Web Service Discovery Based on Information Gain Theory and BiLSTM with Attention Mechanism

Xiangping Zhang, Jianxun Liu, Buqing Cao(✉), Qiaoxiang Xiao, and Yiping Wen

Key Laboratory of Knowledge Processing and Networked Manufacturing, Hunan University of Science and Technology, Xiangtan, Hunan, China
zxpkpnm@gmail.com, ljx529@gmail.com, buqingcao@gmail.com, l8390219693@163.com, ypwen81@gmail.com

Abstract. Web service discovery is an important problem in service-oriented computing with the increasing number of Web services. Clustering or classifying Web services according to their functionalities has been proved to be an effective way to Web service discovery. Recently, semantic-based Web services clustering exploits topic model to extract latent topic features of Web services description document to improve the accuracy of service clustering and discovery. However, most of them don't consider deep and fine-grained level information of description document, such as the weight (importance) for each word or the word order. While the deep and fine-grained level information can be fully used to argument service clustering and discovery. To address this problem, we proposed a Web service discovery approach based on information gain theory and BiLSTM with attention mechanism. This method firstly obtains the effective words through information gain theory and then adds them to an attention-based BiLSTM neural network for Web service clustering. The comparative experiments are performed on ProgrammableWeb dataset, and the results show that a significant improvement is achieved for our proposed method, compared with baseline methods.

Keywords: Web service clustering · Mashup creation · Information gain · Attention layer · BiLSTM

1 Introduction

With the wide adoption of Service-Oriented Architecture (SOA), the quantity and diversity of published web services on the Internet have been rapidly growing [1]. For example, programmableweb.com, the largest API services repository, has collected almost 19000 API services with various functionalities. This provides a convenient way for developers to build up software by using and compositing existing services. However, developers have to spend a lot of time for searching suitable Web services to complete the development process of software if they don't know what the functionalities of the Web services is [2]. In order to reduce software development cycle,

H. Gao et al. (Eds.): CollaborateCom 2018, LNICST 268, pp. 643–658, 2019.
https://doi.org/10.1007/978-3-030-12981-1_45

mashup technology has emerged as a promising Web services development method, which allows Web service developers to create new or value-added Web services with existing Web APIs [3]. The key challenge of creating a new mashup is that find appropriate web services according to user's complex functional requirements.

Web service discovery is to identify the most relevant services in response to a requester's query by comparing Web services and the requester's requirement [21]. Clustering or classifying Web services according to their functionalities has been proved to be an effective way to Web service discovery. The Web services are grouped into clusters if they are similar in functionality aspect, which can improve the retrieval efficiency and avoid to return a set of Web services not similar to user's query term. But the difficulty of Web services discovery is still existing, that service providers and requesters have different perspectives and knowledge of the same service. For example, a service was classified as "education" by the provider while a requester may send query term "teaching", this may make the discovery engine can't return a satisfying searching result. However, it's impractical to expect that service providers and requesters share an identical understanding of a Web service.

To solve this problem, several semantic-based Web service discovery frameworks are proposed. They consider that the contextual information of Web service description documents to improve Web service discovery. For example, [4] exploits Latent Dirichlet Allocation (LDA) to extract the latent topic information of Web service description documents for Web service discovery. Several of them have integrated functional documents and user-contributed tagging information to enhance the ability of LDA for service clustering and mining [4]. [21] proposes a sentence representation model to consider both the word-level semantics and the sentence-level semantics with BiLSTM for text classification. [24] have propose a hierarchical attention network for document classification. However, most of them don't consider deep and fine-grained level information of description document, such as the weight (importance) for each word or the word order, the word order information of the documents. We consider that the deep and fine-grained level information can be exploited to improve the performance of service clustering. In our prior work [10], we use the accurate word clusters information extracted from the original description to enhance the ability of LDA model. On this basis, we propose a Web service discovery method based on information gain theory and BiLSTM with attention mechanism, named as IGBA. To be our best knowledge, this is the first work combines information gain theory and BiLSTM with attention mechanism for Web service clustering. The contributions of this paper are summarized as follows:

- We perform information gain theory to extract the importance of each word in the description document. And we rank them according to the value size to make an effective words list. Each word of Web service description document will be assigned a score by information gain what can be considered as a weight represents the importance to the description document, and an effective words list is archived.
- We exploit an attention-based BiLSTM to argument the performance of Web service clustering. The model combines the original Web service description documents and the effective words list together for modeling, it can capture the deep

information, such as the weight for each word and the word order in description documents which is valuable for Web service clustering.
- We conduct comparative experiments on a real-world dataset which is crawled from programmableweb.com. The experimental results indicate that the proposed approach indeed improve the accuracy of Web services discovery.

