Web Services Classification with Topical Attention Based Bi-LSTM

Yingcheng Cao[1,2], Jianxun Liu[1,2(✉)], Buqing Cao[1,2], Min Shi[1,2], Yiping Wen[1,2], and Zhenlian Peng[1,2]

[1] Hunan University of Science and Technology, Xiangtan, China
caoyingchengl2l38@gmail.com, ljx529@gmail.com
[2] Key Laboratory of Knowledge Processing and Networked Manufacturing, HNUST, Xiangtan, China

Abstract. With the rapid growth of the number of Web services on the Internet, how to classify web services correctly and efficiently become more and more important in the development and application of Web services. Existing function-based service clustering techniques have some problems, such as the sparse document semantics, unconsidered word order and the context information, so the accuracy of service classification needs to be further improved. To address this problem, this paper exploits the attention mechanism to combine the local implicit state vector of Bi-LSTM and the global LDA topic vector, and proposes a method of Web services classification with topical attention based Bi-LSTM. Specifically, it uses Bi-LSTM to automatically learn the feature representation of Web service. Then, it utilizes the offline training to obtain the topic vector of Web service document and performs the topic attention strengthening processing for Web service feature representation, and obtains the importance or weight of the different words in Web service document. Finally, the enhanced Web service feature representation is used as the input of the softmax neural network layer to perform the classification prediction of Web service. The experimental results validate the efficiency and effectiveness of the proposed method.

Keywords: Web services · Bi-directional Long Short-Term Memory · LDA topic model · Web service classification · Attention model

1 Introduction

Web service is an application that exposes to the outside world as an API that can be called over the Web. With the rapid development of the Internet, a large number of Internet applications based on SOA (Service-oriented Architecture) have been created, and Web services have gradually become the mainstream technology for implementing SOA. Web services are published by service providers on private or shared Internet platforms. Users search in a flood of Web services in order to find those Web services can meet their actual business needs. In this process, users do not need to know the specific implementation of the Web services to get satisfactory results [1, 2].

X. Wang et al. (Eds.): CollaborateCom 2019, LNICST 292, pp. 394–407, 2019.
https://doi.org/10.1007/978-3-030-30146-0_27

With the rapid growth of the number and diversity of Web services on the Internet, it is necessary to manage Web services through Web services-based applications such as service discovery [3, 4] and service composition or Mashup [5, 6]. Hence, it is unrealistic to organize Web services manually. For such a large number of Web services, how to enable machines to automatically recognize, manage, and use Web services has been attracted from many researchers [7]. The first step in implementing automated management is to classify Web services correctly and efficiently. At present, the research on Web service classification mainly includes function-based service clustering and QoS-based service clustering. Among them, the main techniques of function-based service clustering are keyword extraction based on TF-IDF algorithm [8, 9], Web service document clustering based on K-Means algorithm [10], and document topic modelling based on LDA [11], etc. These methods and techniques improve the accuracy of service clustering, but we are still facing the following two problems:

- Considering the Web service description document is usually short and their corpus is limited, some investigations exploit the auxiliary information, such as tags and word clustering, to augment the service description document to optimize the service clustering process. However, the relevance of the expanded content is insufficient and semantics sparsity problem of service document still exists.
- Document modeling techniques, such as TF-IDF and LDA, only perform statistical learning on the probability of occurrence of each word in the service document, and do not utilize the word order and context information between words, which are essential to fully characterize services functional semantics.

