ISSN 1330-3651(Print), ISSN 1848-6339 (Online)

https://doi.org/10.17559/TV-20200918143701
Original scientific papers

# Clustering Algorithm Based on Sparse Feature Vector without Specifying Parameter

Huixia HE, Guiying WEI, Sen WU*, Xiaonan GAO

**Abstract:** Parameter setting is an essential factor affecting algorithm performance in data mining techniques. CABOSFV is an efficient clustering algorithm which can cluster binary data with sparse features, but it is challenging to specify the threshold parameter. To solve the difficulty of parameter decision, a clustering algorithm based on sparse feature vector without specifying parameter (CASP) is proposed in this paper. The calculation method of an upper limit of threshold is firstly defined to determine the range of threshold. Furthermore, we use the sparseness index to sort the data and conduct the clustering process based on the adjusted sparse feature vector after data sorting. An interval search strategy is adopted to find a suitable threshold within the defined threshold range, and the clustering result with the selected suitable parameter is the outcome. Experiments on 7 UCI datasets demonstrate that the clustering results of the CASP algorithm are superior to other baselines in terms of both effectiveness and efficiency. CASP not only simplifies the parameter decision process, but also obtains desirable clustering results quickly and stably, which shows the practicability of the algorithm.

**Keywords:** CABOSFV; clustering; sparse feature; threshold parameter

## 1 INTRODUCTION

Clustering is an important method in identifying the natural structures of datasets [1]. As a fundamental technique in data mining [2, 3], clustering analysis aims at dividing data objects into several groups such that data objects in each group are similar to one another and dissimilar to data objects in different groups [4, 5]. Over the years, clustering algorithms are widely used in data analysis in different domains, such as text data [6, 7], customer data [8, 9], image data [10, 11] and medical data [12, 13].

Clustering algorithm based on sparse feature vector (CABOSFV) [14] is an efficient clustering algorithm for binary data with sparse features. The similarity of a set is measured by the defined Sparse Feature Dissimilarity (*SFD*).Moreover, Sparse Feature Vector (*SFV*) is exploited to compress sparse data effectively. Using the additivity of *SFV*, clustering can be completed only by scanning the data once. Hence, CABOSFV algorithm is of high computing efficiency and good clustering performance [15]. Subsequently, CABOSFV has attracted extensive attention, and it is exercised in many applications such as customer knowledge discovery [15], text mining [16, 17], traditional Chinese medicine [18] and Intelligent Miner (I-MINER) [19].

However, CABOSFV algorithm has two shortcomings: firstly, clustering result is sensitive to data input order; secondly, a threshold parameter needs to be given in advance, which directly affects the final clustering result. Some improved algorithms have been developed to solve these problems. Improved CABOSFV clustering considering data sort (CABOSFV_CS) [20] defines the sparseness index of the object, and it is verified through experiments that the accuracy of clustering results will be improved if the data objects are sorted in ascending order according to the sparseness index. CABOSFV_CS gives a solution to weaken the sensitivity of data order, but the problem of threshold parameter determination has not been substantially broken.

The threshold $b$ of CABOSFV is a crucial parameter, similar to the number of clusters -$k$- in the clustering problem [21, 22], which is the upper limit of the *SFD* within a cluster. If the threshold is too large, it is easy to merge different clusters; if the threshold is too small, it is easy to split the same cluster. As a result, the selection of threshold $b$ plays a decisive role in the clustering process. However, this parameter is usually determined empirically, and there are no criteria for determining it. Thus, how to determine the threshold more reasonably becomes an essential and challenging task of the CABOSFV-based algorithms.

Hierarchical clustering algorithm for binary data based on cosine similarity (HABOC) [23] is an improved algorithm of CABOSFV which exploits hierarchical clustering procedure and does not need to specify the threshold parameter in advance. Although HABOC can get clustering results without pre-setting threshold parameter, the time complexity of hierarchical clustering program is high, which changes the efficiency advantage of CABOSFV that can complete clustering in one scan. High dimensional data clustering algorithm based on extended dissimilarity (CABOSFV_D) [24] proposes a calculation method of extended dissimilarity, which makes the clustering process more accurate. Also, the literature [24] presents a method to determine the threshold $b$, which subjectively set the initial threshold range to (0, 3). In fact, the threshold parameter may be taken on $(0, +\infty)$ and varied from data to data. Thus, the determination of the threshold parameter remains to be further studied.

Therefore, a clustering algorithm based on sparse feature vector without specifying parameter (CASP) is proposed in this paper. Firstly, the calculation method of an upper limit of the threshold is defined to determine the threshold range. Next, the sparseness index and adjustment index are used to improve the stability and reliability of clustering. Then, a certain search strategy is adopted to find a suitable parameter within the initial threshold range, and the final partition result is obtained. Finally, the experimental results demonstrate the superiority of the proposed method.

The key contributions of this paper are as follows: (1) We give a method to calculate an upper limit of threshold, which theoretically reduces the threshold range to a definite interval. The proposed CASP method includes parameter decision, which improves the practicability of

the algorithm. (2) The CASP algorithm combines the sparseness index and adjusted sparse feature vector to make the clustering results stable and reliable. (3) We evaluate the performance of CASP with several UCI datasets, and the experiments verify that CASP outperforms existing improved CABOSFV algorithms and classical categorical data clustering algorithm K-modes [25]. Moreover, CASP shows high computing efficiency. Therefore, the proposed CASP is promising for its practical application value.

The remaining chapters of this paper are organized as follows. Some preliminaries are firstly presented in Section 2. Section 3 defines an Upper Bound of the SFD Threshold and describes the details of the proposed method. Then, extensive experiments are presented in Section 4. Seven UCI datasets and three external clustering validation indices are used to verify the performance of algorithms. As a final part, conclusions are summarized in Section 5.

