Data-sparsity Service Discovery using Enriched Neural Topic Model and Attentional Bi-LSTM

Li Yao, Bing Li, Jian Wang
School of Computer Science,
Wuhan University,
Wuhan, China
e-mail: jianwang@whu.edu.cn

Abstract—**In recent years, the amount of Web services has increased dramatically, and service discovery aiming to help users identify appropriate services matching their requirements thus becomes increasingly important. Many studies based on machine learning techniques have been reported to improve the performance of service discovery. A major obstacle in Web service discovery is the data sparsity in service descriptions. Towards this issue, in this paper, we propose a novel approach based on enriched neural topic model (NTM) and attentional Bi-LSTM. To alleviate the data sparsity issue, we enrich the semantics of each word in service descriptions and queries using external knowledge sources like Wikipedia and combine NTM and the attention mechanism to minimize the noise brought in the enrichment process. Experiments conducted on a real-world dataset show that our approach outperforms several state-of-the-art methods.**

Keywords-Web service; service discovery; attention mechanism; neural topic model; Bi-directional LSTM

I. INTRODUCTION

Service-oriented architecture (SOA) facilitates a new paradigm for system development and integration, where system functionalities are encapsulated as loosely coupled and interoperable services. A growing number of Web services or cloud services have thus been created and published to meet the interoperability and flexibility requirements of modern software development in the cloud. The proliferation of Web services offers convenience for developers; however, it also brings difficulties in quickly selecting appropriate candidate services from large scale service registries.

In existing service registries, Web services are usually described in WSDL (Web Service Description Language) or simple natural language texts. Since keyword matching adopted in most service search engines may suffer from retrieving irrelevant services or missing relevant ones, many efforts have been made to address the service matching problem [1-5]. These approaches could be approximately classified into two types. The first group of approaches annotates services and queries using domain ontologies and leverage ontology reasoning for service matching. However, building such problem-specific ontologies and annotating services is time-consuming and sometimes impractical. The other group of approaches using machine learning or deep learning techniques for service matching. For example, topic models like Latent Dirichlet Allocation (LDA) [8] are introduced to obtain the topic distribution vectors of services and queries and calculate their similarities. Topic models are also combined with word embeddings [11] to alleviate the sparsity of semantic information by leveraging the advantages of embedding models in transforming words from discrete representations into high dimensional continuous vector space. Furthermore, many sequential models like LSTM [9] are also adopted as encoders of input texts.

These machine learning or deep learning techniques have made remarkable progress in many NLP (natural language processing) tasks [3, 4, 17]. However, there are still many obstacles to leveraging them in service discovery. On the one hand, the words in service descriptions and queries are limited and extremely sparse, which makes the complex models hard to extract effective feature vectors from input texts. On the other hand, the useful words extracted from service descriptions represented in service description languages like WSDL can hardly form a natural sentence and are lack of context information, which are necessary for the model training of these sequential models. To address this issue, we propose a novel deep learning-based service matching model. In our model, we enrich the semantics of each word in service descriptions and queries using external knowledge sources (e.g., Wikipedia). Because the enrichment process will inevitably bring noise, we leverage a neural topic model to extract topic distributions from enriched service descriptions and employ the attention mechanism in weighing the various words in texts according to their topic distributions. The contributions of this paper are summarized as follows:

- We propose a novel service discovery approach by combining Bi-LSTM and the neural topic model.
- We present a semantics enrichment technique to generate better topic distributions for service descriptions.

The rest of the paper is organized as follows. Section II discusses related work. Section III introduces the details of our approach, and Section IV shows the results of experiments. Finally, Section V concludes the paper and puts forward our future work.

II. RELATED WORK

As a fundamental topic in services computing, Web service discovery or matching has been extensively investigated in the

DOI reference number: 10.18293/SEKE2020-113

past decade. Web service discovery aims to identify appropriate services according to user requirements, which could be functional or nonfunctional. In this paper, we focus our attention on matching functional requirements. Existing studies on service discovery can be approximately categorized into two types: ontology-based approaches, and machine learning or deep learning-based approaches.

To overcome the limitation of keyword matching based on the structure of WSDL documents, many ontology-based matching techniques have been proposed. SAWSDL-MX [22] and OWLS-MX [1] are the representatives of this type. These approaches firstly annotate services and queries using domain ontologies and then leverage ontology-based semantical reasoning for service matching, which can obtain ideal discovery performance based on ontological reasoning. However, the major obstacle is the absence of appropriate ontologies for the matching tasks since general-purpose ontologies will not work, and it is time-consuming and sometimes impractical to construct such problem-specific ontologies and annotate services and queries using the ontologies.

