2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovations

Web Services Clustering Based on HDP and SOM Neural Network

Qiaoxiang Xiao[1], Buqing Cao[1,*], Xiangping Zhang[1], Jianxun Liu[1], Rong Hu[1], Bing Li[2]
[1](Key Lab. of Knowledge Processing & Networked Manufacturing, Hunan University of Science & Technology, Xiangtan, Hunan, China)
[2](School of Computer, Wuhan University, Wuhan, Hubei, China)
18390219693@163.com, {buqingcao, zxpkpnm, ljx529}@gmail.com, ronghu@126.com
bingli@whu.edu.cn

***Abstract*—Discovering appropriate, user-desired Web services from massive Web services quickly and accurately for users to build valuable Web applications has become a significant challenge. Web services clustering has been proved to be an effective way for Web service discovery. Some recent research works exploit additional information (such as, tags or word clusters information) to argument LDA-based topic modeling for better service clustering. Although they definitely boost service clustering via mining more implicit topics and semantic correlation of service document, their performance still can be improved due to inherent disadvantage of LDA topic model and adopted clustering algorithms. To address this problem, we propose a Web services clustering method based on HDP topic model and SOM neural network. This method, firstly uses Word2Vec to expand the original Web services description document from Wikipedia English corpus and applies HDP topic model to model the expanded Web services description document to obtain the document-topic vector. At last, it employs SOM algorithm on the document-topic vector to achieve Web services clustering. The comparative experiments are performed on ProgrammableWeb dataset. The experimental results show that the proposed method respectively achieves significant improvements of 486%, 54.0%, 35.8%, 47.1%, 39.0%, 28.6%, and 9.4%, compared with other seven service clustering methods.**

Keywords-Web Services; Word2Vec; HDP topic model; Self Organizing Maps; Web Service Clustering

I. INTRODUCTION

In recent years, with the rapid development of the Internet, more and more Web services have been emerged. How to combine Web services from different domains with simple structure and single content to build more powerful Web applications has attracted lots attention [1]. In this case, Mashup which allows software developers to compose existing Web service to create new or value-added composite Web services has emerged as a promising development technology. At the same time, with the developing of Web service repositories, the number of Web services has increasing. For example, nowadays, a representative Web service repository, ProgrammableWeb.com has more than 6300 Mashups and 17200 APIs. It is hard for user to compose appropriate Web services to complete a complex Web service application. Thus, how to discover appropriate, user-desired Web services from massive Web services quickly and accurately has become a significant challenge.

Clustering Web services according to their functionalities has been proved to be an effective way for Web service discovery [1]. Performing cluster algorithm on the vector which contains functionality information obtained from the description text of Web services can cluster the Web services which are similar together. This can not only narrow down the searching space but also avoids to return an irrelevant searching result [2]. However, these Web services are developed by different organizations, and the description documents of them usually short in length, and not standard in format. For example, it's only 24.16 average words for each description document in the ProgrammableWeb.com Web services dataset. It becomes difficult that extracts useful and valuable information from the original description document of Web service to build high-quality, informative vectors for Web service clustering. Recently, some researchers focus on this issue and exploit the topic model LDA (Latent Dirichlet Allocation) as latent functional factors to improve the performance of service clustering [1-4]. LDA is used to identify latent functional factors of service documents and discover implicit semantic correlation among service documents. The capability of the basic LDA model for service clustering is limited to some degree. Some auxiliary features can be used to improve the usage of LDA. Several of them integrated functional document and user-contributed tagging data or word clusters information obtained by Word2vec through augmented LDA for service clustering [5-6]. However, compared with the traditional LDA model, these augmented LDA topic models have not achieved significant improvement in the accuracy of service clustering. The main reasons for this are: (1) traditional or augmented LDA topic models usually need to large document corpus as input data, while service description document is still short, resulting in insufficient model training; (2) the number of topics need to be tuned manually in the LDA model, which leads to model training with very long time and low efficiency; (3) general clustering algorithms, such as k-means, need to determine the number of clusters in advance, and its initial points selection affects the clustering resulting. To address these problems, we propose a Web services clustering method based on HDP (Hierarchical Dirichlet Processing) topic model and SOM (Self-Organizing Map) neural network. In this method, we firstly use Word2Vec tool to expand the original Web service description documents from Wikipedia English corpus, and then adopt HDP to model the topic information of each Web service description document. Finally, we perform SOM clustering algorithm on the document-topic vector to cluster Web services. The contributions of this paper are summarized as follows:

978-1-5386-9380-3/18/$31.00 ©2018 IEEE
DOI 10.1109/SmartWorld.2018.00097

- We exploit the Wikipedia English corpus as the external information source to enhance the capability of information representation which the original, short Web service description documents contain.
- We perform HDP topic model on the expanded Web service description document to model the latent topic information comprehensively. HDP can automatically learns the optimal number of topics to afford more accurate semantic similarity measurement among service description documents.
- We apply SOM neural network clustering algorithm to cluster Web services together which have similar functionality. SOM adopts the topology of the input data and obtains a better performance than other cluster algorithms. The experimental results on a real-world dataset show that our approach indeed improve the accuracy of Web service discovery.

