Information Sciences 586 (2022) 239–260

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A Web service clustering method based on topic enhanced Gibbs sampling algorithm for the Dirichlet Multinomial Mixture model and service collaboration graph

Qiang Hu [a,*], Jiaji Shen [a], Kun Wang [a], Junwei Du [a], Yuyue Du [b]

[a] College of Information Science and Technology, Qingdao University of Science and Technology, Qingdao 266061, China
[b] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

ARTICLE INFO

Article history:
Received 25 May 2021
Received in revised form 28 November 2021
Accepted 29 November 2021
Available online 4 December 2021

Keywords:
Web service
Service clustering
Topic model
Service collaboration graph
GSDMM

ABSTRACT

A method to enhance Web service clustering is proposed in this paper. Since current service clustering methods usually face low quality of service representation vectors and lack consideration of service collaboration, we try to provide an improved topic model to generate high-quality service representation vectors and design a service clustering method to integrate function similarity and collaboration similarity. First, by introducing feature word extraction and probability distribution correction into GSDMM, we present a model called TE-GSDMM (topic enhanced Gibbs sampling algorithm for the Dirichlet Multinomial Mixture model). Then, a service collaboration graph is put forward to model cooperation relationships and generate service collaboration vectors. Collaboration similarity is assessed by the similarity of service collaboration vectors. Finally, the K-means++ algorithm is employed to cluster Web services by evaluating service function similarity and collaboration similarity. Experiments show that TE-GSDMM outperforms other topic models in generating high-quality service representation vectors for service clustering. Moreover, service clustering performance is further improved by integrating collaboration similarity. Thus, the proposed method effectively enhances Web service clustering by improving the quality of service representation vectors and integrating service collaboration similarity.

1. Introduction

In the fast-paced development of SOA (Service Oriented Architecture), the number of Web services has increased dramatically. SOA is widely used in the network-based software systems [1]. Many Enterprises have encapsulated their business or products as Web services and published them on the Internet. Users can invoke their required services to quickly build the value-added application systems [2]. It can make up for the lack of their business capabilities and improve the users' competitiveness.

Web service is a kind of Web API code encapsulated by standardized protocol, which can be divided into two types: SOAP Web service and Restful Web service [3]. Early Web Services are mainly SOAP type. They used WSDL to describe their functions. Currently, most of Web services are Restful type. Short natural language texts are employed to describe their functions

* Corresponding author.
E-mail address: huqiang200280@163.com (Q. Hu).

https://doi.org/10.1016/j.ins.2021.11.087
0020-0255/

for current Web services. Service discovery is becoming a challenge for the difficulty in extracting accurate function information from unstructured natural language. In addition, with the popularization of SOA applications, a large number of enterprises continue to launch new Web services. It makes the number of Web services increase rapidly and further aggravates the difficulty of service discovery.

Service clustering groups Web services into different service clusters [4]. It can effectively reduce search space and improve the efficiency of service discovery. Service function similarity is the primary basis to cluster Web services. Since there are various labels in WSDL, the feature words to describe service functions in SOAP Web services are easily extracted. Then service function similarity can be easily evaluated by computing the semantic similarity or word frequency of these feature words for early Web services [5].

Due to the lack of explicit labels to identify the semantic information of service descriptions, it is not easy to extract feature words for current Web services. To obtain the key functional feature in the service description text, many researchers have applied topic models to generate topic vectors for Web services. The topic model is a kind of statistical model that clusters the implicit semantic structure of the text by unsupervised learning. It generates the topic vector for a piece of text in probability distribution [6]. The topic vector generated by topic models for service description text is called the service representation vector in this paper.

Currently, it is a popular way to evaluate the functional similarity between Web services by calculating the similarity of service representation vectors. Topic models used to generate service representation vectors mainly include LSA, LDA, BTM, HDP and GSDMM [7]. Aiming to enhance Web service clustering quality, we propose a clustering method based on the improved GSDMM (TE-GSDMM) and service collaboration graph. The main contributions of this paper are as follows.

(1) Feature word extraction is embedded in the proposed TE-GSDMM. The words that contribute less to the service description are excluded from generating service representation vectors. It helps to alleviate the sparsity and looseness of topics in the service representation vectors.
(2) A correction factor is designed in the proposed TE-GSDMM. The probabilities of non-key topics in service representation vectors are amended by the correction factor. It helps to increase the discrimination of service representation vectors via balancing their topic distributions.
(3) Service collaboration similarity is integrated into the proposed clustering method to enhance service clustering quality further. Collaboration similarity can be easily computed based on service collaboration vectors generated from the service collaboration graph.
(4) We conduct a series of experiments on the real-world datasets to verify the effectiveness of the proposed service clustering method.

The rest of this paper is organized as follows: Section 2 is devoted to the related work. How to generate service representation vectors based on the proposed TE-GSDMM model is detailed in Section 3. We introduce service collaboration graph and the method of generating service collaboration vectors in Section 4. In Section 5, we present a service clustering method integrating the similarity evaluation of service function and collaboration based on the K-means++ algorithm. The performances of the proposed method are evaluated and discussed in Section 6. Finally, conclusions and some perspectives are given in Section 7.

2. Related work

Fig. 1 shows an example of the WSDL document fragment. Early service clustering research mainly focused on extracting the service description information used to compute service similarity from structured WSDL. For example, Liang presented a clustering method called WCCluster, which clustered the WSDL documents and extracted keywords in the form of bipartite

```
<portType name="Example">
<operation name="toSayHello" parameterOrder="userName">
<input message="tns:toSayHello"></input>
<output message="tns:toSayHelloResponse"></output>
</operation>
<operation name="sayHello" parameterOrder="person arg1">
<input message="tns:sayHello"></input>
<output message="tns:sayHelloResponse"></output>
<fault message="tns:HelloException" name="HelloException"></fault>
</operation>
</portType>
```

Fig. 1. An example of the WSDL document fragment.

graph partition [8]. WSTRec was proposed to handle the clustering performance limitation caused by uneven label distribution and noisy labels. It employed label co-occurrence, label mining, and semantic relevance measurement for label recommendation. Compared with other WSDL clustering methods, this method improved the clustering accuracy by about 14% [9]. Agarwal put forward a length feature weight method for the vectorized representation of service followed by K-Means clustering. Experiments proved that the proposed method outperforms the clustering achieved by using the TF-IDF method for vector space representation of web services [10]. In the above methods, feature words were generally extracted according to different labels in the WSDL document. Then the semantic or word frequency of these feature words was obtained to compute service similarity.

Currently, most Web services adopt the short text for function description. A Web service with the name of "VINdecoder API" in PW (ProgrammableWeb [11]) is shown in Fig. 2. There are four types of elements used to describe services, i.e., service name, service tag, service category and service description. Service tags are a group of words to indicate the service's application areas. Commonly, the first word in the service tag is designated as the service category. The service description is a natural language text to describe service function and usage.

Many works use topic models to vectorize service descriptions in service clustering and discovery. Cao constructed a mashup service network by the collaborations in mashup services. He used a two-level topic model to mine potential topics and presented an integrated content and network-based service clustering method [12]. Web service structure was modeled as Weighted Directed Acyclic Graph (WDAG) by Baskara. Then Bi-term Topic Model was employed to mine the topic on the WDAG for high precision service similarity calculation [13]. To improve topic modeling accuracy, an SP-BTM that only chooses the words with specific parts-of speech to form biterms was proposed by Hu and Liu [14]. After using the HDP model to solve service vectors' dimension problems, Cao adopted SOM neural network to cluster these service vectors [15]. Fletcher deployed the HDP technique to extract topics from service description and user requirements to enhance the discovery of services [16].

Das investigated a method of using the Gaussian LDA model to process text word embedding [17]. On this basis, Lizzaralde added service description to the Gaussian LDA model to obtain service description representation, and finally ranked services by the correlation between the user query and service description representation [18]. A Sen-LDA that learns topics of words, sentences and descriptions in service description was presented by Shi. It can bridge the vocabulary gap between services and user queries with the collective semantic similarity of sentences and descriptions [19]. Bukhari chose the LDA technique to reduce the dimension of the service representation. After that, K-Means clustering was applied to topic vectors for enhancing the accuracy score of clustering [20]. Many similar works can be found to use LDA and its variants as topic models to generate service representation vectors [21–22].

Generally, the service description text is short. Early topic models represented by LDA usually face the problem that the words in most topics of service representation vectors are sparse. It leads to a significant influence of prior parameters on the results and low classification accuracy. The GSDMM proposed by Yin is more suitable to perform topic extraction for short texts [23]. Agarwal constructed a service clustering method based on GSDMM and K-means [24]. He used the service representation vectors generated by GSDMM for service clustering. The clustering quality is significantly improved compared

Fig. 2. An example of Web service in PW.

with other topic models. Although GSDMM is excellent in generating service representation vectors, we think that the performance of GSDMM can be further improved by reducing the topic sparsity and balancing the topic distribution.

With SOA application promotion, more attention is paid to the non-functional aspects of Web services. Service quality [25], trust [26,27] and reputation [28,29] are the most concerned non-functional attributes in service clustering, recommendation and composition. The accuracy of service discovery, service clustering quality, and service composition efficiency are enhanced after considering these attributes [30]. It can be inferred from the social theory that services with similar partners are more likely to be divided into the same service cluster [31]. Therefore, compared with service quality, trust and reputation, we are more concerned about service collaboration in service clustering.

Service collaboration mostly appears in service discovery and composition. For example, Mezni constructed a context-aware service knowledge graph and designed a recommendation algorithm to deliver the top-rated services according to the target user's context [32]. Li investigated the composition of a resource service chain with interorganizational collaboration. An algorithm called ARSCCOrg was proposed to cope with interorganizational collaboration based on collaborative tasks [33].

Service collaborations are also given more attention in manufacturing services for their resources sharing [34]. Xue proposed a networked collaboration mode in manufacturing service composition increasing the number of the available service composition solution [35]. Xie presented an efficient two-phase way for reliable collaboration-aware service composition in cloud manufacturing. The method considered both the QoS stability and service collaboration ability to reduce the probability of service composition failure [36]. Although the study on service collaboration has achieved fruitful results, few of them integrated service collaboration into service clustering directly. We try to present a simple method to quantify service collaboration and integrate it into service clustering.

