Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 7225552, 12 pages
https://doi.org/10.1155/2021/7225552

Research Article

Service Partition Method Based on Particle Swarm Fuzzy Clustering

Hong Xia,[1,2,3] **Qingyi Dong**,[1] **Hui Gao**,[1] **Yanping Chen**,[1,2,3] **and ZhongMin Wang**[1,2,3]

[1]*School of Computer Science and Technology, Xi'an University of Posts and Telecommunications, Xi'an, 710121 Shaanxi, China*
[2]*Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi'an, 710121 Shaanxi, China*
[3]*Xi'an Key Laboratory of Big Data and Intelligent Computing, Xi'an, 710121 Shaanxi, China*

Correspondence should be addressed to Yanping Chen; chenyp@xupt.edu.cn

Received 3 September 2021; Revised 27 October 2021; Accepted 17 November 2021; Published 8 December 2021

Academic Editor: Yueshen Xu

It is difficult to accurately classify a service into specific service clusters for the multirelationships between services. To solve this problem, this paper proposes a service partition method based on particle swarm fuzzy clustering, which can effectively consider multirelationships between services by using a fuzzy clustering algorithm. Firstly, the algorithm for automatically determining the number of clusters is to determine the number of service clusters based on the density of the service core point. Secondly, the fuzzy *c*-means combined with particle swarm optimization algorithm to find the optimal cluster center of the service. Finally, the fuzzy clustering algorithm uses the improved Gram-cosine similarity to obtain the final results. Extensive experiments on real web service data show that our method is better than mainstream clustering algorithms in accuracy.

1. Introduction

With the development of service-oriented architecture technology, web service has become a vital software resource on the Internet. The number, scale, and types of services have grown rapidly, and services with similar functions have also increased. In the current situation, the difficulty of managing services and assisting in service discovery is time-consuming. Therefore, how to manage web services more conveniently and quickly and accurately find the service that meets the needs of users in a large number of services is a big challenge [1].

The service clustering method can effectively help manage services and assist in service discovery. Web service clustering method has become a key method for service discovery, service recommendation, and service management, which can help web service search engines search services and reduce their search space [2]. Service clustering is aimed at dividing multiple services into different clusters based on similarity. In the same cluster, each service is as similar as possible, while in different clusters, each service is as different as possible. Service clustering can better classify services, compress search space, shorten search time, help quickly manage services, and provide users with accurate and efficient services.

Most related research shows that the service clustering method is based on a topic model, which can improve the efficiency of search services. Many scholars have studied the service clustering method based on the topic model. Paper [3] first applies BTM to learn the latent topics of web service description corpus and then uses the *k*-means algorithm to cluster web services. Considering the web service's description on text is short and lacks enough adequate information. Paper [4] proposes a web service clustering method based on Word2vec and Latent Dirichlet Allocation (LDA) topic model. Word2vec expands the content of web service description documents. Then, use the topic model to model the extended description document.

Most topic models cause web service clustering with low accuracy because most topic models cannot build a well model with short text. Paper [5] proposes a web service clustering with multifunctionality based on LDA and fuzzy *C* -means algorithm. LDA topic model is used to model description documents of web service, and fuzzy *c*-means

algorithm clusters web services into different functional classes. Paper [6] proposes a semantic web service discovery based on fuzzy clustering optimization. As the preprocessing part, the improved fuzzy *c*-means clustering algorithm cluster services into a different class. In this preprocessing process, the improved fuzzy clustering algorithm considers four functional parameters of service input, output, premise, and effect of service as the clustering parameters.

In summary, the existing web service cluster method only focuses on the cluster of individual services and does not consider the connection between services. In fact, there is a mutual invocation relationship between services, and they are not independent individuals. If the interconnection between services is not considered, the accuracy of service clustering will be affected. In addition, most of the current clustering algorithms select the cluster position randomly. However, in a real web environment, this usually leads to poor accuracy of the clustering algorithm. Therefore, it is more difficult to dig out the interconnections between services. Now, most service clustering technology mainly uses LDA model and *k*-means algorithms to work in the same field. Generally, the existing work has the following two shortcomings. Firstly, the semantic relationship between words is not fully considered, leading to unsatisfactory service discovery results. Secondly, the interconnection between services is not fully considered, resulting in low service clustering accuracy.

We propose a service partition method based on particle swarm fuzzy clustering (NFC-NSPO). Firstly, this method first preprocesses web service description and fully considers the semantic relationship between words. Then, use the automatic clustering algorithm to determine the number of service clusters. Secondly, fuzzy clustering algorithm combines particle swarm optimization, in order to avoid fuzzy clustering algorithm random selected of cluster location and random selection of cluster location caused poor accuracy of fuzzy clustering algorithm. Finally, the fuzzy clustering algorithm is based on Gram-improved cosine to measure the similarity of services. The function based on Gram-improved cosine similarity is used to control the sliding window to compare the service description one by one.

2. Background and Related Work

2.1. Web Service Clustering. In the past, the number diversity of web services has increased rapidly and still keeps emerging [7]. Many researchers pay much attention to service-oriented tasks [8, 9], and service computing developed so fast. Web service clustering is one of the most classical and important tasks in service computing [10, 11].

Service clustering is an integral approach to manage services and assist service discovery [12]. Service clustering is an essential part of service matching, service recommendation, service composition, and service discovery. Service clustering decomposes so many services into a set of smaller clusters to help service engineers manage the system effectively. Service engineers match or recommend a set of services in different clusters according to customer demand.

Web service clustering work can be classified into two types: semantic web services and nonsemantic web services [13]. For the semantic web services, combine keywords extracted from the web service document languages (WSDL). This method describes semantic levels through users' queries and searches by keywords. Clustering based on semantic web services is relatively mature.

Bo et al. present a service clustering based on the functional semantics requirements (SCFSR). It extracts functional information in the service requirement documents by natural language processing, then calculated the similarity between functional information matrix. Finally, apply *k* -means to cluster these services [1]. In paper [14] since web service description documents are short, they use Word2vec to expand the content of description document and then use the LDA topic model to find web services.

Sheeba et al. propose mathematical web service semantic description and registration by ontology. They use an ontology tree to catalog the mathematical web service characteristics, about functional as well nonfunctional [15]. Nguyen and Kuo present a web service discovering through ontology matching semantic relationships. The ontology is built to represent the relationship between semantics with keywords matching, and keywords matching method can find best suitable service for user request [16].

Hsu and Chiu propose a semantic Latent Dirichlet Allocation, which obtains synonym table and then acquisition domain feature word set by Word2vec model. It clusters same domain services, and based on this, builds a framework domain semantic-aided web service clustering [17]. Paper [18] proposes an improved multirelational topic model for web service clustering. Since web service description documents contain limited words, the existing LDA model is hard for short text documents. They care about web service multirelational network, so build a model called MR-LDA. This model consider relationship and annotation relationships and then apply a clustering algorithms to get final results.