## 2 RELATED WORK AND PRELIMINARIES

In this section, some techniques for parameter determination in clustering algorithms are reviewed firstly. Then, we briefly review some preliminaries including CABOSFV, CABOSFV_CS and CABOSFV_D.

### 2.1 Techniques for Parameter Determination in Clustering Algorithms

As a fundamental technique in data mining, clustering can help humans understand and utilize data. In clustering analysis, parameter selection is one of the key factors to determine whether the clustering algorithm is effective. The number of clusters, usually notated as $k$, is a vital parameter in most clustering algorithms [26]. Thus, most researches on clustering algorithm parameters focus on how to determine the number of clusters $k$. A typical method is to evaluate a clustering validity index and optimize it as a function of the number of clusters [27]. Some studies use likelihood-based methods, such as Bayesian Information Criterion (BIC) and Akaike's information criterion (AIC), to estimate the correct $k$ value in the context of likelihood function [28, 29]. Recently, machine learning techniques are employed to estimate the number of clusters. Ünlü et al. proposed a weighted consensus clustering scheme which uses four different indices to estimate the correct number of clusters [30]. Pimentel et al. proposed a new methodology using meta learning to recommend the number of clusters [31]. Another direction is to use methods that do not require the a priori definition of the clusters number. Instead of defining the number of clusters, the CABOSFV algorithm needs to set a threshold parameter. Most of the existing researches focus on the estimation of parameter $k$. However, these existing research results cannot be applied to CABOSFV. This is because the candidate set of parameter $k$ is usually a finite set, while the threshold is an infinite set. Therefore, the CASP algorithm is proposed, which includes the determination of threshold parameter. Users can get desirable clustering results without specifying parameter.

### 2.2 CABOSFV Algorithm

CABOSFV is a sparse feature-based clustering algorithm which can cluster sparse data described by binary variables [14]. The binary variable is a kind of categorical variable with only two values (usually expressed as 0 and 1). In real-life data sets, the categorical variable is usually with two or more values. The multivalued variable can be converted into binary variables by one-hot encoding, as shown in Tab. 1. Therefore, the CABOSFV algorithm can also be used to cluster categorical data. All of the following descriptions are based on the assumption that categorical data is converted to binary data.

In CABOSFV, Sparse Feature Dissimilarity is defined to measure the similarity of data objects in the cluster. The algorithm also applies Sparse Feature Vector to compress the data effectively. To make it more concrete, given a dataset with $n$ objects, there are $m$ attributes describing each object, with the value of 1 or 0 (known as the sparse feature). $X$ is a subset of the dataset. The number of objects in $X$ is marked as $|X|$. In subset $X$, the number of attributes with sparse feature values of 1 for all objects is $a$, and the corresponding attribute number set $\{j_{s_1}, j_{s_2}, ..., j_{s_a}\}$ is represented by $S$; the number of attributes with sparse feature values that are not all the same is $e$, and the corresponding attribute number set $\{j_{ns_1}, j_{ns_2}, ..., j_{ns_e}\}$ is represented by $NS$. The Sparse Feature Dissimilarity ($SFD$) of set $X$ is defined as:

$$SFD(X) = e / (|X| * a) \tag{1}$$

The Sparse Feature Vectors ($SFV$) is defined as:

$$SFV(X) = (|X|, S(X), NS(X), SFD(X)) \tag{2}$$

Moreover, by using the additivity of $SFV$, the $SFV$ of the merged new set is calculated directly. It is worth mentioning that the CABOSFV algorithm does not need to calculate and compare the differences between every two data objects one by one. It only needs one data scan to get the clustering results, so CABOSFV is particularly efficient. However, the clustering results are affected by the data input order and threshold parameter. CABOSFV_CS discusses the sensitivity of data input order. CABOSFV_D mentions the selection of threshold parameter. These two algorithms will be introduced in the following subsection.

### 2.3 A Review of CABOSFV_CS and CABOSFV_D

To solve the problem that the clustering quality of CABOSFV is affected by the order of data input, CABSOFV_CS proposes the concept of sparseness index to describe the sparse feature of data [20]. The real data experiments show that the clustering performance can be improved effectively by sorting data in ascending order of sparseness index. CABOSFV_CS provides a practical and straightforward solution to the data order sensitivity problem. Therefore, this sorting method is employed in this paper.

CABOSFV_D is a high dimensional data clustering algorithm based on extended dissimilarity [24]. CABOSDV_D introduces the adjustment index $p$ to expand the original Sparse Feature Dissimilarity. The extended dissimilarity can prevent data objects from being assigned to a larger cluster, which makes the clustering process more accurate. In addition, CABOSFV_D is implemented by bit set to improve the efficiency of the algorithm. At the end of the literature list, the authors present a method for determining the threshold $b$. The method first sets the initial threshold range to an interval, such as (0, 3). Then, take a step length as increment, such as 0.1, conduct multiple experiments, and select the parameter with the best clustering result as the final input parameter of the algorithm. The problem with this approach is that the initial threshold range is set empirically. Furthermore, the fixed threshold is not suitable for all datasets, so it is unreasonable to set the threshold range to the same interval without considering the actual structure of the dataset. In this paper, we give a method to determine the threshold range according to the specific dataset and expect to simplify the user's attempt to determine the threshold parameter.

## 3 CASP ALGORITHM

In this section, the proposed CASP algorithm will be introduced in detail. Firstly, we define an Upper Bound of the SFD Threshold (*TUB*) to determine the threshold range. Then, we introduce the adjusted sparse feature vector after data sorting. Finally, the specific steps of CASP algorithm are described.

### 3.1 Determination of SFD Threshold Range

The Sparse Feature Dissimilarity (*SFD*) describes the similar degree of all objects in a set. The threshold $b$ is a parameter of the algorithm, which represents the upper limit of *SFD* in a set. *SFD* and $b$ jointly determine whether the current object can be added to a cluster. For different datasets, the calculated *SFD* is quite different. The fixed threshold is not adaptive on different datasets. As a result, there is no unified empirical standard for the selection of threshold $b$. It is necessary to find the appropriate threshold range from the given dataset.