With the prevalence of machine learning techniques, some researchers leverage machine learning algorithms in service discovery. At first, many clustering techniques are employed to group similar Web services in advance. For example, Liu et al. [3] and Elgazzar et al. [4] extracted functionality-related elements, including content, type, message, port, and service name from WSDL documents, which are regarded as the input of subsequent clustering and matching processes. Zhang et al. clustered service goals to improve discovery results [2, 5]. Recently, deep learning has become a mainstream tool for NLP tasks, and there are also some attempts to employ techniques such as word embedding in service discovery. For example, Tian et al. [7] combined word embedding and LDA to improve the performance of service discovery. Xiong et al. [17] leveraged the strategy in service recommendation to generate textual features from service descriptions.

To produce better feature vectors of service descriptions and queries, attention mechanisms, together with LSTM, are also widely adopted. For example, Cao et al. [15] applied attention mechanisms and LSTM in Web service classification. Shi et al. [16] leveraged attention-based LSTM in service recommendation. In their work, service descriptions are enriched internally to address the semantic sparsity. Besides, Yang et al. [14] proposed a service classification approach by combining CNN with LSTM.

Inspired by the studies that attempt to enrich service descriptions and queries [6], in this paper, we introduce explanations for each word according to its description on Wikipedia into service descriptions and queries as the external knowledge and enrich the descriptions to alleviate the data sparsity problem. What's more, in our model, a neural topic model based on VAE (variational auto-encoder) [20] is employed to extract topics, and an attention-based Bi-LSTM is used as the encoder of service descriptions, which can help obtain more accurate vector representations.

III. PROBLEM DEFINITION AND SOLUTION

In this section, we first state the problem to be studied. In Section III.B, an overview of our approach is presented. In the rest parts, we introduce modules of the proposed model in detail. In Table I, we list the symbols frequently used in this section and their corresponding meanings.

TABLE I. SYMBOLS USED IN THIS PAPER

Symbol	Meaning
S	A set of descriptions of services and queries
s_i	A description of a service or a query
X_{BoW}	The bag-of-words vector of s
E_s	The embedding matrix of s
H_s	The output hidden state matrix of s in LSTM
A_s	The weight vector of s
O_s	The feature vector of s
θ_s	The topic distribution of X_{Bow}
L	The number of words in s
dim	The dimension of word embeddings
V	The size of the vocabulary in S
K	The topic number in a topic model
k	The number of top-ranked services in $\widehat{R_S}$
h	The hidden size of LSTM
$\widehat{r_i}$	The matching score of s_i with a query
$\widehat{R_S}$	A list of matching scores of candidate services

A. *Problem Definition*

Suppose there are some candidate services in a registry. Given a user query, the problem to be addressed in this paper is how to properly match and rank services in the registry according to the user query. During this process, descriptions of services and queries (represented in WSDL or natural language texts) are the only information we can leverage. More formally, let O_q denote the feature vector of a query q and given a service description s_i (consisting of L words) in S, the feature vector O_i of s_i can be extracted. The matching score between s_i and q is $\widehat{r_i}=cosine(O_i, O_q)$. Consequently, the matching scores of all services are $\widehat{R_S} = \{\widehat{r_1}, \widehat{r_2}, \ldots, \widehat{r_n}\}$, and our proposed approach aims to return a list of services S_k with k highest scores.

B. *Overview of Our Approach*

Towards this problem, we proposed a Web service discovery approach based on attentional Bi-LSTM and enriched NTM (neural topic model), named as AENTM. As shown in Fig. 1, our model consists of three parts:

1) Neural Topic Model

Each description of a service or a query is represented as a bag-of-word vector X_{BoW}, and then the topic distribution θ_s is generated from X_{BoW} by a multi-layer perceptron. Besides, the module needs to reconstruct X_{BoW} from θ_s.

2) Attention-based Bi-LSTM Encoder

A description of a service or a query, s_i, is converted into an embedding matrix, and then fed into Bi-LSTM. The output is the hidden state matrix H_s, which contains the context feature of each word in s_i.