For the nonsemantic web service clustering. Service clustering methods do not consider the semantic relationship among services; they pay more attention to the service clustering method. In paper [19], they propose a cluster feature-based latent factor model for Qos prediction. Divide users and services into different groups based on historical records. And that is the same group; users and services have the same latent feature. Furthermore, they design an integrated latent factor model to cluster. In [20], a *k*-means clustering method based on principal component analysis was proposed to predict web services. Solve the problem of low quality accuracy of service prediction caused by the sparse web service matrix.

In short, the particle swarm fuzzy clustering proposed in this paper adopts the fuzzy clustering algorithm based on Gram-improved cosine similarity to consider the connection between services in more detail. At the same time, combining with the particle swarm algorithm can find the optimal service cluster center. It also avoids the fuzzy *c*-means algorithm to randomly select the cluster center, thereby improving the accuracy of service clustering.

2.2. Particle Swarm Optimization Algorithm. The particle swarm optimization algorithm (PSO) simulates the predation behavior of a flock of birds. A flock of birds is searching for food at random, and there is only one piece of food in this area. All the birds do not know where the food is. But they know how far they are from the food. The best way to find food is to search for birds around the area close to the food. The particle swarm optimization algorithm is a kind of bionic evolutionary algorithm [21].

The mathematical model of PSO [21] supposes that in a search space D, the position of the particle is $X_i = \{x_{i1}, x_{i2}, \cdots, x_{\text{in}}\}$, the particle velocity is $V_i = \{v_{i1}, v_{i2}, \cdots, v_{\text{in}}\}$, and p_{best} represents the personal best position of current particle, and g_{best} represents the global best position of current particle, the particle update velocity, and position according to the equation:

$$\begin{aligned} v_{id}^{m+1} &= v_{id}^{m} + c_1 r_1 (p_{id}^{m} - v_{id}^{m}) + c_1 r_1 \left(p_{gd}^{m} - p_{id}^{m}\right), \\ v_{id}^{m+1} &= v_{id}^{m} + v_{id}^{m+1}. \end{aligned} \tag{1}$$

v_{id} represents the velocity of particles, x_{id} represents the position of the particle, m is number of iterations, p_i represents the personal best position of current particle, p_g represents the global best position of current particle, r_1, r_2 represents the acceleration factor which is a random value between 0 and 1, and c_1, c_2 represents the influence degree of personal best and global best position on particle moving direction.

2.3. Fuzzy C-Means Clustering Algorithm. In many problems, the result is only two possibilities, 0 or 1. For example, a student is either a boy or a girl. But this cannot describe the attributes of many things, such as the degree of hot or cold weather. There is no clear definition of what temperature is hot and what is cold. The reason is that in many cases, the boundaries between multiple categories are not absolutely clear. It is needed to use vague words to judge. Fuzzy logic extends the general concept of taking only 1 or 0 (belonging to/not belonging) to taking real numbers between 0 and 1, "Degree of Membership Function." The "Degree of Membership Function" is used to describe the relationship between elements and sets, and the degree of membership is used to express the probability of a sample belonging to a certain class.

Fuzzy c-means clustering algorithm (FCM) is a partition-based clustering algorithm. FCM combines the essence of fuzzy theory. Compared with the hard clustering of k-means, FCM provides more flexible clustering results. Because in most cases, the objects in the dataset cannot be divided into clearly separated clusters, assigning an object to a specific cluster is a bit blunt, and errors may also occur. Therefore, a weight is assigned to each object and each cluster, indicating the degree to which the object belongs to the cluster. Of course, probability-based methods can also give such weights, but sometimes, it is difficult for us to determine a suitable statistical model. Therefore, it is a better choice to use FCM with natural and nonprobabilistic characteristics.

FCM work for service clustering base on the similarity between services in dataset and service clusters c through the iteration of the objective function; the final service clustering result is obtained. The objective function is as follows:

$$\begin{aligned} \text{Min } J_m(U, V, X) &= \sum_{i=1}^{c} \sum_{j=1}^{n} \left(u_{ij}\right)^m d_{ij}^2, \\ \text{s.t.} \sum_{i=1}^{c} u_{ij} &= 1, 1 < j < n, \\ 0 < u_{ij} &< 1, 1 < i < c, 1 < j < n. \end{aligned} \tag{2}$$

$X = \{x_1, x_2, \cdots, x_n\}$ represents service dataset, n is the number of service, $V = \{v_1, v_2, \cdots, v_n\}$ represents c cluster center of service clustering, u_{ij} represents degree of membership of the i service sample belong to cluster c, and d_{ij} represents distance between service sample i and service cluster j. In this paper, d_{ij} applies improved cosine similarity based on Gram.

3. Method

We introduce the presented framework in Section 3.1 and details from Section 3.2 to Section 3.6.

3.1. Framework. The flowchart of the service partitioning method based on particle swarm fuzzy clustering is proposed in this chapter. The method is divided into the preprocessing part of web service description and the service partition method NFC-NSPO based on particle swarm fuzzy clustering (as shown in Figure 1). The left part is the preprocessing part of the web service description. First, crawl the web service description from the programmable website and write it into excel, then extract keywords from excel and filter stop words from it, and finally, restore the word to the stem and use TF-IDF to calculate the frequency of each word. Among them, the preprocessing part is an important part of the service partition method based on particle swarm fuzzy clustering. The right part is the main introduction of NFC-NSPO, a service partition method based on particle swarm fuzzy clustering, which is divided into the following steps. The first step is to identify the number of service clusters and use it as the number of particles in the particle swarm algorithm, where each particle is designed into two parts. The first part is the control variable used to identify the number of the service cluster. The second part of the function is the service distribution of the cluster. The second step is to initialize the speed and position of the particles and calculate the fitness value of each particle. The fitness function is a linear combination of the overall compactness evaluation function and the fuzzy separation function. The third step is to update the speed and position of each particle and repeat the process; if the output condition is met, the output is performed; if the condition is not met, return to the third step. The output result is based on

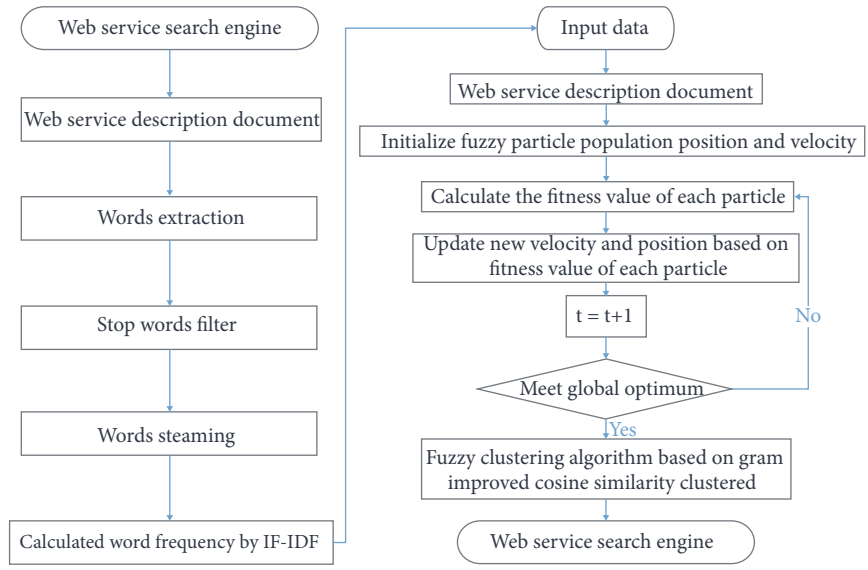

Figure 1: Flow chart of service partition method based on particle swarm fuzzy clusterin.

