Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.060090

ARTICLE

Coordinate Descent K-means Algorithm Based on Split-Merge

Fuheng Qu[1], Yuhang Shi[1], Yong Yang[1,*], Yating Hu[2] and Yuyao Liu[1]

[1]College of Computer Science and Technology, Changchun University of Science and Technology, Changchun, 130022, China

[2]College of Computer Science and Technology, Jilin Agricultural University, Changchun, 130118, China

*Corresponding Author: Yong Yang. Email: yy@cust.edu.cn

Received: 23 October 2024 Accepted: 20 November 2024 Published: 19 December 2024

ABSTRACT

The Coordinate Descent Method for K-means (CDKM) is an improved algorithm of K-means. It identifies better locally optimal solutions than the original K-means algorithm. That is, it achieves solutions that yield smaller objective function values than the K-means algorithm. However, CDKM is sensitive to initialization, which makes the K-means objective function values not small enough. Since selecting suitable initial centers is not always possible, this paper proposes a novel algorithm by modifying the process of CDKM. The proposed algorithm first obtains the partition matrix by CDKM and then optimizes the partition matrix by designing the split-merge criterion to reduce the objective function value further. The split-merge criterion can minimize the objective function value as much as possible while ensuring that the number of clusters remains unchanged. The algorithm avoids the distance calculation in the traditional K-means algorithm because all the operations are completed only using the partition matrix. Experiments on ten UCI datasets show that the solution accuracy of the proposed algorithm, measured by the E value, is improved by 11.29% compared with CDKM and retains its efficiency advantage for the high dimensional datasets. The proposed algorithm can find a better locally optimal solution in comparison to other tested K-means improved algorithms in less run time.

KEYWORDS

Cluster analysis; K-means; coordinate descent K-means; split-merge

1 Introduction

Clustering is an unsupervised machine-learning method [1] that does not depend on data labels. It has a wide range of applications in many fields [2], such as image segmentation [3], target recognition [4] and feature extraction [5]. Among these applications, the K-means algorithm is one of the most commonly used clustering algorithms due to its simplicity and interpretability [6,7].

The K-means clustering problem is a non-deterministic polynomial-time hardness (NP-hard) problem [8]. The traditional K-means clustering algorithm is a greedy algorithm used to optimize the K-means clustering model. However, its sensitivity to the choice of the initial centroids leads to its tendency to fall into local optimal solutions.

To address this problem, researchers have proposed many improved methods [9–10] for enhancing ***the solution accuracy***[1] of the K-means clustering algorithm. Gul et al. proposed the R-K-means algorithm, which uses a two-step process to find the initialization centroids, making the K-means algorithm effective in solution accuracy on large-scale high dimensional datasets [11]. Biswas et al. used the computational geometry for cluster center initialization to make cluster centers uniformly distributed [12]. Layeb et al. proposed two deterministic initialization methods for K-means clustering based on modified crowding distances, which are capable of selecting more uniform initial centers based on the modified crowding distances [13]. Arthur et al. proposed the K-means++ algorithm, which is based on a random seeding technique to make the initialization centroids as dispersed as possible [14]. Lattanzi et al. improved K-means++ by adding a new local search strategy to it, which can optimize the location of the center of bad clusters [15]. Şenol proposed a method to find the optimal initial centers using kernel density estimation so that the initial centroids are distributed in regions with a high density of data points [16]. Reddy et al. selected the initial cluster centers by constructing a Voronoi diagram using the data points, which reduces the problem of the K-means algorithm's overdependence on initial centroids and improves the convergence time of the subsequent K-means algorithm [17].

In addition to the above methods, a large number of improvements for initialization have been proposed in the literature [18]. The final result of K-means clustering is determined by initialization and iteration. Iteration and initialization are similar. They can both make improvements to the solution accuracy. However, there is little research on iterative processes. Recently, Nie et al. improved the iterative process of the K-means algorithm. They rewrote the objective function of K-means and introduced the Coordinate Descent Method [19,20] to optimize the K-means clustering model. This new algorithm is called the Coordinate Descent Method for K-means (CDKM) algorithm. Experimental results show that under the same initialization conditions, the CDKM algorithm has a more significant improvement in solution accuracy than the K-means algorithm and runs more efficiently on high dimensional datasets [21].

Although CDKM is able to find smaller local optimal solutions than K-means, CDKM is still sensitive to initialization. There is a large room for improvement in the solution accuracy of CDKM. Since selecting the appropriate initial center is not always feasible [21], we attempt to improve its iterative process. Specifically, we introduce the split-merge criterion into CDKM. The split-merge criterion was proposed by Kaukoranta et al. in the iterative split-and-merge algorithm in 1998 [22]. The iterative split-and-merge algorithm aims to optimize codebook generation by utilizing the split-merge criterion. The split-merge criterion is also an excellent operation that can significantly enhance the solution accuracy of K-means. Based on the split-and-merge criterion, many improved algorithms have been proposed, such as the iterative split-and-merge algorithm [22], split algorithm [23], random swap algorithm [24] and I-K-means-+ algorithms [25]. However, the split-merge criterion primarily operates under the original K-means objective function, which may not apply to the CDKM objective function. One of the key strengths of CDKM is its efficiency in high dimensional data. This efficiency arises from its objective function, which eliminates the need for distance calculations during the search process. This feature is not typically available in traditional K-means algorithms and its improved algorithms, such as the algorithms mentioned above [11–14] and the split-merge algorithm [22–25] analyzed earlier.

[1] In this paper, ***when we say that the solution accuracy of solution*** $C^{(1)}$ ***is higher than that of solution*** $C^{(2)}$***, it means that the value of the objective function corresponding to solution*** $C^{(1)}$ ***is less than the value of the objective function of solution*** $C^{(2)}$.

The challenge is how to apply the split-merge criterion based on the original K-means objective function to the CDKM algorithm, which is based on the CDKM objective function. We propose a Coordinate Descending method for K-means algorithm based on Split-Merge (CDKMSM) algorithm from the perspective of improving the iterative process of CDKM. First, an existing algorithm based on split-merge criterion, specifically, the I-K-means+ algorithm, is modified with the aim of obtaining an excellent solution. Then, the proposed split-merge criterion is converted into a partition matrix operation to make it applicable to the CDKM clustering model. In order to retain the efficiency advantage of the original CDKM under high dimensional data, the proposed algorithm avoids distance computation, and its computation process can be accomplished only by using the partition matrix.

