WE-LDA: A Word Embeddings Augmented LDA Model for Web Services Clustering

Conference Paper · June 2017

DOI: 10.1109/ICWS.2017.9

Min Shi
Harvard University

Jianxun Liu
Hunan University of Science and Technology

Mingdong Tang
Hunan University of Science and Technology

Bu-Qing Cao
Hunan University of Science and Technology

WE-LDA: A Word Embeddings Augmented LDA Model for Web Services Clustering

Min Shi, Jianxun Liu, Dong Zhou, Mingdong Tang and Buqing Cao
School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, China
{toshimin132, ljx529, mingdongtang, buqingcao}@gmail.com, dongzhou1979@hotmail.com

***Abstract*—Due to the rapid growth in both the number and diversity of Web services on the web, it becomes increasingly difficult for us to find the desired and appropriate Web services nowadays. Clustering Web services according to their functionalities becomes an efficient way to facilitate the Web services discovery as well as the services management. Existing methods for Web services clustering mostly focus on utilizing directly key features from WSDL documents, e.g., input/output parameters and keywords from description text. Probabilistic topic model Latent Dirichlet Allocation (LDA) is also adopted, which extracts latent topic features of WSDL documents to represent Web services, to improve the accuracy of Web services clustering. However, the power of the basic LDA model for clustering is limited to some extent. Some auxiliary features can be exploited to enhance the ability of LDA. Since the word vectors obtained by Word2vec is with higher quality than those obtained by LDA model, we propose, in this paper, an augmented LDA model (named WE-LDA) which leverages the high-quality word vectors to improve the performance of Web services clustering. In WE-LDA, the word vectors obtained by Word2vec are clustered into word clusters by K-means++ algorithm and these word clusters are incorporated to semi-supervise the LDA training process, which can elicit better distributed representations of Web services. A comprehensive experiment is conducted to validate the performance of the proposed method based on a ground truth dataset crawled from ProgrammableWeb. Compared with the state-of-the-art, our approach has an average improvement of 5.3% of the clustering accuracy with various metrics.**

Keywords-tags; Web services, clustering, Word2vec, LDA, K-means++

I. INTRODUCTION

Due to the fast-developed Internet technologies, the past few years have witnessed an explosive growth of Web services, which are software systems designed to support the interoperable machine-to-machine interaction over a network with low cost and high efficiency [1]. In this paper, Web services also represent Web Application Programming Interfaces (APIs). When developing a new Web service, a provider will describe it using the Web Services Description Language (WSDL) [2] and publish this description in a public Universal Description Discovery and Integration (UDDI) registry which is available to all service users. Because of the accelerating development of Web 2.0 technologies and the boom of Web services in various functional domains, many developers prefer to create multi-functional or value-added Internet-based applications called Mashups by compositing existing RESTful Web services in a loosely coupled style. The key challenge of creating a new Mashup is to discover the desired and appropriate Web services according to a customer's complex functional requirements. However, it is not an easy task as many existing Web service searching engines primarily focus on keyword-based matching methods [3-4]. In addition, the rapid increase in the number and diversity of Web services on the web also poses a challenge on search engines to return diverse Web services that are able to cover customers' overall functional properties with high quality.

Clustering Web services according to their domains has been proved to be an efficient way to facilitate the process of discovering domain-related Web services while creating Mashup-based software applications [5-6]. Firstly, all Web services are grouped into clusters, where all services in a specific cluster provide very closed and similar functionalities. Then, the query terms from Mashup developers are submit to the corresponding domains to obtain their satisfactory results. This can not only narrow down the searching space and improve the retrieval efficiency but also diversify the Web services discovery results and avoid returning a set of very similar services, especially for a user's complex queries. For example, suppose a user's query be described as "*Search music based on the location*", it is obvious this Mashup provides two basic functions, including the music service and location service. Then the query terms "music" and "location" can be sent to the related domains and they finally return diverse relevant Web services.

To improve the efficiency of Web services discovery, many methods have been hence proposed for clustering Web services [4][7-9]. Methods, which leverage key features being mined from the WSDL documents (a file that provides a set of definitions that describe a Web service in WSDL format) to represent the functionalities of a service or Mushup, have been widely adopted [7-8]. For example, many of these work first extract key features to constitute a feature vector for each Web service from its WSDL document like the *WSDL content*, *WSDL types*, *WSDL messages*, *WSDL ports* and the *Web service name*, since these features can abstract the behaviors and characteristics of Web services. They then compute similarities between these feature vectors by using some similarity methods, such as the cosine similarity. And finally Web services are clustered into functionally similar groups according to their feature vectors similarities. However, due to the WSDL descriptions of services usually contain very limited numbers of terms which are even not proper words, these algorithm may lead to low clustering quality [4-6]. In addition, WSDL-

1. http://www.programmableweb.com.