As the threshold $b$ changes, the clustering results will be various. When the value of $b$ is very large, the clustering result will not change with $b$ any more. In this case, $b$ has no limiting effect on the *SFD* of a set. That is, we can find such a relatively large value of $b$ as an upper bound of the threshold. Assuming that the maximum *SFD* of any subset of a dataset is $SFD_{\max}$, we expect to find a value equal to or slightly greater than $SFD_{\max}$. According to the definition of $SFD: SFD(X) = e/(|X| * a)$, the larger the $e$ in the numerator is, and the smaller the $|X|$ and $a$ in the denominator are, the larger *SFD* is. From this perspective, we define the calculation method of an Upper Bound of the SFD Threshold range as follows.

Definition 1(An Upper Bound of the *SFD* Threshold, *TUB*) Given a dataset $X$ with $n$ objects, each of them is described by $m$ binary attributes (with values of 0 or 1). The number of attributes that equal 1 for all objects in $X$ is represented as $a$. The number of attributes that equal 0 for all objects in $X$ is indicated by $z$. An Upper Bound of the SFD Threshold, denoted as *TUB*, is defined as:

$$TUB = \begin{cases} \dfrac{m-z-a}{2 \cdot a}, a > 0 \\ \dfrac{m-z-1}{2 \cdot 1}, a = 0 \end{cases} \tag{3}$$

In Eq. (3), the "2" in the denominator means that when two sets are merged, there are at least two objects in the new set. The "$a$" refers to the number of attributes with all values of 1 in the dataset $X$.

If $a > 0$, when two sets are merged, the number of attributes with all values of 1 in the new set is at least $a$. The "*m-z-a*" means the number of attributes that equal 1 for some objects and equal 0 for other objects in the dataset $X$. Then the number of attributes with values that are not all the same in a subset of $X$ does not exceed *m-z-a*.

If $a = 0$, when two sets can be merged into one set, there is at least one attribute with all values of 1 in the new set; otherwise, the two sets are considered entirely different and cannot be merged into a new set. The number of attributes with values that are not all the same in the merged new set does not exceed *m-z*-1.

*TUB* represents the maximum set dissimilarity that a subset of a dataset may achieve. Obviously, the minimum dissimilarity of a set is 0. Therefore, the range of the SFD threshold can be obtained as (0, *TUB*). The following case shows how to calculate the *TUB* in detail.

Suppose that $X = \{x_1, x_2, ..., x_5\}$ is a dataset described by four attributes: $\{A_1, A_2, A_3, A_4\}$. After transforming categorical data to binary data, there are eleven binary attributes. The details of each data object are shown in Tab. 1. According to Eq. (3), $TUB = (11-0-1)/(2 \cdot 1) = 5$. Thus, the threshold range of dataset $X$ is (0, 5).

**Table 1** Converting categorical attributes to binary attributes

| | $A_1$ | $A_2$ | $A_3$ | $A_4$ | → | $A_{1\_1}$ | $A_{1\_2}$ | $A_{1\_3}$ | $A_{2\_1}$ | $A_{2\_2}$ | $A_{2\_3}$ | $A_{2\_4}$ | $A_{3\_1}$ | $A_{4\_1}$ | $A_{4\_2}$ | $A_{4\_3}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 1 | 1 | 1 | 2 | → | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| $x_2$ | 1 | 2 | 1 | 3 | → | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| $x_3$ | 2 | 3 | 1 | 1 | → | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| $x_4$ | 3 | 4 | 1 | 3 | → | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| $x_5$ | 1 | 2 | 1 | 1 | → | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

### 3.2 Adjusted Sparse Feature Vector after Data Sorting

Due to the sensitivity of CABOSFV to the order of data input, the proposed CASP algorithm firstly sorts the data objects according to the sparseness index [20], which is described as follows.

Definition 2 (The Sparseness Index of an Object, *SIO*) Suppose a dataset $X$ has $n$ objects, each of which is

characterized by binary attributes. For object $i$, its sparseness index is denoted as:

$$SIO_i = m_b \tag{4}$$

where $m_b$ represents the number of attributes whose value is equal to 1 in object $i$.

For dataset $X$, the sparseness index of each object is calculated and sorted in ascending order. The sorted dataset is $X_{sort}$. The adjusted sparse feature vector after data sorting will be introduced next.

Definition 3 (The Adjusted Sparse Feature Vector, *ASFV*) For the sorted dataset $X_{sort}$, $X'$ is one of its subsets, and the number of objects in $X'$ is recorded as $|X'|$. The number of attributes that equal 1 for all objects in $X'$ is represented as $A$ and the corresponding attribute set is $S$. The number of attributes that equal 1 for some objects and equal 0 for other objects in $X'$ is denoted as $E$ and the corresponding attribute set is $NS.p$, namely adjustment index is a constant integer greater than or equal to 1.

The adjusted sparse feature vector, namely *ASFV*, is defined as:

$$ASFV(X') = \left(|X'|, S(X'), NS(X'), ESFD(X')\right) \tag{5}$$

Where $ESFD(X')$ represents the extended sparse feature dissimilarity of $X'$, which is defined as:

$$ESFD(X') = \frac{E}{\sqrt[p]{|X'|} \cdot A} \tag{6}$$

According to [24], the adjustment index $p$ in Eq. (6) is usually between (1, 4). When two sets, $X'$ and $Y'$, are merged, the *ASFV* of the new set can be calculated directly as follows:

$$\begin{aligned} &ASFV(X' \cup Y') = ASFV(X') + ASFV(Y') = \\ &= (N, S(X' \cup Y'), NS(X' \cup Y'), ESFD(X' \cup Y')) \end{aligned} \tag{7}$$

in which:

$$N = |X'| + |Y'|;$$

$$S(X' \cup Y') = S(X') \cap S(Y');$$

$$\begin{aligned} NS(X' \cup Y') = &(NS(X') \cup NS(Y') \cup S(X') \cup S(Y')) / \\ &/((S(X') \cap S(Y'));\end{aligned}$$

$$ESFD(X' \cup Y') = |NS(X' \cup Y')| / \left(\sqrt[p]{N} \cdot |S(X' \cup Y')|\right)$$

The CASP algorithm exploits the sparseness index to weaken the sensitivity of data order. At the same time, the clustering based on the adjusted sparse feature vector can make the clustering process more accurate and improve the clustering effectiveness. CASP considers both data order sensitivity and rationality of data allocation. Therefore, it will be more stable and reliable than traditional CABOSFV algorithms.