2 Related Work

2.1 CDKM Algorithm

Suppose $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \in R^{d\times n}$ is a data set with N individual elements. The goal of the K-means clustering model is to divide the dataset into disjoint clusters $C = \{C_1, C_2, \ldots, C_k\}$. The K-means algorithm uses the traditional error sum of squares (SSE) function within clusters as a measure of clustering effectiveness. For a given solution $C = (C_1, C_2, \ldots, C_k)$, its SSE value is shown as follows:

$$\mathrm{SSE}\,(C) = \sum_{i=1}^{k} \mathrm{SSE}\,(C_i), \tag{1}$$

where, $\mathrm{SSE}\,(C_i)$ of the i-th cluster is shown as follows:

$$\mathrm{SSE}\,(C_i) = \sum_{x_j \in C_i} \left\| \boldsymbol{x}_j - m_i \right\|^2, \tag{2}$$

where, m_i is the center of the cluster C_i. The calculation of m_i is shown as follows:

$$m_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} \boldsymbol{x}_j, \tag{3}$$

The clustering model of K-means can be described as a minimization problem is shown as follows:

$$\min_{C} \mathrm{SSE} = \min_{C} \sum_{i=1}^{k} \sum_{x_j \in C_i} \left\| \boldsymbol{x}_j - m_i \right\|_2^2. \tag{4}$$

CDKM rewrites the K-means clustering problem as Eq. (5) by using the partition matrix.

$$\max_{e \in \{1,2,\ldots,k\}} obj\left(\boldsymbol{F}^{(e)}\right) = \max_{e \in \{1,2,\ldots,k\}} \sum_{l=1}^{k} \frac{\left(\boldsymbol{f}_l^{(e)}\right)^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_l^{(e)}}{\left(\boldsymbol{f}_l^{(e)}\right)^T \boldsymbol{f}_l^{(e)}} \tag{5}$$

In Eq. (5), $\boldsymbol{F}^{(e)}$ $(e = 1, 2, \ldots, k)$ is a membership matrix, e represents the position information of the updated element when the $\boldsymbol{F}^{(e)}$ matrix is updated in rows, $\boldsymbol{f}_l^{(e)}$ is the l-th column of $\boldsymbol{F}^{(e)}$, $obj\left(\boldsymbol{F}^{(e)}\right)$ represents the objective function value of the CDKM algorithm. Note: since both the expression form and the objective function value of the rewritten CDKM solution are different from those in the original K-means, different mathematical symbols are used here to distinguish them.

In order to reduce the amount of calculation, CDKM defines $\psi(e)$ is shown as follows:

$$\psi(e) = \begin{cases} \dfrac{\boldsymbol{f}_e^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_e}{\boldsymbol{f}_e^T \boldsymbol{f}_e} - \dfrac{\boldsymbol{f}_e^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_e - 2\boldsymbol{x}_i^T \boldsymbol{X} \boldsymbol{f}_e + \boldsymbol{x}_i^T \boldsymbol{x}_i}{\boldsymbol{f}_e^T \boldsymbol{f}_e - 1} e = p \\ \dfrac{\boldsymbol{f}_e^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_e + 2\boldsymbol{x}_i^T \boldsymbol{X} \boldsymbol{f}_e + \boldsymbol{x}_i^T \boldsymbol{x}_i}{\boldsymbol{f}_e^T \boldsymbol{f}_e + 1} - \dfrac{\boldsymbol{f}_e^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_e}{\boldsymbol{f}_e^T \boldsymbol{f}_e} e \neq p \end{cases}, \tag{6}$$

where, $i \in \{1, 2, \ldots, N\}$ represents the row number of the currently processed row when updating the partition matrix $\boldsymbol{F}^{(e)}$, p represents the column number of the element with the value of 1 in the row i when the row is updated. The calculation of $\psi(e)$ is divided into two cases: $e = p$ and $e \neq p$.

According to the coordinate descent method and the property of the partition matrix $\boldsymbol{F}^{(e)}$, the i-th row of $\boldsymbol{F}^{(e)}$ is updated as follows:

$$\boldsymbol{f}_{iq} = \begin{cases} 1 & \arg\max_e \psi(e) \\ 0 & otherwise \end{cases}, \tag{7}$$

The variables that need to be updated after the i-th row of $\boldsymbol{F}^{(e)}$ are updated as follows:

$$\begin{aligned} &\boldsymbol{X}\boldsymbol{f}_p = \boldsymbol{X}\boldsymbol{f}_p - \boldsymbol{x}_i; \boldsymbol{X}\boldsymbol{f}_q = \boldsymbol{X}\boldsymbol{f}_q + \boldsymbol{x}_i; \\ &\boldsymbol{f}_p^T \boldsymbol{f}_p = \boldsymbol{f}_p^T \boldsymbol{f}_p - 1; \boldsymbol{f}_q^T \boldsymbol{f}_q = \boldsymbol{f}_q^T \boldsymbol{f}_q + 1. \end{aligned} \tag{8}$$

$$\begin{aligned} &\boldsymbol{f}_p^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_p = \boldsymbol{f}_p^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_p - 2\boldsymbol{x}_i^T \boldsymbol{X} \boldsymbol{f}_p + \boldsymbol{x}_i^T \boldsymbol{x}_i; \\ &\boldsymbol{f}_q^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_q = \boldsymbol{f}_q^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{f}_q + 2\boldsymbol{x}_i^T \boldsymbol{X} \boldsymbol{f}_q + \boldsymbol{x}_i^T \boldsymbol{x}_i. \end{aligned} \tag{9}$$

Note: The CDKM method presents new formulas to modify the objective function and replace the original K-means iterative process with a coordinate descent method. These formulas clarify the improvement process and detail the iterative method used in the newly proposed algorithm (Algorithm 1). This paper includes the objective function and iterative process of CDKM, which is why the formulas are presented.

Algorithm 1: CDKM algorithm

```
Input: data matrix $\boldsymbol{X} \in R^{d \times n}$, cluster number $k$.
Initialize c cluster centers by an initial strategy and get initial $\boldsymbol{F} \in R^{n \times k}$.
Compute and store $\boldsymbol{X}\boldsymbol{f}_e, \boldsymbol{f}_e^T\boldsymbol{f}_e, \boldsymbol{f}_e^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{f}_e$ $(e = 1, 2, \ldots, k)$, $\boldsymbol{x}_i^T\boldsymbol{x}_i$ $(i = 1, 2, \ldots, n)$.
repeat
  for i = 1 to n
      Calculate $\psi(e)$ $(e = 1, 2, \ldots, k)$ by Eq. (6).
      Update the i-th row of $\boldsymbol{F}$ by Eq. (7);
      if $p \neq q$ then
          Update $\boldsymbol{X}\boldsymbol{f}_e$ and $\boldsymbol{f}_e^T\boldsymbol{f}_e$ $(e = p, q)$ by Eq. (8).
          Update $\boldsymbol{f}_e^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{f}_e$ $(e = p, q)$ by Eq. (9).
      end if
  end for
until convergence
Output: partition matrix $\boldsymbol{F} \in R^{n \times k}$
```

2.2 *The Problems of CDKM Algorithm*

When the initial center position is not ideal (as shown in the left part of Fig. 1), the initial centers in different regions may be too few or too many. As shown in the right part of Fig. 1, there may be a local optimum problem after CDKM clustering. The datasets in the clusters are too sparse after clustering in regions with too few centers, and the datasets in the clusters are too dense after clustering in regions with too many centers. The division of these clusters is not accurate enough, which leads to the solution accuracy needing to be improved.