978-1-5386-0752-7/17 $31.00 © 2017 IEEE
DOI 10.1109/ICWS.2017.9

based clustering methods fail to capture the semantic relationships among Web services for in absence of the textual information, too [6]. Therefore, the probabilistic topic model LDA (Latent Dirichlet Allocation), which represents Web services as latent functional factors, is used to improve the performance of clustering [4-5][10]. LDA is more robust and flexible than the key features-based models [4][6]. However, the power of the basic LDA model for clustering is limited to some extent, too. Some auxiliary features can be exploited to enhance the ability of LDA. For example, a user tagging augmented LDA model is proposed and investigated in [4]. Since the word vectors obtained by Word2vec is with higher quality than those obtained by the LDA model [11-12], we propose, in this paper, an augmented LDA model (named WE-LDA) which leverages the high-quality word vectors to improve the performance of Web services clustering. In WE-LDA model, the word vectors, which are outputs of the Word2vec tool by taking description documents (a textual description that the developer created for a Web service to describe its functionalities) of Web services as inputs, are first clustered into word clusters by K-means++ algorithm. These word clusters are then incorporated to semi-supervise the LDA training process, which can elicit better distributed representations of Web services since the training of the basic LDA model is only in an unsupervised way. Finally, the performance improvement of Web services clustering is achieved. The augmentation is that, in WE-LDA, it exploits the probabilistic topic distributions of the friend words of a word to refine that of this word in LDA during the training process. Here, we treat the words within a same word cluster as friends to each other. We conduct comparative experiments on a real world Web services dataset crawled from ProgrammableWeb, which demonstrate the effectiveness of our proposed model.

The rest of this paper is organized as follows. Section Ⅱ presents related work on Web services clustering technologies. Preliminary knowledge is described in Section Ⅲ. Section Ⅳ describes our approach in detail. In Section Ⅴ, we describe the experimental settings and analyze results obtained. Finally, Section Ⅵ concludes the paper and discuss the future work.

II. RELATED WORK

Many previous works focus on exploring new methods to promote the Web services clustering performance for supporting the process of Web services discovery and replacement [13]. Generally speaking, existing literatures can be roughly classified into three categories [14]: non-functional-based Web services clustering [15-16], function-based Web services clustering [17-18] and improvement of the Web services clustering [8][19].

Non-functional-based clustering methods primarily focus on the quality of service (QoS) [16][20]. These methods usually calculate the similarities of Web services based on the service name, capability, interface and QoS value. Xia et al propose a method to cluster a large number of atomic Web services into many groups according to their QoS properties [20], which takes non-functional properties such as the cost, execution time and reliability into account.

Though Non-functional-based methods have a relatively low-computation time [20], they do not pay much attention to users' functional requirements. Therefore, function-based Web services clustering methods were adopted which focus on mining functional properties of Web services for clustering them into diverse service domains [17-18]. Xie et al propose a clustering method for Web services discovery purpose, which mainly employs the Web services ontology for service matching using an accurate semantic concept of the domain ontology [17].

To improve the Web services clustering performance, many improved algorithms have been proposed to mine the semantics from WSDL documents [7-8]. They first extract key features from WSDL documents that can capture behaviors of Web services. Web services are then clustered into functionally similar groups by computing the semantic similarities between key features with cosine similarity method or others [7][21]. Factor analysis [22], topic model [23], and matrix factorization [24] are tools being used to identify latent functional factors and discover implicit semantic correlations among Web service documents. For instance, Chen et al. propose a method for Web service clustering by integrating both WSDL documents and tags based on the LDA model [4]. However, most existing topic model-based methods elicit the latent topic information based on the WSDL documents [4][25], which is difficult to obtain a well-performed topic model, especially when numbers of terms in description documents are limited, thus may lead to unsatisfactory clustering accuracy [6]. In this paper, the proposed approach trains the latent topic information of description documents with the aid of Word2vec tool, which can obtain relatively quality latent topic vectors in the case of lacking textual information.

III. PRELIMINARY KNOWLEDGE

To better expound the proposed approach, this section briefly describes preliminary knowledge about the LDA model and the Word2vec model.

A. LDA

Latent Dirichlet Allocation (LDA), a generative probabilistic model for collections of discrete data such as text corpora [26]. It is now a well-known unsupervised statistical model for natural language processing and has been demonstrated to be an effective tool in topic modeling and document clustering in the text domain [5]. Given a description document corpus of Web services, LDA model can not only automatically predict which words in this description document set are relevant to a given topic, but also predict topic distributions (or the weights of different topics) of a description document.

The reason why probabilistic topic model LDA can be adopted is that LDA uses topic features other than traditional feature words (keywords) to represent a Web service or Mashup. It is then more robust and flexible than that of feature words.

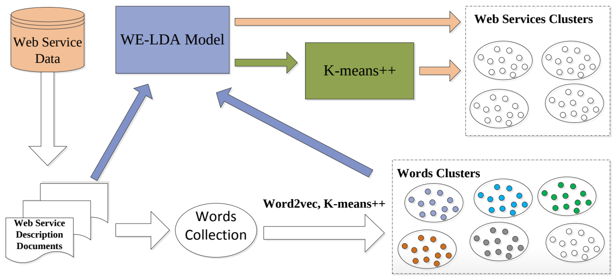

Figure. 1. The Framework of our Web services clustering approach.

B. Word2vec

The Word2vec model is proposed by Google [11] and its applications have recently attracted a great deal of attentions from the machine learning community. It is a neural network language model that can process the text data and generate word vectors [11]. The dense vector representations of words learned by Word2vec can capture the semantic meanings, e.g., the logical relationships between words, where words that we know to be synonyms are more likely to have very closed vectors and antonyms tend to have dissimilar vectors. Besides, these vectors of words adhere surprisingly well to our friend word intuition and obey the laws of analogy. For example, consider the analogy "woman is to queen as man is to king". It turns out to be that:

$$v_{\text{queen}} - v_{\text{woman}} + v_{\text{man}} \approx v_{\text{king}} \tag{1}$$

where the v_{queen}, v_{woman}, v_{man} and v_{king} are the word vectors for queen, woman, man and king respectively.