The rest of this paper is organized as follows: Sect. 2 presents related work on Web services discovery technologies. Section 3 describes our approach in details. Section 4 presents experimental evaluation and results. Finally, Sect. 5 concludes the paper and discusses some future works.

2 Related Work

Web services discovery is a process of retrieving appropriate Web services from service repository to satisfy users' requirement. Web services providers or developers publish their Web services in services repository with their description documents. Then Web services search engine can use the information in which the description documents contain to match the users' requirements. Generally speaking, existing literatures can be roughly classified into two categories: keyword-based Web services discovery methods [7], semantic-based Web services discovery methods [6].

Keyword-based Web services discovery methods depend on selecting appropriate keywords and making a query that matches the selected keywords with the Web service descriptions [7]. [25] which leverage key features being mined from the WSDL documents to represent the functionalities of a service. Due to the different knowledge background between providers and requesters it is difficult for them to select suitable keywords. Besides, keywords can't capture the latent semantics of Web services description documents, which may lead to some irrelevant searching results.

Semantic-based Web services discovery methods usually extract some important the information from Web services description through topic model, such as Latent Dirichlet Allocation (LDA) topic model for Web services clustering [8]. Liu and Fulia [22] incorporated user, topic, and service related latent factors into service discovery and recommendation by combining topic model and matrix factorization. Cao et al. [3] propose a Web service clustering method based on an integration of service content and network via exploiting a two-level topic model. Chen et al. [5] integrates tagging data and WSDL documents through augmented LDA to model Web services for clustering. Shi et al. [10] leverages the accurate word clusters information obtained by Word2vec to enhance the ability of LDA topic model then cluster Web services together if they are similar. However, due to the description documents of Web services usually contain limited numbers of terms which are even not useful or valuable words, these approaches may lead to low clustering results [11]. Actually, the latent semantic correlation behind the terms of service document can be exploited to improve the accuracy problem. In addition, LDA-based clustering methods are inappropriate for modeling short text which the description documents of Web services are.

Recently, many deep learning methods have been proposed to solve the text classification task [23]. [21] propose a model for extracting a sentence representation

that considers both the word-level semantics and the sentence-level semantics with BiLSTM for text classification. Our work differs from them in that we firstly use information gain theory to capture the top-N words which have contain most useful and valuable information from the original Web service description documents to enhance the performance of Web service discovery. In this paper, we model the original Web service documents and the effective words together by using BiLSTM neural network with attention mechanism that can capture the word with a high value weight and the information of word order. The deep and fine-grained level information can improve the performance of service clustering effectively.

3 Methodology Overview

This section consist of four sub-sections, respectively describes the architecture, pre-process, information gain theory and neural network model of the proposed method with details.

3.1 The Architecture of Web Service Discovery

The overall architecture of our Web service discovery method is shown in Fig. 1. In this architecture, we first crawl the web service data which contain the web service name, description and category from programmableweb.com. Then the Web services description documents for each web services are extracted and the information gain algorithm are performed for them to distill the effective words which contains more useful and valuable information for Web services clustering. Then, the original document and the effective words are integrated as the input of the attention-based BiLSTM neural network model. Finally, the web services with similar functionality are classified into the same cluster.

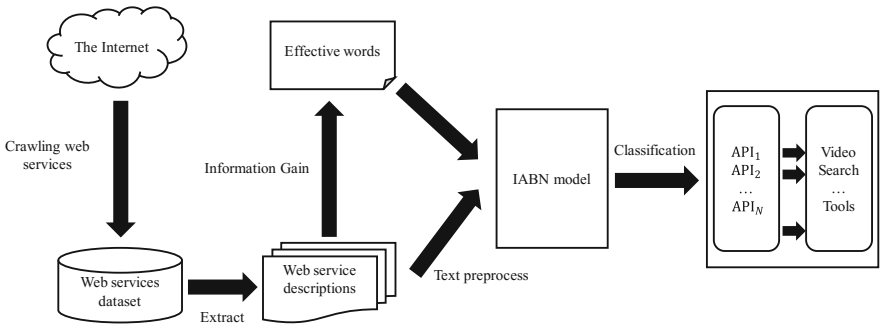

Fig. 1. The architecture of web services discovery approach

3.2 Web Service Document Preprocess

The Web service developer usually provides a brief description document about the Web services' function when publishing it on service repository, such as, programmableweb.com. As we can see in Fig. 2, it's a structure diagram of a Web service, including its name, description document and category.