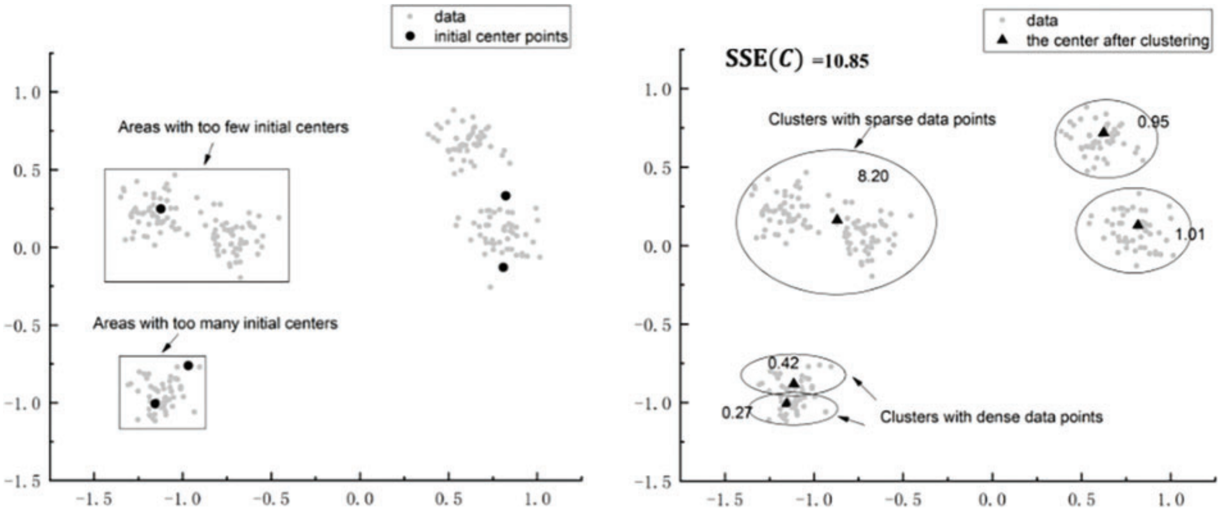

Figure 1: The left part is initial center distribution. The right part is the division of datasets after CDKM clustering

3 Design of Coordinate Descent K-Means Algorithm Based on Split-Merge

For the problem described in Section 2.2, in order to improve the solution accuracy of the CDKM algorithm, we introduce the split-merge criterion and transform it to apply to the clustering model of CDKM. Split-merge criterion has already been introduced in the iterative split-and-merge algorithm in 1998 [22]. The iterative split-and-merge algorithm begins with an initial codebook and enhances it through merge and split operations. Merging small neighboring clusters frees up additional code vectors, which can then be reallocated by splitting larger clusters. We introduce the split-merge criterion for the traditional K-means algorithm in Section 3.1. In Section 3.2, we analyze the shortcomings of the split-merge criterion for the traditional K-means algorithm and introduce the modified method in this paper. Since the split-merge criterion for the traditional K-means and the modified criterion we proposed in Section 3.2 are used to optimize the objective function value SSE (C) of the original K-means model, they cannot be directly applied to solve the objective function value obj $\left(\boldsymbol{F}^{(e)}\right)$ in the CDKM clustering model. Moreover, the original split-merge process involves a large number of distance calculations. In Section 3.3, we introduce a method that transforms the formulas and operations corresponding to the improved split-merge criterion proposed in Section 3.2 of this paper so that it can be applied to the optimization of CDKM models. This method relies solely on the partition matrix and avoids distance calculation to improve the time efficiency of the algorithm in high dimensional datasets.

3.1 Split-Merge Criterion for the Traditional K-Means Algorithm

The idea of split-merge criterion has been used to improve K-means, resulting in many proposed algorithms [22–25]. These algorithms generally share a common approach, utilizing split-merge criterion to enhance the original K-means algorithm. In this paper, we build upon one of the more recent split-merge criterion algorithms, specifically the I-K-means-+ algorithm proposed in 2018. It improves the K-means' solution accuracy by merging two clusters, splitting another cluster, and regrouping in each iteration, which is capable of gradually approaching the global optimal solution. The cluster C_v with the smallest cost selected for deletion when merging, and the cluster C_j with the largest gain selected for division when splitting. The calculation of cost and gain is shown as follows:

$$\text{cost}\,(C_v) = \text{SSE}\,(C_v) - \sum_{\forall p \in C_v} \text{dis}\left(p, Z_p\right)^2, \tag{10}$$

$$\text{gain}\left(C_j\right) \approx \frac{3}{4} \times \text{SSE}\left(C_j\right), \tag{11}$$

where, Z_p in Eq.(10) is the sub-proximal centroid of the data point p.

3.2 Split-Merge Criterion in This Paper

Since the calculation of the gain value of the I-K-means-+ algorithm (Eq. (11)) is an approximate calculation method, it may produce a misclassification when splitting and merging. So in this paper, the split-merge criterion of the I-K-means-+ algorithm is modified in the expectation of obtaining a solution with higher solution accuracy. We have changed the method of calculating the gain to an exact one, and also improved the method of calculating the cost of the I-K-means-+ algorithm.

The merging strategy in this paper aims to find two clusters that are close to each other, which are too densely populated with datasets, and merge them to minimize the cost is shown as follows [22]:

$$\text{cost}_{A,B} = \text{SSE}\left(C_{A,B}\right) - (\text{SSE}\,(C_A) + \text{SSE}\,(C_B)), \tag{12}$$

where, $C_{A,B}$ denotes the new cluster after merging two clusters of C_A and C_B. The merging operation is shown in Fig. 2, assuming that there are k clusters, calculate the SSE values of all the clusters, find the $\lfloor \frac{k}{2} \rfloor$ clusters with the smallest SSE values, calculate the cost of these $\lfloor \frac{k}{2} \rfloor$ clusters merged with the rest of the clusters two by two, respectively, and selecting the two clusters, A and B with the smallest cost values to be merged, i.e., assigning the data points in A to B, and deleting the center a of the original cluster A.

The objective of the splitting criterion presented in this paper is to identify clusters characterized by sparse datasets. Upon splitting such a cluster into two distinct clusters, a significant reduction in the SSE value is observed. Eq. (13) is employed in this study to facilitate an accurate calculation of the gain resulting from splitting [22].

$$\text{gain} = \text{SSE}\,(C_s) - \left(\text{SSE}\left(C_s^1\right) + \text{SSE}\left(C_s^2\right)\right) \tag{13}$$

where, $\left\{C_s^1, C_s^2\right\}$ represents the two clusters produced by each cluster C_s division. The splitting operation is shown in Fig. 2. In the splitting operation, a random point from the selected cluster is chosen as a new center. Use this new center and the original cluster center to perform clustering on the cluster using the K-means method. Traverse all clusters. Calculate the gain of all clusters after splitting. Then, find the cluster with the largest gain H. Finally, a random data point in H is used as the second center so that H splits into two clusters H^1 and H^2.

The advantage of this split-merge criterion is that it avoids too dense or sparse division of clusters and improves the solution accuracy.