IV. THE METHODOLOGY

The overall framework of our Web services clustering approach is shown in Fig.1, the Web service data we crawled from ProgrammableWeb include Web services and their categories information. In the proposed method, we first learn the word vectors of all terms in description documents of Web services based on the Word2vec tool. Then, we perform the K-means++ [27] algorithm to cluster all these words into different groups based on the similarities of their word vectors. Subsequently, the words clusters information is used to help to train the distributed representations of Web services based on the LDA model [26]. Finally, all Web services can be clustered into different functional domains based on the clustering algorithms like K-means, or according to their latent topics purely, i.e., Web services belonging to the same topics are clustered together. In the following, we describe our approach for clustering Web services in detail.

A. Words Clustering of Description Documents

Here, we focus on mining the description documents of Web services, which are given by developers to briefly describe the functionalities of Web services. We take all description documents as inputs of Word2vec to elicit the word vectors of all terms in description documents. It is denoted by $W = \{w_1, w_2, w_3, ..., w_n\}$, where n is the size of all terms.

Word2vec has two model architectures for obtaining the continuous vector representations of words from very large datasets [11-12]: Continuous Bag-of-Word model (CBOW) that predicts the current word based on the words around it, and Continuous Skip-Gram model that predicts the surrounding words based on the current word. In our proposed method, we use the Skip-Gram model to obtain the word vectors. More detail information about Skip-Gram model can be found in the literature [11] where Word2vec proposed. In this paper, we choose top 20 categories which involve 6916 Web services, 14218 (remove the duplicate vocabulary) words in total to evaluate the proposed approach. We hence cluster these words into 20 clusters, where each cluster of words is related to a corresponding Web service domain.

We perform the words clustering process based on the K-means++ algorithm [27]. K-means method is a widely used clustering technique that seeks to minimize the average squared distance between points in the same cluster. However, due to the fact that K-means adopts a very simple and randomized seeding technique, it is guaranteed only to find a local optimum, which can often be quiet poor [27]. K-means++ has been proved to outperform K-means in term of both accuracy and speed as it chooses a point p as a center with probability proportional to p's contribution to the overall potential [27]. The Euclidean distance is used to measure the distance between each point and the cluster center. Suppose the word vector of ith term in W be $\mathbf{v}_i = \{p_1, p_2, p_3, ..., p_k\}$ and the jth cluster center be $\mathbf{c}_j = \{q_1, q_2, q_3, ..., q_k\}$, where k is the dimensions of word vectors. Then the distance between $\mathbf{v}_i$ and $\mathbf{c}_j$ can be calculated by:

$$d(\mathbf{v}_i, \mathbf{c}_j) = \left(\sum_{r=1}^{k}(p_r - q_r)^2\right)^{\frac{1}{2}} = \|\mathbf{v}_i - \mathbf{c}_j\| \tag{2}$$

We implement the K-means++ algorithm based on the *Apache Commons Math package*[1]. Take all terms in W and the value k as inputs, the words clustering results are then obtained, denoted by $W_C = \{C_1, C_2, C_3, ..., C_T\}$, where T is the number of topics trained by the proposed topic model in the next section. Table I presents some sample results of four Clusters. We can observe that domain-related words are grouped together. For example, it is obvious that words in Cluster 4 are related to the service domain of *Traveling*. Then, these clusters information is used to help to train the document-level representations of Web services based on a

TABLE I. SAMPLE RESULTS OF FOUR WORDS CLUSTERS

Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**
json	*card*	*newsletter*	*cheap*
xml	*fund*	*telephone*	*book*
plain	*check*	*message*	*transport*
html	*seller*	*fax*	*travel*
txt	*transfer*	*mail*	*holiday*
format	*wallet*	*chat*	*park*
http	*purchase*	*email*	*airport*
java	*bill*	*channel*	*guest*
php	*payment*	*caller*	*airline*
protocol	*paypal*	*signal*	*hostel*

1. http://commons.apache.org/proper/commons-math/userguide/ml.html.

novel probabilistic topic model proposed in this paper.

B. Learning Document-level Representations

Here, we propose a novel probabilistic topic model named WE-LDA to learn the latent topic vectors of description documents, which uses the words clusters as auxiliary information to improve the performance of LDA model. Here, we first introduce the WE-LDA model, and then describe the differences between the WE-LDA and LDA models.

The plate notation of WE-LDA model is shown in Fig.2. Denote the number of topics by T, the number of words in corpus dictionary of Web services by N. $\boldsymbol{\theta}$ is a length T vector indicating the proportions over all topics for the description document W. $\boldsymbol{\varphi}$ is a length N vector denoting the distributions over all words. C_i represents the words cluster which the current sampled word W_i belongs to and $|C_i|$ is the word size of cluster C_i. In this model, when sampling the topic Z_i of word W_i, the corresponding topic probability of word $W_{j,\neg i}$ is required to be taken into account, where $W_{j,\neg i} \in C_i$ and $W_{j,\neg i} \neq W_i$. The generative process of WE-LDA model is summarized in Table Ⅱ.

