectively. We set the size of the word embedding as 50. In order to train Wikipedia, we use the default configuration of Gensim where the *size* is set to 50, *windowsize* is 5 and *min_count* equals 5. For SAWSDL-TC3 (TC3, for short), we pre-process them by parsing the WSDL documents, removing stop words and lemmatizing the remaining words. Note that, we adopt TC3 to verify the proposed approach, albeit it is still a WSDL service set, not a short text based service set. There are two reasons for doing this. On one hand, as far as we know, there are no standard test dataset for short text based Web service discovery, while TC3 is a widely used dataset in the field of Web service discovery. On the other hand, we apply word embeddings to extend the query, which can solve the semantic sparsity issue. That is, the main purpose of the experiments is to validate the query extension performance of the proposed approach. Thus, a WSDL based test set plays the similar role as a short text based test set.

In our experiments, we used precision p, recall r and F-Measure f as the evaluation criterion for the proposed approach. The larger the F-Measure is, the better the performance of the discovery is. The TC3 dataset contains different queries (i.e., 42 requests), which are also represented in WSDL documents. Furthermore, a binary and graded relevance set for each query is provided which is viewed as the grounding truth in computing the precision or recall.

4.2 Performance of the Proposed Approach

To demonstrate the performance of our method, we compare the proposed method with four Web service discovery approaches. These approaches are illustrated as follows:

1. TF-IDF: We represent each description of the Web service with TF-IDF, and calculate the similarity with Cosine similarity.

[2] https://github.com/rajarshd/Gaussian_LDA.

Table 1. Performance of the proposed approach

Query	TF-IDF			PLSA			LDA			GLDA			GLDA+QE		
	p	*r*	*f*	*p*	*r*	*f*	*p*	*r*	*f*	*p*	*r*	*f*	*p*	*r*	*f*
@10	0.67	0.30	0.41	0.73	0.34	0.46	0.64	0.30	0.40	0.76	0.37	0.50	0.81	0.42	0.55
@15	0.57	0.35	0.43	0.66	0.41	0.50	0.57	0.35	0.43	0.69	0.43	0.53	0.74	0.49	0.59
@20	0.50	0.38	0.44	0.59	0.46	0.52	0.50	0.38	0.44	0.61	0.47	0.53	0.67	0.53	0.60
@25	0.46	0.45	0.44	0.56	0.51	0.53	0.45	0.31	0.43	0.58	0.51	0.54	0.64	0.57	0.60
@30	0.37	0.46	0.42	0.53	0.56	0.54	0.41	0.44	0.43	0.55	0.54	0.54	0.61	0.62	0.62
@35	0.34	0.48	0.40	0.49	0.60	0.53	0.38	0.46	0.42	0.51	0.59	0.55	0.57	0.66	0.61
@40	0.30	0.52	0.38	0.46	0.62	0.51	0.36	0.49	0.42	0.49	0.61	0.54	0.54	0.68	0.60
Average	0.459	0.42	0.439	0.574	0.50	0.534	0.472	0.39	0.427	0.599	0.503	0.547	0.654	0.567	0.61

Notes: GLDA denotes Gaussian LDA; GLDA+QE denotes Gaussian LDA + Query Extension.

2. PLSA: In this approach, Web services are clustered according to the latent factors learned from the Web service descriptions [2].
3. LDA: LDA is another commonly used latent factor based model for service clustering. Services are also grouped according to their latent factors [2].
4. Gaussian LDA: For Gaussian LDA, we first train the word embeddings by Word2vec from the prepared corpus such as Wikipedia. Then we change the words in each Web service into embedding vectors and take the word embeddings in the service description as the input of the Gaussian-LDA. Finally, we take it as the discovery model according to the methods in Sect. 3 except for extending the query.

For the PLSA and LDA models, we learned the topics from the descriptions of the Web services following the procedures described in the work [2], and we tune each algorithm to its best parameter setting by cross validation and the parameters influence will be discussed in Sect. 4.5.

According to these experiments, we have several observations. Firstly, the F-Measure performance of the proposed approach (Gaussian LDA + Query Extension) outperforms other approaches as shown in Table 1, which demonstrates the effectiveness of the proposed approach.

Secondly, the Gaussian LDA based approach without query extension has better performance than LDA based approach. The results show that Gaussian LDA which takes continuous embeddings as input may capture more semantically coherent topics compared to the LDA based approach.

Thirdly, extending the queries by using word embeddings contributes to the performance improvement of the Web service discovery. As shown in Table 1, the approach of Gaussian LDA + Query Extension which uses Algorithm 1 to extend the query has a better performance in all conditions compared with the Gaussian LDA approach. A possible explanation for this may be that more contextual information of the query is introduced to enrich the semantics of the query.