### 3.3 Clustering Process without Specifying Parameter

When the threshold $b$ takes different values within the given range, we can get different partitions. We find the best result from these partitions and the corresponding threshold $b$ is the selected parameter. The proposed CASP algorithm mainly contains the following procedures: firstly, calculate the *SFD* threshold range according to Definition 1; then, sort the data according to the sparseness index; next, conduct the clustering process based on adjusted sparse feature vector and search for a suitable parameter in the defined threshold range; finally, output the final clustering result. The detailed steps of CASP algorithm can be outlined as follows:

**Algorithm.** The CASP Algorithm
**Input**: dataset $X$
**Output**: threshold $B_{p0}$ and clustering result $\pi_{p0}$

**Step1:** For dataset $X$, an upper bound of the *SFD* threshold is calculated as *TUB* according to Eq. (3) in Definition 1.
**Step2:** Calculate the sparseness index for each object according to Eq. (4). Sort the objects in ascending order by sparseness index to get the dataset $X_{sort}$. Set adjusting exponent $p = 1$.
**Step3:** Divide the interval (0, *TUB*) into ten equal parts, and get the corresponding points: $b = \{b_1, b_2, \ldots, b_{11}\}$. For the parameters ($p$, $b_i$) and dataset $X_{sort}$, conduct the clustering process based on *ASFV* and return the partition $\pi_i$. The evaluation of cluster validity is denoted as $CVI_i$, $i \in \{1, 2, ..., 11\}$.
**Step4:** Calculate $I_i = (CVI_i + CVI_{i+1}) / 2, i \in \{1, 2, ..., 10\}$. Sort $I$ in descending order and record it as $I'$. The corresponding threshold intervals of $I_1'$ and $I_2'$ are respectively ($b_{\min1}$, $b_{\max1}$) and ($b_{\min2}$, $b_{\max2}$). If the interval length, $b_{\max1} - b_{\min1}$, approaches 0, go to step 6; otherwise, go to step 5.
**Step5:** Divide the interval ($b_{\min1}$, $b_{\max1}$) and ($b_{\min2}$, $b_{\max2}$) into five equal parts, respectively. And perform operations similar to step 3 and 4.
**Step6:** The current optimal threshold $b$ is $b_{\min1}$, denoted as $B_p$. Clustering result $\pi_p$ and evaluation index $CVI_p^1$ are recorded when the threshold parameter is $B_p$.
**Step7:** Determine whether $p$ is equal to 4. If so, go to step 8; Otherwise, $p = p + 1$, go to step 3.
**Step8:** Search $p_0$ in order that $CVI_{p0}^1 = \max_{p \in \{1, 2, 3, 4\}} CVI_p^1$. The final clustering result and threshold are $\pi_{p0}$ and $B_{p0}$, respectively.

The computational complexity of CASP is $O(I \times k \times n)$, where $I$ is the number of iterations, $k$ is the number of clusters, and $n$ is the number of data objects. $I$ is generally small. Therefore, as long as $k$ is significantly less than $n$, the computational time of CASP is linearly related to $n$, which is effective and simple.

The proposed CASP algorithm combines the strengths of CABOSFV_CS and CABOSFV_D. Moreover, when using CASP for clustering, the clustering results can be obtained by inputting only the datasets without setting parameter in advance. During the parameter determination, our method narrows the threshold search scope from (0, +∞) to (0, *TUB*). As a result, the appropriate parameter can

be located quickly and accurately, and then the ideal clustering result can be obtained.

## 4 EXPERIMENTS

In order to verify the validity of our proposed CASP algorithm, extensive experiments are carried out based on several UCI datasets. In section 4.1, seven UCI datasets and three evaluation metrics are introduced. Section 4.2 describes benchmarks and experimental design. Section 4.3 presents the experimental results and evaluates the performance of CASP algorithm.

### 4.1 Datasets and Evaluation Metrics

In the experiment, seven datasets viz., Zoo, Soybean (Small), Congressional Voting Records, Solar Flare, Audiology (Standardized), Lymphography, and Breast Cancer are selected from UCI Machine Learning Repository [32] for algorithm verification. These datasets are all categorical data with binary or categorical attributes. We remove data objects with missing values. The detailed information of datasets is described in Tab. 2.

The true class labels of these seven UCI datasets are known, so several external clustering validation indices are used to evaluate the clustering performance. The Rand index (RI), Fowlkes-Mallows scores (FMI) and Normalized Mutual Information (NMI) are employed in the experiment, as listed in Tab. 3. These indices are commonly used to compare the matching degree of clustering partitions and external standards.

More concretely, RI indicates the proportion that two objects originally in the same cluster are allocated to the same cluster and originally in the different clusters are correctly separated now; FMI is the geometric average of accuracy and recall; NMI reflects the consistency of the true label distribution and the clustering result label distribution. RI, FMI and NMI are all between 0 and 1. The greater the value of RI/FMI/NMI is, the more consistent the clustering result is with the real situation.