Gram-improved cosine similarity fuzzy clustering algorithm. Clustering obtains the final clustering result.

3.2. Identify the Number of Service Clusters (K). In the service clustering algorithm, the number of clusters plays a vital role in the accuracy of service clustering. Most existing clustering algorithms used empirical rules that is $k \leq \sqrt{n}$ to determine the number of clusters, k represents the number of service clusters, and n is the number of service samples. There are some drawbacks about empirical rules to identify k the number of service clusters, for large datasets. The number of service clusters k will be very large, which will increase the time complexity of service clustering. For small datasets, k, the number of service clusters will be greater than or equal to the number of samples.

In the study of how to identity the number of service clusters [22], most of the ideas about identifying the number of service clusters is based on the local density of service sample points. The center of the service cluster is surrounded by other services, so the local density center of the service cluster is larger than noncenter density. For example, [23] proposes a clustering algorithm using relative KNN kernel density called RECOME. Firstly, this algorithm is to determine the core object, also known as the cluster center. Secondly, sort according to the local density of the core object. Finally, the point with the highest density is selected as the first cluster center, so that the adjacent noncore object data points form a cluster, and the other data points repeat this process became clusters.

Since RECOME algorithm [23] is only suitable for numerical data, this paper solves the problem of how to determine the number of service clustering clusters. The service description data WSDL belongs to the text, so the formula for calculating the density of core numerical data should be modified to the density formula for calculating the core points of service.

Let X is a set of n service data objects with m attributes. Each service x_i can be represented as a set with m classification attributes as features. Therefore, a certain service x_i can be expressed as $x_i = \{x_{i1}, x_{i2}, \cdots, x_{im}\}$, and the core point density of the service x_i is expressed as $\mathrm{Dens}(x_i)$, so density of each objects can be defined as follows [24]:

$$\mathrm{Dens}(x_i) = \frac{\sum_{l=1}^{m} \mathrm{Dens}_l(x_i)}{m},$$
$$\mathrm{Dens}_l(x_i) = \frac{\left|\left\{x_j \in X \mid x_{il} = x_{jl}\right\}\right|}{n}. \tag{3}$$

For each attribute $l \in m$, $\mathrm{Dens}(x_i) = 1/n$, if:

$$\left|\left|\left\{x_j \in X \mid x_{il} = x_{jl}\right\}\right| = 1 x_{il} = x_{jl}\right| = 1. \tag{4}$$

Otherwise, $\mathrm{Dens}(x_i) = 1$ if:

$$\left|\left|\left\{x_j \in X \mid x_{il} = x_{jl}\right\}\right| = n x_{il} = x_{jl}\right| = n. \tag{5}$$

Therefore, the density of a categorical object is limited at $1/n \leq \mathrm{Dens}(x_i) \leq 1$. However, $\mathrm{Dens}(x_i) = 1$ is very rare because it means that the two services are completely similar.

Services adjacent to the service core point are defined as services in noncore areas. dc is a cutoff distance, which represents the distance between the service core point and the service noncore point. To calculate the similarity between the two services using Gram-improved cosine similarity, choose the number of nearest neighbor services about 1% to 2% of the total number of services by Rodriguez and Laio [22]. Specifically, first calculate the density $\mathrm{Dens}(x_i)$ of service x_i and calculate the neighboring service $Neib_{xi}$ of service x_i, and sort $\mathrm{Dens}(x_i)$ in descending order. Second, determine the cutting distance dc. Finally, by identifying the core object and its neighbors several times to form atom clusters, get the number of clusters in each service dataset. The algorithm for identifying the number of services in this paper adopts the literature [16].

Table 1: Representation of particles.

	C	1	2	3	⋯	N
1	c_1	w_1	w_2	w_3	⋯	w_n
…	…	…	…	…	…	…
k	c_k	w_{k1}	w_{k2}	w_{k3}	⋯	w_{kn}

Table 2: The particle representation of $k_{\max} = 4$, $n = 6$.

	C	1	2	3	4	5	6
1	0.6	0.9	0.8	0.8	0.4	0.8	0.5
2	0.1	0	0	0	0	0	0
3	0.8	0.7	0.7	0.3	0.6	0.2	0.5
4	0.4	0	0	0	0	0	0

3.3. Particle Swarm Representation in Web Service. As shown from Table 1, the particle swarm represents a web service clustering fuzzy matrix whose size is $k \times (1 + n)$, where k represents the maximum number of clusters $k_{\max}$ and where n is the number of web services. Cluster control variable in the particle is used to identify the number of clusters that should be defined in web service, which is from 0 to 1. In this case, if the cluster control variable is larger than 0.5 or equal to 0.5, it means the fuzzy membership function will assign web service objects to cluster based on the control variable. On the other hand, if the cluster control variable is less than 0.5, there does not exist a cluster in the variable, and the fuzzy membership values are 0. The web service clusters' assignment is a fuzzy membership matrix $W = (w_{ij})$, $j = 1, 2, \cdots k_{\max}$, when $k_{\max} = 4$, $n = 6$, the specific way of expressing particles as fuzzy matrix is shown in Table 2.

3.4. Fitness Function. The fitness function is the linear combination of the compactness function and the fuzzy separation function, clustering compactness (π), and fuzzy separation (sep) to evaluate the clusters. If clustering compactness (π) is smaller, it represents the clusters is tighter. If fuzzy separation (sep) is larger, it means the distance between clusters is larger, and the gap in different clusters is larger. This function is calculated as follows:

$$\mathrm{Fit}(x_i) = \pi + \mathrm{sep}, \tag{6}$$

$$\pi = \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} w_{ij}^{\alpha} d(x_i, z_j)}{\sum_{i=1}^{n} w_{ij}^{\alpha}}. \tag{7}$$

$$\mathrm{sep} = \sum_{j=1}^{k} \sum_{l=1, l \neq j}^{k} w_{ij}^{\alpha} d(z_j, z_l), \tag{8}$$

where $W = (w_{ij})$ is the fuzzy membership matrix, $k = k_{\max}$, $Z = \{z_1, z_2, \cdots, z_k\}$ is the set of web service cluster centers, α is the weight, $d(x_i, z_j)$ presents the distance between web services i with clusters j, and $d(z_i, z_j)$ is the distance from cluster j and l.