TABLE II. THE GENERATIVE PROCESS FOR WE-LDA

Require: *words clusters information* W_C
Require: *all Web services description documents D*
1. for each topic $t \in [1, T]$, draw a distribution over words $\boldsymbol{\varphi} \sim Dirichlet(\beta)$
2. **for** each document $d \in D$ **do**
3. draw a vector of topic proportions for this document $\boldsymbol{\theta}|\alpha \sim Dirichlet(\alpha)$
4. **for** each word W_i in document d **do**
5. conditional on $\boldsymbol{\theta}$ choose a topic $Z_i|\boldsymbol{\theta} \sim Mult(\boldsymbol{\theta})$
6. conditional on Z_i choose a word $W_i|Z_i \sim Mult(Z_i)$
7. locate the words cluster C_i, where $W_i \in C_i$
8. for each word $W_{j,\neg i}$ $(j \neq i)$ in C_i, draw $\delta_{i,j} \sim \psi(Z_{i,j}, \lambda)$, where $Z_{i,j}$ is the topic probability of word $W_{j,\neg i}$ at topic Z_i
9. **end for**
10. **end for**

The function ψ represents the influence of words in C_i exerting on the current topic sampling of word W_i. α, β and λ are prior parameters of WE-LDA. Taking all words clusters information and description documents as inputs, the posterior distribution of latent variables $\boldsymbol{\theta}$, $\boldsymbol{\varphi}$ and Z_i can be approximated by Gibbs sampling method [28]. In the training process, a Markov chain is established and topic samples are taken from the chain which in turn changes the state of the chain. The update rule for words in description documents by utilizing the latent topic information of word clusters is as follow:

$$p(z_i = t|W_c, \lambda) \propto \frac{n_{t,\neg i}^{(w_i)} + \beta}{n_{t,\neg i}^{(\cdot)} + N\beta} \times \frac{v_{t,\neg i}^{(d)} + \alpha}{v_{\cdot,\neg i}^{(d)} + T\alpha} \times \prod_{W_{j,\neg i} \in C_i} \exp\left(\frac{\lambda}{|C_i|} \times z_t^{(W_{j,\neg i})}\right) \qquad (3)$$

In this formula, for each sampled description document $d \in D$, $n_{t,\neg i}^{(w_i)}$ denotes the number of times word w_i has been observed with topic t, $n_{t,\neg i}^{(\cdot)}$ denotes the number of times words in the vocabulary dictionary are assigned to topic t, $v_{t,\neg i}^{(d)}$ represents the number of times words in document d have been assigned to topic t, and $v_{\cdot,\neg i}^{(d)}$ is the number of all words in document d. After above sampling process, we can obtain the document-level latent topics by Formula 4:

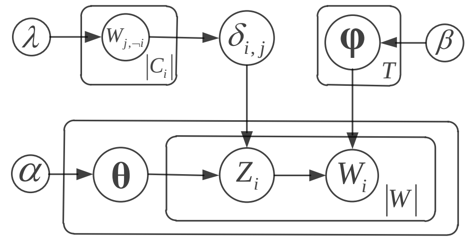

Figure. 2. Plate Notation of the WE-LDA Model .

$$p(z_i = t) \propto \frac{v_t^{(d)} + \alpha}{v_{\cdot}^{(d)} + T\alpha} \qquad (4)$$

$p(z_i = t)$ indicates the topic probability of topic t in the topic vector of document d. Due to the high-quality word vectors learned by the Skip-Gram model [11], the words clusters information is very helpful to elicit better distributed representations of documents based on the LDA model.

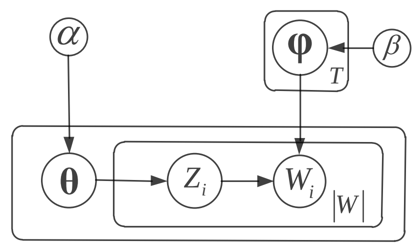

Figure. 3. Plate Notation of the LDA Model .

Fig.3 shows the plate notation of LDA model [26]. Different from the WE-LDA model, LDA takes only the description documents as inputs. During the training process, the update rule of words is:

$$p(z_i = t|W_c, \lambda) \propto \frac{n_{t,\neg i}^{(w_i)} + \beta}{n_{t,\neg i}^{(\cdot)} + N\beta} \times \frac{v_{t,\neg i}^{(d)} + \alpha}{v_{\cdot,\neg i}^{(d)} + T\alpha} \qquad (5)$$

Then the document-level latent topics of LDA can also be obtained by Equation (4). The most significant difference between WE-LDA and LDA is that the words clusters information obtained based on the Word2vec tool is used to semi-supervise the training of WE-LDA model, while the training of LDA is in an unsupervised way. To have a deep insight into the influence of this difference, we cluster the words of all Web services in top 20 categories based on their topics obtained by WE-LDA and LDA, respectively, i.e., words belonging to the same topics are clustered together. We then mark all the words to determine whether these words have been correctly clustered, i.e., tanking W_C

TABLE III. THE WORDS CLUSTERS RESULT BASED ON WE-LDA

Cluster 1		Cluster 2		Cluster 3		Cluster 4	
json	√	*card*	√	*newsletter*	√	*cheap*	√
duigxml	×	*fund*	√	*telephone*	√	*book*	√
plain	√	*check*	√	*message*	√	*transport*	√
html	√	*seller*	×	*fax*	√	*travel*	√
txt	√	*transfer*	×	*mail*	√	*holiday*	√
format	√	*wallet*	√	*chat*	√	*park*	×
http	√	*purchase*	√	*email*	√	*airport*	√
java	√	*bill*	√	*channel*	×	*guest*	√
php	√	*payment*	√	*caller*	√	*airline*	√
protocol	√	*paypal*	√	*signal*	×	*hostel*	√