4.3 Validation of Embedding

In the proposed approach, the embedding set plays two important roles: one is to change the Bag of Words model into continuous embedding space, and the other is to extend the query by providing additional contextual information. Figure 2 shows the F-Measure performance of the proposed approach with different word embeddings trained by word2vec model using two different corpus TC3 and Wikipedia, as described in Sect. 4.1.

According to these results, we can see that the proposed method using TC3 has a better performance than using Wikipedia. However, the improvement of the performance is not very significant. There are several possible explanations for this result: the word embeddings trained from TC3 may be more domain specific than that of Wikipedia, that is, the domain that the words in a Web service description belong to may have different word distribution compared with Wikipedia. Another possible reason may be that some words, which are parsed from the WSDL files but do not have enough occurrence time in the Wikipedia corpus, are removed when training the embeddings though they are very informative. For example, the term 'lendingduration' composed of words 'lending' and 'duration' is informative in many descriptions, but it is dismissed in the embeddings as it is not parsed into individual words.

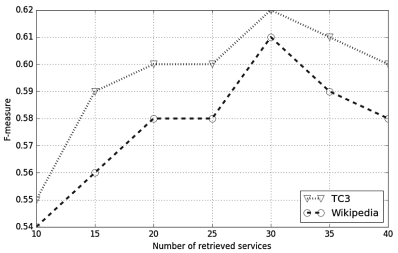

Fig. 2. Influence of different embedding sets.

4.4 Validation of Query Extension and Semantic Sparsity

Query extension illustrated in Sect. 3.3 is introduced to enrich the contextual information for the query. Figure 3(a) shows the F-Measure performance of two approaches using Gaussian LDA: one with query extension and the other one without extending the query. According to these results, we can discover that the performances of Gaussian LDA with query extension outperforms the approach without query extension in all conditions of different size of retrieved Web services. The results of this study indicate that integrating more contextual information will contribute to the performance improvement of the proposed approach. However, in our approach, we did not distinguish the relation between the added words and the original one, which will be further investigated in our future work.

An experiment on validating the sparsity problem is also conducted and the result is shown in Fig. 3(b). We truncated each WSDL file by randomly choosing feature words according to a certain scale to simulate the sparsity context. For example, we choose 10% words of a WSDL to create a new semantic sparsity file to stand for the original one when the number of retrieved services is 30. Based on the new file, we conducted the similar retrieval task to check the performance of the proposed approach in dealing with the semantic sparsity problem by following the process discussed in above sections. As shown in Fig. 3(b), the retrieval performances of both approaches are getting better with the increase of the percentage of WSDL involved in the experiments, which shows that the semantic sparsity problem does affect the performance of the retrieval model. Moreover, the more percentage of the WSDL files are involved in, the better performance is achieved, which validates that the proposed model can handle the semantic sparsity problem. The results also show the F-Measure under all settings of the proposed model are better than that of Guassian LDA without query extension, which further indicates that query extension contributes to retrieval task.

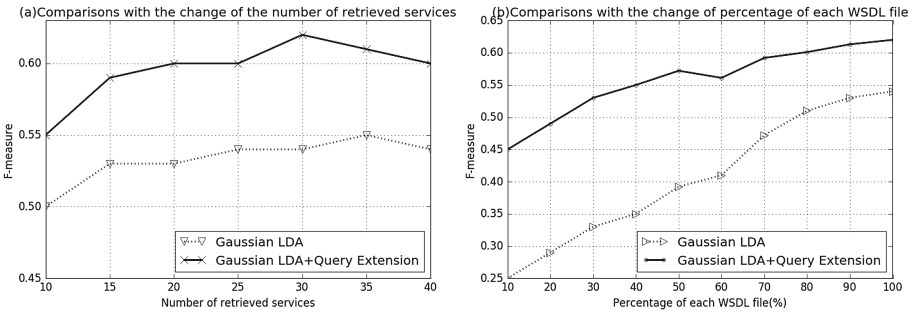

Fig. 3. Validation of semantic sparsity

4.5 Influence of Parameters

Hyperparameters: In Gaussian-LDA, the parameter α denotes the weight of language model contribution while μ and Σ control the document contribution.

In our work, some parameters are set as $\alpha = 1/K$, $\mu =$ zero mean and $\Sigma = 3 * I$ as defined in the work [4]. Here K is the number of topics and I is the identity matrix.

Topic Numbers: We computed an estimate of $P(e|k)$ for different K values. For all values of k, we ran the Gaussian LDA model until it converges. In that case, the log-likelihood values stabilized within a few hundred iterations, as shown in Fig. 4(a). The results suggest that the data are best accounted for by a model incorporating 6 topics.

Similarity Weight τ: As illustrated in Algorithm 1, the value τ is used to control the similar neighbors of each word to extend a query. A higher value results in less words to be chosen. On the other hand, a small value may bring more irrelevant contextual information into the queries. Thus we tune the parameter according to the performance of the proposed approach by cross validation.