In recent years, neural network models have been proved with great potential in various information processing tasks. They can learn effective feature representation of documents and achieve optimal performance in many natural language processing (NLP) tasks. Among a variety of neural network models, Bi-directional Long Short-Term Memory Network (BiLSTM) [12] is widely used since it can comprehensively consider context information in sequence information learning. Here, we hope to use the Bi-LSTM model to learn the feature representation of Web service documents via taking into account word order and contextual information. For this reason, inspired by the work of Li et al. [13], when learning the feature representation of Web services, we improve Bi-LSTM by incorporating topic modeling into the Bi-LSTM architecture through an attention mechanism. The topic distribution of Web service documents is incorporated into the learning process to discriminate the importance of different words in Web service documents. To this end, we propose a novel topical attention based Bi-LSTM for Web services classification, called as LAB-Bi-LSTM. This method firstly takes the description document and the topic distribution of the Web service as input and then outputs the feature vector of the Web service. Finally, a softmax neural network is used for the classification prediction of the feature vector of the Web service.

The rest of the paper is structured as follows. Section 2 presents the proposed model. The experiment is described in the Sect. 3. Section 4 introduces related work of service classification methods. Section 5 concludes this paper and gives the future work.

2 Proposed Model

In this paper, we consider the Web services classification as a text multi-classification problem. A common practice is to use Bi-LSTM to learn vector representations of Web service documents, and then to classify Web service documents based on this vector representation. However, a potential problem with this approach is that all the necessary information in the Web service document needs to be compressed into a vector with fixed-dimension size. The existing method is to perform an average pooling operation on the hidden state vector of Bi-LSTM. Despite great promise, we argue that Bi-LSTM can be hindered by its modelling of all words in Web service documents with the same weight which means that each word in the Web service document contributes to the generation of the document vector equally. In real-world applications, different words usually have different representation capability, not all words contain useful information for generation of the document vector. As such, the words with less useful features should be assigned a lower weight as they contribute less to the generation of the document vector. Nevertheless, Bi-LSTM lacks such capability of differentiating the importance of words, which may result in sub-optimal document vector. In this work, we improve Bi-LSTM by discriminating the importance of different words. We propose a novel model, named as LAB-BiLSTM, which utilizes the recent advance in neural network - the attention mechanism [14] to enable words to contribute differently to the generation of the document vector. More importantly, the importance of a word is automatically learned from data without any human domain knowledge. Firstly, through Bi-LSTM, the eigenvector representation of the Web service description document is learned. Next, the attention mechanism is used to capture the topic relevance of different words in the document. Finally, by modeling the local interactions between the words and the global topics, the proposed model can learn effective representations of services for classification. As shown in Fig. 1, the LAB-BiLSTM architecture is a multi-layer feedforward neural network. It includes five specific function layers: Input layer, Embedding layer, Bi-LSTM layer, Topical attention layer, and Output layer. In its forward propagation process, the output of each layer is used as the input of the next layer. The following parts describe each layer and its processing in detail.

2.1 Input Layer

Considering that Web services' tags contain a wealth of functional information, we take the Web Services' tags together with the description documents as the input of LAB-BiLSTM model. The description document of Web service a is denoted as $W^{(a)} = \left\{w_1^{(a)}, w_2^{(a)}, \cdots, w_{|W^{(a)}|}^{(a)}\right\}$, where $w_i^{(a)}$ is the i-th word of the document, and $\left|W^{(a)}\right|$ is the number of words. Tags annotated with Web service a are defined as $T^{(a)} = \{t_1, t_2, \ldots, t_m\}$, where t_i is the i-th tag of the Web service, and m is the number of tags. $Z^{(a)} = \left\{z_1^{(a)}, z_2^{(a)}, \cdots, z_K^{(a)}\right\}$ is the topic distribution of a, where K represents the number of topics, and $Z^{(a)}$ is a vector with a length K. Since the number of words and the number of tags in each Web service are all different, we fix the length of input sequence to N.