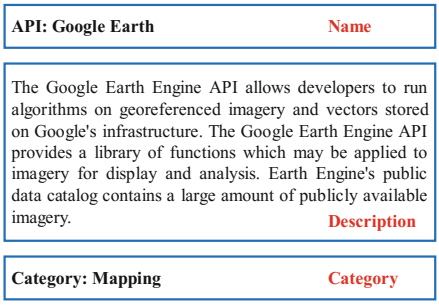

Fig. 2. API structure diagram

However, the Web service description documents always contain lots of noise which negatively affects text analysis. We should clear the description documents first. We extract all Web service description documents from the crawled dataset and concatenate them to a text file. Then we denote it by $D = \{d_1, d_2, \ldots, d_n\}$, n is the number of documents. In order to get a better performance, we perform a text preprocess for it, which contains below steps:

- Tokenization. Chopping a sentence into pieces, each piece is a word or a punctuation mark.
- Remove stop words. There are some common short function words which contain little lexical meaning but occur frequently in the Web services description documents. Those words can't afford useful information for web service discovery, such as, *the, in, a, an, with,* those words should be removed.
- Stemming. For grammatical reasons, documents have to use different forms of a word, such as *provide*, *providing*, *provides*. The common word endings for English words, such as *es*, *ed* and *ing* should be removed. Then *providing* become *provid*.

After performing text preprocessing, we obtain the processed Web services documents for further steps.

3.3 Information Gain

As shown in the original Web services description documents in the Fig. 2, those words, such as "*the*", "*to*", "*and*", which can't afford enough information for classifying the API "*Google Earth*" into category Mapping obviously. By contrast, when we observe those words, such as "*earth*", "*engine*", "*display*", we conjecture the API

"*Google Earth*" can be classified into category Mapping naturally. Information gain (IG) is the difference of the information entropy that a characteristic word appears or does not appear in the text documents [12]. The IG score is greater means that the information carried by the characteristic word is greater. IG score for each word is calculated using the formula below:

$$IG(w) = -\sum_{i=1}^{M} P(C_i) logP(C_i) + P(w) \sum_{i=1}^{M} P(C_i|w) logP(C_i|w) + P(\bar{w}) \quad (1)$$

where M is the number of categories, $P(C_i)$ is the probability of class C_i, $P(w)$ and $P(\bar{w})$ are respectively the probabilities of presence and absence of word w, $P(C_i|w)$ and $P(C_i|\bar{w})$ are the conditional probabilities of class C_i given presence and absence of word w, respectively. Then we can get the effective words for augmenting the model we proposed. We set up an experiment to study how to select the number of effective words appropriately.

3.4 The Neural Network Model of Web Service Clustering

Inspired by the success of neural network model work on text classification [20], we present an attention-based BiLSTM approach to classify Web services into different clusters bases their description documents. In this section, we first give an overview of our attention-based BiLSTM model for web service clustering. The architecture of our model is illustrated in Fig. 3, which contains Input layer, Embedding layer, LSTM layer, Attention layer, and Output layer.

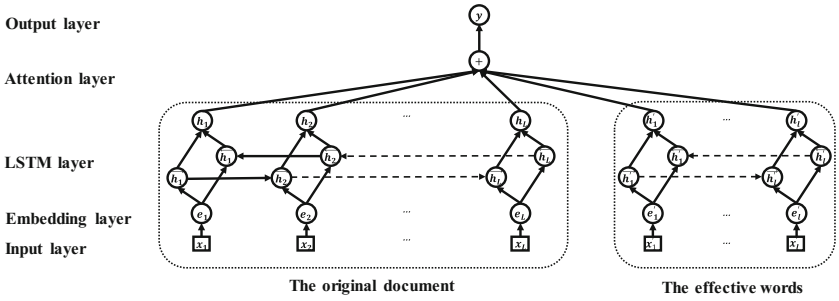

Fig. 3. The architecture of IGBA model

(1) *Input layer:* the original Web service description document which have been preprocessed and the extracted effective words are connected together as input data of BiLSTM model in the input layer.
(2) *Embedding layer:* we concatenate the Web service description document $d_i = \{x_{i1}, x_{i2}, \ldots, x_{iL}\}$ and its effective words $E_i = \{x'_{i1}, x'_{i2}, \ldots, x'_{il}\}$ as a sentence S_i. We use the pretrained model from Wikipedia corpus to map each word into a

fixed length vector by word2vec [19]. For each word x_{ij} in S_i, we transform into its word embedding e_{ij}. Then S_i is feed into the next layer as a real-valued vector $emb_i = \{e_{i1}, e_{i2}, \ldots, e_{i(L+l)}\}$.