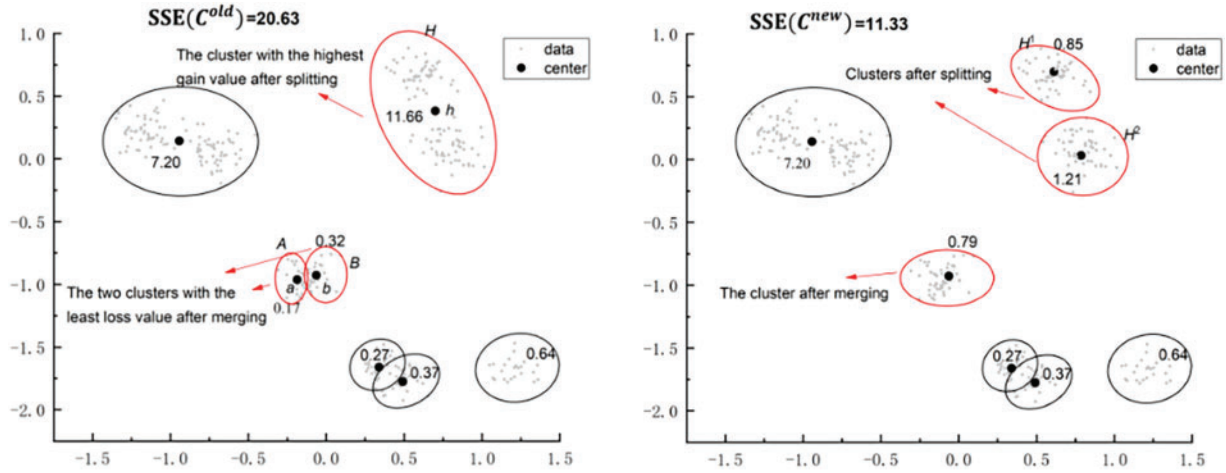

Figure 2: The left part is the division results corresponding to the original solution C^{old} with SSE values. The right part is the result after merging and splitting

3.3 Split-Merge Criterion for CDKM Model Optimization

Section 3.2 introduces the improved split-merge criterion, however, this criterion is based on the original K-means model, which is not directly applicable to the CDKM model since the form of the solution and the objective function value of the CDKM are different from the original K-means. Moreover, this criterion needs to adjust the existing centers and redistribute the data points for the split-merge operation, which involves more distance calculations, and the more the centers change, the more the data points need to be calculated, and the higher the time complexity, especially in the high-dimensional dataset, which is even more prominent. To address the above problems, this paper uses the partition matrix to design a new split-merge criterion to make it applicable to the CDKM model and reduce the time complexity at the same time.

1) Merging operation

We perform the merging operation by integrating the two columns of the partition matrix. The split-merge criterion in the previous section used the objective function values of the K-means algorithm for calculating the cost gain, which is modified here to use the CDKM form of the objective function values. The values of the objective function before and after merging are calculated using Eqs. (14) and (15), respectively, and the cost after merging is calculated as clusters using Eq. (16). Where, obj $(\boldsymbol{f}_o,\boldsymbol{f}_w)$ represents the sum of the CDKM objective function values of the columns $\boldsymbol{f}_o$ and $\boldsymbol{f}_w$ corresponding to the clusters before merging, obj $(\boldsymbol{f}_{o,w})$ represents the sum of the CDKM objective function values of the new column $\boldsymbol{f}_{o,w}$ after the merging of these two clusters, and cost $(\boldsymbol{f}_{o,w})$ represents the cost after the merging of these two clusters. Using Eq. (5), calculate and find the $\lfloor \frac{k}{2} \rfloor$ clusters with the smallest obj $(\boldsymbol{F}^{(e)})$ value, and calculate the cost after merging this $\lfloor \frac{k}{2} \rfloor$ cluster with the other clusters separately. The two clusters with the smallest cost after merging are merged, assuming that these two clusters correspond to columns $\boldsymbol{f}_o$ and $\boldsymbol{f}_w$ in the partition matrix $\boldsymbol{F}$. This is shown in Fig. 3. Find the labels of the rows in $\boldsymbol{f}_o$ with value 1, and set the values of these rows in $\boldsymbol{f}_w$ to 1. Delete $\boldsymbol{f}_o$ after the operation, and complete the merging of the two clusters by the above steps.

$$\text{obj}\left(\boldsymbol{f}_o,\boldsymbol{f}_w\right) = \frac{\left(\boldsymbol{f}_o\right)^T \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{f}_o}{\left(\boldsymbol{f}_o\right)^T \boldsymbol{f}_o} + \frac{\left(\boldsymbol{f}_w\right)^T \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{f}_w}{\left(\boldsymbol{f}_w\right)^T \boldsymbol{f}_w}, \tag{14}$$

$$\text{obj}\left(\boldsymbol{f}_{o,w}\right) = \frac{\left(\boldsymbol{f}_{o,w}\right)^T \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{f}_{o,w}}{\left(\boldsymbol{f}_{o,w}\right)^T \boldsymbol{f}_{o,w}}, \tag{15}$$

$$\text{cost}\left(\boldsymbol{f}_{o,w}\right) = \text{obj}\left(\boldsymbol{f}_{o,w}\right) - \text{obj}\left(\boldsymbol{f}_o,\boldsymbol{f}_w\right). \tag{16}$$

$$\begin{array}{c} \begin{array}{cccccccc} \downarrow o & & & & & w\downarrow & & \end{array} \\ \left[\begin{array}{cccccccc} 1 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ & & & \vdots & & & & \\ 1 & 0 & 0 & & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & & 0 & 0 & \cdots & 0 \\ & & & \vdots & & & & \\ 1 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{array}\right]_{n\times k} \end{array} \Rightarrow \begin{array}{c} \begin{array}{ccccccc} \downarrow o+1 & & & & w\downarrow & & \end{array} \\ \left[\begin{array}{ccccccc} 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ & & \vdots & & & & \\ 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \\ & & \vdots & & & & \\ 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \end{array}\right]_{n\times(k-1)} \end{array}$$

Figure 3: Use membership matrix to merge two clusters

2) Splitting operation

First, the gain of each cluster is calculated. Then, the cluster splitting operation is accomplished by splitting the columns of the partition matrix. The specific method is as follows. For each cluster $\boldsymbol{C}_i$ before the splitting operation, find its corresponding column $\boldsymbol{f}_i$ in the partition matrix $\boldsymbol{F}$. As shown in Fig. 4, we create a matrix $\boldsymbol{g} = \left(\boldsymbol{g}_1, \boldsymbol{g}_2\right)$, where $\boldsymbol{g}_1 = \boldsymbol{f}_i$ and $\boldsymbol{g}_2$ is 0 for all elements except row u, which is 1. u is the row label of a non-zero element randomly selected in $\boldsymbol{f}_i$. Clustering to convergence using the CDKM algorithm for $\boldsymbol{g}$ yields the membership matrix $\boldsymbol{g}'$ corresponding to the two clusters after splitting. The objective function values before and after splitting are calculated using Eqs. (17) and (18), respectively, and the gain after cluster splitting is calculated using Eq. (19), where, obj $\left(\boldsymbol{f}_i\right)$ represents the CDKM objective function value before splitting of the cluster C_i, obj $\left(\boldsymbol{g}_1', \boldsymbol{g}_2'\right)$ represents the CDKM objective function value after splitting of the corresponding cluster of the column C_i into two clusters. gain $\left(\boldsymbol{f}_i\right)$ represents the gain value of the column $\boldsymbol{f}_i$ after splitting the corresponding cluster.

$$\text{obj}\left(\boldsymbol{f}_i\right) = \frac{\left(\boldsymbol{f}_i\right)^T \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{f}_i}{\left(\boldsymbol{f}_i\right)^T \boldsymbol{f}_i}, \tag{17}$$

$$\text{obj}\left(\boldsymbol{g}_1', \boldsymbol{g}_2'\right) = \frac{\left(\boldsymbol{g}_1'\right)^T \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{g}_1'}{\left(\boldsymbol{g}_1'\right)^T \boldsymbol{g}_1'} + \frac{\left(\boldsymbol{g}_2'\right)^T \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{g}_2'}{\left(\boldsymbol{g}_2'\right)^T \boldsymbol{g}_2'}, \tag{18}$$

$$\text{gain}\left(\boldsymbol{f}_i\right) = \text{obj}\left(\boldsymbol{g}_1', \boldsymbol{g}_2'\right) - \text{obj}\left(\boldsymbol{f}_i\right). \tag{19}$$