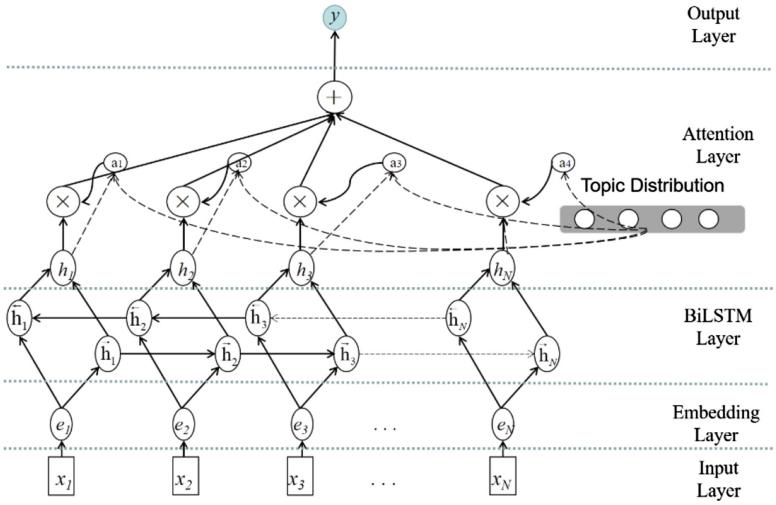


Fig. 1. The graphical illustration of the proposed topical attention-based BiLSTM model (LAB-BiLSTM)

Given an input sequence, it will be filled with the word "null" for a length less than N and the part whose length is greater than N will be truncated.

2.2 Embedding Layer

The embedding layer is above the input layer. Given a service document consisting of N words $S = \{x_1, x_2, ..., x_N\}$, every word x_i is converted into a real-valued vector e_i. For each word in S, we first look up the embedding matrix $w^{wrd} \in R^{d^w|V|}$, where V is a fixed-sized vocabulary, and d^w is the size of word embedding. The matrix w^{wrd} is a parameter to be learned, and d^w is a hyper-parameter to be chosen by user. We transform a word x_i into its word embedding e_i by using the matrix-vector product shown in the below formula:

$$e_i = W^{wrd} v^i \tag{1}$$

where v^i is a vector with size $|V|$ which has value 1 at index e_i and 0 in all other positions. Then the sentence is feed into the next layer as a real-valued vectors $emb_s = \{e_1, e_2, \ldots, e_N\}$.

2.3 Bi-LSTM Layer

Recurrent Neural Network (RNN) is a special kind of feed-forward neural networks that has gained significant success to tackle many problems in the NLP area, such as question answering [15] and sentiment prediction [16]. It is famous for its ability to process sequential data of arbitrary length like text. However, during the gradient computation step, RNN tends to suffer from the vanishing/exploding gradient problem

[17], which means when long-term dependency needs to be learned, the back-propagation algorithm will make the weight parameters too small or too huge, making it hard to learn complex functions. LSTM [12] was proposed to mitigate this issue by using a memory unit to keep the dependent information contained in sequential data for a long period of time. Figure 2 shows the structure of a single LSTM cell, where f_t, i_t and i_t represent the forget, input and output gates, respectively. And h_t indicates the cell output at time t, and o_t is the global cell state that enables the sharing of different cell outputs throughout the LSTM networks. These parameters are updated by:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{3}$$

$$g_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + W_{cc}c_{t-1} + b_c) \tag{4}$$

$$c_t = i_t g_t + f_t c_{t-1} \tag{5}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{6}$$