(3) *LSTM layer:* LSTM units are firstly proposed to overcome gradient vanishing problem. The main idea of it is to introduce an adaptive gating mechanism, which decides the degree to which LSTM units keep the previous state and memorize the extracted features of current data input. The flow of data is shown in Fig. 4 [13].

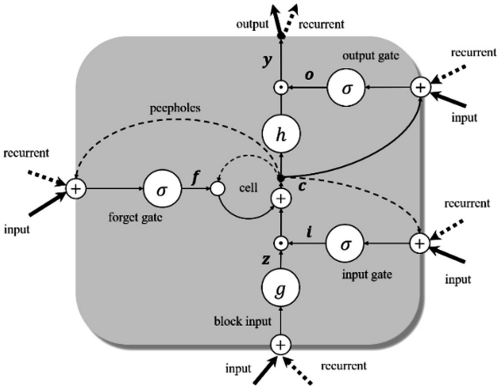

Fig. 4. LSTM unit

Let e_{ij} be the input vector of S_i, N is the number of LSTM blocks and M is the number of inputs. Then, we have the following weights for an LSTM unit.

1. Input Weights: W_z, W_s, $W_f, W_o \in \mathbb{R}^{N \times M}$.
2. Recurrent Weights: R_z, R_s, $R_f, R_o \in \mathbb{R}^{N \times M}$.
3. Peephole Weights: $p_s, p_f, p_o \in \mathbb{R}^N$.
4. Bias Weights: $b_z, b_s, b_f, b_o \in \mathbb{R}^N$.

Then the vector formulas for a LSTM layer can be written as follow:

$$z^t = g(W_z x^t + R_z y^{t-1} + b_z) \qquad \textit{block input}$$
$$i^t = \sigma(W_i x^t + R_i y^{t-1} + p_i \odot c^{t-1} + b_i) \qquad \textit{input gate}$$
$$f^t = \sigma\left(W_f x^t + R_f y^{t-1} + p_f \odot c^{t-1} + b_f\right) \qquad \textit{forget gate}$$
$$c^t = z^t \odot i^t + c^{t-1} \odot f^t \qquad \textit{cell}$$
$$o^t = \sigma(W_o x^t + R_O y^{t-1} + p_o \odot c^t + b_o) \qquad \textit{output gate}$$
$$y^t = h(c^t) \odot o^t \qquad \textit{block output}$$

Where σ, g, h are pointwise nonlinear activation functions and $\sigma(x) = 1/(1+e^{-x}), g(x) = h(x) = tanh(x)$. $\odot$ is present pointwise multiplication of two vectors.

The whole LSTM layer contains two sub-layers for feature selection of two different text respectively [14]. The output of the k^{th} word in LSTM layer is shown in the following equation:

$$h_k = \left[\overrightarrow{h_k} \oplus \overleftarrow{h_k}\right] \tag{2}$$

Here, we use element-wise sum to combine the forward and backward pass outputs.

(4) *Attention layer:* Since not all words contribute equally to the representation of the sentence meaning, we use attention mechanism to extract such words that are important to the web service clustering. Let H be a matrix consisting of output vectors $[h_1, h_2, \ldots, h_T]$ generated by the LSTM layer, where $T = L + l$ is the sentence length. The representation r of the sentence is formed by a weighted sum of these output vectors:

$$M = tanh(H) \tag{3}$$

$$\alpha = softmax\left(w^T M\right) \tag{4}$$

$$\mathrm{r} = H\alpha^T \tag{5}$$

We obtain the final sentence-pair representation, that is used for web services clustering from [14]:

$$h^* = \tanh(r) \tag{6}$$

(5) *Output layer:* In this model, we use a softmax classifier to predict category $\hat{y}$ from a discrete set of classifications Y for a Web service document [17]. The classifier takes the hidden state h^* as input:

$$\hat{p}(y|D) = softmax\left(W^{(D)}h^* + b^{(D)}\right) \tag{7}$$

$$\hat{y} = arg\,max_y\, \hat{p}(y|D) \tag{8}$$

The cost function is the negative log-likelihood of the true category $\hat{y}$:

$$J(\theta) = -\frac{1}{m}\sum\nolimits_{i=1}^{m} t_i \log(y_i) + \lambda||\theta||_F^2 \tag{9}$$

Where $t \in \Re^m$ is the one-hot represented ground truth and $y \in \Re^m$ is the estimated probability for each class by softmax (m is the number of target classes), and λ is an L2 regularization hyper-parameter which prevent the model from overfitting the training data.