TABLE IV. THE WORDS CLUSTERS RESULT BASED ON LDA

Cluster 1		Cluster 2		Cluster 3		Cluster 4	
json	√	*card*	√	*newsletter*	×	*cheap*	×
duigxml	×	*fund*	√	*telephone*	√	*book*	√
plain	×	*check*	×	*message*	√	*transport*	√
html	√	*seller*	×	*fax*	√	*travel*	√
txt	×	*transfer*	×	*mail*	×	*holiday*	√
format	√	*wallet*	√	*chat*	×	*park*	×
http	×	*purchase*	×	*email*	×	*airport*	√
java	×	*bill*	√	*channel*	×	*guest*	√
php	×	*payment*	√	*caller*	√	*airline*	√
protocol	√	*paypal*	√	*signal*	×	*hostel*	×

as the standard words clusters result, if a word is correctly clustered based on WE-LDA and LDA respectively, then marked with √, otherwise marked with ×. The statistic results demonstrate that 78% of the total words have been correctly clustered based on the WE-LDA model, while that for LDA model is only 64%. For example, Table Ⅲ and Table Ⅳ present the cluster results of words obtained respectively by the WE-LDA and LDA models, while compared those in Table I. We can observe that for all four sample words clusters, WE-LDA performs better than LDA, i.e., the words *java* and *php* are correctly placed into Cluster 1 based WE-LDA, while LDA fails to correctly cluster them.

C. *Web Services Clustering*

In this section, we adopt two widely used methods to cluster the Web services, i.e., topic model-based methods [4] and clustering algorithm-based methods [6]. We denote the topic distribution of the *i*th Web service obtained by WE-LDA as $Z = (z_1^i,\ z_2^i, \ldots, z_T^i)$, calculated from its description document.

In the topic model-based methods, each service belongs to a unique topic with the maximum probability value. Then Web services having the same topics are clustered together. Compared with the clustering algorithm-based methods, e.g, K-means, topic model-based methods pay attention to the topic similarity purely between Web services, while K-means pays more attention to the topic distribution similarity between Web services during the clustering process.

1. https://www.programmableweb.com/mashup/holidayenl.

In the clustering algorithm-based methods, K-means is widely adopted for Web service clustering [6][8]. Based on the above obtained latent topic vectors of Web services, we perform the Web services clustering based on the K-means++ algorithm. Suppose the cluster points be presented by $P = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots, \mathbf{v}_{|D|}\}$, where each point $\mathbf{v}_i$ is a *T*-dimension latent topic probability vector, $|D|$ is the size of all Web services, *T* is the number of latent topics. We aim to partition these Web services into *h* sets $S = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \ldots, \mathbf{s}_h\}$ so as to minimize the objective function:

$$\Gamma = \sum_{j=1}^{h} \sum_{i=1,\mathbf{v}_i \in \mathbf{s}_j}^{D} d\left(\mathbf{v}_i, \boldsymbol{\mu}_j\right)^2 \qquad (6)$$

where $\boldsymbol{\mu}_j$ is the mean of the points in $\mathbf{s}_j$, and $d\left(\mathbf{v}_i, \boldsymbol{\mu}_j\right)$ is calculated by Equation (2). K-means++ is guaranteed to find a solution that is O(log *S*) competitive to the optimal k-means solution [27].

After above clustering process, all Web services are clustered into different functional groups according to their semantic distances. In the next section, we conduct a comprehensive evaluation to validate the performance of the proposed approach.

TABLE V. STATISTICS INFORMATION OF CRAWLED DATASET

Items	Values
Number of Web services	12920
Number of Web services categories	384
Average number of Web services per category	33.73
Average number of words in description document	43.18

V. EVALUATION

A. *Dataset Description*

To evaluate the proposed approach, we had been crawling ProgrammableWeb.com until the end of Oct. 2016 and obtained 12920 Web services (or API services) and there are totally 384 Web service categories. The detailed statistic data is shown in Table V. This crawled dataset is now accessible at *http://kpnm.hnust.cn/xstdset.html*.

As can be seen from Table V, there are totally 384 categories for 12920 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. As categories with less Web services would result in poor clustering performance, we only choose here the top 20 categories, which involve 6916 Web services, as our experiment dataset. Table VI shows the distribution data of the top 20 Web services categories.

B. *Evaluation Metrics*

In our experiments, we evaluate the clustering performance by four metrics: Precision and Recall, Purity and Entropy. Suppose the standard classification of Web services in top *M* categories be presented by *RSC* = {*RC₁*,

TABLE VI. THE DISTRIBUTION OF WEB SERVICES IN TOP 20 CATEGORIES

CATEGORY	*NUMBER*	*CATEGORY*	*Number*
Tools	*790*	*Telephony*	*285*
Financial	*586*	*Reference*	*278*
Enterprise	*487*	*Advertising*	*248*
eCommerce	*435*	*Email*	*240*
Social	*403*	*Travel*	*237*
Messaging	*388*	*Search*	*234*
Payments	*374*	*Video*	*216*
Government	*306*	*Security*	*216*
Mapping	*295*	*Education*	*208*
Science	*287*	*Transportation*	*205*

RC_2, ..., RC_M}, which is available in the crawled dataset. We represent the experimental Web services clustering results as $ESC = \{EC_1, EC_2, ..., EC_V\}$. The Precision and Recall metrics are defined as follows:

$$Recall(EC_i) = \frac{|EC_i \cap RC_i|}{|RC_i|}, \quad (7)$$

$$Precision(EC_i) = \frac{|EC_i \cap RC_i|}{|EC_i|}, \quad (8)$$

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

The Purity of cluster EC_i and the mean Purity are defined as follows:

$$Purity(EC_i) = \frac{\max_j |EC_i \cap RC_j|}{|EC_i|}, 1 \leq j \leq M, \quad (9)$$

$$Purity(ESC) = \sum_{i=1}^{TK} \frac{|EC_i|}{N} \times Purity(EC_i), \quad (10)$$

where N is the total number of Web services in *RSC*, and *TK* represents the top k $(1 \leq k \leq V)$ clusters in the experiment.