Fig. 3 is the outline of the research route for this paper. We first crawl Web services and their collaboration information from cloud platforms, such as PW, Cosmoplat [37] and Casicloud [38]. The formats of Web service or service collaboration in different platforms are diverse. For example, mashup services and service processes are respectively used to organize the

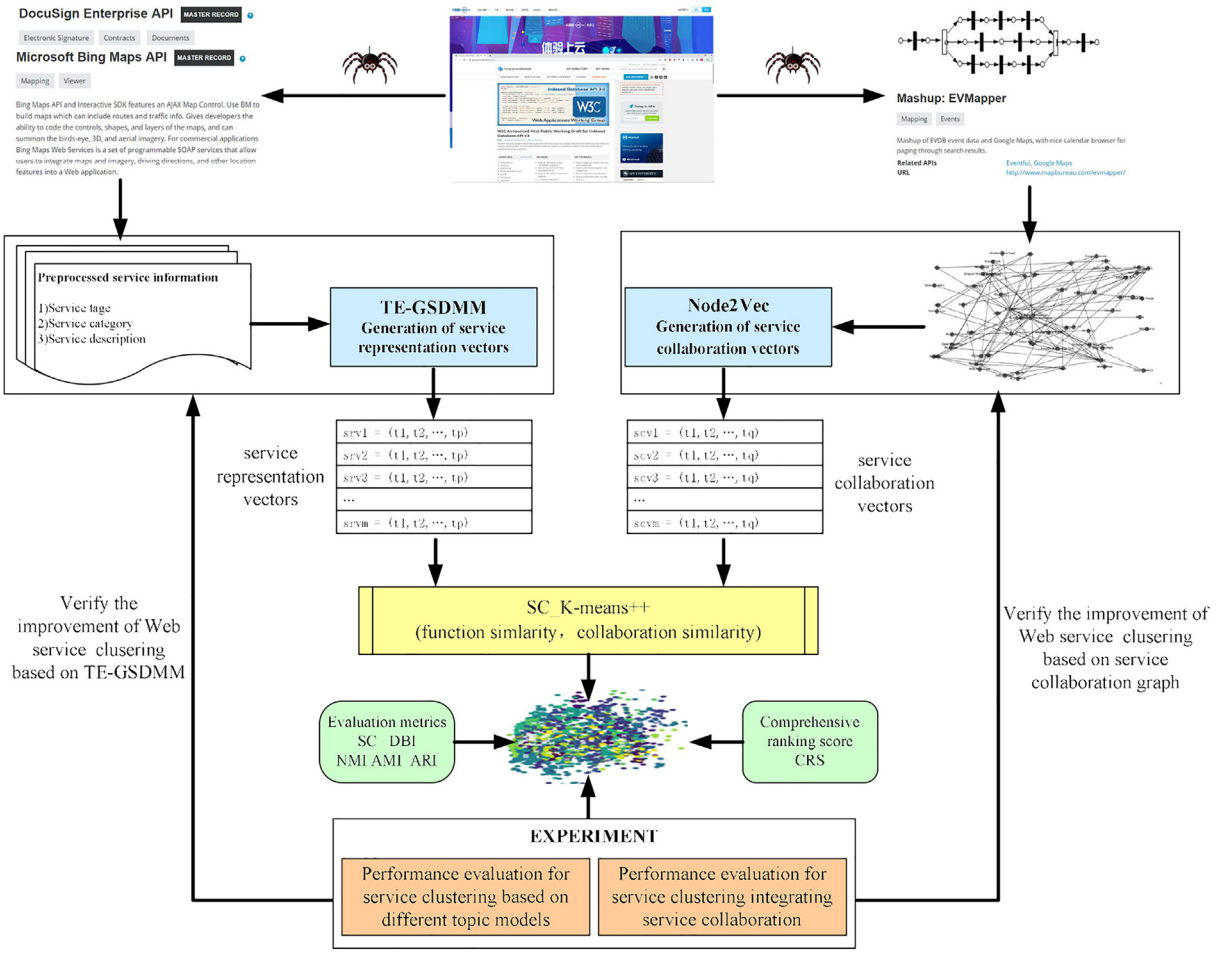

Fig. 3. Outline of the research route.

service collaborations in PW and Cosmoplat. So the information about Web services and service collaboration should be pre-processed before service clustering. The processed information of Web service mainly includes service tag, category and description. Service collaboration is described by service collaboration graph. Then TE-GSDMM and Node2Vec are employed to generate service representation vectors and collaboration vectors. K-means++ algorithm is utilized to cluster Web services based on service function and collaboration similarity. Finally, we present a comprehensive ranking way to assess the clustering performance based on five evaluation metrics. A series of experiments are conducted to verify the proposed model and method.

3. TE-GSDMM and generation of service representation vectors

3.1. TE-GSDMM

In this section, we propose an improved topic model to generate high-quality service representation vectors. Our clustering study objects are Web services that adopt short natural language texts to organize their service descriptions. The description format of these Web services can refer to the example in Fig. 2. Some formal definitions about Web service and the concepts used in service clustering are first presented.

Definition 1 (*Web service*). *A Web service is a* 5*-tuple s*=(*Id*, *n*, *t*, *d*, *c*), *where*

(1) *Id* is the ID number of service;
(2) *n* is the name of service;
(3) *t* is the set of service tags;
(4) *d* is the service description text; and
(5) *c* is the category of the service;

Definition 2 (*Service representation vector*). *Given a Web service s, if srv*=(v_{k1}, v_{k2}, …, v_{ki},…,v_{kn}) *is the topic vector generated based on s.d, then srv is called the service representation vector of s. Here, the dimension of the vector srv is n.* v_{ki} *is the probability value in the topic ki.*

We introduce the concept of key topic in service representation vector. The topic k_j is designated as the key topic of *srv* if $v_{kj} = \max(v_{k1}, v_{k2}, \ldots, v_{ki},\ldots,v_{kn})$. Further, all topics except the key topic are called non-key topics or secondary topics.

Definition 3 (*Semantic similarity of words*). w_i *and* w_j *are two words in a piece of text T,* V_{w_i} *and* V_{w_j} *are the word vectors of* w_i *and* w_j *respectively, the semantic similarity of words* w_i *and* w_j *is defined as* $SemSim(w_i, w_j) = \frac{V_{w_i} \cdot V_{w_j}}{|V_{w_i}| \times |V_{w_j}|}$.

Definition 4 (*Context fitness of word*). *d is a document including m different words.* w_i *is a word in d. The context fitness of word* w_i *in d is defined as* $Context\_Fitness(w_i, d) = \sum_k SemSim(w_i, w_k)/(m-1)$. $k = 1, 2, \ldots, m$ *and* $k! = i$.

Definition 5 (*TF-IDF of word*). *d is a document in the document set D.* w_i *is a word in d. The TF-IDF of word* w_i *in d is defined as* $TF\text{-}IDF(w_i, d, D) = TF_{wi}{}^{*}IDF_{wi}$.

Here, $TF_{w_i} = N_{w_i}/N_w$. N_{w_i} and N_w are the number of w_i in d and the total word number of d respectively. $\mathrm{IDF}_{w_i} = lg\frac{|D|}{|\{j:w_i \in d_j\}|}$, $|D|$ is the number of documents in D and j is the number of documents including the word w_i.

Definition 6 (*Context weight of word*). *d is a document in the document set D.* w_i *is a word in d. The context weight of word* w_i *in d is defined as* $Context\_Weight(w_i, d) = Context\_Fitness(w_i, d)^{*}TF\text{-}IDF(w_i, d, D)$.

In the above Definitions, the semantic similarity of two words is obtained by calculating the cosine value between their word vectors. The context fitness is the average value of the semantic similarity between this word and all the other words in a document. To get the context weight of a word, we first employ the *TF-IDF* formula to compute the *TF-IDF* of the word. Then the context weight of a word is obtained by calculating the product between its *TF-IDF* and context fitness. We can see that the context weights comprehensively consider the words' frequency and their semantic similarities, which can better reflect the importance of the words.

After a large number of experiments, Agarwal points out that the quality of service representation vectors generated by GSDMM is higher than other topic models [24]. GSDMM is a probabilistic unsupervised model. It generates documents based on the Dirichlet multinomial mixtures (DMM) model. DMM can model the topic information of documents and divide them into different categories. Gibbs sampling algorithm is employed to extract the topic information for DMM in GSDMM. DMM model is shown in Fig. 4. The probability of getting document d from subject k is calculated by the formula (1).

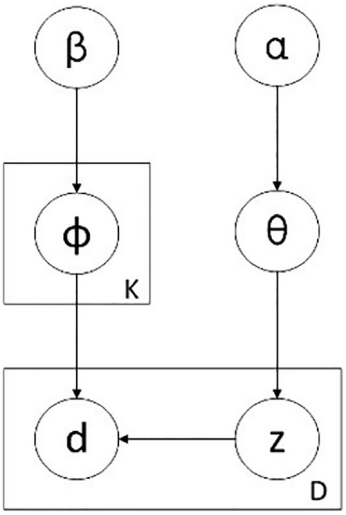

Fig. 4. Graphical model of DMM.

$$p(d|z=k)=\prod_{w\in d}p(w|z=k) \tag{1}$$

Assuming that the topic is polynomial distribution on the word, the word-topic distribution in the document can be computed by formula (2) according to reference [32].

$$p(w|z=k)=p(w|z=k,\Phi)=\phi_{k,w} \tag{2}$$

here, Φ is the word-topic distribution matrix. It describes the probability distribution of the word w in different topics. $\phi_{k,w}$ is denoted as the probability distribution of word w in topic k. The sum of topic distribution of all words in the same document is 1, i.e., $\sum_{w=1}^{V}\phi_{k,w}=1$.

Similarly, the probability of each document-topic also follows the polynomial distribution shown in formula (3).

$$p(d|z=k)=p(d|z=k,\Theta)=\theta_{k,d} \tag{3}$$

here, Θ is the document-topic distribution matrix. It describes the probability distribution of the document d on the different topics. $\theta_{k,d}$ is denoted as the probability distribution of document d on topic k. The sum of topic distribution in the same document is also equal to 1, i.e., $\sum_{k=1}^{K}\theta_{k,d}=1$.

Each word is sampled continuously on different topics in the process of Gibbs sampling. The document-topic matrix $\Theta=D\times Z$ and the word-topic matrix $\Phi=w\times z$ are obtained after all the words are sampled by Gibbs sampling algorithm. In Gibbs sampling, the probability formula of a text belonging to a certain topic is as follows (4).

$$p(z_d=z|\overrightarrow{z}_{\neg d},\overrightarrow{d})\propto\frac{m_{z,\neg d}+\alpha}{D-1+K\alpha}\frac{\prod_{w\in d}\prod_{j=1}^{N_w^d}(n_{z,\neg d}^{w}+\beta+j-1)}{\prod_{i=1}^{N_w^d}(n_{z,\neg d}^{w}+V\beta+i-1)} \tag{4}$$

where, K is the number of topics D is the number of documents. m_z and n_z are the number of document and word under topic z respectively. $n_{z,\neg d}^{w}$ represents the number of occurrences of word w under topic z while $\neg d$ means document d is excluded from the set of texts.

Although service representation vectors generated by GSDMM have an excellent topic discrimination [39–45], the quality of these vectors will be improved once the topic balance and sparsity are ameliorated. To further improve the quality of service representation vectors, a model called TE-GSDMM is proposed in this study. We improve the GSDMM model from two aspects to enhance the quality of service representation vectors. One is to improve the quality of the text that participates in the generation of service representation vectors, and the other is to balance the probability distribution of different topics in the service representation vector. To achieve the above two aspects of improvement, feature word extraction and probability distribution correction are integrated with DMM. The graphical model of TE-GSDMM is presented in Fig. 5.