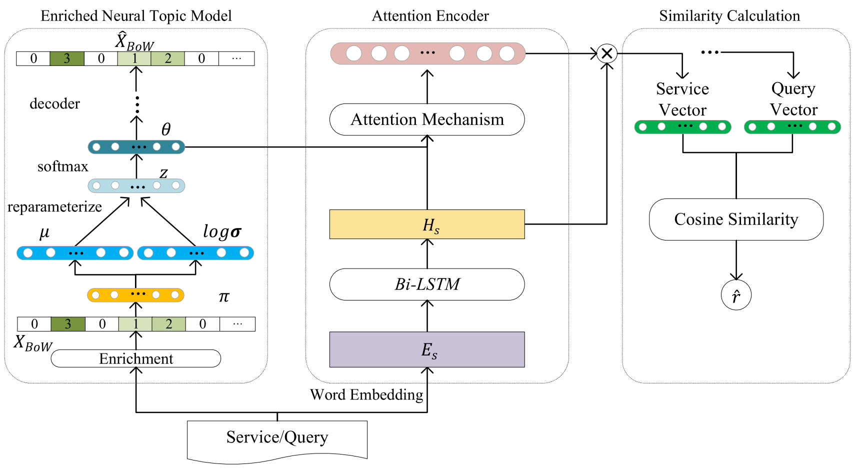

Figure 1. The overall framework of the proposed model.

An attention mechanism takes H_s along with θ_s as input, and produces a weight vector A_s for s_i. Consequently, the feature vector O_s is produced by the multiplication of A_s and O_s.

3) Similarity Calculation:

This module calculates the cosine similarity between feature vectors of service descriptions and queries, and the results are viewed as matching scores between them.

C. Enriched Neural Topic Model

NTM [20] is firstly proposed in a short text classification model to address data sparsity. According to the successful application of the neural variational inference mechanism in topic modeling [18, 19, 21], topic distributions generated by the neural topic model is analogous to Bayesian non-parametric topic models. Inspired by [20], the neural topic model in our approach is based on a variational automatic encoder trained on the overall corpus composed by descriptions of services and queries. Assume that the description of s_i consists of words $\{w_1, w_2, \dots, w_L\}$, it is represented as a bag-of-word vector X_{BoW} before being fed into the neural topic model. In the neural topic model, the topic vector extracted by the model is represented as $\theta \in \mathbb{R}^K$. Note that in this paper, we only describe the structure and the data flow. More details can be found in [19, 20].

The first layer of NTM extracts hidden vector π, from which the input of reparameterization is generated. Then, the latent vector z is the result of a reparameterization with parameter pair (μ, σ). Both μ and σ are generated from π through a multi-layer perceptron. To normalize z, a softmax function is applied to generate topic vector $\theta \in \mathbb{R}^K$. Given the vector θ, the following parts of the topic model are supposed to reconstruct the input vector X_{BoW}, which will be employed as a decoder.

The calculation process of the neural topic model is described as follows.

$$\pi = relu(\boldsymbol{W}^{\pi} \cdot X_{BoW} + b_{\pi}), \tag{1}$$

$$\mu = relu\left(\boldsymbol{W}^{\mu} \cdot \pi + b_{\mu}\right), \tag{2}$$

$$log\boldsymbol{\sigma} = relu(\boldsymbol{W}^{\sigma} \cdot \pi + b_{\sigma}), \tag{3}$$

$$Draw\ \boldsymbol{z} \sim \boldsymbol{\mathcal{N}}(\mu, \sigma^2), \tag{4}$$

$$\theta = softmax\left(relu(\boldsymbol{W}^{\boldsymbol{\theta}} \cdot \boldsymbol{z} + b_{\theta})\right), \tag{5}$$

$$\hat{X}_{BoW} = relu\left(\boldsymbol{W}^{\boldsymbol{\phi}} \cdot \theta + b_{\phi}\right), \tag{6}$$

where $\boldsymbol{W}^*$ and $\boldsymbol{b}^*$ are parameters to be learned, $relu$ is an activation function, and softmax is the normalized function for outputting topic distributions.