The rest of this paper is organized as follows: Section II presents related work on Web services clustering for supporting the process of Web services discovery. Section III overviews the proposed method in details. Section IV describes experimental evaluation and results. Finally, Section V concludes this paper and discusses some future works.

II. Related Work

Web service discovery is a process of identifying services which can match users' requirements [7]. It usually returns a series of related candidate Web services to users. Through service clustering, similar services in terms of functionality are grouped into clusters in order that they can be searched and discovered together, and so improve both the efficiency and accuracy of service discovery [4]. Nowadays, existing Web service clustering methods can be divided into two categories [9]: syntax-based, and semantic-based methods.

Syntax-based Web services clustering method is based on the related keywords. It requires that the selected keywords should represent the relevant functions of the Web service accurately. The functional feature vectors of Web service generally are characterized as a term-based vector space model by analyzing and processing service document (WSDL or Mashup functional description) [9-10]. The similarity among services were measured by using similarity methods, such as cosine similarity. However, the similarity generally measured by calculating the number of co-occurrences terms among service description documents is inaccurate, without considering latent semantic correlation behind them. It is possible that two service documents are similar even if the number of co-occurrences terms of them is small. Due to the "word gap" among keywords, the performance of these syntax-based Web service clustering methods can be improved.

Semantic-based Web service clustering methods usually identify latent functional factors of service description documents and discover implicit semantic correlation among service documents for clustering Web services. Some typical topic models, for example, LDA, BTM (Biterm Topic Model), and unsupervised clustering algorithms are used to model the topic information of Web service description document and cluster Web service into different function clusters, respectively [11]. On the one hand, they can extract latent, relevant semantic topic information of Web services for better service clustering [11]. On the other hand, the Web services which contain different functions can be categorized according to clustering algorithms. Recently, some research works integrate additional information (tags or word clusters information) to argument the topic modeling process for better service clustering [5-7]. These existing LDA-based methods definitely boost service clustering by mining the implicit topics and semantic correlation of service document. However, the performance of them still can be improved due to inherent disadvantage of LDA topic model and clustering algorithms. Generally speaking, the large document corpus means the more performance in LDA model training. But, service description document is still short although some additional information are extended. Not only that, LDA model needs to spend much time to find an optimal number of topics, which maybe cause to unstable training process and an unsatisfied classification result. Besides, some used clustering algorithms [3, 5] require to manually identify the number of clusters in advance, which greatly affects the performance of service clustering. In our previous work [13], we exploit HDP topic model to capture the topic distribution of description documents to recommend Web APIs for Mashup development. As well, in this paper, we adopt HDP to model the latent topic distribution of Web service for identifying Web services with similar functionality. HDP topic model can automatically find the optimal number of topics for more efficient model training and more better topic clustering. What's is more, we use SOM neural network to cluster the Web services together which have similar functionality. SOM adopts neurons based a topological structure on top of prior knowledge to perform stable model training. Through the processes of HDP topic modeling and SOM service clustering, the accuracy of Web service discovery can be greatly improved.

III. METHOD OVERVIEW

The overview of the proposed method is shown in Fig.1. This framework is divided into two parts: data collection and preprocessing part, and model training and clustering part.

In the data collection and preprocessing part, we collect the data from Wikipedia and ProgrammableWeb.com and perform processing on them. This part mainly exploits Word2Vec tools to expand the description document of Web service. Firstly, we use WikiExtractor to extract and clean Wikipedia data to get the extracted file *F*. Then we exploit Word2Vec tool to train the document F to generate word vector model of the Wikipedia corpus. Then we use the trained word vector model to expand the description document of Web services and obtain the description documents with different extension cases.