There are two improvements in TE-GSDMM. The first improvement is adding a module that can extract feature words from the initial document (service description text). A super parameter γ is set to adjust the proportion of feature words extracted from the document. Before extracting feature words, we need to compute the context weight for each word in the document. Then the top γ words are selected as feature words by their context weights. The words with low relevance to the document are filtered out by extracting feature words. We use the feature words to generate service representation

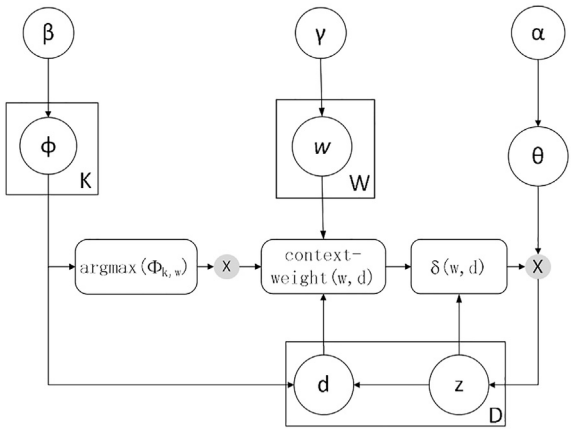

Fig. 5. Graphical model of TE-GSDMM.

vectors so that the words contributing less to the service description are excluded. It helps to alleviate the sparsity and looseness of topics in the service representation vectors.

The second improvement is that probability distribution correction is embedded. We design a correction factor for each feature word in different documents. It is used to correct the probability distribution of non-key topics in service representation vectors. For the word w in document d, the value of correction factor $\delta(w,d)$ is assigned as $1 + max(\phi_{k,w}) * ContextWeight(w,d) * \lambda$. That is, the incremental correction consists of three parts: $max(\phi_{k,w})$, $ContextWeight(w,d)$ and λ. Here, $max(\phi_{k,w})$ refers to the maximum probability value of all the topics for w in the word-topic distribution matrix Φ. $ContextWeight(w,d)$ is the context weight of word w in the document d. λ is an adjusting parameter and its value is 0.1 in this paper.

The usage of correction factors can improve the probability of non-key topics to a certain extent, and promote the balance of topic distribution. Although the probabilities of non-key topics are slightly enlarged, the value difference between key topics and non-key topics is still huge. So, the correction factor will not affect the topic differentiation of generated service representation vector. Experiments in the following Section have also shown that the quality and accuracy of service clustering are improved. This further confirms the validity of the correction factor.

3.2. Generation of service representation vectors

We need to preprocess the service description for each Web service by the following steps before service clustering.

(1) Irrelevant character removal: The irrelevant characters such as punctuation marks, URLs, newline, special symbols, quotes and some pronouns are removed from descriptions because they do not play any roles for service clustering.
(2) Text splitting: The words in the service description are separated by spaces.
(3) Stopword removal: The unnecessary words like ‘a’, ‘an’, ‘the’, ‘of’, ‘what’, ‘that’, etc. are removed.
(4) Stemming: The words with tense or voice changes in the service description are restored to the original word state.
(5) Lowercase conversion: All the words are converted into lowercase.

How to generate service represent vectors for a group of Web services based on TE-GSDMM is presented in Algorithm1. Two empty sets, which are named *Corpus_w* and *SRV*, are initialized in the first line of Algorithm1. *Corpus_w* is used to store service description texts while *SRV* is adopted to archive the service representation vectors. From line 2 to line 5, the service description texts of all the Web services are added into *Corpus_w*. Meanwhile, an empty set *fw_s* for storing feature words is initialized for each service. In line 6, the Word2Vec is adopted to generate word vectors for all words in *Corpus_w*. Then we compute the context weight for each word in a service description text from line 8 to line 10. How to obtain the value of $Context\_Fitness(w,s.d)$ and $TF-IDF(w,s.d,Corpus\_w)$ can refer to Definition 4 and Definition 5. In line 11, the words with top γ context weight in service description text and service tags are extracted as the feature words of a Web service.

GSDMM model is employed to generate the initial service representation vectors, the topic-word matrix Φ and the topic-service matrix Θ from line 12 to line 15. Probability distribution correction for the non-key topics in service representation vectors is conducted since line 16. For each feature word w, we first find the maximum topic probability distribution $max(\phi_{k,w})$ and its related topics (denoted as k-max-w) in Φ. Then the correction factor of feature word w for service s is presented as $\delta(w,s)$ by Definition 6. Meanwhile, the key topic of service s is also obtained as k-max-s. Finally, the value of $\theta_{k-max-w,s}$ in $srv(s)$ will be updated by $\theta_{k-max-w,s} * \delta(w,s)$ once k-max-s and k-max-w are not the same topic.

All the amended service representation vectors are included in the set SRV. Algorithm 1 returns the set of service representation vector SRV for all the Web services in S in line 26. These service representation vectors are utilized to compute the service functional similarity in the Section 5.

Algorithm 1 srv_TE-GSDMM

Input: the set of Web services S;
Output: the set of service representation vector SRV for service s in S;
1. $Corpus\_w = SRV = \varnothing$;
2. for each $s \in S$
3. $Corpus\_w = Corpus\_w \cup s.d$;
4. $fw\_s = \varnothing$;
5. endfor
6. Train the vector $V(w)$ for each word w in $Corpus\_w$ by Word2Vec;
7. for each $s \in S$
8. for each word w in $s.d$
9. $ContextWeight(w, s.d) = Context\_SemSim(w, s.d)^{*}\ TF\text{-}IDF(w, s.d, Corpus\_w)$;
10. endfor
11. $s.fw = s.t \cup Rank(s.d, ContextWeight(w, s.d), \gamma)$;
12. feed each s_fw to GSDMM for ten rounds and get
13. {the initial service representation vector $srv(s)$;
14. the topic-word matrix Φ ;
15. the topic-service matrix Θ ;}
16. for each word w in s_fw
17. $k\text{-max-}w = argmax(\phi_{k,w})$
18. $\delta(w, s) = 1 + \max(\phi_{k,w})^{*}ContextWeight(\mathrm{w, s.d})^{*}\lambda$;
19. $k\text{-max-}s = argmax(\theta_{k,s})$;
20. if (k-max-s!= k-max-w)
21. update $srv(s)$ by $\theta_{k-max-w,s} = \theta_{k-max-w,s}\ ^{*}\delta(w,s)$;
22. endif
23. endfor
24. $SRV = SRV \cup srv(s)$;
25. endfor
26. return(SRV);

4. Service collaboration graph and generation of service collaboration vector

To facilitate service discovery and invocation, enterprises usually adopt fine-grained function encapsulation for Web services. These Web services can achieve simple function responses. However, it is necessary to combine several services to respond to some complex service requests. For example, the map service Google Maps and the auction service eBay are composed to create a specific mashup called BidNearBy, which provides a local auctions search based on a map view [40]. The component services in a service composition form a kind of cooperative relationship when responding to service requests. They are usually complementary in business functions, computing power, resource sharing and so on. Meanwhile, these cooperative Web services have good interaction compatibility [41].

Service collaboration similarity is used to evaluate the extent to which two services can cooperate with other similar Web services. Typically, two Web services may have more common partner services if they are with a higher service collaboration similarity. In the case of same service functional similarity, Web services with higher service collaboration similarity are more likely to be grouped into the same service cluster. So service collaboration similarity should be taken into account while we cluster Web services.

The service collaboration graph is built to model the collaboration relationship of Web services in this study. Web services are mapped as the nodes of the service collaboration graph. An edge in the service collaboration graph indicates a cooperative relationship between the services represented by the nodes it connects. We can obtain many service process models from various cloud platforms. The services and their process transfer dependencies can be extracted from these models and used to build a service collaboration graph. We have extracted the collaborations of mashup services in PW. Fig. 6 shows the service collaboration graph, which is built on these service collaborations. The nodes in Fig. 6 represents the Web services while the arcs mean that the collaboration between two services.

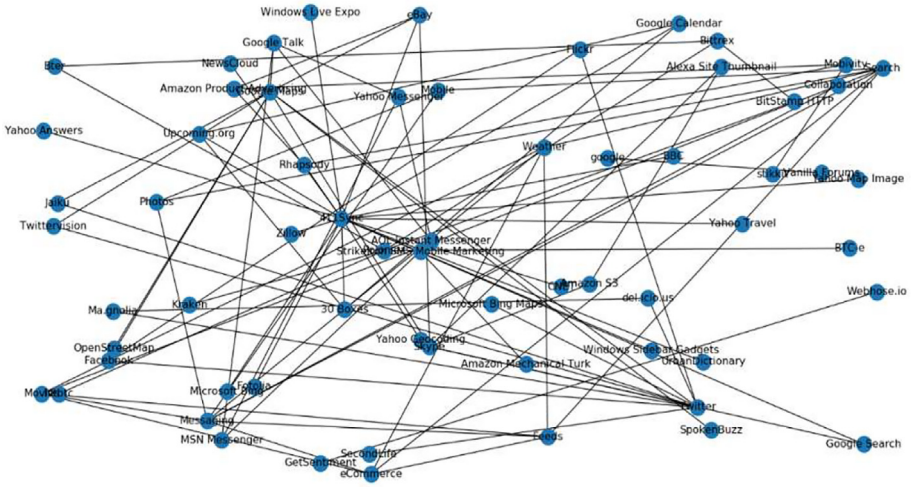

Fig. 6. A fragment of service collaboration graph.

Definition 7 (*Collaboration dependency*). *s_i and s_j are two Web services in a service process* (*or service composition*). *s_j is called collaboration dependency with s_i, denoted as $s_j \rightarrow s_i$, if s_i is the precursor service of s_j.*

Definition 8 (*Service collaboration graph*). *Service collaboration graph is an undirected graph SCG=(V, E), where*

(1) $V=\{v_1, v_2, v_3, \ldots, v_n\}$ is a set of nodes, each node v in V is used to represent a Web service;
(2) $E=\{e=(v_i, v_j)|1 \le i, j \le n\}$ is a set of edges, the $e=(v_i, v_j)$ denotes that there is a collaboration dependency between the services represented by v_i and v_j.

The social network theory shows higher collaboration similarity for two Web services once they have more common partner services than others. Moreover, two services will also have a higher collaboration similarity once they have a smaller distance with the same Web service in service collaboration graph. Thus, the co-occurrence and distance between nodes are two key factors to evaluate collaboration similarity in the service collaboration graph.

Node2vec is a graph embedding representation method [42]. It is an efficient scalable algorithm for feature learning in networks which optimizes a novel network-aware, neighborhood preserving objective using stochastic gradient descent. Node2vec performs a biased random walk in a graph by the preset probability strategy to form a series of node sequences. These node sequences are used to generate the representation of network nodes by the way of processing word vectors.