Due to the requirements of backward propagation, it is important to adjust Equation (4) such that gradients of parameters in the model are able to propagate. The reparameterization can be described as:

$$u = Draw\ u \sim \boldsymbol{\mathcal{N}}(0,1), \tag{7}$$

$$z = u * \sigma + \mu. \tag{8}$$

Equations (9) and (10) reveal the transformation from input to a latent topic distribution, which is regarded as an encoder in VAE. In addition, a decoder is employed to infer the output in the form of bag-of-words from θ. In our model, a multilayer perceptron is treated as a decoder, and according to basic VAE, the loss function is defined as:

$$\mathcal{L} = D_{KL}(q(z)||p(z|x)) - \mathbb{E}_{q(z)}[p(x|z)], \tag{9}$$

where $p(z|x)$ represents the distribution of z with the input x, $q(z)$ is the standard normal distribution, and $p(x|z)$ is the probability distribution output by the decoder when taking z as input.

As mentioned in the previous section, the descriptions of services and queries are extremely sparse. To alleviate the issue, we introduce explanations from Wikipedia for each word in the description. In other words, for each word w_i in a description, our approach will search its corresponding explanation page in Wikipedia and extract the first paragraph explaining w_i. Note that there are a few words that have no corresponding explanation pages, and we ignore these cases and do not enrich them.

The enrichment operation can thus bring auxiliary information to the NTM module, which aims to improve the quality of topic distribution as well as the attention module further.

D. *Attention-based Bi-LSTM*

LSTM is widely applied in extracting features from texts. Our model leverages a bi-directional LSTM (Bi-LSTM) to capture information in the context of each word in the input sequence. Furthermore, an attention mechanism is adopted to weigh each word in the input sequence. More specifically, with the latent topics induced by the NTM described previously and the output of Bi-LSTM, the attention module produces a weight vector in the output of Bi-LSTM. Finally, the weight vector A_s and the hidden states H_s are multiplied to obtain the feature vector of the service description or query s_i.

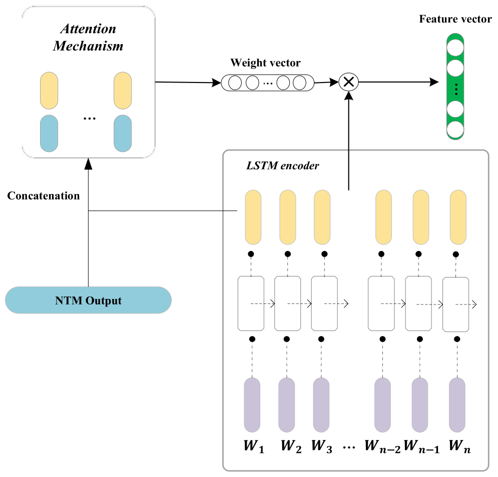

Figure 2. Attention-based Bi-LSTM Module

As shown in Fig 2, firstly, given a sequence of words $s_i = \{w_1, w_2, w_3, \dots, w_L\}$, we take the embedding of each word as the input of Bi-LSTM. The output of Bi-LSTM is the hidden state of each word. Next, hidden states of Bi-LSTM are concatenated with topic distribution θ_s, and then processed into an unnormalized weighted vector $a_s \in \mathbb{R}^{L\times 1}$.

$$E_s = embedding(s_i), \tag{12}$$

$$(\overleftarrow{H_s}, \overrightarrow{H_s}) = BiLSTM(E_s), \tag{13}$$

$$H_s = [\overleftarrow{H_s}, \overrightarrow{H_s}], \tag{14}$$

$$a_s = \boldsymbol{W^a} \cdot tanh(\boldsymbol{W^\theta} \cdot \theta_s + \boldsymbol{W^h} \cdot h_i), \tag{15}$$

$$A_s = softmax(a_s), \tag{16}$$

$$O_s = A_s^T \cdot H_s, \tag{17}$$

where $embedding$ denotes the operation that maps words to vectors using a pre-trained embedding model, and H_s is the result of the concatenation of two direction hidden states of Bi-LSTM $(\overleftarrow{H_s}, \overrightarrow{H_s})$. A_s in Equation (16) is the normalized a_s. As the result of Equation (17), $O_s \in \mathbb{R}^{L\times 2h}$ represents the feature vector of s_i, which is fed into the similarity module subsequently.

Note that the encoder part of queries and services share the same attention-based Bi-LSTM module.

E. *Similarity Calculation Module*

The similarity calculation module produces matching scores from the input feature vectors of services and queries. Since the corpus or words of queries and services can be processed together, services and queries share the same NTM as well as the same attention-based Bi-LSTM. We adopt the widely-used cosine similarity to calculate their matching scores:

$$\hat{r} = R \cdot cosine(O_s, O_q), \tag{18}$$

where O_s and O_q denote the feature vectors of a service and a query, respectively. In particular, the result of the cosine function belongs to the interval $[-1,1]$. Nevertheless, in some circumstances, different levels that measure the similarity between the query and the service may exist. Here, R is a coefficient that scales the output of the cosine function to match different situations.