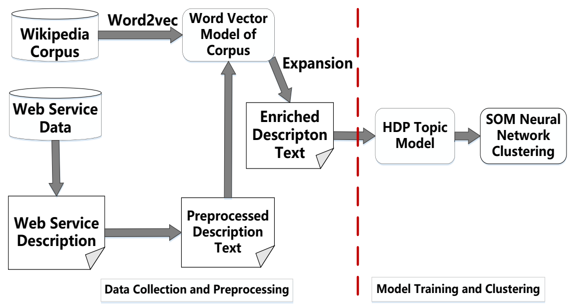
Wikipedia Corpus
Word2vec
Word Vector Model of Corpus
Expansion
Web Service Data
Enriched Descripton Text
HDP Topic Model
SOM Neural Network Clustering
Web Service Description
Preprocessed Description Text
Data Collection and Preprocessing
Model Training and Clustering

Figure.1. The Framework of the Proposed Approach

In the modeling training and clustering part, we use the HDP to model the expanded description document of each Web service to get their topic distribution. The topic distribution vector of each Web service is clustered by using SOM neural network according to the topic similarity among Web service description documents. In the following, we describe our approach for clustering Web services in detail.

A. Wikipedia dataset and word vector training process

Wikipedia corpus is recognized as the most comprehensive, authoritative encyclopedia on the Internet with a rich corpus. The English Wikipedia corpus is available at https://dumps.wikimedia.org/enwiki/lateste*-nwikilatest-pages-articles.xml.bz2*.

In this paper, we use the Word2vec in the gensim which is a python library to train the Wikipedia corpus for generating the word vector model of the Wikipedia corpus [14]. Word2vec is an NLP tool developed by Google in 2013. The basic idea of it is that each word in a document is represented as a word vector with a unified meaning and uniform dimension. In our proposed method, we use the Continuous Bags-of-Word model (CBOW) to obtain the word vectors. More detail information about CBOW can be found in the literature [15] where Word2vec proposed. The parameter settings we used are shown in Table I.

TABLE I. THE PARAMETER SETTINGS OF WORD2VEC

Parameters	***Values***
Size (Dimension)	*200*
Window (Window length)	*10*
Sample (Sampling threshold)	*0.001*
Negative (Number of noise word)	*5*
Sg (Whether to use Skip-gram)	*0(Negate)*
Hs (Whether to use the hierarchical Softmax)	*0(Negate)*

The dataset is crawled from the ProgrammableWeb.com, which is an authoritative platform for the release and retrieval of API. Until December 2017, the number of mashups on this website has exceeded 6,300 and the number of Web APIs has approached 19,000. Each mashup service is consisting of service names, description texts and tags. In the experiment, we choose top 5 categories, which include 1923 mashup services as our experiment dataset. Their detailed distributions are shown in Table II. Here, we use Word2vec for the second time, and focus on mining the description of documents of Web service. We take all of them as inputs of Word2vec to extract the word vectors of all terms in description documents. It is denoted by $D_{text} = \{w_1, w_2, \dots w_n\}$, where n id the size of all terms.

TABLE II. MASHUP SERVICE DISTRIBUTION OF TOP 5 CATEGORIES

Categories	***Number***
Deadpool	*786*
Mapping	*734*
Fun	*155*
Messaging	*133*
Music	*115*

B. Preprocessing on the Description Document of Mashup service

To improve the accuracy of Web services clustering, we perform text preprocessing on the description document of mashup service, which includes below steps:

- Tokenization. Chopping a sentence into pieces. Each piece is a word or a punctuation mark.
- Remove stop words. Stop words are some useless, meaningless words, such as *the*, *in*, *a*, *an*, *with*, and so on. They should be removed.
- Stemming. In English text, the same word has different forms of expression because of its difference of person denotation and tense representation, such as *provide*, *providing*, *provides*. So, we should extract the stemming of these words, i.e., *"providing"* becomes *"provid"*. Besides, the common word endings for English words, such as *"es"*, *"ed"* and *"ing"* should be removed.

The above steps are performed by using a python library NLTK (Natural Language Toolkit) [15] to obtain a preprocessed description document of mashup service D_{text}.

C. Word Vector Enrichment

We use trained Wikipedia corpus word vector models to enrich the description documents of Web service. Firstly, we expand the top-N words which are most similar to the word in the original Web service description document in the word vector space. N is the number of expansion words, for example, *top-3, top-5, top-10*. Table III is an enrichment example of words *Earth*/*Google*, in which top-10 expansion words of them are ranked from top to bottom, according to the similarity between the expanded word and *Earth*/*Google*. We input the preprocessed description document of mashup service

D_{text} into the trained Wikipedia corpus word vector model, and expand each word w_i in the D_{text} by using its corresponding top-N similar words $S_{w_i} = (s_1, s_2, \dots, s_N)$. The enriched description document of mashup service is denoted as $D_{enrichedText-N} = \{(w_1, S_{w_1}), (w_2, S_{w_2}), \dots, (w_n, S_{w_n})\}$, where n is the number of total words in the D_{text}.