4 Evaluation and Results

4.1 Experiment Datasets

To evaluate the performance of different Web service discovery method, we have crawled 12919 Web APIs and their related information from programmableweb.com during Oct. 2016. We select top 10 categories which involve 4351 web services for experimental evaluation. The detailed distribution data of the top 10 Web services categories is shown in Table 1.

We also use English Wikipedia corpus for mapping each word in Web services description documents into 300 dimensions vector by using gensim [15] which is a python natural language process library. The latest English Wikipedia corpus can be downloaded from https://dumps.wikimedia.org/enwiki/.

Table 1. The distribution of web services in top-10 categories

Category	Number	Category	Number
Tools	790	Messaging	388
Financial	586	Payments	374
Enterprise	487	Government	306
eCommerce	435	Mapping	295
Social	403	Science	287

4.2 Evaluation Metrics

In this section, we evaluate the proposed approach by using precision, recall, purity and entropy. Suppose that $AC = \{AC_1, AC_2, \ldots, AC_K\}$ represents the standard classification of Web services in top K categories. We denote the predicted Web services classification results as $PC = \{PC_1, PC_2, \ldots, PC_M\}$. The precision and recall metrics calculated as follows:

$$precision(PC_i) = \frac{|AC_i \cap PC_i|}{|PC_i|} \tag{10}$$

$$recall(PC_i) = \frac{|AC_i \cap PC_i|}{|AC_i|} \tag{11}$$

where $|AC_i|$ is the number of Web services which category is AC_i, $|PC_i|$ is the number of Web services that are classified into the category PC_i, and $|AC_i \cap PC_i|$ is the number of Web services in PC_i that are classified into AC_i correctly.

The purity of PC is calculated as follow:

$$purity(PC_i) = \frac{\max\left(\left|PC_i \bigcap AC_j\right|\right)}{|PC_i|}, 1 \leq j \leq K \tag{12}$$

$$purity(PC) = \sum_{i=1}^{top_k} \frac{|PC_i|}{N} \times purity(PC_i) \tag{13}$$

where N is the number of Web services in AC, and top_k represents top k($1 \leq k \leq M$) clusters in prediction clusters. The entropy of PC is calculated as follow:

$$entropy(PC_i) = -\sum_{j=1}^{K} \frac{|PC_i \bigcap AC_j|}{|PC_i|} \times log_2\left(\frac{|PC_i \bigcap AC_j|}{|PC_i|}\right) \tag{14}$$

$$entropy(PC) = \sum_{i=1}^{top_k} \frac{|PC_i|}{N} \times entropy(PC_i) \tag{15}$$

For each predicted cluster PC_i, we calculate their recall, precision, purity and entropy, respectively.

4.3 Baseline Methods

To demonstrate the effectiveness of the proposed method, we compare with several competitive approaches which is related to our work:

- TF-IDF [16]: The similarity calculation between Web services is based on the term frequency and inverse document frequency. Then it exploits the K-means algorithm to cluster all Web services.
- LDA-K [3]: The similarity calculation between services is based on the document-topic vector which obtained by LDA model. The number of topics needs manual adjustment. Then it adopts K-means algorithm to cluster all Web services.
- HDP-K [17]: The similarity calculation between Web services is based on the document-topic vector which obtained by Hierarchical Dirichlet process. The number of topics can be unbounded and learnt from the Web service description documents. Then it performs K-means algorithm on Web services to cluster them.
- WE-LDA [9]: This method leverages the high-quality word vectors to improve the performance of Web services clustering. The word vectors are obtained by word2vec.
- WT-LDA [5]: This method integrates tagging data and WSDL documents to augment Latent Dirichlet Allocation. Then it uses K-means algorithm to cluster all web services.
- BiLSTM [18]: This approach divides all Web services documents into two parts, i.e., train set and test set. It firstly captures the most important semantic information in the Web service description document-based LSTM on the train set and then predicts the Web service category on the test set.
- IGBA: The proposed method in this paper, which uses the effective words that are selected by information gain to argument attention-based BiLSTM neural network training process and improve the clustering performance of Web services.

4.4 Evaluation Results

In this section, we firstly investigate the impact of the parameter of λ which is the L2 regularization hyper-parameter and study the effect of the number of effective words. Then the performance of the proposed approach is compared with other Web services discovery approaches.