3.5. Particle Swarm Algorithm Procedure. Kennedy et al. proposed a particle swarm algorithm by observing the trajectory of birds looking for food and conducting research [25]. In the particle swarm algorithm, individuals are called particles. The algorithm includes the following steps:

Step1. Initializing particle swarm

N particle population, each particle consists of two parts, control variables and cluster assignment. Control variables identify how many clusters are active. Next, set the velocity and position of the initial particle. The initialization process of particles: first, randomly generate control variables $C = (c_1, c_2, \cdots, c_k)$ from 0 to 1 for all particles, denoted as $C(p)$, $k = k_{\max}$. After calculating the number of active clusters in each particle, when $c_j \geq 0.5$, according to the initialization process in [26], a fuzzy membership matrix with active clusters $h(p)$ is generated; otherwise, no partitioning is performed. Finally, use $h(p)$ to get the $W(p)$ allocated by the cluster. The initialization process of particle velocity: first, randomly generate control variables $V_C = (v_1, v_2, \cdots, v_k)$ for all particles, denoted as $V_c(p)$, $k = k_{\max}$. Finally, use $h(p)$ to get the cluster distribution speed Vw. Note that during the initialization process, the number of cluster activities $h(p)$ must ensure that $k_{\min} \leq h(p) \leq k_{\max}$.

Step 2. Use formula (8) to calculate the fitness function value of each particle and record the number of iterations

Step 3. For each particle, compared with the fitness value, $\mathrm{Fit}(x_i)$ is compared with p_{best}(individual extremum), if $\mathrm{Fit}(x_i) > p_{\mathrm{best}}$, $\mathrm{Fit}(x_i)$ value replaced p_{best}; otherwise, p_{best} keeps before value

Step 4. For each particle, the fitness value $\mathrm{Fit}(x_i)$ is compared with g_{best} (local extremum), if $\mathrm{Fit}(x_i) > g_{\mathrm{best}}$, $\mathrm{Fit}(x_i)$ value replaced g_{best}; otherwise, g_{best} keeps before value

Step 5. Update new position and velocity for all particle; the new position and velocity are updated as follows:

$$V_c^{t+1}(p) = w \times V_c^t(p) + c_1 r_1 \left(p_{\mathrm{best}_c}(p) - C^t(p)\right) + c_2 r_2 \left(g_{\mathrm{best}_c} - C^t(p)\right),$$

$$C^{t+1}(p) = C^t(p) + V_c^{t+1}(p). \tag{9}$$

w is the inertia weight, $C^t(p)$, $V_c^t(p)$, respectively, represent particles p position and speed, c_1,c_2 is positive acceleration constant, representing local and global learning ability of particles, and r_1,r_2 are random numbers in intervals. In the update process, the value of the control variable will be greater than 1 or less than 0 as the velocity value in the particle changes. If $C(p) > 1$, then adjust to 1. If $C(p) > 1$, adjusted to 0. The update position of control variables may lead to changes in the number of active clusters. Therefore, the number of active clusters will also be updated (as shown in formula (13)):

$$h^{t+1}(p) = \mathrm{count}\left(C^{t+1}(p) \middle| c_j > 0.5\right). \tag{10}$$

In addition, in order to increase the flexibility of membership function, this paper uses the degree of hesitation of

IFS Sugenos [27] to add to the speed of cluster allocation. Like the traditional particle swarm algorithm, the algorithm for updating the velocity and position of the particles adopts literature [16].

Step 6. If the condition is satisfied, exit; otherwise, return step 2

3.6. Improved Cosine Algorithm Based on Gram. Gram algorithm [28] considers that the topic of the service is closely related to the words in the description of the service. So, the probability of the words in the service description is used to describe the topic of the service, and the probability of the topic data is its Gram value. If the Gram value is higher, the services are more similar. The Gram algorithm process is as follows:

(1) Dataset preprocessing includes deleting special characters and filtration stop words

(2) Establish corpus, count the number of each word, and record it as N

(3) Each service topic is closely related to the word. Markov probability formula is used to calculate the probability of a topic word as shown in

$$P(S) = P(c_1)P(c_2 \mid c_1)P(c_3 \mid c_2c_1) \cdots P(c_{n-1} \mid c_{n-2} \cdots c_1). \tag{11}$$

$P(S)$ is the probability of the entire data, and c represents the relative topic words.

Since the probability of a certain word in the service is only related to the previous word. Formula (12) can be simplified to formula (13), reducing unnecessary operations.

$$P(S) \approx P(c_1)P(c_2 \mid c_1)P(c_3 \mid c_2) \cdots P(c_n \mid c_{n-1}), \tag{12}$$

$$P(S) \approx \frac{N(c_1c_2)}{N(c_2)} \times \frac{N(c_2c_3)}{N(c_2)} \times \frac{N(c_3c_4)}{N(c_3)} \cdots \frac{N(c_{n-1}c_n)}{N(c_{n-1})}. \tag{13}$$

Gram algorithm uses a sliding window to assist the measure of service similarity. When the Gram value is small, the service window is expanded to accelerate the measure of service similarity. When the Gram value in the sliding window is large, the window is narrowed to improve the accuracy of service similarity calculation and improve the accuracy of a service clustering algorithm. Gram dynamic sliding window size calculation is shown in formula (14).

$$W_{i+1} = \begin{cases} W_i + n, S_{i+1} < S_i, \\ W_i, S_{i+1} = S_i, \\ W_i - n, S_{i+1} > S_i, \end{cases} \tag{14}$$

where n is the dynamic change of the window. W_i represents the size of the window of i the service data, which is updated by the variance of the service data i. S_i represents the variance of the Gram value of the service data in the window.

The cosine similarity value is obtained from the word frequency vector. As shown in formula (15),

$$\cos(\theta) = \frac{\sum_i^n (a_i + b_i)}{\sqrt{\sum_i^n (a_i)^2 \times \sum_i^n (b_i)^2}}. \tag{15}$$

n is the number of service samples, and the vector a_i, b_i represented as two services.

4. Experiment

Firstly, the details of preprocessing and evaluation metrics were introduced. Secondly, compared with other algorithms in terms of entropy, accuracy, recall, and F value.

4.1. Preprocessing

4.1.1. Remove Stop Words. According to our observation of WSDL documents. We found some words do not have practical meaning as stop words ("be," "the," "a," and "an"). We will filter the stop word in order to filter the noise of the data.

4.1.2. Stemming. WSDL descriptions are written in English, and the same words will be different for different people and tenses. For example, "change" and "changed" have the same meaning, but the computer will consider them to have different meanings. Therefore, we process such words to improve the accuracy of NFC-NSPO through python NLTK (Natural Language Toolkit).

4.1.3. TF-IDF. The TF-IDF algorithm calculates feature words in a document while it offers the frequency. In this paper, we use TF-IDF to calculate the frequency of web service document words and then generate a word frequency matrix.

$$\begin{aligned} tf_{ij} &= \frac{n_{ij}}{\sum n_{ij}}, \\ idf_i &= \log\left(\frac{N}{n_i}\right). \end{aligned} \tag{16}$$

n_{ij} represents the number of j^{th} word in the i^{th} service, $\sum n_{ij}$ donates the total number of words, and idf_i measures the importance of word. N represents the number of web service descriptions, and n_i presents the number of n_{ij} in the web service description.