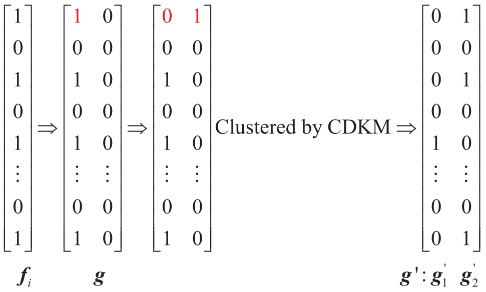

Figure 4: Using partition matrix to split clusters

The following describes the method of splitting the cluster with the largest gain, i.e., column $\boldsymbol{f}_m$ in the $\boldsymbol{F}$ matrix. As shown in Fig. 5, a new column $\boldsymbol{f}_r$ is added at the end of the membership matrix $\boldsymbol{F}$. The initial values of the column are all set to 0, and an element with a value of 1 is randomly selected in $\boldsymbol{f}_m$, which is aligned with the element values of the corresponding rows in $\boldsymbol{f}_r$. Clustering was again performed using CDKM to group the data points to the nearest centers, thus completing the merging operation by the partition matrix.

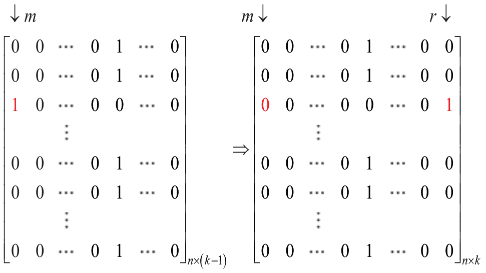

Figure 5: Using partition matrix to split clusters

3.4 Algorithm Description

Algorithm 2 is the proposed algorithm in this paper. The following steps provide a clear representation of the algorithm's workflow, illustrating how it progresses from start to finish. This includes performing the CDKM, followed by the split-merge criterion, and concludes with the final completion of the algorithm.

Algorithm 2: CDKMSM algorithm

```
Input: Dataset $\boldsymbol{X}'_{n\times d}$ clustering number $k$
Step 1. Calculate $\boldsymbol{F}_{n\times k}$. $\boldsymbol{F}_{n\times k} \leftarrow$ CDKM $(\boldsymbol{X}_{n\times d}, k)$.
Step 2. $\boldsymbol{F}_{n\times k} \rightarrow \boldsymbol{F}'_{n\times(k-1)}$. Perform the merging operation on $\boldsymbol{F}_{n\times k}$ to obtain $\boldsymbol{F}_{n\times k} \rightarrow \boldsymbol{F}'_{n\times(k-1)}$:
        For each $f_i$, calculate and find $\left\lfloor \frac{k}{2} \right\rfloor$ columns with the smallest obj $\left(\boldsymbol{F}^{(e)}\right)$ value by Eq. (5)
```

(Continued)

Algorithm 2 (continued)

```
        Find the two columns with the smallest cost from these $\left\lfloor \frac{k}{2} \right\rfloor$ columns and all other columns by Eq. (16)
        Merge their corresponding columns in the partition matrix to create a new merged matrix $\boldsymbol{F}'_{n\times(k-1)}$.
Step 3. $\boldsymbol{F}'_{n\times(k-1)} \rightarrow \boldsymbol{F}'_{n\times k}$. Perform the splitting operation on $\boldsymbol{F}'_{n\times(k-1)}$ to obtain $\boldsymbol{F}'_{n\times k}$:
        For each $f_i$, calculate the gain using Eq. (19).
        Split the $f_i$ with the highest gain using the splitting operation in Section 3.3, updating the partition matrix $\boldsymbol{F}'_{n\times(k-1)}$.
Step 4. Update $\boldsymbol{F}_{n\times k}$ by using $\boldsymbol{F}'$ and $k$ value as inputs, and run the CDKM algorithm until convergence to obtain $\boldsymbol{F}_{n\times k}$.
Output: partition matrix $\boldsymbol{F}_{n\times k}$
```

3.5 Computational Complexity Analysis

Our proposed algorithm is mainly composed of four parts. Let t_1, t_2, t_3 denote the number of iterations for the first, third and fourth parts of running CDKM, respectively. The first part is clustering using CDKM, and its time complexity is $O(ndkt_1)$. The second part is the stage of finding suitable clusters for consolidation, in which the time complexity of finding the smallest $\left\lfloor \frac{k}{2} \right\rfloor$ clusters by sorting each cluster according to the objective function value is $O\left(k^2\right)$. The time complexity of calculating the cost and merging the two clusters corresponding to the minimum value is $O\left(k^2d + nk\right)$. The total time complexity of this part is $O\left(k^2d + nk\right)$. The third part is to find the cluster with the largest gain for splitting, and the time complexity is $O(ndkt_2)$. The fourth part is the re-clustering using CDKM with time complexity $O(ndkt_3)$. So the time complexity of this algorithm is $O\left(ndk\left(t_1 + t_2 + t_3\right) + k^2d\right)$.

4 Experiment

4.1 Experimental Environment and Experimental Dataset

The hardware experimental platform for all the algorithms used Intel (R) Core (TM) i7-9750H CPU @ 2.60 GHz processor. The results of the split algorithm and random swap algorithm were obtained from the C++ software available at **https://cs.uef.fi/ml/software/(data=May.26.2010)** (accessed on 19 November 2024), and they were executed in Linux environment. The other algorithms' codes are written in C++. They were executed in Windows environment. Ten sets of UCI (The University of California, lrvine) data sets are selected in the experiment, which can be downloaded from **https://archive.ics.uci.edu/(data=September.30.1985)** (accessed on 19 November 2024). The specific data set information is shown in Table 1.

Table 1: Information of seven UCI datasets

	Data set	Data amount	Feature dimension
High-dimensional data sets	Tox	171	1203
	DARWIN	174	450
	Driv	606	6400

(Continued)

Table 1 (continued)

	Data set	Data amount	Feature dimension
	arcene	900	10,000
	USPS	1854	256
	TUAN	4464	241
	isolet	7797	617
Low-dimensional data sets	Liver	345	6
	Ionosphere	351	34
	Page	5473	10

Note: **Abbreviation:** Tox = Toxicity; Driv = DrivfaceD; USPS = USPSdata_20; TUAN = TUANDROMD; Liver = Liver_Disorders; Page = Page_Blocks.

In order to objectively evaluate the clustering performance of each algorithm, the evaluation index adopted in this paper is SSE, and the SSE value of each algorithm is calculated by Eq. (1) and compared. For the algorithms that rewrite the objective function, including CDKM and CDKMSM, we run the algorithm to get its solution first, and then calculate the SSE value of the corresponding K-means model objective function based on this solution.