Given the current node v, the probability of visiting the next node x can refer to formula (5). π_{vx} is the unnormalized transition probability between nodes v and x, and Z is the normalizing constant. Considering a random walk that just traversed edge (t, v) and now resides at node v, the next visited node is decided by the transition probabilities π_{vx} on edges (v, x) leading from v. $\pi_{vx} = \alpha_{pq}(t, x).\omega_{vx}$, where ω_{vx} is the edge weight between vertex v and x, and $\alpha_{pq}(t,x)$ is computed by formula (6) with d_{tx} denotes the shortest path distance between nodes t and x.

$$P(c_i = x|c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} & if \ \ (v,x) \in E \\ 0 & otherwise \end{cases} \tag{5}$$

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} if d_{tx} = 0; \\ 1 if d_{tx} = 1; \\ \frac{1}{q} if d_{tx} = 2; \end{cases} \tag{6}$$

Node2vec takes two factors into account when sampling and generating the node sequence. It is suitable for computing the collaboration similarity of Web services. Therefore, node2vec is employed to train the service collaboration vectors. Service collaboration similarity of two Web services can be obtained by evaluating the similarity of theirs service collaboration vectors. How to generate service collaboration vectors is presented in algorithm 2.

The first line of the algorithm initialized a service collaboration graph *SCG*. We use the codes from line 2 to line 8 to build *SCG*. By traversing all the service processes in *SP*, the collaboration dependencies between any two services are extracted. The nodes and edges representing the newly discovered collaboration dependency are updated or added into *SCG*. The transition

probabilities of a node to its neighbors are computed by the parameters p, q and formula (6). *SCG'* is a service collaboration graph with the transition probabilities (line 9 and line 10). The *Walk* is used to store node sequences in one round. All node sampling sequences are added into *walks*. The detailed node sampling process is presented in line 11 to line 22. The algorithm performs r rounds of node sequence sampling. The node sequence length is l. A new starting node is selected for each round of in sampling node sequence. The key step to generate node sequence is to visit the next node by alias sample method according to the transition probability. Finally, we use the SGD method to train walks and generate the service collaboration vectors of all the nodes in line 23 to line 26.

Algorithm 2 scv_Node2Vec

Input: the set of service process *SP*;
Output: the set of service collaboration vector *SCV*;
1.Initialize a service collaboration graph *SCG*;
2. for each service process *sp* in *SP*
3. for each service $s \in sp$
4. if $\exists\ s' \in sp$, s.t. $s \rightarrow s' \vee s' \rightarrow s$
5. add the nodes $v(s)$,$v(s')$ and the edge < $v(s)$,$v(s')$ > into *SCG*.
6. endif
7. endfor
8. endfor
9. π = PreprocessModifiedWeights*(SCG, p, q)*
10. SCG' = (V, E, π)
11. Initialize *walks* to Empty
12. for *iter* = 1 to *r* do
13. for all nodes $u \in V$ do
14. Initialize *walk* to [u]
15. for walk_*iter* = 1 to *l* do
16. *curr* = *walk*[-1]
17. V_{curr} = GetNeighbors(*curr*, G')
18. *s* = AliasSample(V_{curr}, π)
19. Append *s* to *walk*
20. endfor
21. Append *walk* to *walks*
22. endfor
23. *scv*[*i*] = *StochasticGradientDescent*(*k*, *d*, *walks*)
24. endfor
25. *SCV*={*scv*[*i*]};
26. return *SCV*.

5. Service clustering algorithm based on evaluation of service function and collaboration similarity

K-means++ algorithm is employed to achieve service clustering [43]. It is one of the most widely used clustering algorithms. K-means++ algorithm can easily support users to design the distance (similarity) between samples according to the clustering scenarios. Compared with other algorithms, it is easier to realize the integration of function similarity and collaboration similarity in K-means++. More importantly, it shows excellent clustering performance with low requirements for computing resources. Following experiments confirm that K-means++ outperforms other clustering algorithms in Web service clustering.

We have proposed the methods of obtaining service representation vectors and collaboration vectors in the previous Sections. Functional similarity for two Web services is evaluated by the Euclidian distance between their service representation vectors in this paper. If the Euclidean distance of two service representation vectors is considerable, it means that their service representation vectors may have significant differences in the topic probability distribution. Further, we can conclude that the two web services are pretty different in the service function. Similarly, Euclidian distance can also be used to determine the collaboration similarity of two Web services based on their collaboration vectors.

Service clustering method based on function and collaboration similarity is presented in algorithm 3. The algorithm is divided into three parts. The first two lines of the algorithm are the first part. In this part, algorithm 1 and algorithm 2 are adopted to generate the set of service representation vector *SRV* and the set of service collaboration vector *SCV* respectively.

The second part of algorithm 3 (line 3 to line 11) is used to generate the k initial centre point services. A Web service is randomly chosen as the first centre point service. The distance SD between each service s and centre point service cps is calculated from line 5 to line 7. Here, $ED(v_i, v_j)$ is used to compute the Euclidian distance between two vector v_i and v_j. $D(s)$ is assigned as the minimum distance between each service s and the centre point service in its nearest service cluster. A service selection probability based on maximum distance is provided in line 9. It is used to choose the next centre point service by the Roulette method in line 10. We can get high-quality initial centre point services for k service clusters by the above steps. It should be noted that some centre point services may be virtual services.

The third part of algorithm 3 (line 12 to line 22) generates the k service clusters. The distance between each pair of Web service and centre point service is computed in line13 to line 15. Web service s will be added to the cluster whose centre point service has the minimum distance with s in line 16. The k centre point services are recomputed from line 17 to line 20. Centre point service is updated by recalculating the mean of service representation vector and service collaboration vectors of all services in the service cluster. Algorithm 3 finishes execution and returns the generated service clusters when all the centre point services are no longer changed.

Algorithm 3. KM_TG_SC

Input: the set of Web service S, the number of clusters k
Output: k service clusters
1. Generate the set of service representation vector SRV for S by algorithm 1;
2. Generate the set of service collaboration vector SCV for S by algorithm 2;
3. select a Web service s randomly in S as the first centre point service $cps1$;
4. *repeat*
5. *for* each service s in S and centre point service cps
6. $SD(s, cps)=(1-q)*ED(srv(s), srv(cps)) + q^* ED(scv(s), scv(cps))$;
7. *endfor*
8. $D(s) = \min(SD(s, cps))$;
9. $P(s)= \frac{D(s)^2}{\sum_{s\in S} D(s)^2}$
10. select a service s as a centre point service by Roulette method based on $P(s)$;
11. *until* k centre point services are generated;
12. *repeat*
13. *for* each service s and centre point service cps
14. $SD(s, cps)=(1-q)*ED(srv(s),srv(cps)) + q^* ED(scv(s), scv(cps))$;
15. *endfor*
16. add service s into the service cluster $SC[j]$ with the $\min(SD(s, cps))$;
17. *for* $i = 1$ to k
18. $srv(cps[i])= \frac{1}{|SC[i]|}\sum_{s\in SC[i]} srv(s)$;
19. $scv(cps[i])= \frac{1}{|SC[i]|}\sum_{s\in SC[i]} scv(s)$;
20. *endfor*
21. *until* no alteration in centre point services.
22. *return* $\{SC[k]\}$

The effect of the proposed method will be affected by the number of words in the service description text. For some services with too few words in the description text, the quality of extracting feature words is low. Then it may degrade the generation quality of service representation vectors. The quality of service clustering will not achieve the effect of this paper if there are many services with too few words in the service description texts participating in service clustering.

In addition, the clustering method is designed based on functional similarity and collaborative similarity. However, since the data in the most cloud platforms cannot support the statistics of collaboration times, the impact of collaboration times on collaboration similarity is not considered in this method. The clustering quality will be further improved once the impact of collaboration times is added. Meanwhile, the proportion between function similarity and collaboration similarity needs to be determined by adjusting parameters in the clustering process. It increases the complexity of the proposed service clustering algorithm.

6. Experiments

Experiments were conducted in this Section to answer two questions:

(1) Whether TE-GSDMM can outperform other topic models in generating high-quality service representation vectors?
(2) Whether the quality of service clustering is further improved by integrating service collaboration similarity?

6.1. Experiment setup

All experiments were run on a PC with i7-8750h, 16 g RAM. The programs were developed by Python.

6.1.1. Evaluation metrics

The evaluation metrics of clustering quality can be divided into two types: external evaluation metrics and internal evaluation metrics. External evaluation metrics use the sample tag to evaluate whether the clustering is reasonable. Internal evaluation metrics evaluate the clustering quality by some parameters related to the structural characteristics of clusters. The following evaluation metrics were chosen to appraise the clustering quality.

1. Internal evaluation metrics

(1) Silhouette Coefficient (SC)

For a sample x, let a be the average distance from other samples in the same category, and b be the average distance from the nearest samples in different categories. The Silhouette Coefficient of x is given by formula (7).

$$SC(x) = \frac{b-a}{\max(a,b)} \tag{7}$$

The range of SC value is [−1,1]. A higher score means the better clustering quality in view of SC.

(2) Davies-Bouldin index (DBI)

DBI is the maximum quotient of the sum of the average distance within a class and the distance between the centres of any two clusters.

$$DBI = \frac{1}{n}\sum_{i=1}^{n}\max_{j\neq i}\left(\frac{\sigma_i+\sigma_j}{d(c_i,c_j)}\right) \tag{8}$$

In formula (8), n is the number of clusters, c_i is the centre of the cluster i, σ_i is the average distance from all the samples in the cluster i to the centre c_i, and $d(c_i, c_j)$ is the distance of two centre c_i and c_j. Smaller DBI means that a smaller distance between samples within the cluster and a more considerable distance between clusters. So the clustering quality will be better if the DBI is smaller.

2. External evaluation metrics

(1) Normalized mutual information (NMI)

NMI is used to evaluate the degree of consistency between two samples. It is the normalization of mutual information (MI) score. The calculation method is shown in formula (9).

$$MNI(X,Y) = \frac{MI(X,Y)}{F(H(X),H(Y))} \tag{9}$$

X=$\{x_1, x_2, \ldots, x_k\}$ is a sample division after clustering. Y=$\{y_1, y_2, \ldots, y_k\}$ is the real categorization. MI(X, Y) is the mutual information of X and Y. It reflects the correlation degree between X and Y. $H(x)$ and $H(y)$ represent the entropies of X and Y respectively. F is the normalized function. The range of NMI value is [0,1]. The higher score means the better clustering quality in view of NMI.

(2) Adjusted mutual information (AMI)

Compared with MI, AMI follows the theory of clustering vector random assignment. Given clustering tags and real tags, AMI can measure the correlation between them and ignore the ranking of tags. The calculation method of AMI is shown in formula (10).

$$AMI(X,Y) = \frac{MI(X,Y) - \mathbb{E}\{MI(X,Y)\}}{F(H(X),H(Y)) - \mathbb{E}\{MI(X,Y)\}} \tag{10}$$

Here, $\mathbb{E}\{MI(X,Y)\}$ is the expected-value of MI(X,Y). The range of AMI value is in [−1.1]. The larger the value of AMI, the more consistent the clustering results are with the real situation.