(1) *The impact of λ and hidden units*

This experiment studies the impact of the hyper-parameter λ on Web services classifying in our model. During this experiment, we change the value of λ from 0.02 to 0.1, and the steps is 0.02. Then we calculate the values of precision, recall, purity and entropy in different hidden units which vary from 100 to 1000. The experimental results are shown in Fig. 5. It can be seen from those figures, the values of precision, recall and purity are becoming bigger and the value of entropy is decreasing while the number of hidden units are increasing from 100 to 900. This is because more hidden units result in more effective information in the neural network, which are used for classifying Web services. And it is over-fitting when the hidden units vary from 900 to 1000, the values of these metrics are decreasing. We can also know that when $\lambda = 0.04$, the performance of our approach reaches the peak point in most instances. The λ is added into the cost function that prevents the model from over-fitting the training data. It can encourage all parameters to become smaller. But there is not an automatic algorithm for selecting the value of λ. So, we choose $\lambda = 0.04$, the number of hidden units is 900 for the next experiment.

(2) *The impact of the number of effective words*

In this experiment, we study the impact of the number of effective words on Web services classifying. We set the number of effective words from 2 to 10, and the steps is 2. According to the previous section, we set the number of hidden units is 900, and $\lambda = 0.04$. Then we calculate the four metrics, precision, recall, purity and entropy.

The results of our proposed method are shown in Fig. 6. It can be seen from Fig. 6 (a), when the number of effective words increasing from 2 to 4, the precision of our proposed method is becoming better. This is because those words carry more useful and valuable information to augment the attention layer to catch more effective information. We can also observe that recall and purity is increase while the number of effective words is increasing. Then the model can get better performance of Web service classification. When the number of effective words increase from 4 to 10, the performance of our approach is becoming worse, such the entropy is getting bigger, since the information of the too much additional words which we call noise make the semantic of Web services description documents becoming indistinct. So, we consider the best number of effective words of our dataset is 4 for next experiment.

(3) *The comparison of Web services discovery methods*

In order to investigate the performance of different Web services discovery approaches, we compare our method with other five methods. We choose the best result of all methods to compare their performance. Table 2 presents the performances of all

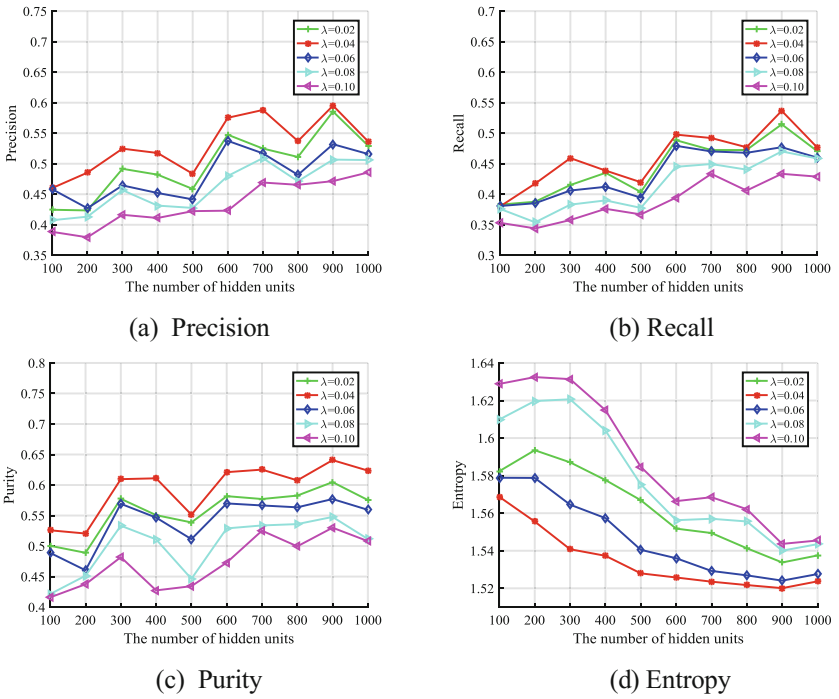

Fig. 5. The performance with the change of λ and the number of hidden units

Web services discovery methods on top 10 categories of our dataset. The bigger Precision, Recall, Purity and the smaller Entropy demonstrate that the method is better. Specifically, we have the following observations:

- The performance of our model is significantly superior to all other methods. Our model IGBA has an average precision improvement of 512% over the TF-IDF, 30.9% over the LDA-K, 16.6% over the WT-LDA, 12.1% over the WE-LDA, 12.7% over the BiLSTM. The reason is that IGBA integrates the original description documents and the effective words together and use an attention-based BiLSTM neural network to model them. The effective words and the attention mechanism promote to capture the latent semantic information for classifying Web services. This model considers deep and fine-grained level information of description document, such as the importance for each word and the information of word order. The performance of the original BiLSTM is worse than our method because the attention layer of our approach assigns different weight to each word in the Web service description documents. The recall metric of IGBA is also higher than other methods significantly that can retrieve more correct results than other methods.
- The topic model-based methods, LDA-K, HDP-K, WT-LDA and WE-LDA show a significant improvement of the precision compared with the lexical matching-based method TF-IDF. The reason is that TF-IDF fails to catch the latent semantic information in the Web services description. It only uses the term-based vector to