4.2 Experimental Result

The experiment compares the results of K-means algorithm (KM), CDKM algorithm [21], split algorithm [23], Random Swap algorithm (RS) [24], I-K-means-+ algorithm (IKM) [25], K-means with new formulation algorithm (KWNF) [26] and CDKMSM algorithm. The K-means algorithm is the original clustering algorithm. The split algorithm is an original algorithm based on splitting. Random swap algorithm is an excellent variant based on the split-merge criterion. The I-K-means-+ algorithm is a modified K-means algorithm based on split-merge criterion. CDKM is a new algorithm that incorporates the coordinate descent method into the iterative process of K-means clustering. CDKM is the algorithm that is to be improved in this paper. The KWNF algorithm is a recent and effective improvement of the K-means algorithm.

In order to verify the effect of the algorithm with different numbers of clusters k, five different values of k are chosen in this paper: 4, 6, 8, 10 and 12, and the corresponding clustering results are counted. For each value of k for each data set, all algorithms run within 50 random times and the results are averaged for comparison. For each algorithm, we used randomized initialization of the centers.

The SSE value is affected by various aspects such as the dataset scale and the dataset size dimension, resulting in large differences in the SSE value for different datasets and k values. By defining a variable to represent the relationship between the SSE values of two algorithms, we can directly compare their clustering performance. It allows us to quantify the difference in performance between the two algorithms. Therefore, in addition to the SSE value (shown in Eq. (1)), we define the following E value, which measures the relative degree of improvement in the solution accuracy of a certain algorithm, named here as “alg1”, with respect to another algorithm “alg2”.

$$E = \frac{\mathrm{SSE}_{a\lg 2} - \mathrm{SSE}_{a\lg 1}}{\mathrm{SSE}_{a\lg 2}} \times 100\%. \tag{20}$$

where, $\text{SSE}_{a\lg 1}$ denotes the SSE value of the algorithm “alg1” to be measured, and $\text{SSE}_{a\lg 2}$ denotes the SSE value of the algorithm “alg2” to be compared.

By defining a variable that represents the relationship between the time values of the two algorithms, we can directly compare their clustering performance. This variable serves as a quantitative measure that encapsulates both the execution time and the effectiveness of the clustering results produced by each algorithm. The index for the time of the algorithm we measured by the following T value, which quantifies the percentage speedup of the running time T_2 of our proposed algorithm relative to the running time T_1 of the rest of the algorithms.

$$T = \frac{T_1 - T_2}{T_1} \times 100\%. \tag{21}$$

The SSE results are shown in Table 2, with the optimal and sub-optimal results shown bolded as well as skewed, respectively. The E value of the remaining algorithms as measured against the solution accuracy of the K-means algorithm is shown in Table 3.

Table 2: Comparison for SSE value

k	Dataset	Algorithm SSE							
		KM	Split [23]	RS [24]	IKM [25]	CDKM [21]	KWNF [26]	CDKMSM	Scaling value[2]
4	Tox	14.87	18.42	**6.37**	13.62	*7.88*	15.21	**6.37**	∗10^14
	DARWIN	5.29	13.24	**4.45**	*5.28*	5.32	5.30	**4.45**	∗10^13
	Driv	6.13	8.93	6.51	6.11	*6.10*	6.18	**5.82**	∗10^04
	arcene	2.07	3.61	1.97	*1.92*	2.04	2.08	**1.84**	∗10^10
	USPS	*8.74*	9.42	**8.72**	*8.74*	*8.74*	8.74	*8.73*	∗10^04
	TUAN	2.58	3.17	**2.30**	2.59	2.67	2.60	*2.41*	∗10^04
	isolet	6.46	6.72	**6.42**	*6.43*	6.46	6.46	**6.42**	∗10^05
	Liver	2.64	2.95	**2.61**	2.63	2.63	2.66	*2.62*	∗10^05
	Ionosphere	2.12	2.33	**2.00**	2.10	*2.05*	2.12	**2.00**	∗10^03
	Page	**2.13**	2.68	**2.13**	**2.13**	**2.13**	**2.13**	**2.13**	∗10^10
6	Tox	5.14	13.79	**2.30**	3.52	3.05	4.62	**2.30**	∗10^14
	DARWIN	4.57	13.20	**3.39**	4.55	3.96	4.58	*3.53*	∗10^13
	Driv	4.97	7.92	5.44	*4.77*	4.93	5.02	**4.72**	∗10^04
	arcene	1.78	3.52	1.68	**1.65**	1.77	1.78	*1.66*	∗10^10
	USPS	7.92	8.90	**7.88**	7.92	7.91	7.92	*7.90*	∗10^04
	TUAN	2.06	3.17	**1.72**	1.96	2.24	2.10	*1.95*	∗10^04
	isolet	5.94	6.57	**5.89**	*5.91*	5.93	5.93	**5.89**	∗10^05
	Liver	2.10	2.44	**1.87**	2.02	2.06	2.10	*1.92*	∗10^05
	Ionosphere	1.94	2.26	**1.81**	1.92	1.85	1.94	*1.83*	∗10^03
	Page	1.64	1.36	**1.02**	1.05	1.63	1.64	*1.03*	∗10^10
8	Tox	46.68	137.79	**3.56**	16.77	22.99	46.17	*8.18*	10^13
	DARWIN	4.04	13.19	**2.89**	3.91	3.51	4.03	*3.02*	10^13
	Driv	4.31	6.96	5.05	*4.14*	4.28	4.31	**4.10**	10^04
	arcene	1.63	3.51	1.63	**1.54**	1.62	1.63	*1.55*	10^10
	USPS	7.29	8.51	**7.25**	7.28	7.28	7.29	*7.27*	∗10^04
	TUAN	1.74	3.15	**1.41**	*1.56*	1.85	1.71	1.62	∗10^04

(Continued)

Table 2 (continued)

k	Dataset	Algorithm SSE							
		KM	Split [23]	RS [24]	IKM [25]	CDKM [21]	KWNF [26]	CDKMSM	Scaling value[2]
	isolet	5.60	6.38	**5.56**	*5.57*	5.59	*5.59*	*5.57*	∗10^05
	Liver	1.77	2.35	**1.48**	*1.57*	1.69	1.74	**1.48**	∗10^05
	Ionosphere	1.76	2.23	**1.67**	1.75	1.70	1.76	*1.69*	∗10^03
	Page	15.44	8.65	*6.49*	7.40	7.60	15.44	**6.47**	∗10^09
10	Tox	48.41	109.20	**1.23**	15.24	22.99	47.88	*8.18*	∗10^13
	DARWIN	3.66	13.19	**2.60**	3.25	3.19	3.67	*2.68*	∗10^13
	Driv	3.95	6.72	4.64	*3.83*	3.94	3.97	**3.82**	∗10^04
	arcene	1.55	3.50	1.51	**1.46**	1.55	1.55	*1.49*	∗10^10
	USPS	6.79	8.43	**6.74**	*6.77*	*6.79*	*6.79*	*6.77*	∗10^04
	TUAN	1.52	3.15	**1.25**	*1.39*	1.65	1.53	1.46	∗10^04
	isolet	5.36	6.36	**5.30**	*5.32*	5.35	5.36	*5.32*	∗10^05
	Liver	1.47	2.28	**1.28**	1.34	1.37	1.44	*1.30*	∗10^05
	Ionosphere	1.67	2.21	**1.55**	1.63	1.60	1.68	*1.57*	∗10^03
	Page	14.33	5.39	**4.59**	5.95	6.29	14.33	*4.68*	∗10^09
12	Tox	4928.50	399.50	**1.35**	1115.30	2289.20	4826.00	*808.68*	∗10^11
	DARWIN	3.39	4.99	**2.40**	2.86	2.96	3.39	*2.42*	∗10^13
	Driv	3.70	6.72	4.28	**3.58**	3.66	3.70	*3.60*	∗10^04
	arcene	1.46	3.50	1.45	**1.40**	1.44	1.46	*1.41*	∗10^10
	USPS	6.47	8.42	**6.41**	6.44	6.46	6.47	*6.43*	∗10^04
	TUAN	1.45	3.15	**1.12**	*1.28*	1.57	1.43	1.36	∗10^04
	isolet	5.17	6.27	**5.09**	*5.12*	5.16	5.16	5.13	∗10^05
	Liver	1.26	2.25	**1.13**	*1.16*	1.19	1.26	*1.14*	∗10^05
	Ionosphere	1.60	2.11	**1.48**	1.56	*1.51*	1.59	**1.48**	∗10^03
	Page	13.87	4.04	**3.49**	4.82	5.86	13.88	*4.14*	∗10^09