$$h_t = o_t \, tanh(c_t) \tag{7}$$

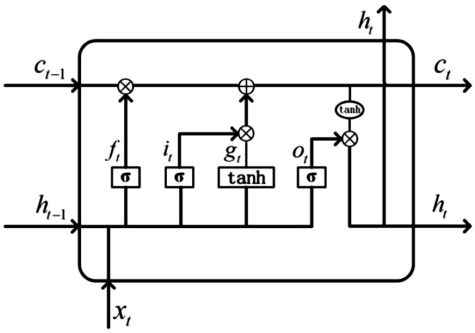


Fig. 2. Structure of the LSTM memory block

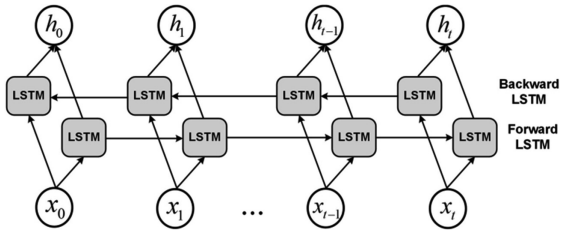


Fig. 3. The Bi-LSTM networks

However, in some situations like modeling description documents in this paper, it is beneficial to capture both the past and future dependencies of words. Bi-directional LSTM (Bi-LSTM) [12] is proposed for this purpose by introducing a second LSTM layer, where the hidden connections flow from the opposite direction. Figure 3 presents the network structure of Bi-LSTM, which contains two LSTM layers to model the sequential data from the opposite directions. For the t-th time step, the forward LSTM and the backward LSTM process the input from the opposite direction, and then output the hidden state vectors $\vec{h}_t$ and $\overleftarrow{h}_t$, respectively. There are many ways to combine $\vec{h}_t$ and $\overleftarrow{h}_t$. Here, we concatenate $\vec{h}_t$ and $\overleftarrow{h}_t$ to form the final representation h_t.

$$h_t = \vec{h}_t + \overleftarrow{h}_t \tag{8}$$

The output of Bi-LSTM layer is a sequence of hidden vectors $[h_1, h_2, \ldots, h_N]$. Each annotation h_t contains information about the whole input document with a strong focus on the parts surrounding the t-th word of the input document.

2.4 Topical Attention Layer

Since the attention mechanism has been introduced to neural network modelling, it has been widely used in many tasks [14]. Its main idea is to allow different parts to contribute differently when compressing them to a single representation. Motivated by the limitation of Bi-LSTM, we propose to employ the attention mechanism on hidden vectors $[h_1, h_2, \ldots, h_N]$ by introducing a series of attention-weighted combinations of these hidden vectors using the external topic distribution.

Taking all hidden states $[h_1, h_2, \ldots, h_N]$ and the external topic vector $\theta_s \in R^{K\times 1}$ as input, the topical attention layer outputs a continuous context vector $vec \in R^{d\times 1}$ for each input document. The output vector is computed as a weighted sum of each hidden state h_j:

$$vec = \sum\nolimits_{j=1}^{N} a_j h_j \tag{9}$$

where d is the hidden dimension of Bi-LSTM, $a_j \in [0, 1]$ is the attention weight of h_j and $\sum\limits_j a_j = 1$.

Next, we will introduce how we obtain $[a_1, a_2, \ldots, a_N]$ in detail. Specifically, for each h_j, we use the following equation to compute scores on how well the inputs around position j match the topic distribution θ_s:

$$g_j = v_a^T \tanh(W^a \theta_s + U^a h_j) \tag{10}$$

where K is the number of topics, $W^a \in R^{d\times K}$, $v_a \in R^{d\times 1}$ and $U^a \in R^{d\times d}$ are the weight matrices. After obtaining $[g_1, g_2, \ldots, g_N]$, we feed them to a softmax function to calculate the final weight scores $[a_1, a_2, \ldots, a_N]$.

2.5 Output Layer

Finally, we use the output from the topical attention layer as the embedding of the service document from our LAB-BiLSTM. We feed the output vector *vec* to a linear layer whose output length is the number of service categories. Then a softmax layer is added to output the probability distributions of all candidate categories. The softmax function is calculated as follows, where M is the number of services categories:

$$softmax(m_i) = \frac{exp(m_i)}{\sum_{i'}^{M} exp(m_i)} \tag{11}$$

3 Experiment

3.1 Dataset Description

To evaluate the proposed approach, we crawled a Web service dataset from ProgrammableWeb.com and obtained 12919 API services as well as their related information, and basic statistics of it are shown in Table 1. This crawled dataset is available at http://kpnm.hnust.cn/xstdset.html. There are totally 384 categories for 12919 Web services and the average size of each category is 33.73. The number of Web services in each category is severely uneven. For example, the category Tools contains 790 Web services while the category law contains 1 service only. Table 2 shows the detailed distribution data of the top 10 Web services categories.