**Table 2** Description of seven UCI datasets

| Dataset Name | Abbreviation | #Instances | #Binary Attributes | #Categorical Attributes | #Classes |
|---|---|---|---|---|---|
| Zoo | ZO | 101 | 15 | 2 | 7 |
| Soybean (Small) | SO | 47 | 16 | 19 | 4 |
| Congressional Voting Records | VO | 435 | 16 | 0 | 2 |
| Solar Flare | SF | 1389 | 5 | 7 | 3 |
| Audiology (Standardized) | AU | 226 | 53 | 16 | 24 |
| Lymphography | LY | 148 | 9 | 9 | 4 |
| Breast Cancer | BC | 286 | 3 | 6 | 2 |

**Table 3** Description of three external validation metrics

| Measure | Description | Formula | Explanation |
|---|---|---|---|
| RI | Rand index | $RI = \frac{TP + TN}{TP + TN + FP + FN}$ | If two data objects with the same true labels are assigned to the same cluster, the number of such object pairs is denoted as *TP*. If two data objects with the different true labels are assigned to the different clusters, the number of such object pairs is denoted as *TN*. If two data objects with the different true labels are assigned to the same cluster, the number of such object pairs is denoted as *FP*. If two data objects with the same true labels are assigned to the different clusters, the number of such object pairs is denoted as *FN*. |
| FMI | Fowlkes-Mallows scores | $FMI = \frac{TP}{\sqrt{(TP + FP) \cdot (TP + FN)}}$ | |
| NMI | Normalized Mutual Information | $NMI = \frac{MI(labels_{true}, labels_{clustering})}{\sqrt{H(labels_{true}) H(labels_{clustering})}}$ | *MI* is the mutual information between the true labels and the result labels, and *H* is the information entropy. |

**Table 4** Description of algorithms involved in the experiment

| Name | Description | Source | Role |
|---|---|---|---|
| CASP | A clustering algorithm based on sparse feature vector without specifying parameter | Section 3 | Proposed method |
| CABOSFV_CS | An improved CABOSFV algorithm considering data sort | Wu et al., 2011 | Baseline |
| CABOSFV_D | An improved CABOSFV algorithm based on extended dissimilarity | Wu et al., 2020 | Baseline |
| HABOC | A hierarchical clustering algorithm for binary data | Gao & Wu, 2018 | Baseline |
| K-modes | A clustering algorithm to extend the k-means paradigm to categorical domains | Huang, 1997 | Baseline |

### 4.2 Benchmarks and Experimental Design

Some binary or categorical data clustering algorithms, including CABOSFV_CS, CABOSFV_D, HABOC and K-modes, are selected to compare with the proposed CASP algorithm. All algorithms are described in Tab. 4. CABOSFV_CS proposes a data sorting method to improve CABOSFV. CABOSFV_D is a high dimensional data clustering algorithm based on extended dissimilarity. HABOC is an improved algorithm of CABOSFV, and it is a hierarchical clustering program that does not require to pre-set threshold parameter. K-modes is a representative partition-based clustering algorithm for categorical data. It should be noted that CABOSFV is not included in the baselines. This is because CABOSFV_CS, CABOSFV_D and HABOC are improved algorithms of CABOSFV and it has been proved in [20, 23, 24] that the clustering effectiveness of these improved algorithms is better than that of CABOSFV.

These algorithms need to pre-set parameters except the CASP algorithm. Since the proposed parameter determination method is suitable for CABOSFV-based algorithms which need to determine the threshold, both CABOSFV_CS and CABOSFV_D use this method to determine the threshold and get the final clustering result. The number of clusters is set to $n$ = {2, 3, …, 25} for HABOC and K-modes. The best clustering results of HABOC and K-modes are selected as the final results for

algorithm comparison. In particular, CABOSFV_D is sensitive to data order, so repeat the algorithm ten times with randomly sorted datasets and take the average of clustering results as the final result.

### 4.3 Results and Discussions

The experiments are carried out on a personal computer with Windows 10 operating system, Intel Core i5 8250u CPU and 8 GB memory. All algorithms are implemented by MATLAB.

The clustering results of CASP algorithm and other baseline algorithms on seven datasets with three metrics are reported in Tab. 5 to Tab. 7, and each table corresponds to one evaluation metric. The best results for each dataset are indicated in bold. The last row of each table represents the average performance of each algorithm on seven datasets.

**Table 5** Evaluation on seven datasets of five algorithms with RI as metric

| RI | CASP | CABOS FV_CS | CABOS FV_D | HABOC | K-modes |
|---|---|---|---|---|---|
| ZO | **0,9818** | **0,9818** | 0,9542 | 0,9667 | 0,9488 |
| SO | 1 | 0,9325 | 0,9801 | **1** | 0,9630 |
| VO | 0,7670 | 0,7670 | 0,7438 | **0,8001** | 0,7700 |
| SF | **0,9425** | 0,9371 | 0,9389 | 0,5018 | 0,5761 |
| AU | **0,8694** | 0,8682 | 0,8678 | 0,8424 | 0,8073 |
| LY | **0,6226** | 0,5587 | 0,5883 | 0,6183 | 0,5917 |
| BC | **0,5861** | 0,4872 | 0,5849 | 0,4997 | 0,5244 |
| Average | **0,8242** | 0,7904 | 0,8083 | 0,7470 | 0,7402 |

**Table 6** Evaluation on seven datasets of five algorithms with FMI as metric.

| FMI | CASP | CABOS FV_CS | CABOS FV_D | HABOC | K-modes |
|---|---|---|---|---|---|
| ZO | **0,9404** | 0,9095 | 0,8667 | 0,9312 | 0,8805 |
| SO | **1** | 0,7839 | 0,9607 | **1** | 0,9055 |
| VO | 0,7660 | 0,7660 | 0,6959 | **0,8005** | 0,7718 |
| SF | **0,9689** | 0,9675 | 0,9663 | 0,7023 | 0,7349 |
| AU | 0,3694 | 0,3688 | 0,3756 | **0,4029** | 0,4113 |
| LY | **0,6692** | **0,6692** | 0,6629 | 0,5330 | 0,5241 |
| BC | **0,6241** | 0,5160 | 0,6240 | 0,5478 | 0,5611 |
| Average | **0,7626** | 0,7116 | 0,7360 | 0,7025 | 0,6842 |