Note: [2]The meaning of "scaling value" is that, for example, 10^14 indicates that the values in this row are multiplied by 10^14.

Table 3: Comparison of the *E* value between various optimization algorithms and K-means algorithm

k	Split [23]	RS [24]	IKM [25]	CDKM [21]	KWNF [26]	CDKMSM
4	−37.70%	9.01%	1.75%	4.88%	−0.53%	10.25%
6	−60.79%	15.05%	8.95%	5.35%	0.70%	14.75%
8	−72.56%	20.35%	15.14%	11.72%	0.45%	20.31%
10	−75.73%	21.95%	16.72%	12.42%	0.14%	20.67%
12	−38.66%	23.19	18.88%	12.88%	0.29%	21.27%
Mean	−57.09%	17.91%	12.29%	9.45%	0.21%	17.45%

In Table 2, as the number of clusters increases, the sum of squared errors (SSE) typically decreases because more clusters can better capture the data characteristics, resulting in a reduced distance between data points and their cluster centers. Fig. 6 represents the comparison of the *E* value between each optimization algorithm and the K-means algorithm. Due to the relatively lower solution accuracy of the split algorithm compared to the other algorithms, it was not included in the comparison in Fig. 6. The *E* value of the CDKM algorithm is 9.45%, the *E* value of split algorithm, I-K-means-+ algorithm, and KWNF algorithm are −57.09%, 12.29%, and 0.21%, respectively, the E value of the proposed algorithm is 17.45%. The E value of the proposed algorithm is 11.29% with respect to the CDKM algorithm, 35.61%, 7.87%, and 17.39% with respect to the split algorithm, I-K-means-+algorithm, and KWNF algorithm, respectively. As the number of clusters increases, the E value of the proposed algorithm compared to the K-means algorithm gradually increases, indicating that the improvement in the SSE relative to K-means and other tested algorithms apart from the random swap algorithm becomes more and more significant. The solution accuracy of the proposed algorithm has achieved an improvement over the K-means algorithm and other tested K-means optimization algorithms apart from the random swap algorithm. Compared to the K-means algorithm, the E value of the random swap algorithm is 17.91%, which is slightly higher than the E value of the proposed algorithm, which is 17.45%.

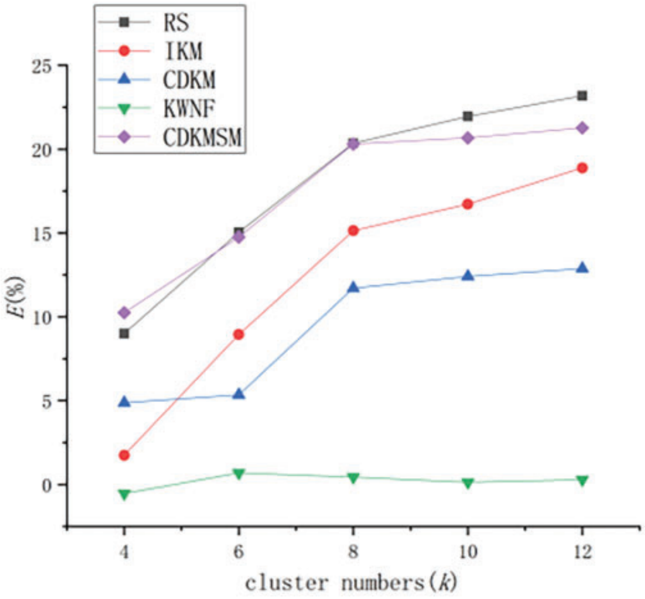

Figure 6: Comparison of the E value between each optimization algorithm and K-means algorithm

The runtime results of the algorithm are shown in Table 4 with the optimal and sub-optimal results shown bolded as well as skewed, respectively. Since the split and random swap algorithms were run on Linux system, we did not compare their execution time. As the number of clusters increases, the time gradually increases. Fig. 7 represents the percentage speedup of our proposed algorithm with other algorithms, for the higher dimensional datasets (first seven datasets) at $k = 4, 6, 8, 10, 12$. The T values of our proposed algorithm compared to another split-merge based K-means improved algorithm I-K-means-+ algorithm are 9.16%, 32.69%, 44.65%, and 50.96%, respectively, 50.51%. Compared to the I-K-means-+ algorithm, the algorithm in this paper operates more efficiently as the value of k increases. Comparing the T value with K-means algorithm and K-means with new formulation algorithm is 11.60% and 89.23%, respectively. As the number of clusters increases, the T value of the proposed algorithm compared to other tested algorithms becomes increasingly larger. It can be concluded that the proposed algorithm still maintains the efficiency advantage of CDKM algorithm in high dimensional data.