IV. EXPERIMENTS

In this section, we evaluate our approach and explore the factors that influence the performance of our model on a public Web service dataset.

A. *Experimental settings*

1) *Dataset Description:* SAWSDL-TC[1] is a WSDL collection for the service retrieval test, which consists of 1080 Web services and 42 queries represented in WSDL documents. These services belong to nine domains. A set of graded relevance (ranging from 1 to 3) for each query is provided, where 3 represents the highest relevance and 1 represents the least relevance. We utilized several preprocessing steps, including spelling correction, tokenization, stopword removal, and lemmatization, to extract words from documents. We employed the GooleNews[2] as the pre-trained embedding. Words absent in the dictionary of pre-trained embeddings were removed in the descriptions. As mentioned before, we leveraged Wikipedia to provide external knowledge for words in descriptions.

2) *Evaluation Metrics:* We adopted several commonly used evaluation metrics, including Precision, Recall, F1, and Normalized Discounted Cumulative Gain (NDCG), to evaluate the performance of our model. We evaluated the performance of top *k* services in the ranking list, where *k* ranges from 5 to 30.

3) *Competing Approaches:* We compared our proposed model with several state-of-the-art approaches.

- LDA [8]: LDA is a representative topic modeling method. We employed LDA to generate topic distributions for service descriptions and calculated the cosine distances of topic distributions between queries and candidate service descriptions.
- Lucene[3]: Lucene is a popular and high-performance text search method, which is the basis of many search engines. In our experiments, service descriptions were indexed according to Lucene.
- Doc2Vec [13]: Doc2Vec is an unsupervised model based on Word2vec. We trained a doc2vec model on queries and service descriptions, and calculated the cosine distance between their feature vectors.

[1] http://projects.semwebcentral.org/projects/sawsdl-tc/
[2] https://code.google.com/archive/p/word2vec/
[3] https://www.elastic.co/

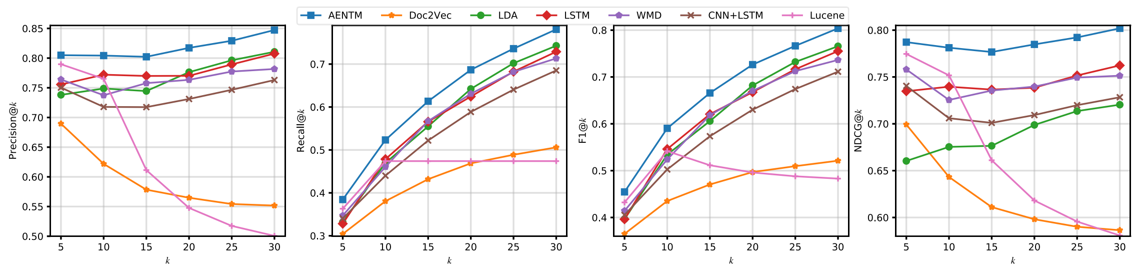

Figure 3. Performance comparison of different models

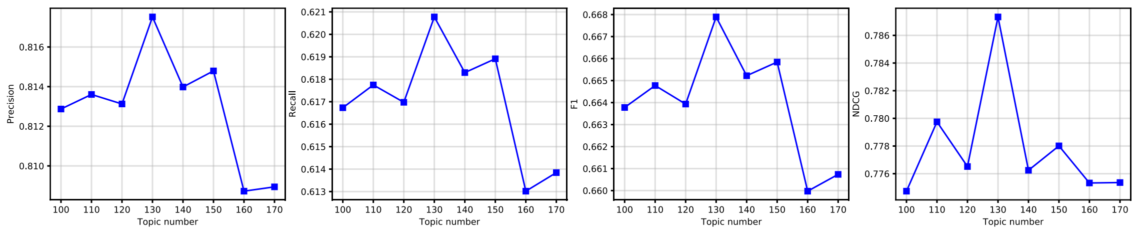

Figure 4. Impact of the topic number

- WMD [12]: WMD is a widely used method to measure the similarity of two documents or sentences based on a pre-trained word embedding. We used the pre-trained embeddings trained on the GoogleNews.
- LSTM [9]: To explore the effect of the attention mechanism and NTM module, we experimented on a Bi-LSTM with the same configuration in our model.
- CNN+LSTM [14]: CNN is a popular encoder in text-similarity calculation tasks [10]. In this experiment, we encoded texts by two CNNs, and then sent the output of the CNN encoder to a Bi-LSTM.