**Table 7** Evaluation on seven datasets of five algorithms with NMI as metric.

| NMI | CASP | CABOS FV_CS | CABOS FV_D | HABOC | K-modes |
|---|---|---|---|---|---|
| ZO | **0,9085** | **0,9085** | 0,8607 | 0,8631 | 0,9056 |
| SO | **1** | 0,8489 | 0,9783 | **1** | 0,9441 |
| VO | 0,4678 | 0,4678 | 0,4398 | **0,5301** | 0,4567 |
| SF | **0,1550** | 0,1446 | 0,1223 | 0,0329 | 0,0557 |
| AU | **0,6959** | 0,6865 | 0,6888 | 0,5098 | 0,5521 |
| LY | **0,4175** | **0,4175** | 0,4164 | 0,3104 | 0,3095 |
| BC | **0,3135** | **0,3135** | 0,3134 | 0,0850 | 0,0893 |
| Average | **0,5655** | 0,5410 | 0,5457 | 0,4759 | 0,4733 |

As seen in Tab. 5 to Tab. 7, CASP shows the best clustering performance on the most datasets among all the comparison methods. More specifically, CASP gets the best results with all three metrics on five of seven datasets, including ZO, SO, SF, LY and BC. HABOC achieves the best results with three metrics on two of the seven datasets viz. SO and VO. With respect to dataset VO, clustering results of CASP are ranking third in terms of RI/FMI and ranking second in terms of NMI. For dataset AU, though CASP does not perform as well as K-modes and HABOC on FMI metric, it performs best on the other two metrics. From the last row of each table, it is clear that CASP gets the best average with all three metrics compared with the baseline algorithms.

Moreover, we plot the average performance of each algorithm in Fig. 1 to compare these algorithms. In Fig. 1, three subgraphs a, b, and c respectively represent the average results of all algorithms on the three metrics RI, FMI and NMI. Fig.1 shows that the performance ranking of each algorithm is consistent on three metrics. No matter which metric is used, CASP outperforms other algorithms, and CABOSFV_D is second only to CASP. CABOSFV_CS and HABOC are ranking third and fourth, respectively. K-modes and HABOC get the approximate clustering results.

It can be inferred from the above analysis that the threshold determination method proposed in this paper is effective for CABOSFV-based algorithms which need to determine the threshold. Furthermore, the comprehensive clustering effectiveness of CASP is better than those baselines, including existing improved CABOSFV algorithms and the classical categorical data clustering algorithm K-modes, which proves that our proposed approach is of superiority.

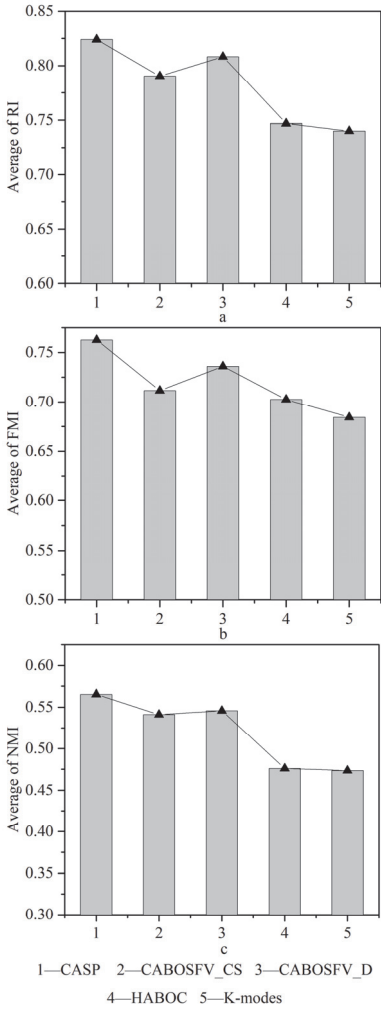


**Figure 1** Average performance of five algorithms over seven datasets with RI/FMI/NMI

In addition, the performance of CASP is compared with other algorithms in terms of execution time. The running time of five algorithms to perform clustering on each dataset is recorded in Tab. 8, and we can see that CASP has the lowest running time on most datasets. In the last row of the table, the average running time of each

algorithm over seven datasets is presented, and we plot the average running time in Fig. 2 for comparison. Tab. 8 shows that CASP has the least average running time. As seen in Fig. 2, the average running time of CASP, CABOSFV_CS, CABOSFV_D and K-modes is approximate and much smaller than HABOC. HABOC adopts the hierarchical clustering framework with time complexity of $O(n^3)$. Therefore, HABOC is expensive in terms of computation. Compared with HABOC, the proposed method not only solves the problem of parameter selection, but also maintains the efficiency of the algorithm.