4.2. Evaluation Measures. This section introduces evaluation measures and comparison algorithms.

It is important to evaluate the performance of the algorithm. We choose widely employed metrics: entropy, recall, accuracy, and F value to assess the performance of NFC-NSPO in web services. The four metrics are shown as follows. We can compare the results with the class label. The accuracy rate is shown in formula (18). The recall rate is

shown in formula (17). The entropy is shown in formula (20). The value of F is as shown in formula (19).

$$\text{Recall} = \frac{\text{succ}(c_i)}{\text{succ}(c_i) + \text{missed}(c_i)}, \tag{17}$$

$$\text{Precision} = \frac{\text{succ}(c_i)}{\text{succ}(c_i) + \text{mispl}(c_i)}, \tag{18}$$

$$F = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{19}$$

$$r = -\sum_{k=1}^{c} P_{ij} \log_2 P_{ij}, \tag{20}$$

$$P_{ij} = \frac{\|C_k^t\|}{\|C_k\|}, \tag{21}$$

where c_i represents the cluster i, $\text{succ}(c_i)$ represents the number of web service put correct cluster c_i, $\text{mispl}(c_i)$ represents the number of web service false put in cluster c_i, and $\text{missed}(c_i)$ represents the number of web service should be in cluster c_i but put it in other clusters. P_{ij} represents the probability that the services belongs to the cluster.

There are several classic clustering methods, such as k-means [29] and PSO-$k$means algorithms [30]. It has been applied to web service recommendations and service combinations. These algorithms have achieved good results. k-means, k-modes, and k-prototype algorithms represent different feature algorithms. For PSO-$k$means, it is similar to our algorithm. We conduct experiments on the same dataset.

K-means: for the k-means algorithm, the first thing to pay attention to is the choice of the k value. Generally speaking, we will choose an appropriate value of k based on the prior experience of the data. k-means divides the sample set into k clusters according to Euclidean distance [29].

K-modes: K-modes is a method used by k-means on nonnumerical sets. Replace the Euclidean distance used by k-means with the Hamming distance between characters [31].

K-prototype: K-prototype is a collective form of k-means and k-modes, which is suitable for data of the numerical type and character type collection. But the web service has only a small amount of numerical data. To some extent, the k-prototype has weakened into k-mode and k-means [32].

PSO-$k$means: the literature [30] proposes a k-means algorithm based on particle swarm optimization (PSO). It makes the k-means algorithm unaffected by the initial cluster centers.

In web service clustering, for k-means and k-modes, they need to give the value of k in advance, and the clustering effect is affected by the cluster center. The particle swarm fuzzy clustering method we proposed uses an automatic clustering algorithm to determine the number of service clusters, and the fuzzy clustering algorithm combines a particle swarm optimization algorithm to determine the location of the cluster center. Although k-means and k-modes have improved, they still cannot get rid of the limitation of k. K-prototype is a combination of k-means and k-modes, and the cluster center is updated by combining K-means and K-modes. PSO-$k$means is similar to our proposed algorithm. It makes the k-means algorithm unaffected by the initial cluster centers. In order to increase the accuracy of clustering, our algorithm uses Gram-improved cosine similarity to measure the similarity between services. We conducted experiments on real web service datasets, and the experiments proved that our algorithm is better than these four algorithms.

4.3. Dataset. In this paper, the dataset of web service text data crawl from a programmable website by python. ProgrammableWeb is a public web service repository. The method of obtaining our dataset is the same as that of the literature [33, 34] using python crawler. The website is https://www.programmableweb.com/. These datasets are statistically calculated, including the number of each service statistics. The number of services is in descending order; it can be seen from Figure 2 that the number of Mapping services is up to nearly 1000. In the dataset, there are more than 200 services that are less than 300, such as search, social, eCommerce, photos, and music. The number of other services is less than 200. Search, social, eCommerce, photos, music, and other services are selected for clustering in this experiment.

The name and number of each service in the service dataset are shown in Table 3. The most significant number of services in the first column is 286 named search, and the smallest number of services is 229 called music. The most significant number of services in the second column is 95 named Government, and the smallest number of services is 52 named Movies. In the third column, the most significant number of services is 46 named Financial, and the smallest number of services is 32 called Books.

The web service sample data obtained includes service names, labels, descriptions, and categories (as shown in Table 3).

4.4. Analysis of Results. In this paper, MATLAB R2016a is used to generate the experimental results. In order to avoid the contingency of the experiment, each clustering algorithm run 10 times, and the average value of the running results is each algorithm's final clustering result. Accuracy, entropy, recall, and F value are used to evaluate each clustering algorithm. The MATLAB code and dataset are available at https://github.com/dqy1122/PSOcmeans.git.

As shown from Table 4, in terms of accuracy, the highest accuracy of NFC-NSPO algorithm is 0.896. The second is PSO-$k$means algorithm. Its accuracy is 0.845. The worst is k-prototype clustering algorithm. The accuracy is 0.743. In terms of recall rate, the highest recall rate of k-modes algorithm is 0.756. Next is the NFC-NSPO algorithm. Its value is 0.734. The worst is k-means algorithm; its value is 0.621. In terms of entropy, k-prototype clustering algorithm has the maximum entropy value of 0.781. The second is PSO-$k$means algorithm; its value is 0.772. The worst is NFC-NSPO algorithm; its value is 0.642. In terms of F value, the F value of NFC-NSPO algorithm is the highest, which is 0.806. The second is