(3) Adjusted rand index (ARI)

ARI is obtained by adjusting RI.ARI is defined in formula (11) while RI is presented in formula (12). RI is to determine the proportion of correct decisions. The range of ARI value is [−1,1]. It reflects the degree of coincidence between the real tags and the clustering results. A higher the consistency between the clustering results and the real situation can be concluded for a larger ARI value in the clustering process.

$$ARI(X,Y) = \frac{RI(X,Y) - \mathbb{E}\{RI(X,Y)\}}{max(RI(X,Y)) - \mathbb{E}\{RI(X,Y)\}} \tag{11}$$

$$RI(X,Y) = \frac{TP(X,Y)+TN(X,Y)}{TP(X,Y)+FP(X,Y)+TN(X,Y)+FN(X,Y)} \tag{12}$$

6.1.2. Dataset and baseline methods

We have crawled more than 24,000 Web services (APIs and mashup services) from PW. Some Web services have too short service description texts to generate reasonable service representation vectors. Meanwhile, some service categories are only containing very few Web services. Web services in such categories are also not suitable for service clustering experiment. After deleting these noise services, we choose 19,241 web APIs and 1253 mashup services to build the datasets.

We rank service categories according to the number of Web services they include. The top-20, 50 and 132 Web service categories are chosen as the classification benchmark category. They are named DS1, DS2, and DS3, respectively. DS4 is established based on the Web services in 1253 mashup services. Table 1 shows the datasets used in our experiments.

We have carried out two experiments. TE-GSDMM is compared with the popular topic models LSA, LDA, BTM, HDP and GSDMM in experiment 1. The clustering algorithms K-means++, BIRCH, GMM and AGNES are employed to evaluate the performance of these topic models.

In experiment 2, we compare our clustering algorithm KM_TG_SC with KM_TG to investigate whether the service clustering quality is improved after the service collaboration similarity is considered. Here, KM_TG refers to the service clustering algorithm which has removed service collaboration similarity from KM_TG_SC.

To consistently compare the internal and external evaluation metrics, we set the number of categories in the datasets as the final numbers of service clusters. So the number of service clusters in the following experiments are 20, 50, 132 and 34 for DS1, DS2, DS3 and DS4, respectively. This is also the common way to determine the number of clusters in many clustering research literatures [12,44,46].

6.2. Performance comparison

6.2.1. Optimal number of topics

When a topic model is used to generate service representation vectors, the number of topics in the topic vector will affect the quality of the service representation vectors. To better compare the performance of different topic models, we should first determine the optimal number of topics for each topic model on different datasets.

We first generate service representation vectors for each dataset with a different number of topics and topic models. Then K-means++ algorithm was employed to cluster the Web services in different datasets based on these service representation vectors.

Table 2 shows SC, DBI, AMI, NMI and ARI scores for the clustering evaluation based on the service representation vectors generated by BTM and DS1. The meaning of each evaluation metric is different. Moreover, the metric spaces of different evaluation metrics are also not consistent. Therefore, the five evaluation metrics cannot be comprehensively quantified by the weight summation.

Table 1
Outline of datasets.

DataSet	Service components	Number of services
DS1	Top 20	9394
DS2	Top 50	14,768
DS3	Top 132	19,241
DS4	Mashup Web API /34 categories	774

Table 2
Topic number and CRSs of BTM in DS1.

Topic	SC	DBI	AMI	NMI	ARI	SC_r	DBI_r	AMI_r	NMI_r	ARI_r	CRS
20	0.365	1.096	0.389	0.397	0.221	1	1	6	14	1	0.957
25	0.334	1.222	0.402	0.416	0.203	2	2	2	10	2	0.944
30	0.300	1.321	0.387	0.409	0.173	3	3	7	12	4	0.966
35	0.269	1.430	0.390	0.416	0.170	4	5	5	9	5	0.964
40	0.237	1.630	0.410	0.435	0.195	6	8	1	1	3	0.947
45	0.238	1.362	0.382	0.432	0.117	5	4	8	2	10	0.966
50	0.212	1.697	0.392	0.428	0.157	7	10	3	4	7	0.968
55	0.206	1.493	0.376	0.427	0.116	8	7	11	5	12	0.977
60	0.177	1.648	0.377	0.423	0.110	10	9	9	6	13	0.979
65	0.178	1.725	0.376	0.418	0.142	9	12	10	8	8	0.979
70	0.152	1.738	0.372	0.422	0.129	11	13	12	7	9	0.981
75	0.150	1.442	0.322	0.394	0.072	13	6	17	16	17	0.986
80	0.149	1.967	0.391	0.429	0.158	14	17	4	3	6	0.977
85	0.151	1.754	0.338	0.394	0.091	12	14	15	15	15	0.986
90	0.146	1.889	0.345	0.399	0.116	15	15	13	13	11	0.985
95	0.130	1.699	0.340	0.411	0.079	16	11	14	11	16	0.985
100	0.126	1.896	0.325	0.382	0.098	17	16	16	17	14	0.988

To synthetically evaluate the clustering quality, we propose a comprehensive ranking method to determine the optimal number of topics. The concept of comprehensive ranking score (*CRS*) is presented as follows.

P={p_1, p_2,. . ., p_n} is a group of evaluation metrics. S={s_1, s_2,. . .s_m} is a set of samples. ps_{ij} is the score of sample $_{sj}$ under the metric p_i. Let rs_{ij} be the rank of ps_{ij} in all the scores of s_j under the metric p_i. Then the comprehensive ranking score of s_j based on the P is

$$CRS(s_j) = 1 - \frac{1}{\sum_{i=1}^{n} rs_{ij}} \tag{13}$$

The set of evaluation metrics is built as P={SC, DBI, AMI, NMI, ARI} while S consists of scoring samples with different topic numbers. The ranking score of each metric is illustrated in Table 2. The comprehensive ranking score for each sample can be computed by the formula (13). The topic with the minimum comprehensive ranking score is the optimal one. We can see that topic number 25 is with the minimum value of comprehensive ranking score (0.944). So the optimal topic number of BTM in DS1 is 25.

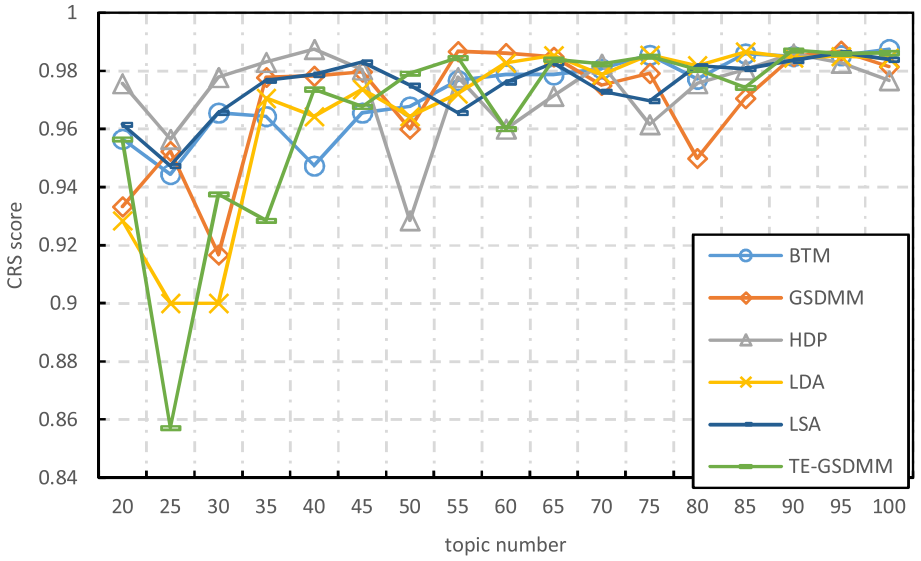

Fig. 7. CRS for different topic number based on DS1.

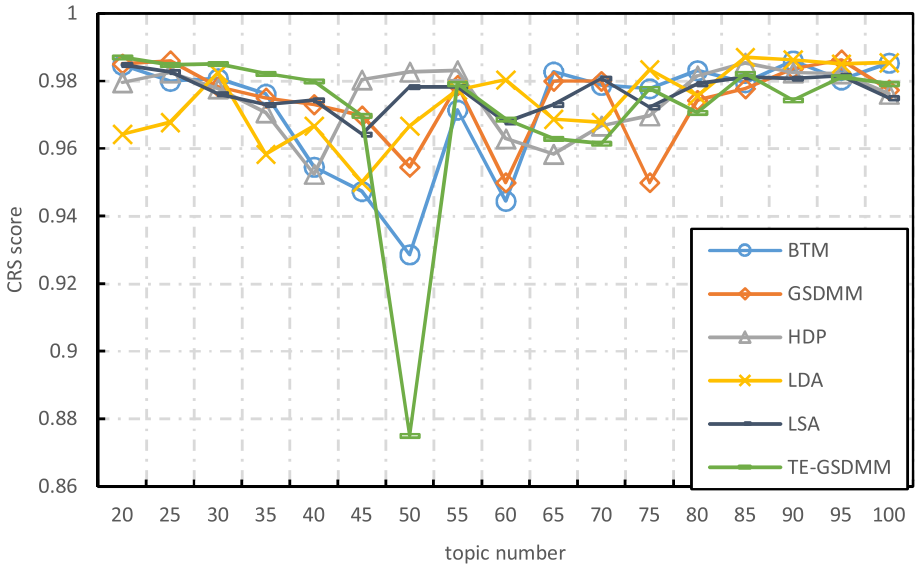

Fig. 8. CRS for different topic number based on DS2.

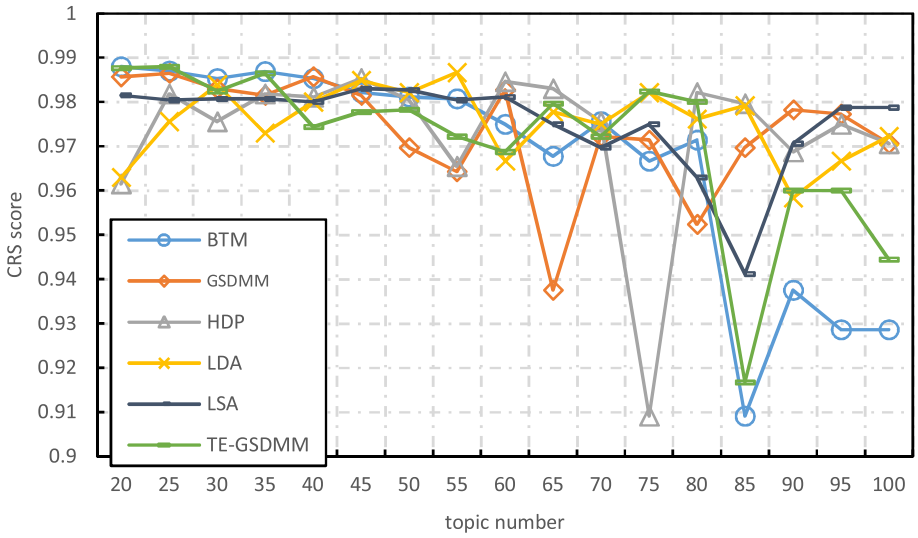

Fig. 9. CRS for different topic number based on DS3.