The Entropy of cluster EC_i and the mean Entropy are defined as follows:

$$Entropy(EC_i) = -\sum_{j=1}^{M} \frac{|EC_i \cap RC_j|}{|EC_i|} \times log_2 \frac{|EC_i \cap RC_j|}{|EC_i|}, \quad (11)$$

$$Entropy(ESC) = \sum_{i=1}^{TK} \frac{|EC_i|}{N} \times Entropy(EC_i). \quad (12)$$

For each cluster EC_i, we calculate the Recall, Precision, Purity and Entropy, respectively. The average results are then reported.

C. *Comparison Approaches*

We choose to compare the proposed approach with the following baseline methods:

- TF-IDF [8]: This method adopts the K-means++ algorithm to cluster all Web services. The similarity calculation between services is based on the term frequency and inverse document frequency (TF-IDF).
- LDA-K [6]: This method adopts the K-means++ algorithm to cluster all Web services. The similarity calculation between services is based on their semantics obtained by the LDA model.
- LDA [4]: The LDA model is used to cluster all Web services, where services having same latent topics are clustered together.
- WELDA-K: This method adopts the K-means++ algorithm to cluster all Web services. The similarity calculation between services is based on their semantics obtained by the WE-LDA model.
- WELDA: The WE-LDA model is used to cluster all Web services, where services having the same latent topics are clustered together.

Since we cluster all Web services into top 20 categories (h = 20), we set the number of topics to 20 (T = 20) for all topic model-based methods, where each latent topic is related to a specific Web services domain. The prior parameters α, β and λ are empirically set to 2.0, 0.1 and 3.0 respectively as these gave the best results [29].

TABLE VII. THE CLUSTERING PERFORMANCES OF METHODS

Methods	**Recall**	**Precision**	**Purity**	**Entropy**
TF-IDF	0.0735	0.2692	0.1446	2.8582
LDA-K	0.4695	0.4700	0.5289	1.6236
LDA	0.5206	0.4806	0.5351	1.5755
WELDA-K	**0.4894**	**0.5086**	**0.5370**	**1.5867**
WELDA	**0.5567**	**0.5221**	**0.5483**	**1.5269**

D. *Performance Evaluation*

In this section, we describe the performance of all clustering methods. Table VII presents the clustering performances of all baseline methods on top 20 categories. The bigger Precision, Recall, Purity and the smaller Entropy, demonstrate that the clustering result is the better. Based on the results, we have the following observations:

- In our methods based on the WE-LDA model, WELDA is significantly superior to all other methods. With various metrics, WELDA has an average accuracy improvement of 269.2% over the TF-IDF, 8.8% over the LDA-K, 5.3% over the LDA and 5.6% over the WELDA-K. The reason is that the document-level topic distribution obtained by WE-LDA is with higher quality than that obtained by the LDA model. As it is well-known in the NLP community, Word2vec is an excellent neural language model used to learn the distributed representations of words which can capture the logical relationships between words where LDA fails [12][29]. It has been proved that Word2vec can be used to the words clustering task more accurate than the LDA model [12]. Our assumption is that the words clusters information obtained based on the Word2vec tool could be used to supervise the training of LDA so as to learn better document-level representations. The experimental results demonstrate the reasonability of this hypothesis.
- Although WELDA-K also performs the clustering based

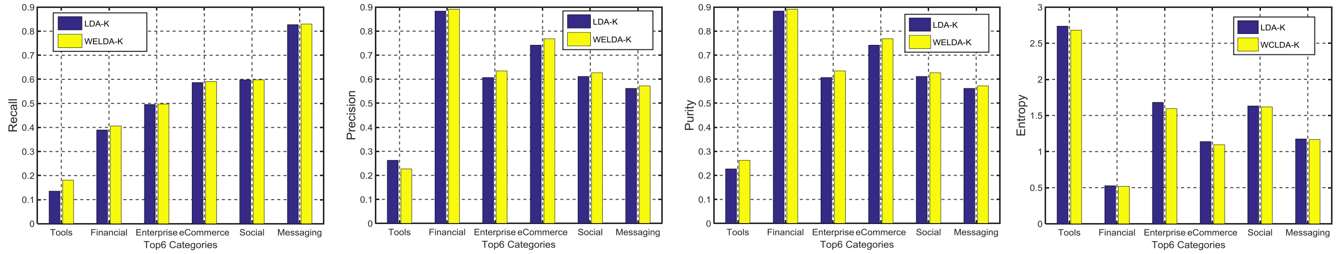

Figure 4. The Performances Comparison between LDA-K and WELDA-K in Top 6 Categories.