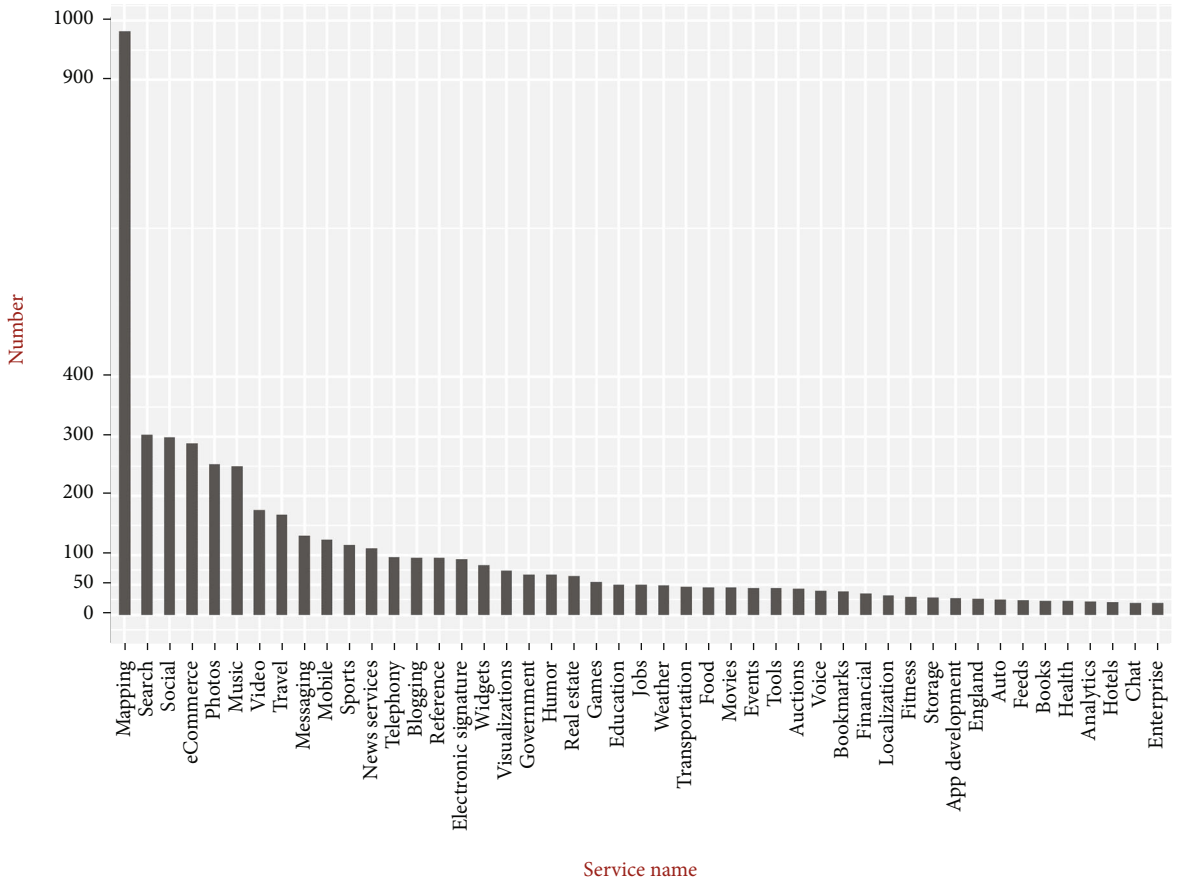

Figure 2: Statistics of service datasets.

Table 3: The number of services.

Name	Number	Name	Number	Name	Number
Search	286	Government	95	Bookmarks	32
Social	276	Humo	97	Financial	46
eCommerce	245	Real estate	93	Localization	46
Photos	236	Games	95	Fitness	45
Music	229	Movies	52	Books	32

Table 4: The accuracy of each algorithm.

Algorithm name	Accuracy	Recall	Entropy	F value
k-means	0.756	0.621	0.721	0.682
k-modes	0.825	0.756	0.762	0.789
k-prototype	0.743	0.738	0.781	0.733
PSO-$k$means	0.845	0.712	0.772	0.772
NFC-NSPO	0.896	0.734	0.642	0.806

the k-modes algorithm, whose value is 0.789. The worst is PSO-$k$means algorithm. Its value is 0.772.

The reason analysis is as follows: NFC-NSPO fuzzy clustering service partition method based on particle swarm optimization. Firstly, the improved cosine similarity calculation based on Gram is used to calculate the similarity between services. Gram algorithm uses a sliding window to assist the service similarity. When the Gram value in the window is small, the service window should be expanded to accelerate the detection of service similarity. When the Gram value in the sliding window is large, the window should be narrowed to improve the accuracy of service clustering. Secondly, using the advantages of the particle swarm optimization algorithm, the optimal global solution can be found through the movement of particles. It avoids the problem that fuzzy clustering algorithm randomly selects the clustering center and falls into the local optimum. Therefore, the clustering accuracy of NFC-NSPO is improved. In k-means, Euler distance is used to measure the similarity between different services. Since Euler distance is not suitable for calculating the similarity of text data, the calculation of service similarity is not accurate enough. At the same

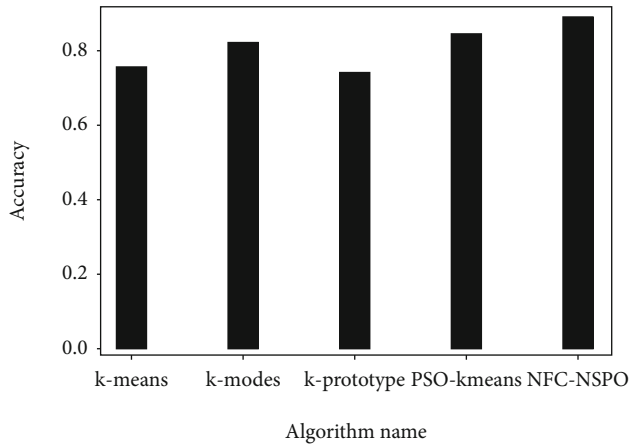

Figure 3: The accuracy of each algorithm.

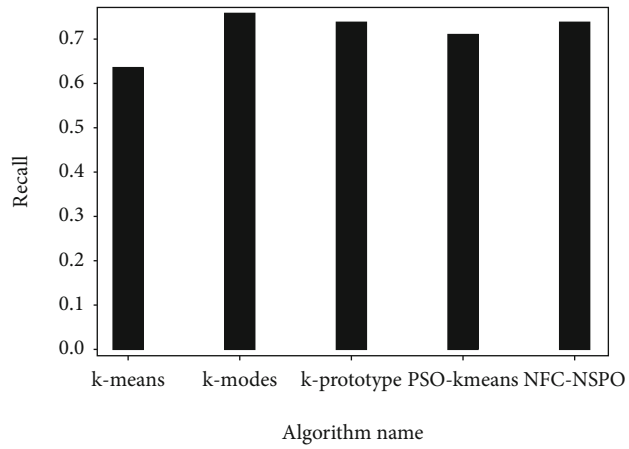

Figure 4: Recall rate of each algorithm.

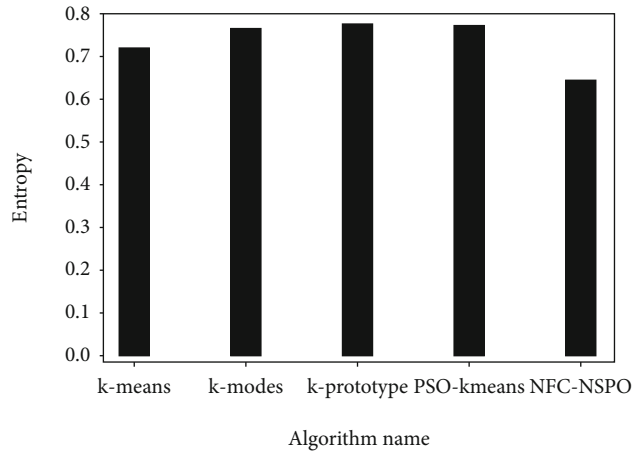

Figure 5: Entropy of each algorithm.

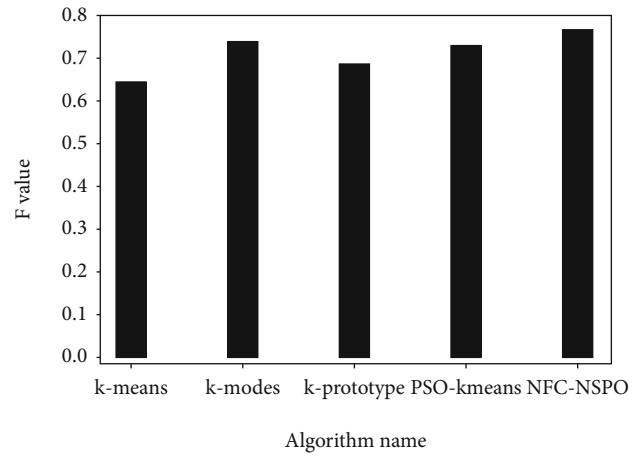

Figure 6: F value of each algorithm.

time, k-means belongs to the partition clustering algorithm. The location and number of clusters are randomly selected so it is easy to fall into the local optimum, thereby affecting the accuracy of the clustering algorithm.