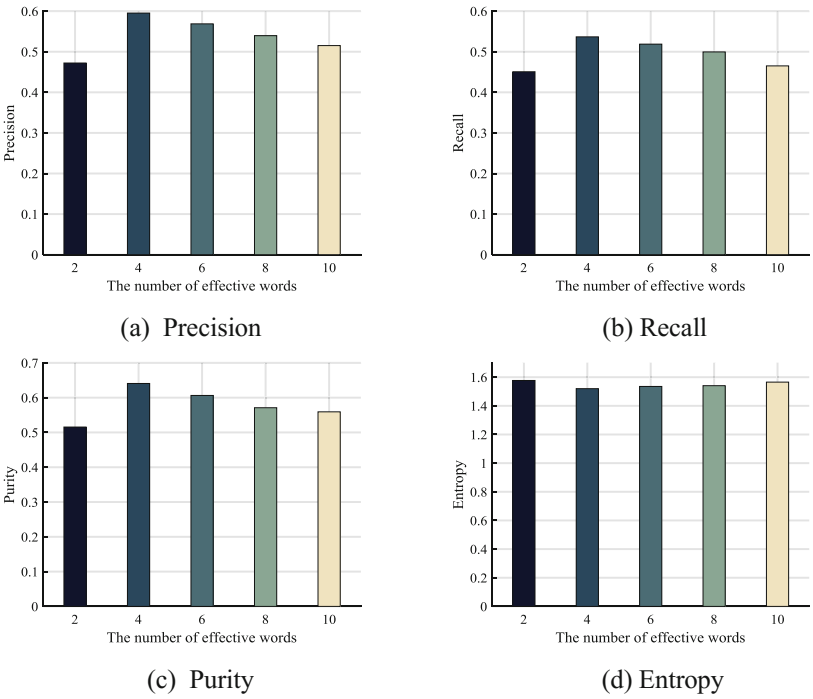

Fig. 6. The performance with the change of the number of effective words

Table 2. The performances of all Web services discovery methods

Methods	Precision	Recall	Purity	Entropy
TF-IDF	0.1037	0.3123	0.1815	2.6735
LDA-K	0.4853	0.5022	0.5347	1.5996
HDP-K	0.5372	0.5211	0.5107	1.5736
WT-LDA	0.5449	0.5306	0.5517	1.5463
WE-LDA	0.5579	0.5321	0.5439	1.5288
BiLSTM	0.5637	0.5407	0.5672	1.5319
IGBA	**0.6352**	**0.5766**	**0.6608**	**1.5101**

represent the functional features of Web services without the latent semantic which the description documents contain. We also observe that the recall of TF-IDF method is unexpectedly low, because the distributions of Web services are uneven that can be proved in Table 1. LDA-K, WT-LDA and WE-LDA exploits the LDA technique to mine the latent functional information from the description documents so they obtain the better results than TF-IDF method. We can also observe that WT-LDA and WE-LDA are better than LDA-K, because those methods exploits the additional information which is considered as function summary to augment LDA model. The additional information can improve the accuracy of Web services

discovery. The performances of WE-LDA are better WT-LDA, because the word cluster obtain by word2vec can provide more information for LDA modeling to capture the latent semantic.

- The LSTM-based methods, BiLSTM and IGBA have a better performance compared with other traditional methods. IGBA has a better performance than BiLSTM because IGBA uses information gain to extract effective words which carry more useful and valuable information for Web services clustering. Those words are exploited to augment the attention mechanism of neural network to catch the latent classifying information. The attention mechanism will assign high weights to the relevant words and low weights to the irrelevant words. For example, it is not the word earth than the word library can afford more useful information for classifying the Web service Google Earth into category earth. And the word earth is assigned to a high weight.

5 Conclusion and Future Work

This paper presents a Web service discovery approach based on information gain theory and attention-based Bidirectional neural network. We extract the effective words from the Web service description documents based on information gain theory. And we use an attention-based BiLSTM neural network for catching important information for Web service classifying. The comparative experiments performed on ProgrammableWeb dataset demonstrate the effectiveness of the proposed approach. The results show that a significant improvement of the clustering accuracy compared with other approaches and illustrates that the information which is carried by the effective words can greatly improve the performance of Web service clustering indeed. In the future work, we will develop new method based on integrated deep neural network model for extracting the multi-functionality information and context of Web service to create novel mashup application.