Table 1. APIs statistics in the web service dataset

Items	Values
Number of APIs	12919
Number of categories	384
Average number of member APIs per category	33.64
Average number of tags per APIs	3.46
Total number of tags	44734

Table 2. Top 10 categories order by number

Category	Number	Category	Number
Tools	790	Messaging	388
Financial	586	Payments	374
Enterprise	487	Government	306
eCommerce	435	Mapping	295
Social	405	Science	287

3.2 Evaluation Metrics

In our experiments, we evaluate the clustering performance by three metrics, *i.e.*, precision, recall, and F-smeasure. Suppose the standard classification of Web services in top M categories as $RSC = \{RC_1, RC_2, \ldots, RC_M\}$, which is available in the crawled dataset. We represent the experimental Web services classification results as $ESC = \{EC_1, EC_2, \ldots, EC_V\}$. The precision and recall metrics are defined as follows:

$$recall(EC_i) = \frac{|EC_i \cap RC_i|}{|RC_i|} \tag{12}$$

$$precision(EC_i) = \frac{|EC_i \cap RC_i|}{|EC_i|} \tag{13}$$

where $|EC_i|$ is the number of Web services in category EC_i, $|RC_i|$ is the number of Web services in RC_i and $|EC_i \cap RC_i|$ is the number of Web services successfully placed into category RC_i.

The average precision and average recall for all Web service categories are respectively:

$$Precison = \frac{1}{M}\sum_{C=1}^{M} precison(EC_i) \tag{14}$$

$$Recall = \frac{1}{M}\sum_{C=1}^{M} recall(EC_i) \tag{15}$$

F-measure: a tradeoff value between the recall and precision, which is denoted as:

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{16}$$

3.3 Baselines

In this section, we compare our approach with the following approaches to verify the effectiveness of the proposed approach.

[1] **Naive Bayes:** The Naive Bayes classifier is a simple classification method based on Bayes theory, which shows excellent performance in many fields. Here, all the words in the Web service description document are one-hot encoded, then the description document of the Web service is converted into a digital sequence as input to the classifier.

[2] **LDA-SVM:** Support vector machine is a common classification method. It is a supervised learning model that is commonly used for pattern recognition, classification, and regression analysis. First, the description text of the Web service is modeled by the LDA topic model, and the topic distribution vector of each Web service document is obtained, then they are used as the input of the SVM for classification prediction.

[3] **Bi-LSTM:** We regard the last hidden vector from Bi-LSTM as the service document representation. Then we feed it to a linear layer whose output length is the number of categories. Finally, a softmax layer is added to output the probability distributions of all candidate categories.

[4] **Tag-BiLSTM:** On the basis of Bi-LSTM, it exploits tags as auxiliary information of the Web service description document to perform Web services classification. The tag contains a wealth of information about the Web service features, which helps improve the result of service classification.

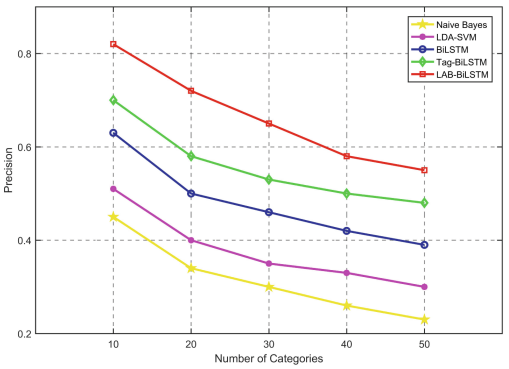
[5] **LAB-BiLSTM:** The proposed method in this paper, which incorporates topic modeling into the Bi-LSTM architecture through an attention mechanism for service classification.