As Figure 3 shown, the accuracy of NFC-NSPO peaked at 0.896. At the same time, the accuracy of the PSO-K means algorithm and k-prototype algorithm is smaller than NFC-NSPO, with 0.845 and 0.734, respectively. Some reasons cause NFC-NSPO higher accuracy. Firstly, NFC-NSPO applies improved cosine similarity based on Gram, which can better calculate the similarity between two services. Secondly, fuzzy clustering algorithm combines particle swarm algorithm, to avoid fuzzy clustering algorithm random selection of cluster location. PSO-means algorithm uses Euler distance to calculate the similarity between two services. It will lead to low service clustering accuracy, because Euler distance is not suitable for calculating service similarity.

In Figure 4, the recall rate of the k-modes algorithm reached the highest, followed by NFC-NSPO and PSO-k means. The similarity calculation of k-modes used Hamming distance to measure the similarity between services. By comparing whether each bit of the vector is the same or not. If the vectors were different, the Hamming distance increased by 1. Otherwise, the Hamming distance remained unchanged.

The entropy value represents the degree of the chaos of an object. If the entropy value is larger, it means that objects are chaotic. If the entropy value is smaller, the object is stable, and the chaos coefficient is low. Figure 5 shows the entropy value of k-prototype reached the largest, followed by PSO-$k$means, and the entropy value of NFC-NSPO algorithm fall the lowest. On the one hand, the k-prototype algorithm is an improved algorithm combining k-means and k-modes, which can deal with both numerical data and categorical data. Since web service description text has only a small amount of numerical data. To some extent, k-prototype is the similarity to k -modes. On the other hand, the k-prototype algorithm is easily affected by the position of the cluster center, which is easy to fall into local optimum. So the k-prototype algorithm is unstable.

The F value is a linear combination of accuracy and recall, which measures the performance of the algorithm in a more stable form. Figure 6 shows that F value of NFC-NSPO reached the highest, followed by k-modes, and F value of k-prototype falls the lowest.

Table 5: The accuracy of each algorithm.

Algorithm name	Accuracy	Recall	Entropy	F value
NFC-NSPO (Gram-cosine)	0.896	0.734	0.713	0.806
NFC-NSPO (Cosine)	0.842	0.710	0.747	0.770
NFC-NSPO (Euler)	0.638	0.653	0.773	0.637
NFC-NSPO (Hamming)	0.801	0.624	0.752	0.701
NFC-NSPO (Manhatten)	0.743	0.612	0.781	0.671

Figure 3 shows that the accuracy of NFC-NSPO is significantly higher than other algorithms. Figure 4 shows that the recall value of NFC-NSPO is lower than that of the k-modes algorithm. The recall rate of NFC-NSPO is still higher than other clustering algorithms, because NFC-NSPO clustering algorithm applies the improved cosine similarity based on Gram, which can better calculate the similarity between two samples. On the other hand, fuzzy clustering algorithm combined particle swarm algorithm, to avoid fuzzy clustering algorithm random selected of cluster location; the random selection of cluster location caused poor accuracy of a fuzzy clustering algorithm.

The similarity measure plays a very important role in the clustering algorithm. Even if the same clustering algorithm uses different similarity measures, the accuracy of the clustering algorithm is different. This paper improves the similarity measure, which combined the Gram algorithm and cosine measure. To verify the performance of the improvement similarity measure. We use the same algorithm and use different similarity functions to compare accuracy, recall, entropy, and F value.

In Table 5, the accuracy of NFC-NSPO (Gram-Cosine similarity) is the highest with 0.896. Followed by NFC-NSPO (Cosine similarity) with 0.842, the accuracy of NFC-NSPO (Euler) falls lowest. In terms of recall, NFC-NSPO (Gram-Cosine similarity) reached the highest with 0.734, followed by NFC-NSPO (Manhatten) with 0.612. About entropy, the entropy of NFC-NSPO (Euler) reaches the highest with 0.773, followed by NFC-NSPO (Gram-Cosine similarity) with 0.713. In terms of F value, the F value of NFC-NSPO (Gram-Cosine similarity) reaches the highest, which is 0.806. The F value of NFC-NSPO (Euler) falls the lowest, which is 0.637.

This paper gave a brief analysis of the point of higher performance of NFC-NSPO (Gram-Cosine similarity), which uses the improved cosine similarity based on Gram to better measure the similarity between two services. This method can adjust the window size between services and clusters. This method can improve the accuracy of service clustering algorithm.

In most existing algorithms for automatically determining the number of clusters, k is obtained by means of running many times, which leads to the fact that k is not an integer. For solving this problem, most scholars choose the rounded method to take the integer k value, because the optimal number of clusters k should be an integer. In this paper, the NFC-NSPO algorithm proposed can determine the number of clusters k on six datasets. The NFC-NSPO algorithm calculates the number of clusters k equal to the number of predetermined classes. On the contrast, the rounding method can only correctly provide the optimal k on five datasets. NFC-NSPO algorithm generates that the number k of clusters on eCommerce dataset is the same as the expected value.

5. Conclusion

Because of the interrelationship between services, it is challenging to assign services to a specific cluster accurately. This paper propose a service partition method based on particle swarm fuzzy clustering. It can avoid the random selection of clustering positions, which leads to poor accuracy of the fuzzy clustering algorithm. The fuzzy clustering algorithm applies based on Gram-improved cosine similarity measure the similarity service. The function based on Gram-improved cosine similarity controls the sliding window to compare the service description one by one. When the Gram value is small, the Gram window is expanded to accelerate the measure service similarity. When the Gram value is large, the window is narrowed to improve service similarity measures' accuracy and service clustering accuracy. Experimental results show that the NFC-NSPO algorithm can better evaluate the interconnection between services and improve the accuracy of service clustering compared with existing algorithms. That can reasonably consider the relationship between service and service. Combined with the particle swarm algorithm, the relationship between the two can find the optimal cluster center position.

Data Availability

The MATLAB code and dataset are available at https://github.com/dqy1122/PSOcmeans.git.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Science and Technology Project in Shaanxi Province of China (program no. 2019ZDLGY07-08), Natural Science Basic Research Program of Shaanxi Province, China (grant no. 2020JM-582), Science and Technology of Xi'an (grant no. 2019218114 GXRC017CG018-GXYD17.9), Scientific Research Program Funded by Shaanxi Provincial Education Department (no. 21JP115), Natural Science Basic Research Program of Shaanxi (program no. 2021JQ-719), and Special Funds for Construction of Key Disciplines in Universities in Shaanxi.

References

[1] J. Bo, H. U. Song, P. Wei-Feng, W. Ye, and S. Bei-Bei, "Service clustering based on the functional semantics of requirements," *Chinese Journal of Computers*, vol. 41, no. 6, pp. 1036–1040, 2015.