Figure 4(b) shows the influence for discovery performance under different settings of similarity weight τ. As shown in Fig. 4(b), an F-measure performance is achieved when τ is 0.96.

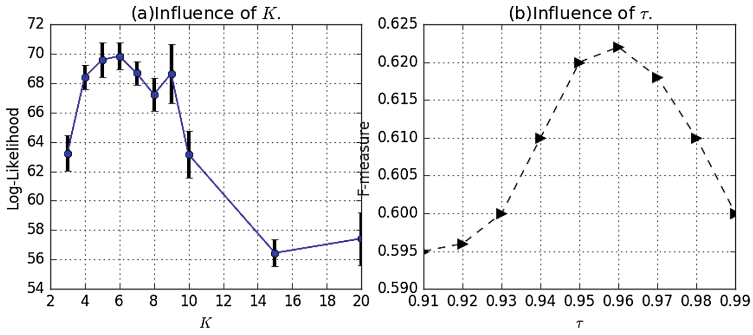

Fig. 4. Influence of parameters.

5 Conclusion

In this paper, we proposed a Web service discovery approach that combines Gaussian LDA with word embeddings to deal with the poor recall problem in searching semantic sparse Web services. We used word embeddings to model the word as continuous embedding, which can extend and enrich the semantics of Web service descriptions. We also introduced the Gaussian LDA which models the Web services as three layers by its two distributions. Based on the Gaussian LDA and word embedding, we proposed a service query and ranking approach.

We validated the proposed approach by using a real-world Web service dataset. Several experiments were conducted to assess the effectiveness of the proposed approach. Experimental results suggested that the proposed approach is feasible, and in particular the inclusion of meaningful word embeddings in the discovery process leads to enhanced performance.

In the future, we plan to further investigate the usefulness of various sources of word embeddings in Web service discovery. We also try to incorporate more linguistic features to provide goal oriented Web service discovery.

Acknowledgment. The work is supported by the National Basic Research Program of China under grant No. 2014CB340404, the National Natural Science Foundation of China under grant Nos. 61672387 and 61373037, the State Key Laboratory of Software Engineering Foundation under the grant No.SKLSE 2014-10-07, University Science and Technology Program of Shandong Province under the grant No.J16LN08, Scientific Research Foundation of Shandong University of Science and Technology for Recruited Talents under the grant No.2016RCJJ045. Jian Wang is the corresponding author.

References

1. Aznag, M., Quafafou, M., Jarir, Z.: Leveraging formal concept analysis with topic correlation for service clustering and discovery. In: Proceedings of the 2014 IEEE International Conference on Web Services (ICWS), pp. 153–160. IEEE (2014)
2. Cassar, G., Barnaghi, P., Moessner, K.: Probabilistic methods for service clustering. In: Proceeding of the 4th International Workshop on Semantic Web Service Matchmaking and Resource Retrieval, Organised in Conjonction the ISWC. Citeseer (2010)
3. Chen, L., Wang, Y., Yu, Q., Zheng, Z., Wu, J.: WT-LDA: user tagging augmented LDA for web service clustering. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 162–176. Springer, Heidelberg (2013). doi:10.1007/978-3-642-45005-1_12
4. Das, R., Zaheer, M., Dyer, C.: Gaussian LDA for topic models with word embeddings. In: ACL 2015 July 26–31, 2015, Beijing, China, vol. 1, Long Papers, pp. 795–804 (2015)
5. Elgazzar, K., Hassan, A.E., Martin, P.: Clustering wsdl documents to bootstrap the discovery of web services. In: Proceedings of the 2010 IEEE International Conference on Web Services (ICWS), pp. 147–154. IEEE (2010)
6. Hu, X., Sun, N., Zhang, C., Chua, T.S.: Exploiting internal and external semantics for the clustering of short texts using world knowledge. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 919–928. ACM (2009)
7. Jin, O., Liu, N.N., Zhao, K., Yu, Y., Yang, Q.: Transferring topical knowledge from auxiliary long texts for short text clustering. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, pp. 775–784. ACM (2011)
8. Kenter, T., de Rijke, M.: Short text similarity with word embeddings. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 1411–1420. ACM (2015)
9. Li, Y., Xu, L., Tian, F., Jiang, L., Zhong, X., Chen, E.: Word embedding revisited: a new representation learning and explicit matrix factorization perspective (2015)
10. Liu, W., Wong, W.: Web service clustering using text mining techniques. Int. J. Agent Oriented Softw. Eng. **3**(1), 6–26 (2009)
11. Mikolov, T., Yih, W.t., Zweig, G.: Linguistic regularities in continuous space word representations. In: HLT-NAACL, pp. 746–751 (2013)
12. Tekli, J.: An overview on xml semantic disambiguation from unstructured text to semi-structured data: background, applications, and ongoing challenges. IEEE Trans. Knowl. Data Eng. **28**, 1383–1407 (2016)