The topic number increases from 20 to 150 by 5 in our experiments. We find that all the optimal number of topic models are less than 100. For clarity, comprehensive ranking scores (CRSs) are presented with the topic number from 20 to 100. The values of CRS for different topic models based on DS1 to DS4 are tested. Fig. 7 shows the CRSs for different topic numbers in DS1. We can see that most of the optimal topic numbers are 25. The CRSs for different topic number in DS2 is illustrated in Fig. 8. The optimal topic numbers for these topic models are about 50. Fig. 8 presents the CRSs for different topic numbers in DS3. The values of optimal topic numbers include 65, 75, 85 and 95 for different topic models. Similarly, the CRSs for other topic numbers in DS4 is provided in Fig. 10. The value 25 is selected as the optimal topic number in DS4. Table 3 presents the optimal topic numbers for each topic model and dataset. The following experiments were conducted based on the topic numbers in Table 3.

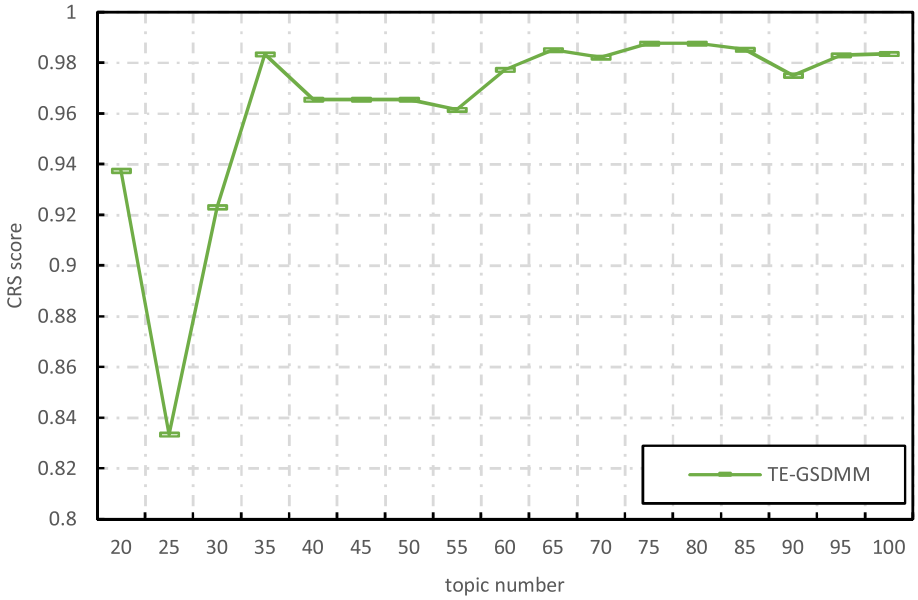

Fig. 10. CRS for different topic number based on DS4.

Table 3
The optimal topic numbers for topic models.

Topic model	DS1	DS2	DS3	DS4
BTM	25	50	85	
GSDMM	30	60	65	
HDP	50	40	75	
LDA	25	45	95	
LSA	25	45	85	
TE-GSDMM	25	50	85	25

6.2.2. Coherence comparison between TE-GSDMM and other topic models

Topic coherences for different topic models are investigated in this Section. To perform more comprehensive analysis, we utilize an automated metric, namely coherence score, proposed by Mimno for topic quality evaluation [21]. Given a topic z and its top T words $V(z) = (v_1^{(z)}, \cdots, v_T^{(z)})$ ordered by $P(w|z)$, the coherence score is defined as formula (14).

$$c\left(z; V^{(z)}\right) = \sum_{t=2}^{T} \sum_{l=1}^{t} \log \frac{DN\left(v_t^{(z)}, v_l^{(z)}\right) + 1}{D(v_l^{(z)})} \tag{14}$$

where $D(v)$ is the document frequency of word v, $DN(v, v')$ is the number of documents word v and v' co-occurred. The coherence score is based on the idea that words belonging to a single concept will tend to co-occur within the same documents. It is empirically demonstrated that the coherence score is highly correlated with human-judged topic coherence.

Since the coherence score only is appropriate for measuring frequent words in a topic, the number of top words T is set as 10 in our calculation of coherence scores. Table 4 shows an overview of coherence scores for different topic models and datasets. For each topic model, we have tested the coherence scores on the data sets DS1, DS2 and DS3. The column data in Tab.4 is the coherence scores of different topic models on three data sets. We can see that TE-GSDMM has scored larger than other topic models in all the datasets except for the BTM on DS2. Compared with the GSDMM, the coherence scores of TE-GSDMM have been increased by 14.6%, 5.1% and 4.3% in DS1, DS2 and DS3, respectively. Experiments confirm that the TE-GSDMM performs better than other topic models in the metric of coherence.

6.2.3. Performance comparison between TE-GSDMM and other topic models

Service representation vector, vector similarity calculation method and clustering algorithm are the three factors that affect service clustering. Therefore, under the premise of using the same similarity calculation method and clustering algorithm, we can evaluate the quality of service representation vectors by the quality of service clusters. Furthermore, under the same data set, if the quality of service representation vectors is higher, the topic model of generating these service representation vectors is better than others. Given these rules, we verify the performance by testing the quality difference of service clusters generated by TE-GSDMM and other topic models.

We compare the performances of TE-GSDMM, LSA, LDA, BTM, HDP and GSDMM. Service representation vectors are trained based on DS1 to DS3 by these topic models. Then four clustering methods including K-means++, BIRCH, GMM and AGNES are run on these service representation vectors. Table 5 presents the scores of evaluation metrics for different topic models in DS1. TE-GSDMM has outperformed other models in four metrics. The evaluation metric SC of GSDMM is higher than that of TE-GSDMM. The scores of evaluation metrics for different topic models in DS2 and DS3 are shown in Table 6 and Table 7, respectively. From the data in Table 6 and Table 7, we can see that TE-GSDMM has outperformed other models in all the evaluation metrics.

We can see that TE-GSDMM and GSDMM have gotten the first and second place according to *CRSs* on the three datasets. This further proves the conclusion that the performance of GSDMM is better than LDA and other traditional topic models in reference [24]. TE-GSDMM and GSDMM were compared 12 times and formed 60 comparison scores on three datasets in Table 1 to Table 3. We can find that TE-GSDMM has achieved 57 higher scores. In DS1, GSDMM got two leading scores in SC based on BIRCH and K-means++ clustering methods. In DS2, GSDMM has gotten one leading value in DBI based on the GMM clustering method. TE-GSDMM has scored ahead in other cases. It proves that the quality of service representation vector generated by TE-GSDMM is better than GSDMM.

We calculated the average scores of GSDMM and TE-GSDMM under different clustering methods in the same dataset. The average scores of evaluation metrics for GSDMM and TE-GSDMM in different datasets are illustrated in Fig. 11, Fig. 12 and Fig. 13. Fig. 11 shows the metric comparison between the GSDMM and TE-GSDMM in DS1. Similarly, the metric comparisons

Table 4
The coherence scores for different topic models based on DS1-DS3.

	GSDMM	TE-GSDMM	LDA	HDP	BTM	LSA
DS1	−2.1153	−1.8066	−4.2617	−15.5530	−1.8112	−2.3228
DS2	−2.5608	−2.4307	−4.2442	−14.7090	−2.0356	−2.5572
DS3	−2.2902	−2.1918	−5.3865	−13.9450	−2.2231	−2.6700

Table 5
Scores of evaluation metrics for different topic models in DS1.

Method	Topic model	SC	DBI	AMI	NMI	ARI	CRS
AGNES	BTM	0.296	1.339	0.398	0.411	0.224	0.941
	GSDMM	0.917	0.935	0.439	0.446	0.310	0.889
	HDP	0.327	0.969	0.059	0.069	0.024	0.958
	LDA	0.216	1.739	0.206	0.212	0.122	0.960
	LSA	0.114	1.571	0.315	0.363	0.099	0.958
	TE-GSDMM	0.903	0.857	**0.508**	0.513	**0.359**	0.833
BIRCH	BTM	0.303	1.369	0.409	0.417	0.240	0.941
	GSDMM	0.915	0.924	0.437	0.444	0.309	0.900
	HDP	0.324	0.963	0.060	0.068	0.023	0.958
	LDA	0.222	1.725	0.208	0.214	0.122	0.960
	LSA	0.115	1.491	0.309	0.360	0.091	0.958
	TE-GSDMM	0.916	**0.837**	0.500	0.505	0.354	0.800
GMM	BTM	0.072	2.483	0.386	0.393	0.229	0.933
	GSDMM	0.742	1.218	0.426	0.434	0.289	0.900
	HDP	−0.073	4.021	0.045	0.071	0.006	0.966
	LDA	−0.058	5.024	0.112	0.122	0.048	0.962
	LSA	−0.018	2.723	0.363	0.374	0.220	0.950
	TE-GSDMM	0.754	1.199	0.493	0.497	0.347	0.800
Kmeans++	BTM	0.334	1.222	0.402	0.416	0.203	0.941
	GSDMM	**0.920**	0.925	0.440	0.446	0.310	0.900
	HDP	0.354	0.886	0.059	0.067	0.022	0.957
	LDA	0.306	1.588	0.222	0.227	0.131	0.960
	LSA	0.131	1.472	0.328	0.367	0.120	0.958
	TE-GSDMM	0.906	0.851	**0.508**	**0.514**	0.358	0.833

Table 6
Scores of evaluation metrics for different topic models in DS2.