**Table 8** Running time of five algorithms on seven datasets (Unit: second)

| Dataset | CASP | CABOS FV_CS | CABOS FV_D | HABOC | K-modes |
|---|---|---|---|---|---|
| ZO | 0,0156 | 0,1406 | **0,0094** | 0,7656 | 0,0269 |
| SO | **0,0035** | 0,0313 | 0,0094 | 0,1875 | 0,0100 |
| VO | **0,0156** | 0,0625 | 0,0141 | 3,9063 | 0,0188 |
| SF | 0,0781 | 1,3438 | 0,1156 | 393,8438 | **0,0691** |
| AU | **0,0781** | 4,4375 | 0,2469 | 2,7344 | 0,1172 |
| LY | **0,0156** | 1,5156 | 0,0313 | 1,5469 | 0,0184 |
| BC | **0,0156** | 0,3281 | 0,0250 | 5,3125 | 0,0216 |
| Average | **0,0344** | 1,1228 | 0,0645 | 58,3281 | 0,0403 |

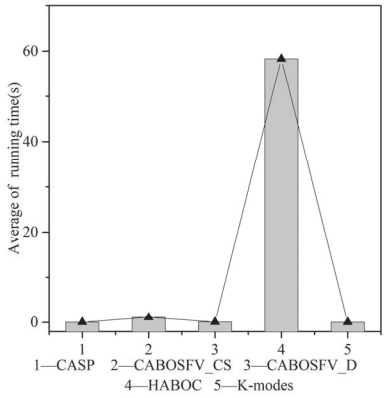


**Figure 2** Average running time of five algorithms over seven datasets

In summary, considering the clustering effectiveness and computation complexity, CASP obtains a better clustering performance than these baseline algorithms. With the development of information technology, the volume of data generated in real-world applications is increasing. CASP has great advantages in processing these large-scale data due to its low computation complexity. Moreover, CASP is able to automatically determine parameters, which provides great convenience for users.

## 5 CONCLUSION

Determining the threshold parameter is an essential but difficult step in CABOSFV-based clustering algorithms, which directly affects the stability of clustering. A clustering algorithm based on sparse feature vector without specifying parameter is proposed in this study to simplify the user's parameter attempt process. By defining an upper bound of the *SFD* threshold, the threshold range can be determined theoretically rather than empirically. In addition, CASP defines the adjusted sparse feature vector (*ASFV*), which combines the sparseness index and adjustment index to improve the stability and accuracy of the clustering. When using the proposed CASP algorithm for clustering, only the input dataset is needed to get the final clustering result, which makes the algorithm simpler and more practical. Based on the experiments on 7 UCI datasets, we compare the performance of the proposed CASP with several existing clustering techniques, including CABOSFV_CS, CABSOFV_D, HABOC, and K-modes. The experimental results with three external evaluation metrics indicate that CASP algorithm has better clustering effectiveness than baseline algorithms. CASP not only solves the difficulty of threshold parameter decision, but also has high computing efficiency. Moreover, the clustering results are of stability and reliability. So, it can be widely used in practical applications.

In the proposed CASP algorithm, a suitable threshold parameter can be determined to acquire final clustering results. However, the parameter found by this method is usually a relatively suitable parameter, not necessarily the optimal one. It is difficult to define an evaluation function that measures the clustering results of CASP. Therefore, the optimization of CASP evaluation function needs further study. In the future research, we will try some intelligent algorithms to solve the above problem.

## Acknowledgements

This work is supported by National Natural Science Foundation of China (71971025).

## 6 REFERENCES

[1] Paul, D., Saha, S., & Mathew, J. (2020). Improved Subspace Clustering Algorithm using Multi-objective Framework and Subspace Optimization. *Expert Systems with Applications*, *158*, 113487. https://doi.org/10.1016/j.eswa.2020.113487

[2] Ritu Gautam, D. P. (2017). A Systematic Review on Educational Data Mining. *International Journal of Science and Research (IJSR)*, *5*(11), 15991-16005. https://doi.org/10.1109/ACCESS.2017.2654247

[3] Liao, S. H., Chu, P. H., & Hsiao, P. Y. (2012). Data mining techniques and applications - A decade review from 2000 to 2011. *Expert Systems with Applications*, *39*(12), 11303-11311. https://doi.org/10.1016/j.eswa.2012.02.063

[4] Zhang, J., Wei, Q., & Chen, G. (2012). An efficient incremental method for generating equivalence groups of search results in information retrieval and queries. *Knowledge-Based Systems*, *32*, 91-100. https://doi.org/10.1016/j.knosys.2011.08.013

[5] Sharma, K. K. & Seal, A. (2020). Clustering analysis using an adaptive fused distance. *Engineering Applications of Artificial Intelligence*, *96*(March), 103928. https://doi.org/10.1016/j.engappai.2020.103928

[6] Janani, R. & Vijayarani, S. (2019). Text document clustering using Spectral Clustering algorithm with Particle Swarm Optimization. *Expert Systems with Applications*, *134*, 192-200. https://doi.org/10.1016/j.eswa.2019.05.030

[7] Zheng, C. T., Liu, C., & Wong, H. S. (2018). Corpus-based topic diffusion for short text clustering. *Neurocomputing*, 275, 2444-2458. https://doi.org/10.1016/j.neucom.2017.11.019

[8] Wang, Y., Ma, X., Lao, Y., & Wang, Y. (2014). A fuzzy-based customer clustering approach with hierarchical structure for logistics network optimization. *Expert Systems with Applications*, *41*(2), 521-534. https://doi.org/10.1016/j.eswa.2013.07.078

[9] Motlagh, O., Berry, A., & O'Neil, L. (2019). Clustering of residential electricity customers using load time series. *Applied Energy*, 237(January), 11-24.