TABLE III. EXAMPLES OF WORD VECTOR ENRICHMENT

Original Word	***Earth***	***Google***
The Expansion Words and Their Similarity	*Planet, 0.8212175378095*	*gmail, 0.7283909558905*
	Martian, 0.7272833017951	*dropbox, 0.7130858015551*
	mars, 0.6625107720393	*evernote, 0.7012767986875*
	venusian, 0.6397825388415	*app, 0.6928009800359*
	planets, 0.6372965069147	*adsense, 0.690444284196*
	spaceship, 0.6258281393796	*yahoo, 0.6894803422971*
	universe , 0.6230481157668	*microsoft, 0.6881834100870*
	planetary, 0.6217788611424	*flickr, 0.6712522269707*
	moon,0.6169323561824	*hotmail, 0.6707652150003*
	deimos, 0.6150976470414	*mapquest, 0.6674713908624*

D. HDP Topic Modeling

Topic modeling is a process of extracting features from the documents, and they can capture the latent topic information of the documents. Latent Dirichlet Allocation (LDA) is a typical topic model, in which the number of topics K need to be adjusted repeatedly, resulting in the huge time complexity and unsatisfied efficiency [16]. HDP is a multi-layer form of the Dirichlet process, which can be considered as a derivation of the LDA in the non-parametric direction [17]. The topics of each document in the HDP model obey a same Dirichlet process, and the document share the same topic sets [18]. In the HDP model, the number of topics in the document corpus is not limited and it can automatically learn the optimal number of topics. Thus, we choose HDP to model the topics distribution of the enriched description document of mashup service, which is shown in Fig.2.

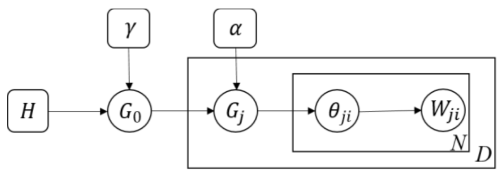

Figure.2 The HDP Topic Model

In the HDP topic model, DP stands for Dirichlet Process, and the global random probability measure G_0 is sampled from the corpus-level DP, with the base measure H and the concentration parameter γ. Each G_j is drawn from document-level DP, with the base distribution G_0 and the parameterα, where j is an index for each document. θ_{ji} indicates a generated topic of the ith word in the jth document. D represents the number of mashup documents, and N represents the number of words in each mashup document. The words can be generated from the topic. Mathematically,

$$G_0 \sim DP(\gamma, H) \tag{1}$$

$$G_j | G_0 \sim DP(\alpha, G_0) \tag{2}$$

$$\theta_{ji} \sim G_j \tag{3}$$

$$W_{ji} \sim Multi(\theta_{ji}) \tag{4}$$

We use Stick-breaking construction [19] to explain the generative process of our HDP model, which helps to infer the posterior distribution of parameters. The corpus-level topic $\emptyset_k$ is distributed as: $\emptyset_k \sim H$, each $\emptyset_k$ has the weight β_k, which is related to γ. The greater the weight, the greater the probability of choosing the current topic. The weights β_k is distributed as $\beta_k' \sim Beta(1, \gamma)$, and we can update it according to $\beta_k = \beta_k' \prod_{l=1}^{k-1}(1 - \beta_l')$, this process is also written as $\beta \sim GEM(\gamma)$ [20]. Therefore, the discrete distribution G_0 can be expressed as: $G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\emptyset_k}$, $\delta_{\emptyset_k}$ represents the atom at $\emptyset_k$, which can be understood as choosing the topic $\emptyset_k$ with the probability β_k, and the number of topics is infinite. The document-level topic ψ_{ji} is drawn from G_0, and the weight of ψ_{jn} comes from $\pi_{jn} = GEM(\alpha)$, $G_j = \sum_{n=1}^{\infty} \pi_{jn} \delta_{\psi_{jn}}$, $\psi_{jn} \subset \emptyset_k$. An indicator variable $C_{jn} \sim Multi(\beta)$ is defined, such that $\psi_{jn} = \emptyset_{C_{jn}}$, and ψ_{jn} maps to $\emptyset_k$. The generation process of the ith word W_{ji} of the jth document is as follows:

$$Z_{ji} \sim Multi(\pi_j) \tag{5}$$

$$\theta_{ji} = \psi_{jZ_{ji}} = \emptyset_{C_{jZ_{ji}}} \tag{6}$$

$$W_{ji} \sim Multi(\theta_{ji}) \tag{7}$$

In the above formulas, Z_{ji} indicates the topic ψ_{jn} and is mapped to the topic $\emptyset_k$ through the variable C_j. We take the enriched description document of mashup service $D_{enrichedText-N}$ as the input data of the HDP topic model, in which uses online variation inference to infer the topic distribution of documents [22]. We set the enriched description documents set of the crawled mashup services as $D = \{D_1, D_2, D_3, \dots, D_{|M|}\}$, $|M|$ is the number of mashup services. We denote the topic distribution information generated by the HDP model as $T = \{t_1, t_2, t_3, \dots, t_K\}$, K is the number of topics. Each D_i in the D have a topic probability distribution $T^i = \{t_1{}^i, t_2{}^i, t_3{}^i, \dots, t_K{}^i\}$. Therefore, the topic distribution vector of all mashup services can be represented as a matrix $Q = <D, T> = \{T^1, T^2, T^3, \dots, T^{|D|}\}$.

E. SOM Neural Network Clustering

SOM is an unsupervised neural network learning algorithm which maps a high-dimensional vector into a low-dimensional space [21]. It has two layers, i.e., input layer and hidden layer [23]. The number of neurons in input layer justly is the dimension of the input data. The number of neurons in hidden layer depends on the size of the input data. There are connection weights between the neurons in the two layers. The network topology of it is shown in Fig.3.

The main function of SOM is to adjust the weight of the network through self-organizing method based on a large number of training data, in order to make the output of the network to reflect the distribution of the sample data and achieve unsupervised learning clustering. Compared with traditional clustering algorithms, such as K-means, SOM does not need to determine the number of category/cluster in advance and can obtain a better clustering performance. The detailed steps of SOM are shown as follows:

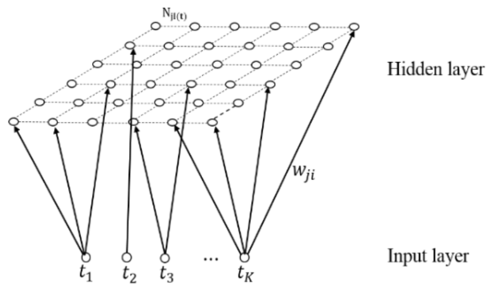

Fig.3 The Network Topology of SOM

(1) Initialization: the input data of the SOM is the topic distribution vector of mashup services Q described in the previous section. The weights w_{ji} between input nodes and output nodes are assigned to random values.

(2) Competition: The winner is the closest neurons to Q, and the similarity is calculated as follows:

$$d_j = \sum_{i=1}^{K}\left(t_i - w_{ji}\right)^2,\ j\epsilon\{1,2,\cdots,M\} \tag{8}$$

$$d_{I(t)} = \min_{j\epsilon\{1,2,\cdots,M\}}\{d_j\} \tag{9}$$

$I(t)$ is the index of the winner.

(3) Calculating the weight of adjacent nodes: The weight of the adjacent nodes $\mathrm{N}_{\mathrm{jI}(\mathbf{t})}$ should be updated when the winner is found. It is shown as formula (10):

$$N_{jI(\boldsymbol{t})} = exp\left(-\frac{S_{j,I(t)}^2}{2\sigma^2}\right) \tag{10}$$

(4) Update weights: Update the weight of the winner and its adjacent nodes until it is convergent:

$$\Delta w_{ji} = \eta(t)N_{jI(\boldsymbol{t})}(t)\left(t_i - w_{ji}\right) \tag{11}$$

We randomly select 10 Mashup services from four categories for showing the result of SOM clustering. The row and column of the hidden layer are set to 2, that is to say, the hidden layer contains four nodes/regions for achieving four categories classifications result. SOM can map each map service according to the similarity into different regions in two-dimension space. As shown in Fig.4, the region (0,0) contains two mashup services which belong to *Fun* category, the region (1,1) shows *Mapping* category, the region (0,1) indicates *Music* category, and the region (1,0) denotes *deadpool* category.