Method	Topic model	SC	DBI	AMI	NMI	ARI	CRS
AGNES	BTM	0.265	1.367	0.411	0.426	0.188	0.933
	GSDMM	0.911	0.962	0.399	0.440	0.211	0.909
	HDP	0.272	1.376	0.115	0.140	0.026	0.960
	LDA	0.175	1.471	0.211	0.231	0.077	0.960
	LSA	0.068	1.975	0.359	0.382	0.126	0.958
	TE-GSDMM	0.941	**0.334**	**0.485**	0.520	0.289	0.800
BIRCH	BTM	0.271	1.348	0.406	0.421	0.181	0.933
	GSDMM	0.886	1.238	0.392	0.433	0.209	0.909
	HDP	0.322	1.468	0.115	0.144	0.028	0.960
	LDA	0.177	1.499	0.213	0.233	0.081	0.960
	LSA	0.070	1.951	0.351	0.376	0.119	0.958
	TE-GSDMM	0.930	0.691	0.483	0.518	0.289	0.800
GMM	BTM	0.094	2.152	0.395	0.415	0.177	0.929
	GSDMM	0.750	1.215	0.392	0.428	0.204	0.900
	HDP	0.049	3.615	0.108	0.151	0.034	0.960
	LDA	−0.097	5.344	0.118	0.142	0.033	0.966
	LSA	−0.027	2.667	0.378	0.394	0.167	0.952
	TE-GSDMM	0.756	1.442	0.460	0.493	0.264	0.833
Kmeans++	BTM	0.307	1.177	0.394	0.413	0.141	0.938
	GSDMM	0.922	0.809	0.398	0.440	0.210	0.900
	HDP	0.318	1.183	0.111	0.137	0.025	0.960
	LDA	0.233	1.554	0.218	0.239	0.084	0.960
	LSA	0.090	1.695	0.337	0.362	0.111	0.958
	TE-GSDMM	**0.946**	0.555	0.484	**0.521**	**0.291**	0.800

in DS2 and DS3 are respectively presented in Fig. 12 and Fig. 13.We can see that all the average scores of TE-GSDMM are higher than that of GSDMM except for SC in DS1. According to our analysis, the main reason for this phenomenon is the imbalance of data distribution in the dataset. With the increasing number of services in the dataset, the performance of TE-GSDMM is stable and better than GSDMM.

Quantitatively compared with the GSDMM model, the clustering quality evaluation metrics SC, DBI, AMI, NMI and ARI were respectively improved by 2.1%, 18.3%, 19%, 16.3% and 23.9% in the average scores of the three datasets. Furthermore, we investigate the best evaluation metric scores of the TE-GSDMM model in different clustering methods based on three datasets. An overview of the optimal metric values related to different clustering methods is presented in Table 8. We have counted the times of the optimal values of four clustering methods on five evaluation metrics. K-means++ has obtained 9 of the 15 best scores of the three datasets. AGNES and BIRCH got 5 and 3 times respectively. The three methods have got the

Table 7
Scores of evaluation metrics for different topic models in DS3.

cluster	Topic model	SC	DBI	AMI	NMI	ARI	CRS
AGNES	BTM	0.189	1.341	0.365	0.436	0.125	0.933
	GSDMM	0.875	0.958	0.347	0.425	0.174	0.923
	HDP	0.345	0.854	0.056	0.139	0.006	0.957
	LDA	0.130	1.700	0.196	0.278	0.052	0.960
	LSA	0.031	2.362	0.328	0.398	0.098	0.958
	TE-GSDMM	0.913	0.817	0.423	0.495	0.214	0.800
BIRCH	BTM	0.184	1.372	0.364	0.436	0.127	0.933
	GSDMM	0.886	0.917	0.347	0.426	0.173	0.923
	HDP	0.375	0.879	0.056	0.137	0.006	0.957
	LDA	0.130	1.515	0.195	0.278	0.050	0.960
	LSA	0.043	2.271	0.330	0.397	0.087	0.958
	TE-GSDMM	0.919	**0.716**	0.422	0.495	0.216	0.800
GMM	BTM	0.155	1.610	0.363	0.435	0.128	0.917
	GSDMM	0.668	1.850	0.354	0.424	0.183	0.923
	HDP	−0.301	2.304	0.061	0.125	0.009	0.964
	LDA	−0.076	3.338	0.150	0.231	0.031	0.962
	LSA	0.028	2.463	0.326	0.397	0.098	0.952
	TE-GSDMM	0.680	1.471	**0.425**	0.491	**0.220**	0.800
Kmeans++	BTM	0.222	1.312	0.358	0.428	0.112	0.933
	GSDMM	0.879	0.914	0.347	0.425	0.175	0.923
	HDP	0.383	0.801	0.056	0.139	0.006	0.957
	LDA	0.172	1.733	0.197	0.281	0.056	0.960
	LSA	0.049	2.137	0.313	0.382	0.084	0.958
	TE-GSDMM	**0.921**	0.744	0.422	**0.496**	0.216	0.800

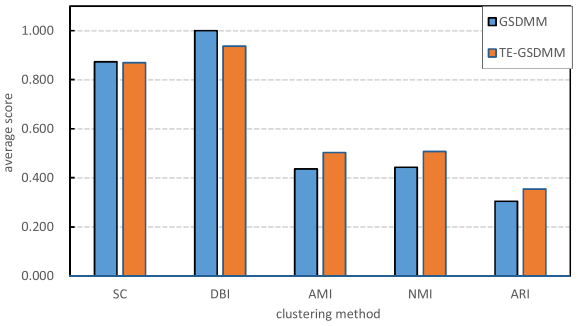

Fig. 11. Comparison of average value for each metric in DS1.

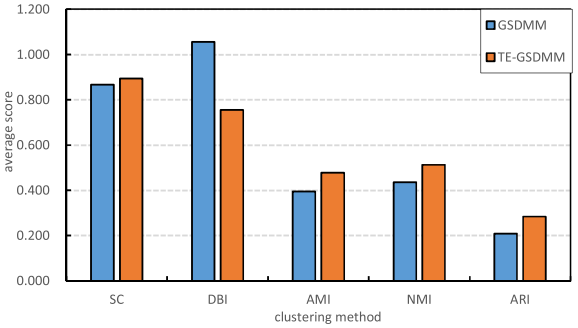

Fig. 12. Comparison of average value for each metric in DS2.

best scores for 17 times because some metrics were tied for the first place. We select the K-means++ as our clustering method due to its excellent performance and low requirement of computing resources.

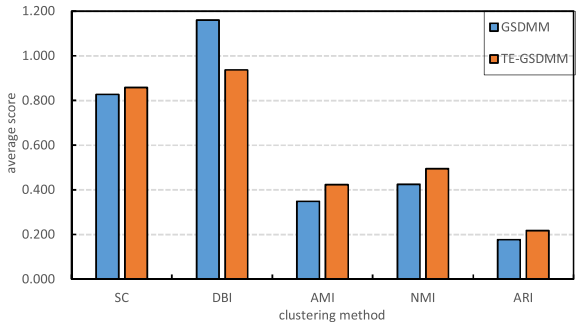

Fig. 13. Comparison of average value for each metric in DS3.

Table 8
Statistical results of the optimal metric value of clustering method.

Method	SC	DBI	AMI	NMI	ARI	SUM
K-means++	3	0	1	3	2	9
AGNES	0	1	3	0	1	5
BIRCH	0	2	0	0	1	3
GMM	0	0	0	0	0	0

6.2.4. Performance comparison between KM_TG and KM_TG_SC

Experiments in the previous Section have proved that the quality of service representation vectors generated by the proposed TE-GSDMM model is significantly higher than other models. Meanwhile, the experiment data showed that the performance of the K-means++ algorithm was better than other algorithms. So we have designed a new service clustering algorithm which was named as KM_TG_SC in Section 5. KM_TG_SC is a K-means++ based clustering method. It determines the service similarity by integrating service function and collaboration similarity.

This section presents an experiment to verify that the service clustering quality is improved by integrating service collaboration similarity into KM_TG_SC. The dataset DS4 is composed of Mashup services in PW. There are 34 categories and total of 774 Web services. We build the service collaboration graph based on these services and their collaborations in the Mashup services. Algorithm 2 is employed to generate service collaboration vectors. Performances of the following service clustering methods are investigated.

(1) K-means++ algorithm with TE-GSDMM (KM_TG);
(2) K-means++ algorithm with TE-GSDMM integrating service collaboration similarity (KM_TG_SC).

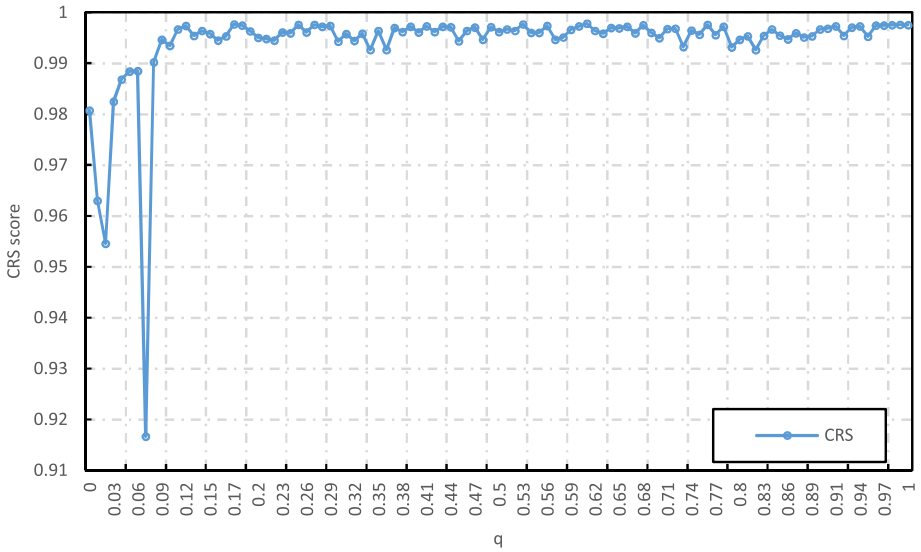

Fig. 14. CRS of KM_TG_SC with different hyper parameter q.

Table 9
Score comparison of KM_TG and KM_TG_SC.

Clustering method	Topic	SC	DBI	AMI	NMI	ARI
KM_TG	25	0.733	1.032	0.408	0.551	0.213
KM_TG_SC	25	0.801	0.921	0.442	0.574	0.260

The testing topic numbers of TE-GSDMM in DS4 and its comprehensive ranking scores are illustrated in Fig. 9. The optimal topic number is set as 25 according to comprehensive ranking scores. Then the experiments are carried out to verify the performances of KM_TG and KM_TG_SC. Fig. 14 shows the comprehensive ranking score of KM_TG_SC with different hyper parameter q. We assigned q with the value 0.07 in order to obtain the best service clustering quality for KM_TG_SC.

In this experiment, the service collaboration similarity only accounts for 7% when calculating the similarity in service clustering. The main reason is that service collaboration in mashup services is relatively sparse. With the increase in the number of service collaborations, the proportion of service collaboration similarity in clustering would be increased to obtain the best clustering quality.

Table 9 summarizes SC, DBI, AMI, NMI and ARI scores for KM_TG and KM_TG_SC based on DS4. Compared with KM_TG, the scores of SC, DBI and AMI, NMI and ARI were respectively improved by 9.3%, 10.8%, 8.3%, 4.2% and 22.1% in KM_TG_SC. It proves that the performance of KM_TG_SC is enhanced by integrating the service collaboration similarity in the clustering method.

Therefore, Experiments in Section 6.2.3 have shown that the quality of service representation vectors generated by the GSDMM model is higher than that of a traditional topic model. Furthermore, the proposed TE-GSDMM has a better performance than GSDMM. Four clustering methods are employed to cluster Web services with different topic models in the experiments. We have verified that the clustering method with the best performance is the K-mean++ algorithm integrating TE-GSDMM. In this section, we prove that the KM_TG_SC has further improved the performance by considering service collaboration similarity. So the proposed method can effectively enhance Web service clustering based on the TE-GSDMM and service collaboration graph.