[2] Z. Hao-quan and Z. Qi, "Web service discovery clustering performance analysis based on clustering LDA method," *Computer Application*, vol. 39, no. 10, pp. 27–30, 2020.

[3] D. Yang and D. He, "Web service clustering method based on word vector and biterm topic model," in *2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, pp. 299–304, Chengdu, China, 2021.

[4] Ö. Çoban and G. T. Özyer, "Word2vec and clustering based twitter sentiment analysis," in *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, pp. 1–5, Adana, Turkey, September, 2018.

[5] Z. Xiangping, J. Liu, Q. Xiao, M. Shi, and B. Cao, "Web services clustering with multi-functionality based on lda and fuzzy c-means algorithm," *Journal of Central South University(Science and Technology)*, vol. 49, no. 12, pp. 2986–2992, 2018.

[6] Y. M. Wang, Y. J. Zhang, B. H. Xie, L. H. Pan, and L. C. Chen, "Semantic web service discovery based on fuzzy clustering optimization," *Computer Engineering*, vol. 39, no. 7, pp. 219–223, 2013.

[7] Q. Feng, L. Chen, C. L. P. Chen, and L. Guo, "Deep fuzzy clustering—a representation learning approach," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 7, pp. 1–1433, 2020.

[8] C. Lv, W. Jiang, S. Hu, J. Wang, G. Lu, and Z. Liu, "Efficient dynamic evolution of service composition," *IEEE Transactions on Services Computing*, vol. 11, no. 4, pp. 630–643, 2018.

[9] Y. Yin, J. Xia, Y. Li, xu, W. Xu, and L. Yu, "Group-wise itinerary planning in temporary mobile social network," *IEEE Access*, vol. 7, pp. 83682–83693, 2019.

[10] H. Gao, K. K. Dluzniak, H. Xia et al., "A service clustering method based on wisdom of crowds," in *2019 IEEE International Congress on Big Data (BigDataCongress)*, pp. 98–105, Milan, Italy, July, 2019.

[11] C. Cho, K. Lee, M. Lee, and C. Lee, "Software architecture module-view recovery using cluster ensembles," *IEEE Access*, vol. 7, pp. 72872–72884, 2019.

[12] B. Cao, X. F. Liu, M. M. Rahman, B. Li, J. Liu, and M. Tang, "Integrated content and network-based service clustering and web apis recommendation for mashup development," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 99–113, 2020.

[13] T. Liang, Y. Chen, W. Gao, M. Chen, M. Zheng, and J. Wu, "Exploiting user tagging for web service co-clustering," *IEEE Access*, vol. 7, pp. 168981–168993, 2019.

[14] Q. Xiao, B. Cao, X. Zhang, J. Liu, and L. I. Yanxinwen, "Web services clustering based on word2vec and lda topic model," *Journal of Central South University(Science and Technology)*, vol. 49, no. 12, pp. 2979–2985, 2018.

[15] A. Sheeba, S. Padmakala, and C. A. Subasini, "Ontology based semantic description and registration of mathematical web services," in *2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pp. 521–525, Coimbatore, India, February, 2019.

[16] T. P. Q. Nguyen and R. J. Kuo, "Automatic fuzzy clustering using non-dominated sorting particle swarm optimization algorithm for categorical data," *IEEE Access*, vol. 7, pp. 99721–99734, 2019.

[17] C.-I. Hsu and C. Chiu, "A hybrid latent dirichlet allocation approach for topic classification," in *2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 312–315, Gdynia, Poland, July, 2017.

[18] M. Shi, J. X. Liu, D. Zhou, B. Q. Cao, and Y. P. Wen, "Multi-relational topic model-based approach for web services clustering," *Chinese Journal of Computers*, vol. 42, no. 4, pp. 820–836, 2019.

[19] S. Chen, Y. Peng, H. Mi, C. Wang, and Z. Huang, "A cluster feature based approach for QoS prediction in web service recommendation," in *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pp. 246–251, Germany, March, 2018.

[20] H. Yang, H. Yan, and C. Dong, "A k-means clustering approach for PCA-based web service QoS prediction," in *2019 IEEE International Conferences on Ubiquitous Computing Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS)*, pp. 129–132, Shenyang, China, October 2019.

[21] P. C. Fourie and A. A. Groenwold, "The particle swarm optimization algorithm in size and shape optimization," *Structural and Multidisciplinary Optimization*, vol. 23, no. 4, pp. 259–267, 2002.

[22] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[23] M. Liang, Q. Li, Y. A. Geng, J. Wang, and Z. Wei, "Remold: an efficient model-based clustering algorithm for large datasets with spark," in *2017 IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 376–383, Shenzhen, China, December, 2017.

[24] F. Cao, J. Liang, and L. Bai, "A new initialization method for categorical data clustering," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10223–10228, 2009.

[25] J. Kennedy, R. C. Eberhart, and Y. Shi, "The particle swarm," *Swarm Intelligence*, pp. 287–325, 2001.

[26] I. Heloulou, M. S. Radjef, and M. T. Kechadi, "A multi-act sequential game-based multi-objective clustering approach for categorical data," *Neurocomputing*, vol. 267, pp. 320–332, 2017.

[27] K. T. Atanassov, "Intuitionistic fuzzy sets," *Fuzzy Sets & Systems*, vol. 20, no. 1, pp. 87–96, 1986.

[28] A. Ahmad, M. Rub Talha, M. Ruhul Amin, and F. Chowdhury, "Pipilika n-gram viewer: an efficient large scale n-gram model for bengali," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, pp. 1–5, Sylhet Bangladesh, September, 2018.

[29] I. Alhadid, S. Khwaldeh, M. al Rawajbeh, E. Abu-Taieh, R.'e. Masa'deh, and I. Aljarah, "An intelligent web service composition and resource-optimization method using K-means clustering and knapsack algorithms," *Mathematics*, vol. 9, no. 17, p. 2023, 2021.

[30] M. Handa, H. Xiaoyu, and M. Renqing, "Parallel PSO-kmeans algorithm implementing web log mining based on hadoop," *Computer Science*, vol. 42, no. S1, pp. 470–473, 2015.

[31] O. S. Soliman, D. A. Saleh, and S. Rashwan, "A hybrid fuzzy particle swarm and fuzzy k-modes clustering algorithm," in *8th International Conference on Informatics and Systems (INFOS)*, pp. 68–75, Giza, Egypt, July, 2012.

[32] X. Chen, "An improved clustering algorithm for mixed attributes data based on K-prototypes algorithm," in *2015 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, pp. 396–399, Krakow, Poland, March, 2015.

[33] Z. Neng, W. Jian, and M. Yutao, "An intelligent web service composition and resource-optimization method using K-means clustering and knapsack algorithms," *IEEE Transactions on Services Computing*, vol. 13, no. 3, pp. 488–502, 2020.

[34] N. Zhang, K. He, J. Wang, and Z. Li, "WSGM-SD: an approach to RESTful service discovery based on weighted service goal model," *Clarivate Analytics Web of Science*, vol. 25, no. 2, pp. 256–263, 2016.