3.4 Experimental Results

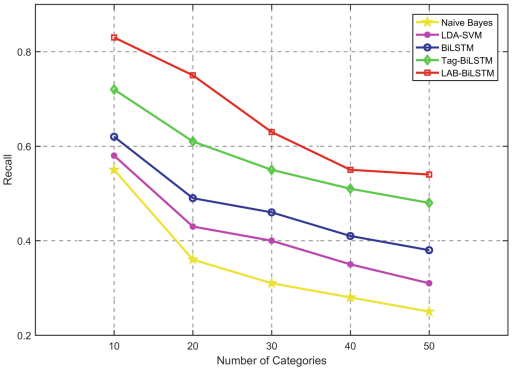
Experimental Setup. In the experiment, we chose 30% data as the training set and 70% data as the test set. As for Bi-LSTM, Tag-BiLSTM and LAB-BiLSTM, we use a same minibatch stochastic gradient descent (SGD) algorithm together with the Adam method to train their model [18]. The hyperparameters β_1 is set to 0.9 and β_2 is set to 0.999. The learning rate is equal to 0.001, and the batch size is equal to 100. For LAB-BiLSTM, we test with different numbers of LDA topic size K and find an optimal setting when $K = 20$.

Performance Comparison. In this section, we select the top 10, 20, 30, 40, and 50 categories of Web services to conduct performance comparison, respectively. The experimental results are shown in Fig. 4. It can be seen that the method proposed in this paper LAB-BLSTM is superior to the other four methods in terms of precision, recall and F-measure. When the number of service categories is 20, LAB-BiLSTM has 38%, 32%, 24%, and 14% improvement in F-measure compared to Naive Bayes, LDA-SVM, BiLSTM, and Tag-BiLSTM, respectively. Specially, we have the following observations:

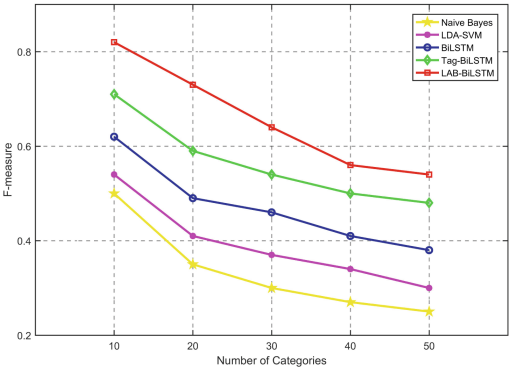
- The neural network-based models (*i.e.*, Bi-LSTM, Tag-BiLSTM, and LAB-BiLSTM) are far superior to Naive Bayes and LDA-SVM. It indicates that deep neural networks can achieve better classification results than traditional machine learning methods.
- Tag-BiLSTM has a significant increase in F-measure value of nearly 10% compared to Bi-LSTM. This is because the tags of the Web service are added as auxiliary information to the service description document, which greatly improves service classification performance.
- Compared to Tag-BiLSTM, LAB-BiLSTM has an increase of 14% in F-measure, which indicates that the introduction of LDA topic attention mechanism is indeed beneficial to Web services classification. In the LAB-BiLSTM method, words in the Web service description document that are highly relevant to the topic are gained greater attention weight, and these words become keywords in generating document-level feature vector representations.



(a) Precision



(b) Recall



(c) F-measure

Fig. 4. Performance comparisons of all the baseline methods listed.

Hyper-Parameter Investigation. Firstly, taking 10 service categories as an example, we select different LDA topics (i.e., 10, 15, 20, 30, and 50) for Web service classification experiments. The precision, recall, and F-measure obtained are shown in Table 3 below. It can be seen from the experimental results that our model shows the best performance when the number of topics is 20. When the number of topics decreases or increases, the service classification effect decreases accordingly. Therefore, choosing the appropriate number of topics is still important for service classification.