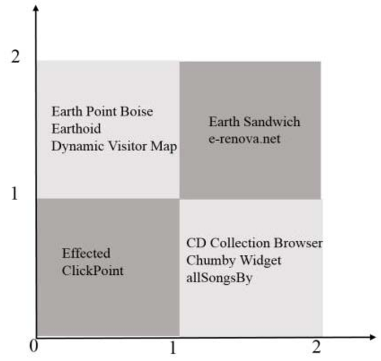

Fig.4 Example of SOM Clustering

IV. Experimental Evaluation And Results

A. Evaluation Metrics

We use precision, recall and F-measure to evaluate the performance of service clustering, which can be denoted as follows:

$$Precision = \frac{succ(c_i)}{succ(c_i)+mispl(c_i)} \tag{12}$$

$$Recall = \frac{succ(c_i)}{succ(c_i)+missed(c_i)} \tag{13}$$

$$F-Measure = \frac{2\times Precision\times Recall}{Precision+Recall} \tag{14}$$

Where $succ(c_i)$ is the number of Mashup services successfully placed into cluster c_i, $mispl(c_i)$ is the number of Mashup services are incorrectly placed into cluster c_i, and $missed(c_i)$ is the number of Mashup services that should be placed into c_i but are placed into another cluster. F-measure is the harmonic means of precision and recall.

B. Baseline Methods

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

- TF-IDF [5]: The similarity calculation between Web services is based on the term frequency and inverse document frequency. K-means algorithm is used to cluster mashup services.
- LDA: The similarity calculation between Web services is based on the document-topic vector which obtained

by LDA topic model. Each Web service belongs to the category with the biggest topic probability.

- HDP [13]: The similarity calculation between Web services is based on the document-topic vector which obtained by HDP topic model. The optimal number of topics can be automatically determined.
- WT-LDA [3]: This method integrates tagging data and WSDL document as the input data of LDA to generate more accurate topic information. K-means algorithm is adopted to cluster Web services.
- HDP-K: The document-topic vector is generated by HDP topic model. K-means algorithm is exploited to cluster Web services.
- HDP-S: The document-topic vector is obtained by HDP topic model. SOM algorithm is used to cluster Web services.
- ELDA-K-i: This method exploits LDA topic model on the enriched description text to get the document-topic vector. K-means algorithm is used to cluster Web services. Where *i* is the number of extended words.
- EHDP-S: This is the proposed method in this paper. It firstly makes use of HDP to model the extended description documents of Web services to obtain their document-topic vector. Then SOM algorithm is used to perform Web services clustering.

C. *Experimental Results and Analysis*

In this section, we describe the performance of all clustering methods. The bigger Precision, Recall and F-measure, demonstrate that the clustering result is the better. Base on the results, we have the following observation:

(1) The Performance Comparison of LDA and HDP

This experiment compares the performance of LDA and HDP on precision, recall and F-measure. We select the top 5, 10, 15, 20, 25 categories of mashups for experimental evaluating. The number of topics for LDA topic model is vary from 20 to 100, and the steps is 20. The values of hyper parameters α and β are assigned to$50/K$, 0.1, respectively, K is the number of topics. The experimental results are shown in Fig.5, Fig.6, and Fig.7.

As can be known from Fig.5, Fig.6, and Fig.7, the performances of HDP model are better than LDA model at most situation. The reason is that LDA has to tune the topic parameters artificially to get a better performance, while HDP can learn the best topic parameters from the dataset to achieve a best performance. We observe that the clustering performances of the two topic models are decreasing with the increasing of the number of categories. This is because more categories bring more noises and fuzziness to topic modeling. Furthermore, we also aware that the cluster performance of LDA topic model is getting better with the increasing of number of topics. This is because the greater number of topics, the more meticulous topic modeling.

(2) The Impact of the Number of Extension Words

This experiment investigates the impact of the number of extension words on mashup services clustering in our proposed method. The number of extension words is 0, 3, 5, 10, respectively. As we can see from Fig.8, the performance of clustering without any extension words is the worst. Since the original description document of mashup service is very short, which is not enough for topic modeling. When the number of extension words increases is 3, the performances of clustering are the best. When the number of extension words progressively increase to 5 and 10, the precision, recall and F-measure are all decreasing. This is because too many extension words make the topic model can't infer the hidden topics accurately and result in the semantics of the extended Mashup description document becoming obscure. Thus, the performance of topic modeling and mashup service clustering is getting worse. We select the number of extension words $n = 3$ for next experiment.