7. Conclusions

In this paper, an improved GSDMM model (TE-GSDMM) is proposed to overcome the low quality of service representation vectors generated by traditional topic models. TE-GSDMM can reduce sparsity and balance of topic distribution in service representation vectors by integrating feature word extraction and probability distribution correction. It can increase the calculation accuracy of service functional similarity. We also put forward the service collaboration graph to model the collaborations between Web services. The collaboration similarity can be obtained based on the service collaboration graph. By integrating function and collaboration similarity into K-means++, an efficient service clustering method named KM_TG_SC is designed. We carried out a series of experiments to verify the performances of the TE-GSDMM and KM_TG_SC algorithms. Experiments proved that the proposed model and method had enhanced the Web service clustering.

In the future, we will investigate the influence of the proportion of verb, noun and adjective in the feature words towards the quality of service representation vectors. The number of service collaborations will also be added into the service collaboration map. Then the calculation accuracy of service collaboration similarity can be improved and the service clustering quality can be further enhanced.

CRediT authorship contribution statement

Qiang Hu: Conceptualization, Writing – original draft. **Jiaji Shen:** Validation, Formal analysis. **Kun Wang:** Investigation. **Junwei Du:** Visualization. **Yuyue Du:** Supervision, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 61973180, the Natural Science Foundation of Shandong Province under Grant ZR2019MF033, ZR2021MF092 and the key research program of Shandong Province (Soft Sciences) under Grant 2021RKY02037.

References

[1] N. Niknejad, W. Ismail, I. Ghani, B. Nazari, M. Bahari, A.R.B.C. Hussin, Understanding Service-Oriented Architecture (SOA): a systematic literature review and directions for further investigation, Inf. Syst. 91 (2020) 101491, https://doi.org/10.1016/j.is.2020.101491.
[2] H. Wang, X. Hu, Q.i. Yu, M. Gu, W. Zhao, J. Yan, T. Hong, Integrating reinforcement learning and skyline computing for adaptive service composition, Inf. Sci. 519 (2020) 141–160.
[3] D. Rathod, Performance evaluation of restful web services and soap/wsdl web services, Int. J. Adv. Res. Comp. Sci. 8 (7) (2017) 415–420.
[4] X. Zhang, J. Liu, B. Cao, M. Shi, Web service classification based on information gain theory and bidirectional long short-term memory with attention mechanism, Concurrency and Computation: Practice and Experience, e6202, 2021.
[5] K. Elgazzar, A.E. Hassan, and P. Martin, Clustering WSDL Documents to Bootstrap the Discovery of Web Services, 2010 IEEE International Conference on Web Services, pp. 147–154.
[6] K. Xiao, Z. Qian, B. Qin, A graphical decomposition and similarity measurement approach for topic detection from online news, Inf. Sci. 570 (2021) 262–277.
[7] G. Costa, R. Ortale, Jointly modeling and simultaneously discovering topics and clusters in text corpora using word vectors, Inf. Sci. 563 (2021) 226–240.
[8] T. Liang, L. Chen, H. Ying, J. Wu, Co-clustering WSDL documents to bootstrap service discovery, in: 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications, 2014, pp. 215–222.
[9] J. Wu, L. Chen, Z. Zheng, M.R. Lyu, Z. Wu, Clustering web services to facilitate service discovery, Knowl. Inf. Syst. 38 (1) (2014) 207–229.
[10] N. Agarwal, G. Sikka, L.K. Awasthi, Enhancing web service clustering using Length Feature Weight Method for service description document vector space representation, Expert Syst. Appl. 161 (2020) 113682, https://doi.org/10.1016/j.eswa.2020.113682.
[11] https://www.programmableweb.com.
[12] B. Cao, X. Liu, MD.M. Rahman, B. Li, J. Liu, M. Tang, Integrated content and network-based service clustering and web APIs recommendation for mashup development, IEEE Trans. Serv. Comput. 13 (1) (2020) 99–113.
[13] A.R. Baskara, R. Sarno, Web service discovery using combined bi-term topic model and WDAG similarity, in: 2017 11th International Conference on Information & Communication Technology and System (ICTS), 2017, pp. 235–240.
[14] H. Rong, L. Jianxun, W. Yiping, SP-BTM: A Specific Part-of-speech BTM for Service Clustering. 2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking, 2020.
[15] B. Cao, Q. Xiao, X. Zhan, J. Liu, An API Service recommendation method via combining self-organization map-based functionality clustering and deep factorization ma-chine-based quality prediction, Chinese J. Comp. 42 (6) (2019) 1367–1383.
[16] K.K. Fletcher, A quality-based web API selection for mashup development using affinity propagation, in: International conference on services computing, 2018, pp. 153–165.
[17] R. Das, M. Zaheer, C. Dyer, Gaussian lda for topic models with word embeddings, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Volume 1: Long Papers, 2015, pp. 795–804.
[18] I. Lizarralde, C. Mateos, A. Zunino, T.A. Majchrzak, T.-M. Grønli, Discovering web services in social web service repositories using deep variational autoencoders, Inf. Process. Manage. 57 (4) (2020) 102231, https://doi.org/10.1016/j.ipm.2020.102231.
[19] M. Shi, Y. Tang, J. Liu, Functional and Contextual Attention-Based LSTM for Service Recommendation in Mashup Creation, in IEEE Transactions on Parallel and Distributed Systems, vol. 30, no. 5, pp. 1077-1090, 2019.
[20] A. Bukhari, X. Liu, A Web service search engine for large-scale Web service discovery based on the probabilistic topic modeling and clustering, SOCA 12 (2) (2018) 169–182.
[21] A. Onan, H. Bulut, S. Korukoglu, An improved ant algorithm with LDA-based representation for text document clustering, J. Inf. Sci. 43 (2) (2017) 275–292.
[22] W. Pan, X. Xu, H. Ming, C.K. Chang, Clustering Mashups by Integrating Structural and Semantic Similarities Using Fuzzy AHP, Int. J. Web Services Res. (IJWSR), 18(1): 34–57, 2021.
[23] J. Yin, J. Wang, A dirichlet multinomial mixture model-based approach for short text clustering, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 233–242.
[24] N. Agarwal, G. Sikka, L.K. Awasthi, Evaluation of web service clustering using Dirichlet Multinomial Mixture model based approach for Dimensionality Reduction in service representation, Inf. Process. Manage. 57 (4) (2020) 102238, https://doi.org/10.1016/j.ipm.2020.102238.
[25] M.E. Khanouche, F. Attal, Y. Amirat, A. Chibani, M. Kerkar, Clustering-based and QoS-aware services composition algorithm for ambient intelligence, Inf. Sci. 482 (2019) 419–439.
[26] K. Zanbouri, N. Jafari Navimipour, A cloud service composition method using a trust-based clustering algorithm and honeybee mating optimization algorithm, Int. J. Commun. Syst., 33(5): e4259, 2020.
[27] O.A. Wahab, R. Cohen, J. Bentahar, H. Otrok, A. Mourad, G. Rjoub, An endorsement-based trust bootstrapping approach for newcomer cloud services, Inf. Sci. 527 (2020) 159–175.
[28] K. Su, X. Su, L. Yang, B. Xiao, A clustering-based user reputation evaluation approach for web service recommendation, in: Fuzzy Systems and Data Mining IV, IOS Press, 2018, pp. 260–267.
[29] Y. Wu, C. Yan, Z. Ding, G. Liu, P. Wang, C. Jiang, M. Zhou, A novel method for calculating service reputation, IEEE Trans. Autom. Sci. Eng. 10 (3) (2013) 634–642.
[30] J. Liu, Y. Chen, A personalized clustering-based and reliable trust-aware QoS prediction approach for cloud service recommendation in cloud manufacturing, Knowl.-Based Syst. 174 (2019) 43–56.
[31] J. Du, E. Gelenbe, C. Jiang, H. Zhang, Y. Ren, H.V. Poor, Peer prediction-based trustworthiness evaluation and trustworthy service rating in social networks, IEEE Trans. Inf. Forensics Secur. 14 (6) (2019) 1582–1594.
[32] H. Mezni, D. Benslimane, L. Bellatreche, Context-aware service recommendation based on knowledge graph embedding, IEEE Trans. Knowl. Data Eng. (2021), https://doi.org/10.1109/TKDE.2021.3059506.
[33] H. Li, M. Liang, T. He, Optimizing the composition of a resource service chain with interorganizational collaboration, IEEE Trans. Ind. Inf. 13 (3) (2017) 1152–1161.
[34] X. Xue, S. Wang, B. Lu, Manufacturing service composition method based on networked collaboration mode, J. Network Comp. Appl. 59 (2016) 28–38.
[35] P. Li, Y. Cheng, W. Song, F. Tao, Manufacturing services collaboration: connotation, framework, key technologies, and research issues, Int. J. Adv. Manuf. Technol. 110 (9-10) (2020) 2573–2589.
[36] N.a. Xie, W. Tan, X. Zheng, L.u. Zhao, L.i. Huang, Y. Sun, An efficient two-phase approach for reliable collaboration-aware service composition in cloud manufacturing, J. Ind. Inf. Integr. 23 (2021) 100211, https://doi.org/10.1016/j.jii.2021.100211.
[37] https://www.casicloud.com/.
[38] https://www.haier.com/haier_cosmoplat/.
[39] J. Qiang, Z. Qian, Y. Li, Y. Yuan, X. Wu, Short text topic modeling techniques, applications, and performance: a survey, IEEE Trans. Knowledge Data Eng., doi: 10.1109/TKDE.2020.2992485.
[40] N. Almarimi, A. Ouni, S. Bouktif, M.W. Mkaouer, R.G. Kula, M.A. Saied, Web service API recommendation for automated mashup creation using multi-objective evolutionary search, Appl. Soft Comput. 85 (2019) 105830, https://doi.org/10.1016/j.asoc.2019.105830.
[41] C. Liu, Q. Zeng, L. Cheng, H. Duan, M. Zhou, J. Cheng, Privacy-preserving behavioral correctness verification of cross-organizational workflow with task synchronization patterns, IEEE Trans. Autom. Sci. Eng. 18 (3) (2021) 1037–1048.

[42] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.
[43] Z. Xu, D. Shen, T. Nie, Y. Kou, N. Yin, X. Han, A cluster-based oversampling algorithm combining SMOTE and k-means for imbalanced medical data, Inf. Sci. 572 (2021) 574–589.
[44] Zhao Y, Qiao Y, He K. A novel tagging augmented LDA model for clustering. Int. J. Web Services Res., 2019,16(3): 59–77.
[45] D. Mimno, H. Wallach, E. Talley, et al. Optimizing semantic coherence in topic models. Proceedings of the 2011 conference on empirical methods in natural language processing. 2011: 262–272.
[46] Y. Cao, J. Liu, M. Shi, et al, Relationship network augmented web services clustering, in: 2019 IEEE International Conference on Web Services (ICWS), 2019, pp. 247–254.