4) Parameter settings: The coefficient *R* in Equation (18) was set to 3 since the highest relevance level is 3. The learning rate for the encoder part was set to 0.0003 and trained with 30 epochs. The hidden size of Bi-LSTM was 150. To ensure the reliability of our experiment, we conducted a 5-fold validation on the dataset.

B. Performance comparison

According to the result presented in Fig. 3, our approach outperforms all competing methods across all ranking positions, which shows the advantages of our approach in addressing the data sparsity issue of Web services. More specifically, as a representative IR technique, Lucene suffers from a recall problem, and thus the performance decreases when *k* increases. It is indicated that LDA addresses the issue and achieves a good result in the precision, recall, and F1, but it still suffers from lower NDCG scores. Compared with LDA, deep learning approaches such as LSTM and WMD can achieve similar performance on F1, while they show better results on NDCG. Nevertheless, CNN+LSTM performs even worse than LDA, which indicates that the architecture is not very suitable for this task.

C. Impact of the topic number

We analyze the effects of parameter settings in our approach. The topic number of a topic model is a vital parameter that affects the performance as well as the quality of the topic distribution. In our approach, the topic number is still a hyperparameter that needs to be tuned on different datasets. Fig. 4 shows the change of performance when the topic number changes on the dataset. It is indicated that when the topic number is set to 130, our approach can obtain the best performance on all four metrics.

D. Response Time

Response time is an important consideration in service discovery. The response time of all approaches is shown in Table II. Note that the deep learning methods were performed with the help of a GPU (1080Ti), and other approaches were calculated on a CPU (Intel Xeon E5-2630).

TABLE II. RESPONSE TIME COMPARISON

Method	**Response Time (*ms*)**
LDA	428.74
LSTM	14.25
CNN+LSTM	23.36
AENTM	41.60
Lucene	2.29
WMD	4259.66
Doc2Vec	169.27

As shown in Table II, as a proven high-performance information retrieval technique, Lucene is the fastest approach. WMD costs the longest response time due to its high time complexity. The response time of LSTM, AENTM, and CNN+LSTM is relatively low. Although our AENTM model is relatively complicated, it does not take a much longer time to match services.

E. *Effects of enrichment*

As mentioned before, the services in the dataset belong to nine domains. To show the effects of the description enrichment, we reduced the topic distributions of descriptions (with 110-dimension vectors) generated by NTM to 2-dimension vectors, and then clustered them into nine clusters, as shown in Fig. 5. Moreover, the quality of clustering is usually measured by the CH score. We found that after the enrichment, the CH score increases from 2638 to 3180. Both the CH score and the visualization can demonstrate that after the enrichment of service descriptions, topic distributions generated by NTM become more distinguishable and cohesive, which further suggests that the enrichment can strengthen the semantics of descriptions to a certain extent.

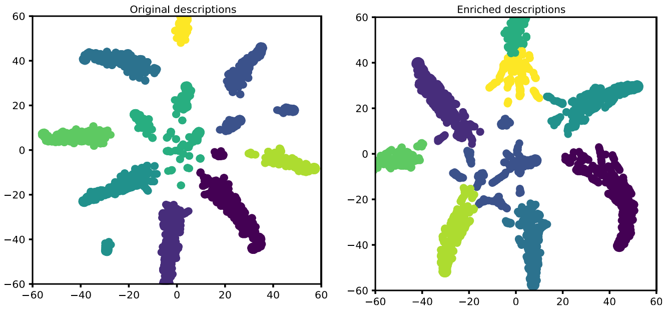

Figure 5. Change of topic clusters before/after the enrichment

V. Conclusions

In this paper, we propose a service discovery model by integrating enriched NTM with attention-based Bi-LSTM. An enrichment method is also adopted to leverage knowledge on Wikipedia to alleviate the lack of sufficient semantics in service descriptions. Experiments conducted on an open dataset show that our approach outperforms several state-of-the-art methods on discovery performance.

In the future, we plan to improve our approach from the following aspects. Firstly, the enrichment in our model employs the data on Wikipedia directly, which also brings lots of noise. The explanation should be selected precisely, and it may be helpful if techniques like attention mechanisms are applied. Secondly, in our experiment, the NTM module is hard to train due to the VAE part it leveraged, which needs to be further investigated.