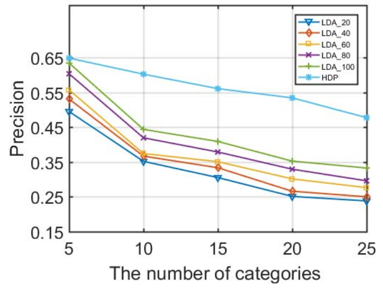

Fig.5. The Precision of Different Methods

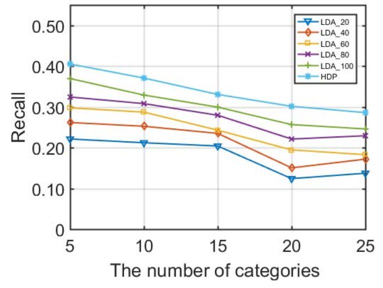

Fig.6. The Recall of Different Methods

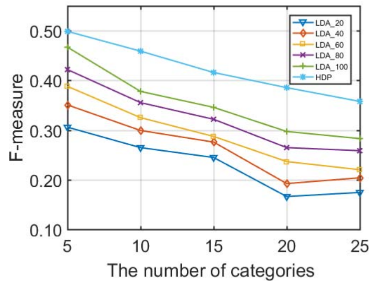

Fig.7. The F-measure of Different Methods

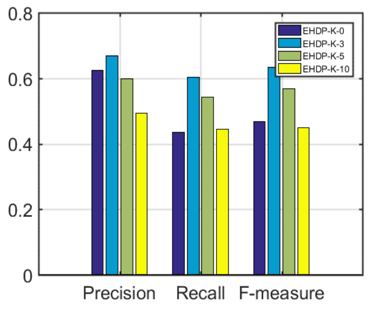

Fig.8. The performance in Different Extension Situation

(3) The Comparison of Web Services Clustering Methods

To investigate the performance of different Web services clustering approaches, we compare our method with other seven methods. The number of extension words is assigned to 3 based on the result of previous experiment.

The following Table IV presents the performances of all Web services clustering methods. The bigger values of the three metrics demonstrate the method is better. Specifically, we have the following observations:

TABLE IV THE PERFORMANCES OF SERVICES CLUSTERING METHODS

Methods	Precision	Recall	F-measure
TF-IDF-K	0.2692	0.0735	0.1155
LDA	0.5429	0.3695	0.4397
HDP	0.6481	0.4053	0.4987
WT-LDA	0.5578	0.3920	0.4604
HDP-K	0.6353	0.3949	0.4871
HDP-S	0.6591	0.4387	0.5268
ELDA-K-3	**0.6932**	**0.5593**	**0.6191**
EHDP-S-3	**0.7101**	**0.6472**	**0.6772**

- The performance of our proposed method is significantly superior to all other methods. Our model EHDP-S-3 has an average improvement of F-measure, i.e., 486% over the TF-IDF, 54.0% over the LDA, 35.8% over the HDP, 47.1% over the WT-LDA, 39.0% over the HDP-K, 28.6% over the HDP-S, 9.4% over the ELDA-K-3. The reason for this is that EHDP-S-3 exploits the additional information to overcome the defect of the original documents which lack enough information. At the same time, it also uses HDP topic model which can learn the best topic parameter from the dataset.
- The topic model-based methods, such as LDA and WT-LDA, show a significant improvement of the precision compared with the lexical matching-based method TF-IDF. The performance of TF-IDF is the worst in most cases. The reason for this is that TF-IDF cannot obtain the latent semantic information in the service description documents. LDA and WT-LDA model and exploit LDA technique to obtain the latent semantic information, so they get a better result than TF-IDF method. But such LDA-based method still cannot get a satisfying performance, because it's hard to select the best parameters for inferring the latent topic of Web service description documents. Due to the usage of the tag information, the performance of WT-LDA is better than LDA. By extending and enriching the description documents of Web services, the performance of ELDA-K-3 obviously is better than those of WT-LDA. This indicates that the extension of Web services description documents indeed improves the performance of Web service clustering. We can also learn that WT-LDA is better than LDA, because WT-LDA has use the users contribute tagging information which is considered as function summary to argument the ability of LDA topic model.
- Though HDP-K use HDP topic model to model the document-topic vector, its performance is worse than HDP-S. This is because the used K-means algorithm in the HDP-K depends on the initial point selection and easily falls into the local minimum solution. SOM algorithm in the HDP-S uses the topology structure and similarity of input vectors to perform clustering and so it can get the better performance.