https://doi.org/10.1016/j.apenergy.2018.12.063
[10] Guo, J., Yuan, X., Xu, P., Bai, H., & Liu, B. (2020). Improved image clustering with deep semantic embedding. *Pattern Recognition Letters*, *130*, 225-233. https://doi.org/10.1016/j.patrec.2018.10.022
[11] Ren, Y., Wang, N., Li, M., & Xu, Z. (2020). Deep density-based image clustering. *Knowledge-Based Systems*, *197*, 105841. https://doi.org/10.1016/j.knosys.2020.105841
[12] Tseng, K. K., Li, J., Tang, Y. J., Yang, C. W., Lin, F. Y., & Zhao, Z. (2020). Clustering Analysis of Aging Diseases and Chronic Habits With Multivariate Time Series Electrocardiogram and Medical Records. *Frontiers in Aging Neuroscience, 12(May)*. https://doi.org/10.3389/fnagi.2020.00095
[13] Li, X., Jiao, H., & Li, D. (2020). Intelligent medical heterogeneous big data set balanced clustering using deep learning. *Pattern Recognition Letters*, *138*, 548-555. https://doi.org/10.1016/j.patrec.2020.08.027
[14] Wu, S. & Gao, X. (2004). CABOSFV algorithm for high dimensional sparse data clustering. *Journal of University of Science and Technology Beijing*, 11.
[15] Liu, X., Yu, D., & Li, Y. (2008). Customer Clustering of CABOSFV Based on Customer Knowledge. *Journal of Intelligence*, (2), 7-10.
[16] WANG, D. & ZHU, D. (2013). Research of Mining Word Category Knowledge Based on CABOSFV. *Computer Science*, *40*(7), 211-215.
[17] Wei, G. Y., Zou, L., & Pan, J. (2014). Improved text classification algorithm for spam filtering based on CABSOFV. *Future Computer and Information Technology*, *86*, 1131-1139. https://doi.org/10.2495/ICFCIT131301
[18] Wang, M., Wang, M., Wang, D., Duan, S., Wang, Y., Huang, Y., & Shang, H. (2015). The syndromes of lung cancer and compatibility of medicine in Traditional Chinese Medicine science treatment based on Clustering Algorithm. *Proceedings - 2015 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2015*, 1070-1077. https://doi.org/10.1109/BIBM.2015.7359830
[19] Zhang, Q. (2013). Research and implementation of clustering analysis algorithms based on I-MINER. *Proceedings - 2013 International Conference on Computer Sciences and Applications, CSA 2013*, 254-257. https://doi.org/10.1109/CSA.2013.65
[20] Wu, S., Wang, J., & Tan, Y. (2011). Improved CABOSFV clustering considering data sort. *Computer Engineering and Applications*, 2010-2012.
[21] Khan, I., Luo, Z., Huang, J. Z., &Shahzad, W. (2019). Variable Weighting in Fuzzy k-Means Clustering to Determine the Number of Clusters. *IEEE Transactions on Knowledge and Data Engineering*, *32*(9), 1838-1853. https://doi.org/10.1109/tkde.2019.2911582
[22] Azhar, M., Huang, J. Z., Masud, M. A., Li, M. J., & Cui, L. (2020). A hierarchical Gamma Mixture Model-based method for estimating the number of clusters in complex data. *Applied Soft Computing Journal*, *87*, 105891. https://doi.org/10.1016/j.asoc.2019.105891
[23] Gao, X. & Wu, S. (2018). Hierarchical Clustering Algorithm for Binary Data Based on Cosine Similarity. *8th International Conference on Logistics, Informatics and Service Sciences, LISS 2018 - Proceeding*, (71271027). https://doi.org/10.1109/LISS.2018.8593222
[24] Wu, S., He, H., & Fan, Y. (2020). High dimensional data clustering algorithm based on extended dissimilarity. *Computer Engineering and Applications*. https://doi.org/10.1007/978-981-15-1145-5
[25] Huang, Z. (1997). A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. *Research Issues on Data Mining and Knowledge Discovery*, 1-8.
[26] Zhang, Y., Yang, Y., Li, T., & Fujita, H. (2019). A multitask multiview clustering algorithm in heterogeneous situations based on LLE and LE. *Knowledge-Based Systems*, 163, 776-786. https://doi.org/10.1016/j.knosys.2018.10.001
[27] Tibshirani, R., Walther, G., & Hastie, T. (2001). Estimating the number of data clusters via the gap statistic. *Journal of the Royal Statistical Society: Series B*, 63, 411-423. https://doi.org/10.1111/1467-9868.00293
[28] Cheong, M. Y. & Lee, H. (2008). Determining the number of clusters in cluster analysis. J*ournal of the Korean Statal Society*, *37*(2), 135-143. https://doi.org/10.1016/j.jkss.2007.10.004
[29] Bondarenko, I., Malderen, H. V., Treiger, B., Espen, P. V., & Grieken, R. V. (1994). Hierarchical cluster analysis with stopping rules built on akaike's information criterion for aerosol particle classification based on electron probe x-ray microanalysis. *Chemometrics and Intelligent Laboratory Systems*, *22*(1), 87-95. https://doi.org/10.1016/0169-7439(93)E0052-6
[30] Ünlü, R. & Xanthopoulos, P. (2019). Estimating the number of clusters in a dataset via consensus clustering. *Expert Systems with Applications*, *125*, 33-39. https://doi.org/10.1016/j.eswa.2019.01.074
[31] Pimentel, B. A. & de Carvalho, A. C. P. L. F. (2020). A Meta-learning approach for recommending the number of clusters for clustering algorithms. *Knowledge-Based Systems*, *195*, 105682. https://doi.org/10.1016/j.knosys.2020.105682
[32] Dua, D. & Graff, C. (2017). {UCI} Machine Learning Repository. Retrieved from http://archive.ics.uci.edu/ml

**Contact information:**

**Huixia HE**, Master degree candidate
School of Economics and Management,
University of Science and Technology Beijing,
No. 30 Xueyuan Road, Haidian District, Beijing, China
E-mail: 18810081025@163.com

**Guiying WEI**, Associate Professor
School of Economics and Management,
University of Science and Technology Beijing,
No. 30 Xueyuan Road, Haidian District, Beijing, China
E-mail: weigy@manage.ustb.edu.cn

**Sen WU**, Full Professor
(Corresponding author)
School of Economics and Management,
University of Science and Technology Beijing,
No. 30 Xueyuan Road, Haidian District, Beijing, China
E-mail: wusen@manage.ustb.edu.cn

**Xiaonan GAO**, PhD candidate
School of Economics and Management,
University of Science and Technology Beijing,
No. 30 Xueyuan Road, Haidian District, Beijing, China
E-mail: gaoxiaonan0001@163.com