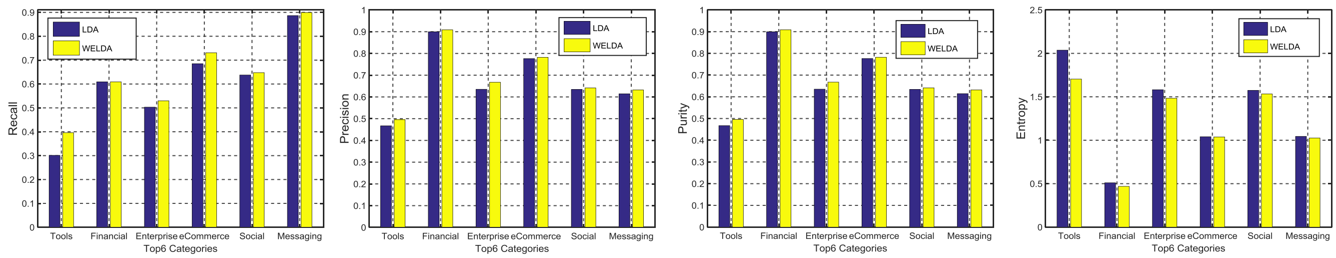

Figure 5. The Performances Comparison between LDA and WELDA in Top 6 Categories.

on the latent information obtained by the WE-LDA mode, the clustering accuracy is unsatisfied by various metrics compared with WELDA. The reason is that WELDA makes all Web services in a cluster according to a related latent topic purely, which stresses the topic similarities among services in a cluster. While WELDA-K takes all topics data into account during the clustering process, which pays more attention to the topic vectors similarities among services. As we know, under the environment of Web services, each service is related to a specific functional domain, which can be viewed as a service to its related topic. Therefore, it is more reasonable to cluster the Web services according to their topics purely instead of their topic distributions. The phenomenon can also be proved by comparing the performances of LDA-K and LDA. As can be seen from Table VII, LDA performs better than LDA-K for all metrics, which also demonstrates the topic similarity calculation is more accurate than the topic vector similarity calculation in terms of the problem of Web service clustering.

- The topic model-based methods, including LDA-K, WELDA-K, LDA and WELDA, show a significant improvement of the clustering accuracy compared with the lexical matching-based method TF-IDF. This is because that TF-IDF only uses the ontology data of Web services for similarity calculation, which fails to capture the implicit semantic correlations among Web services. Since the number of words in a description document is limited, it is more necessary to learn the latent topic information to improve the clustering performance [2][4]. In addition, we observe that the Recall of TF-IDF method is unexpectedly low. This is caused by the fact that in its clustering results the distributions of Web services are extremely uneven, which results in low Recall performance but high Precision performance in some categories containing very few Web services.

To further illustrate the effectiveness of the proposed novel topic model, we conduct the following methods comparisons: WELDA-K vs. LDA-K and WELDA vs. LDA. In Fig. 4, we compare the clustering performances of WELDA-K and LDA-K in terms of the Top 6 Web service categories. We calculate the Recall, Precision, Purity and Entropy performances of the top 6 Web service categories respectively. As can be seen from these pictures, with all evaluation metrics WELDA-K shows better performance than LDA-K. This is because in the proposed topic model the words clusters information obtained by the Word2vec tool have been utilized to help the samplings of topics of words based on the WE-LDA model. The experiment results demonstrate the words clusters information is helpful to help to train high-quality document-level latent topic vectors. Likewise, the clustering performances of WELDA and LDA are compared in Fig. 5. From these pictures, we can also observe that WELDA obtains more accurate clustering results in all top 6 categories than the LDA model with various metrics. Both WELDA and WELDA-K perform the clustering based on the WE-LDA model and they have better clustering results than the LDA and LDA-K, respectively. Therefore, these two groups of comparison further demonstrate the effectiveness of the WE-LDA model.

VI. Conclusion and Future Work

In order to improve the performance of Web services clustering, we propose an augmented LDA model, called WE-LDA, by leveraging the accurate word clusters information obtained based on the Word2vec tool in this paper. The reason why it works is that Word2vec model can output word vectors in a higher quality than that of the basic LDA model. And these word clusters obtained from the word vectors by K-mean++ algorithm are incorporated to semi-supervise the training process of LDA model. The better probabilistic representations of Web services is then elicited, which finally improves the clustering performances.

We conduct experiments to evaluate the proposed Web services clustering method and the WE-LDA model. The experimental results demonstrate that topic model-based methods show a significant improvement of the clustering accuracy compared with the lexical matching-based method TF-IDF. The comparative experiments show that WE-LDA outperforms LDA in all the performance metrics. The results also illustrates that the WE-LDA model performs better than the LDA model in learning the document representations.

Since Web services usually provide more than one aspect of utility, adopting a fuzzy clustering way is more reasonable in some environments, especially when customers desire few Web services that can satisfy their multiple functional requirements.

ACKNOWLEDGMENTS

The work described in this paper was supported by the National Natural Science Foundation of China under Project No. 61572187, 61300129, 61272063, 61572186 and 61402168, China National Key Technology R&D Program under Grant No. 2015BAF32B01, Scientific Research Fund of Hunan Provincial Education Department of China under Grant No. 16K030, Hunan Provincial Natural Science Foundation of China under Grant No. 2017JJ2101, 2017JJ2098.

REFERENCES

[1] K. Wan, P. Lei, C. Chatwin and R. Yong, "Service-Oriented Architecture". 2006, pp. 1-6.

[2] “Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language,” June 26, 2007. [Online]. Available: http://www.w3.org/TR/wsdl20. [Accessed: Feb. 26, 2010].

[3] E. Al-Masri, Q.H. Mahmoud, “Investigating web services on the World Wide Web”. International World Wide Web Conference, 2008, pp. 795–804 (2008).