V. CONCLUSION AND FUTURE WORK

This paper presents a Web service clustering approach based on HDP topic model and SOM neural network. This method, firstly uses the Wikipedia English corpus as the external information source to expand the original description documents of mashup services. Then, it exploits HDP topic model to model the expanded description documents of mashup services to obtain the document-topic vectors. At last, it exploits SOM algorithm on the document-topic vector to achieve Web services clustering. The comparative experiments performed on ProgrammableWeb dataset demonstrate the proposed method achieves a significant improvement in service clustering accuracy. In the next work, we will explore integrated deep neural network model, such as BiLSTM & CNN or others, to extract context information of service document for further improving the performance of service clustering.

ACKNOWLEDGEMENTS

The work was supported by the Hunan Provincial Natural Science Foundation of China under grant No. 2017JJ2098, 2017JJ4036, National Natural Science Foundation of China under grant No. 61572187, 61772193, 61702181, Open Foundation of State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) under grant No. SKLNST-2016-2-26.

REFERENCE

[1] Xia B, Fan Y, Tan W, et al. Category-Aware API Clustering and Distributed Recommendation for Automatic Mashup Creation. IEEE Transactions on Services Computing, 2015, 8(5): 674-687.

[2] Elgazzar K, Hassan A E, Martin P. Clustering WSDL Documents to Bootstrap the Discovery of Web Services. ICWS2010, pp. 147-154.

[3] Chen L, Wang Y, Yu Q, et al. WT-LDA: User Tagging Augmented LDA for Web Service Clustering. ICSOC2013, pp. 162-176.

[4] Yu Q, Wang H, and Chen L. Learning Sparse Functional Factors for Largescale Service Clustering. ICWS2015, pp. 201-208.

[5] Chen L, Hu L, Zheng Z, et al. WTCluster: Utilizing Tags for Web Services Clustering. ICSOC2011, pp. 204-218.

[6] Shi M, Liu J, Zhou D, et al. WE-LDA: A Word Embeddings Augmented LDA Model for Web Services Clustering. ICWS2017, pp. 9-16.

[7] Zhang Y, Sun J, Ding Y. Topic Mining for Microblog Based on MB-LDA Model[J]. Journal of Computer Research and Development, 2011, 48(10):1795-1802.s

[8] Chen F, Li M, Wu H, et al. Web service Discovery Among Large Service Pools Utilizing Semantic Similarity and Clustering. Enterprise Information Systems, 2015, 11(3): 452-469.

[9] Cheng B, Zhao S, Li C, et al. A Web Services Discovery Approach Based on Mining Underlying Interface Semantics. IEEE Transactions on Knowledge & Data Engineering, 2017, 29(5): 950-962.

[10] Kang G, Liu J, Tang M, Liu X, Cao B, and Yu X. AWSR: Active Web Service Recommendation Based on Usage History. ICWS2012, pp. 186-193.

[11] Cao B, Liu J, Tang M, and Zheng Z. Mashup Service Recommendation based on User Interest and Social Network. ICWS2013, pp. 99-106.

[12] Li C, Zhang R, Huai J, et al. A Probabilistic Approach for Web Service Discovery. ICWS2013, pp.101-108.

[13] Cao B, Li B, Liu J, et al. Web APIs Recommendation for Mashup Development Based on Hierarchical Dirichlet Process and Factorization Machines. CollaborateCom2016, pp. 3-15.

[14] Radim Řehůřek, PetrSojka. Software Framework for Topic Modelling with Large Corpora. Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks.

[15] Bird S, Klein E, Loper E. Natural language processing with Python: [analyzing text with the natural language toolkit] [J]. 2009

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

[17] Yan R, Tao Z, Li L. An automatic topic literature recommendation method based on HDP topic model. Information Studies: Theory & Application, 2016, 39(1):128-132.

[18] Ferguson T S. A Bayesian Analysis of Some Nonparametric Problems. Annals of Statistics, 1973, 1(2): 209-230.

[19] Liu S, Yin J, Ouyang J, et al. Topic Mining from Microblogs Based on MB-HDP Model. Chinese Journal of Computers, 2015, (7): 1408-1419.

[20] Wang C, Paisley J W, Blei D M. Online Variational Inference for the Hierarchical Dirichlet Process. Journal of Machine Learning Research, 2011, 15: 752-760.

[21] Kohonen T. The Self-organizing Map. Proceeding of the IEEE,1990,78(9):1464-1480.

[22] Yee Whye Teh, Michael I Jordan, Matthew J Beal, et al. Hierarchical Dirichlet Processes. Journal of the American Statistical Association, 2006,101(476): 1566-1581.

[23] Kohonen, Teuvo. Self-organization and associative memory: 3rd edition. Applied Optics, 2006, 8(1): 3406-3409.