Acknowledgements. The work was supported by the Hunan Provincial Natural Science Foundation of China under grant No. 2017JJ2098, 2017JJ4036, 2018JJ2139, 2018JJ2136, Open Foundation of State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) under grant No. SKLNST-2016-2-26, National Natural Science Foundation of China under grant No. 61572187, 61772193, 61702181, 61872139, 61873316, Innovation Platform Open Foundation of Hunan Provincial Education Department of China under grant No. 17K033.

References

1. Xia, B., Fan, Y., Tan, W., et al.: Category-aware API clustering and distributed recommendation for automatic mashup creation. IEEE Trans. Serv. Comput. **8**(5), 674–687 (2015)
2. Samanta, P., Liu, X.: Recommending services for new mashups through service factors and top-k neighbors. In: ICWS 2017, pp. 381–388 (2017)

3. Cao, B., Liu, X., Li, B., et al.: Mashup service clustering based on an integration of service content and network via exploiting a two-level topic model. In: ICWS 2016, pp. 212–219 (2016)
4. Li, C., Zhang, R., Huai, J., et al.: A probabilistic approach for web service discovery. In: ICWS 2013, pp. 101–108 (2013)
5. 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
6. Rodriguez Mier, P., Pedrinaci, C., Lama, M., et al.: An integrated semantic web service discovery and composition framework. IEEE Trans. Serv. Comput. **9**(4), 537–550 (2016)
7. Cheng, B., Zhao, S., Li, C., et al.: A web services discovery approach based on mining underlying interface semantics. IEEE Trans. Knowl. Data Eng. **99**, 1–18 (2017)
8. Lu, Y., Mei, Q., Zhai, C.: Investigating task performance of probabilistic topic models: an empirical study of PLSA and LDA. Inf. Retrieval **14**(2), 178–203 (2011)
9. Shi, M., Liu, J., Zhou, D., Cao, B., et al.: WE-LDA: a word embeddings augmented LDA model for web services clustering. In: ICWS 2017, pp. 9–16 (2017)
10. Chen, F., Lu, C., Wu, H., et al.: A semantic similarity measure integrating multiple conceptual relationships for web service discovery. Expert Syst. Appl. Int. J. **67**(C), 19–31 (2017)
11. Zhu, L., Wang, G., Zou, X.: Improved information gain feature selection method for Chinese text classification based on word embedding. In: ICSCA 2017, pp. 72–76 (2017)
12. Greff, K., Srivastava, R.K., Koutnik, J., et al.: LSTM: a search space odyssey. IEEE Trans. Neural Netw. Learn. Syst. **28**(10), 2222–2232 (2017)
13. Zhou, P., Shi, W., Tian, J., et al.: Attention-based bidirectional long short-term memory networks for relation classification. In: ACL 2016, pp. 207–212 (2016)
14. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of LREC 2010 Workshop New Challenges for NLP Frameworks (2010)
15. Chen, L., et al.: WTCluster: utilizing tags for web services clustering. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.) ICSOC 2011. LNCS, vol. 7084, pp. 204–218. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25535-9_14
16. Tian, G., He, K., Sun, C., et al.: Ontology learning from web service descriptions. J. Front. Comput. Sci. Technol. **9**(5), 575–585 (2015)
17. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. Comput. Sci. (2015)
18. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. Comput. Sci. (2013)
19. Wang, Z., Hamza, W., Florian, R.: Bilateral multi-perspective matching for natural language sentences (2017)
20. Cao, B., Liu, X., Rahman, M.D., Li, B., Liu, J., Tang, M.: Integrated content and network-based service clustering and web APIs recommendation for mashup development. IEEE Trans. Serv. Comput. https://doi.org/10.1109/tsc.2017.2686390. Accepted 22 Mar 2017
21. Wu, Z., Zheng, X., Dahlmeier, D.: Character-based text classification using top down semantic model for sentence representation (2017)
22. Liu, X., Fulia, I.: Incorporating user, topic, and service related latent factors into web service recommendation. In: ICWS 2015, pp. 185–192 (2015)
23. Zhang, X., Zhao, J., Lecun, Y.: Character-level convolutional networks for text classification, pp. 649–657 (2015)

24. Yang, Z., Yang, D., Dyer, C., et al.: Hierarchical attention networks for document classification. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1480–1489 (2017)
25. Elgazzar, K., Hassan, A., Martin, P.: Clustering WSDL documents to bootstrap the discovery of web services. In: ICWS 2010, pp. 147–154 (2010)