[4] L. chen, Y. Wang, Q. Yu, Z. Zheng, J. Wu. “ WT-LDA: User tagging augmented LDA for web service clustering”. International Conference on Service Oriented Computing (ICSOC), 2013, Springer, pp.162-176.

[5] B. Xia, Y. Fan, W. Tan, K. Huang, J. Zhang, and C. Wu. “Category-aware API Clustering and Distributed Recommendation for Automatic Mashup Creation”. *IEEE Transactions on Services Computing*, 8(5): 674-687, 2015.

[6] B. Cao, et al. "Mashup Service Clustering Based on an Integration of Service Content and Network via Exploiting a Two-Level Topic Model". IEEE International Conference on Web Services (ICWS), 2016, pp. 212-219

[7] K. Elgazzar, A. Hassan, and P. Martin. “Clustering WSDL Documents to Bootstrap the Discovery of Web Services”. IEEE International Conference on Web Services (ICWS), 2010, pp. 147-154.

[8] L. Chen, L. Hu, J. Wu, et al. “WTcluster: utilizing tags for web service clustering”. International Conference on Service Oriented Computing (ICSOC), *2011*, Springer, pp. 204-218.

[9] N. Richi, and B. Lee. "Web service discovery with additional semantics and clustering". The IEEE/WIC/ACM International Conference on Web Intelligence, 2007, pp, 555-558.

[10] Q. Yu, H Wang, and L Chen. “Learning Sparse Functional Factors for Largescale Service Clustering”. IEEE International Conference on Web Services (ICWS), 2015, pp. 201-208.

[11] Mikolov T, Chen K, Corrado G, et al. “Efficient estimation of word representations in vector space “[J]. arXiv preprint arXiv:1301.3781, 2013.

[12] Q. V. Le, T. Mikolov. “Distributed Representations of Sentences and Documents[C]”. The International Conference on Machine Learning(ICML), 2014, 14: 1188-1196.

[13] Z. Zhou, M. Sellami, W. Gaaloul, M. Barhamgi and B. Defude, "Data Providing Services Clustering and Management for Facilitating Service Discovery and Replacement," IEEE Transactions on Automation Science and Engineering, 10(4):1131-1146, 2013.

[14] M. H. Hasan, J. Jaafar and M. F. Hassan. “Fuzzy-based clustering of web services’ quality of service: A review[J]”. Journal of Communications, 2014, 9(1): 81-90.

[15] M. Zhang, X. Liu, R. Zhang, and H. Sun, "A Web service recommendation approach based on QoS prediction using fuzzy clustering," in Proc. IEEE 9th International Conference on Services Computing, 2012, pp. 138-145.

[16] J. Zhu, Y. Kang, Z. Zheng, and M. R. Lyu, "A clustering-based QoS prediction approach for Web service recommendation," in Proc. 15th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, 2012, pp. 93-98.

[17] L.-L. Xie, F.-Z. Chen, and J.-S. Kou, "Ontology-based semantic web services clustering," in Proc. IEEE 18th International Conference on Industrial Engineering and Engineering Management, 2011, pp. 2075-2079.

[18] Z. Zhu, H. Yuan, J. Song, J. Bi, and G. Liu, "WS-SCAN: A effective approach for Web services clustering," in Proc. International Conference on Computer Application and System Modeling, 2010, pp. V5-618-V5-622.

[19] A. S. Sukumar, J. Loganathan, and T. Geetha, "Clustering Web services based on multi-criteria service dominance relationship using Peano Space filling curve," in Proc. International Conference on Data Science and Engineering, 2012, pp. 13-18.

[20] Y. Xia, P. Chen, L. Bao, M. Wang, and J. Yang, "A QoS-aware Web service selection algorithm based on clustering". IEEE International Conference on Web Services (ICWS), 2011, pp. 428-435.

[21] G. Kang, J. Liu, M. Tang et al.. “AWSR: Active Web Service Recommendation Based on Usage History”. IEEE International Conference on Web Services (ICWS),2012, pp.186-193.

[22] C. Bishop. “Pattern Recognition and Machine Learning” (Information Science and Statistics). Springer-Verlag New York. Inc., Secaucus, NJ, USA, 2006.

[23] D. Blei, A. Ng and M. Jordan. “Latent Dirichlet Allocation.” *Journal of Machine Learning Research*. 3: 993-1022, 2003.

[24] Z. Zheng, H. Ma, M. Lyu, and I. King. “Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization”. *IEEE Transactions Services Computing*, 6(3): 289-299, 2013.

[25] T. Wen, G. Sheng, Y. Li, and Q. Guo, "Research on Web service discovery with semantics and clustering," in Proc. 6th IEEE Joint International Information Technology and Artificial Intelligence Conference, 2011, pp. 62-67.

[26] D. Blei, A. Ng and M. Jordan. “Latent Dirichlet Allocation.” *Journal of Machine Learning Research*. 3: 993-1022, 2003.

[27] D. Arthur, S. Vassilvitskii. “k-means++: The advantages of careful seeding[C]“. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2007: 1027-1035.

[28] G. Heinrich “Parameter estimation for text analysis,” Technical report, vsonix. GmbH and University of Leipzig, Germany, 2004.

[29] M. Shi, J. Liu, D. Zhou, M. Tang, F. Xie, and T. Zhang, "A Probabilistic Topic Model for Mashup Tag Recommendation" . IEEE International Conference on Web Services (ICWS), 2016, pp. 444-451.