Table 3. Performance *w.r.t* different number of topics *K*

K	Precision	Recall	F-measure
10	0.7777	0.7764	0.7770
15	0.7963	0.7806	0.7883
20	0.8106	0.8074	0.8071
30	0.8012	0.8001	0.8006
50	0.7837	0.7769	0.7802

Secondly, taking top 10 service categories as an example, different word embedding dimensions (i.e., 64, 128, 256 and 512) are selected for Web service classification experiments. The precision, recall and F-measure obtained are shown in Table 4 below. It can be seen from the experimental results that the word embedding dimension has limited influence on the classification effect of Web services. Moreover, the increasing of word embedding dimension can slightly improve the service classification effect, but the time complexity of the model will also increase greatly.

Table 4. Performance *w.r.t* different embedding size

Embedding size	Precision	Recall	F-measure
64	0.8077	0.8064	0.8037
128	0.8106	0.8074	0.8071
256	0.8249	0.8219	0.8224
512	0.8289	0.8284	0.8281

4 Related Work

With the development of service computing and cloud computing, a variety of network services have emerged on the Internet. Among them, the discovery and mining of Web services has become a hot research direction. Some research works show that efficient Web service classification can effectively improve the performance of Web service discovery [19]. As investigated, the researches on automatic Web services classification have attracted the attention of many researchers, mainly based on function-based service clustering and QoS-based service clustering.

At present, there are a lot of researches on function-based service clustering methods. Among them, Web service clustering method based on functional similarity [8, 9], which uses the key features of Web services extracted from WSDL documents to discriminate the similarity between Web services according to the cosine similarity. Liu et al. [20] proposed to extract content, context, host name and service name from Web service description document to perform Web service clustering. Huang et al. [10] proposed a K-Means algorithm based on Mashup service similarity for service clustering based on the description document and corresponding tags of Mashup service. Shi et al. [11] proposed an enhanced LDA topic modeling algorithm for Web service clustering. The word vectors trained by word2vec [21] were clustered to obtain clusters of words and merged into LDA training process. However, some obvious issues of the above approaches exist: (1) only utilizing services description documents; or (2) do not utilize the word order and context information between words, which are essential to fully characterize services functional semantics. Our approach incorporates topic modeling into the Bi-LSTM architecture through an attention mechanism and takes over the advantages of the both.

In addition, QoS-based Web service clustering mainly used the QoS (Quality of Service), which includes throughput, availability, execution time, and so on, to perform service clustering. Xia et al. proposed to cluster large number of atomic Web services into many groups according to their QoS properties [22], which takes non-functional properties such as cost, execution time and reliability into account. Xiong et al. [23] used the long short-term memory neural network LSTM model and collaborative filtering algorithm to predict the QoS value of Web services from the historical call records of services. Wang et al. [24] designed an online QoS prediction method that uses the LSTM model to learn and predict the reliability of the service system in the future. However, QoS of services are dynamic with time and many existing methods fail to take this property into account. In addition, QoS are usually hard to be obtained. Therefore, there are many works focus on content-based or functional-based services clustering, as we do in this paper.

5 Conclusion and Future Work

This paper proposes a novel topical attention-based Bi-LSTM model for Web service classification. We first adopt the architecture of Bi-LSTM to capture the most important semantic information in a service document. Then, our model incorporates topic modeling into the Bi-LSTM architecture through an attention mechanism and takes over the advantages of the both. We evaluate the proposed approach on a real-world dataset and the experimental results show it has an improvement of 34% in F-measure over the standard Bi-LSTM model. In the future, we will consider to exploit other valuable auxiliary information, such as service relationship information, into our model to further improve the accuracy of service classification.

Acknowledgment. This work was supported in part by the National Natural Science Foundation of China under Grant 61572187, Grant 61872139, and Grant 61702181, in part by the Natural Science Foundation of Hunan Province under Grant 2017JJ2098, Grant 2018JJ2136, and Grant 2018JJ2139, and in part by the Educational Commission of Hunan Province of China under Grant 17C0642.