Acknowledgment

This work is supported by the National Key R&D Program of China (No. 2018YFB1402800), the National Natural Science Foundation of China (No. 61832014), and the Natural Science Foundation of Hubei Province of China (Nos. 2017CKB894 and 2018CFB511).

References

[1] M. Klusch, B. Fries, and K. Sycara, "Automated semantic web service discovery with OWLS-MX," in Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, May 2006, pp. 915-922.

[2] N. Zhang, J. Wang, K. He, and Z. Li, "An approach of service discovery based on service goal clustering," in 2016 IEEE International Conference on Services Computing (SCC), June 2016, pp. 114-121. IEEE.

[3] W. Liu, and W. Wong, "Web service clustering using text mining techniques," IJAOSE, 3(1), 2009, pp. 6-26.

[4] K. Elgazzar, A.E. Hassan, and P. Martin, "Clustering wsdl documents to bootstrap the discovery of web services," in 2010 IEEE International Conference on Web Services, July 2010, pp. 147-154. IEEE.

[5] N. Zhang, J. Wang, Y. Ma, K. He, Z. Li, and X.F. Liu, "Web service discovery based on goal-oriented query expansion," Journal of Systems and Software, 142, 2018, pp. 73-91.

[6] X. Hu, N. Sun, C. Zhang, and T.S. Chua, "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, November 2009, pp. 919-928.

[7] G. Tian, J. Wang, Z. Zhao, and J. Liu, "Gaussian LDA and word embedding for semantic sparse web service discovery," in International Conference on Collaborative Computing: Networking, Applications and Worksharing, November 2016, pp. 48-59. Springer, Cham.

[8] D.M. Blei, A.Y. Ng, and M.I. Jordan, "Latent dirichlet allocation," Journal of machine Learning research, 3(Jan), 2003, pp. 993-1022.

[9] S. Hochreiter, and J. Schmidhuber, "Long short-term memory," Neural computation, 9(8), 1997, pp. 1735-1780.

[10] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A Convolutional Neural Network for Modelling Sentences," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014, pp. 655-665.

[11] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient estimation of word representations in vector space," ICLR Workshop, 2013.

[12] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in International conference on machine learning, 2015, pp. 957-966.

[13] Q. Le, and T. Mikolov, "Distributed representations of sentences and documents," in International conference on machine learning, 2014, pp. 1188-1196.

[14] Y. Yang, W. Ke, W. Wang, and Y. Zhao, "Deep Learning for Web Services Classification," in 2019 IEEE International Conference on Web Services (ICWS), 2019, pp. 440-442. IEEE.

[15] Y. Cao, J. Liu, B. Cao, M. Shi, Y. Wen, and Z. Peng, "Web services classification with topical attention based Bi-LSTM," in International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2019, pp. 394-407. Springer, Cham.

[16] M. Shi, and J. Liu, "Functional and contextual attention-based LSTM for service recommendation in Mashup creation," IEEE Transactions on Parallel and Distributed Systems, 30(5), 2018, pp. 1077-1090.

[17] R. Xiong, J. Wang, N. Zhang, and Y. Ma, "Deep hybrid collaborative filtering for web service recommendation," Expert systems with Applications, 110, 2018, pp. 191-205.

[18] Z. Cao, S. Li, Y. Liu, W. Li, and H. Ji, "A novel neural topic model and its supervised extension," in 29th AAAI Conference on Artificial Intelligence, IAAI, 2015, pp. 2210-2216.

[19] Y. Miao, E. Grefenstette, and P. Blunsom, "Discovering discrete latent topics with neural variational inference," in Proceedings of the 34th International Conference on Machine Learning-Volume 70, 2017, pp. 2410-2419. JMLR. org.

[20] J. Zeng, J. Li, Y. Song, C. Gao, M.R. Lyu, and I. King, "Topic Memory Networks for Short Text Classification," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 3120-3131.

[21] Y. Xiao, T. Zhao, and W.Y. Wang, "Dirichlet variational autoencoder for text modeling," arXiv preprint arXiv:1811.00135.

[22] M. Klusch, P. Kapahnke, and I. Zinnikus, "Hybrid adaptive web service selection with SAWSDL-MX and WSDL-analyzer," in European Semantic Web Conference, May 2009, pp. 550-564. Springer.