References

1. Zhang, L., Zhang, J., Hong, C.: Service-oriented architecture. Serv. Comput. 89–113 (2007)
2. Gottschalk, K., Graham, S., Kreger, H.: Introduction to web services architecture. IBM Syst. J. **41**(2), 170–177 (2002)
3. Nayak, R., Lee, B.: Web service discovery with additional semantics and clustering. In: International Conference on Web Intelligence, pp. 555–558 (2007)
4. Zhang, X., Yin, Y., Zhang, M.: Web service community discovery based on spectrum clustering. In: International Conference on Computational Intelligence and Security, pp. 187–191 (2009)
5. Zhang, L.J., Li, B.: Requirements driven dynamic services composition for web services and grid solutions. J. Grid Comput. **2**(2), 121–140 (2004)
6. Dustdar, S., Schreiner, W.: A survey on web services composition. Int. J. Web Grid Serv. **1**(1), 1–30 (2005)
7. Bruno, M., Canfora, G., Penta, M.D.: An approach to support web service classification and annotation. In: IEEE International Conference on E-Technology, E-Commerce and E-Service, pp. 138–143 (2005)
8. Chen, L., Hu, L., Zheng, Z., Wu, J., Yin, J., Li, Y., Deng, S.: WTCluster: utilizing tags for web services clustering. In: Kappel, G., Maamar, Z., Motahari-Nezhad, Hamid R. (eds.) ICSOC 2011. LNCS, vol. 7084, pp. 204–218. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25535-9_14
9. Elgazzar, K., Hassan, A.E., Martin, P.: Clustering WSDL documents to bootstrap the discovery of web services. In: IEEE International Conference on Web Services, pp. 147–154 (2010)
10. Huang, X., Liu, X., Cao, B.: MSCA: mashup service clustering approach integrating K-means and agnes algorithms. J. Chin. Mini-Micro Comput. Syst. **36**(11), 2492–2497 (2015)
11. Shi, M., Liu, J., Zhou, D.: WE-LDA: a word embeddings augmented LDA model for web services clustering. In: IEEE International Conference on Web Services, pp. 9–16 (2017)
12. Zhou, P., Shi, W., Tian, J.: Attention-based bidirectional long short-term memory networks for relation classification. In: Meeting of the Association for Computational Linguistics, pp. 207–212 (2016)
13. Li, Y., Liu, T., Jiang, J., Zhang, L.: Hashtag recommendation with topical attention-based LSTM. In Proceedings of COLING, pp. 943–952 (2016)
14. Chen, J., Zhang, H., He, X., Nie, L., Liu, W., Chua, T.: Attentive collaborative filtering: Multimedia recommendation with feature- and item-level attention. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 335–344 (2017)
15. Feng, M., Xiang, B., Glass, M.R., Wang, L., Zhou, B.: Applying deep learning to answer selection: a study and an open task. In: IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 813–820 (2015)

16. Tang, D., Qin, B., Liu, T.: Document modeling with gated recurrent neural network for sentiment classification. In: Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1422–1432 (2015)
17. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**, 157–166 (1994)
18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. Comput. Sci. (2014)
19. 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). https://doi.org/10.1007/978-3-642-45005-1_12
20. Liu, W., Wong, W.: Web service clustering using text mining techniques. Int. J. Agent-Oriented Softw. Eng. **3**(1), 6–26 (2009)
21. Mikolov, T., Chen, K., Corrado, G.: Efficient estimation of word representations in vector space. Comput. Sci. (2013)
22. Xia, Y., Chen, P., Bao, L., Wang, M., Yang J.: A QoS-aware web service selection algorithm based on clustering. In: IEEE 9th International Conference on Web Services, pp. 428–435 (2011)
23. Xiong, W., Wu, Z., Li, B.: A learning approach to QoS prediction via multi-dimensional context. In: International Conference on Web Services, pp. 164–171 (2017)
24. Wang, H., Yang, Z., Yu, Q.: Online reliability prediction via long shortterm memory for service-oriented system. In: International Conference on Web Services, pp. 81–88 (2017)