Table 4: Comparison for time

k	Dataset	Algorithm time					
		KM	IKM [25]	CDKM [21]	KWNF [26]	CDKMSM	Scaling value
4	Tox	*6.76*	8.38	**5.66**	49.74	8.14	∗10^00
	DARWIN	*3.00*	3.48	**2.18**	14.10	3.66	∗10^00
	Driv	*1.70*	1.93	**1.27**	16.06	2.02	∗10^2
	arcene	*4.29*	5.48	**3.16**	30.41	4.58	∗10^02
	USPS	4.44	4.67	**2.62**	49.04	*3.79*	∗10^01
	TUAN	*5.70*	7.72	**4.22**	71.85	8.58	∗10^01
	isolet	6.94	8.86	**2.69**	75.71	*4.72*	∗10^02
	Liver	**1.40**	*2.80*	16.40	25.60	27.00	∗10^−01
	Ionosphere	**4.80**	*6.40*	12.00	34.80	25.40	∗10^−01
	Page	**3.88**	*4.52*	25.02	69.18	50.48	∗10^00
6	Tox	15.00	18.36	**8.04**	127.56	*12.14*	∗10^00
	DARWIN	*5.82*	6.96	**3.56**	29.32	6.18	∗10^00
	Driv	2.85	3.78	**1.70**	25.45	*2.75*	∗10^02
	arcene	8.01	14.99	**4.34**	58.69	*7.25*	∗10^02
	USPS	8.22	8.80	**3.55**	63.08	*5.42*	∗10^01
	TUAN	*8.18*	11.18	**5.23**	11.95	10.42	∗10^01
	isolet	9.38	14.73	**3.53**	90.06	*5.97*	∗10^02
	Liver	**2.80**	*3.80*	22.20	39.80	43.40	∗10^−01
	Ionosphere	**9.20**	*12.20*	19.20	66.00	34.20	∗10^−01
	Page	**9.72**	*16.36*	67.66	203.16	136.88	∗10^00
8	Tox	3.13	4.47	**1.38**	27.99	*2.05*	∗10^01
	DARWIN	8.18	9.72	**4.52**	40.48	*7.70*	∗10^00
	Driv	*3.31*	5.38	**2.14**	28.97	3.50	∗10^02
	arcene	11.79	23.04	**5.43**	89.00	*9.15*	∗10^02
	USPS	11.44	12.66	**5.27**	82.45	*7.40*	∗10^01
	TUAN	*12.06*	17.78	**6.77**	192.50	12.31	∗10^01
	isolet	13.14	26.22	**5.22**	132.27	*7.81*	∗10^02
	Liver	**3.40**	*7.60*	29.40	66.60	59.80	∗10^−01
	Ionosphere	**1.18**	*2.06*	2.50	8.66	4.46	∗10^00
	Page	**1.95**	*2.99*	18.24	42.35	30.76	∗10^01
10	Tox	2.82	4.95	**1.33**	26.39	*2.22*	∗10^01
	DARWIN	10.30	13.30	**5.46**	52.26	*9.22*	∗10^00
	Driv	5.03	8.72	**2.69**	48.49	*4.23*	∗10^02
	arcene	14.34	32.16	**6.63**	129.26	*11.25*	∗10^02
	USPS	15.56	18.90	**7.08**	121.81	*10.08*	∗10^01
	TUAN	15.49	24.56	**8.63**	338.22	*15.43*	∗10^01
	isolet	14.83	34.47	**5.89**	161.62	*10.16*	∗10^02
	Liver	**5.00**	*9.20*	34.80	77.60	64.20	∗10^−01
	Ionosphere	**1.30**	*2.90*	3.00	9.78	5.74	∗10^00

(Continued)

Table 4 (continued)

k	Dataset	Algorithm time					
		KM	IKM [25]	CDKM [21]	KWNF [26]	CDKMSM	Scaling value
	Page	**3.28**	*4.69*	27.60	76.59	44.42	∗10^01
12	Tox	3.09	5.59	**1.57**	25.54	*2.54*	∗10^01
	DARWIN	*10.66*	14.86	**6.30**	51.76	10.92	∗10^00
	Driv	5.80	10.67	**3.02**	45.26	*4.72*	∗10^02
	arcene	18.43	35.97	**8.44**	158.30	*13.77*	∗10^02
	USPS	16.51	21.19	**7.12**	109.16	*11.18*	∗10^01
	TUAN	18.67	27.45	**9.97**	394.83	*17.46*	∗10^01
	isolet	20.07	39.58	**7.02**	192.58	*11.28*	∗10^02
	Liver	**5.20**	*12.00*	41.60	88.60	81.00	∗10^−01
	Ionosphere	**1.78**	*3.30*	3.42	11.26	6.72	∗10^00
	Page	**5.56**	*7.62*	33.50	145.30	63.64	∗10^01

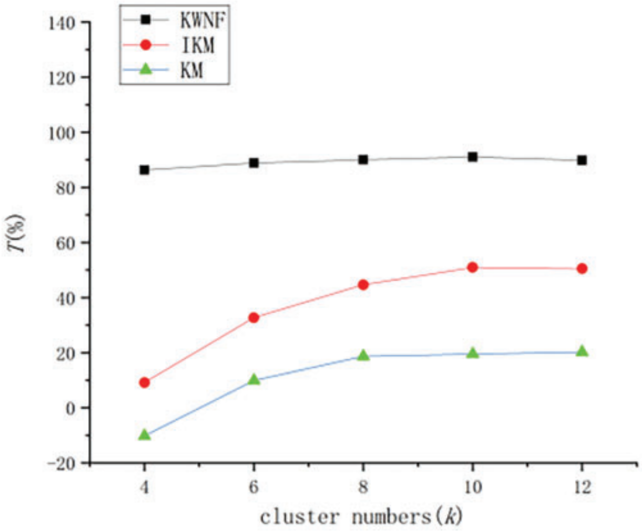

Figure 7: Comparison of *T* value between the proposed algorithm and other algorithms under high dimensional data

The experimental results show that the solution accuracy of the proposed method is higher than that of the K-means algorithm and the two algorithms based on split-merge, namely split and I-K-means-+. The proposed algorithm improves the solution accuracy of CDKM while retaining its computational efficiency advantage by using partition matrix under high dimensional data. Additionally, it outperforms the recently proposed KWNF algorithm. However, its solution accuracy is slightly lower than that of the random swap algorithm. The random swap algorithm is a variant of the split-merge criterion that incorporates the swap operation into the traditional split-merge criterion. This suggests that the swap operation can effectively improve the solution accuracy, which is a key focus of our future research.

5 Conclusions

The CDKM algorithm employs a coordinate descent method to optimize the K-means model and improve the solution accuracy of the original K-means algorithm. However, it is sensitive to the initial centers, which makes its solution accuracy not high enough. In this paper, the iterative process of CDKM is modified, and a coordinate descent K-means algorithm based on Split-merge is proposed. The proposed algorithm obtains the partition matrix by CDKM. Then the partition matrix is optimized by the proposed split-merge criterion to improve the solution accuracy. Since the introduced split-merge operations are completed entirely using the partition matrix, which avoids the distance calculation in traditional K-means and related improved algorithms. So, like CDKM, it has an efficiency advantage over other algorithms on high dimensional datasets.

The experimental results show that in terms of solution accuracy. The *E* value of the proposed algorithm relative to K-means is 17.45%, which is higher than that of the CDKM algorithm (9.45%), and also higher than that of the I-K-means-+ algorithm, split algorithm, and KWNF algorithm. This result indicates that incorporating basic split-merge criterion into the CDKM algorithm can achieve better solution accuracy than traditional split-merge algorithm. We also observed that a variant of the split-merge algorithm, known as the random swap algorithm, has a slight *E* value advantage (17.91%) over the proposed algorithm (17.45%), which is significantly higher than the other two algorithms, the I-K-means-+ algorithm and the split algorithm, which are based on the basic split-merge criterion. This suggests that the swap operation plays a significant role in further enhancing the solution accuracy of the split-merge criterion. As the number of clusters increases, the proposed algorithm shows increasingly greater improvements in the SSE index compared to both K-means and other tested algorithms apart from the random swap algorithm. In terms of computational efficiency, the proposed algorithm is more efficient on high dimensional data than another split-merge based algorithm, i.e., I-K-means-+ algorithm, and it is also more efficient than other tested algorithms. The percentage of time speedup for the proposed algorithm relative to the time of I-K-means-+, K-means, and the latest KWNF algorithm is 37.59%, 11.60%, and 89.23%, respectively. As the number of clusters increases, the *T* value of the proposed algorithm gradually increases compared to other algorithms, indicating that its time efficiency improves progressively.

The split-merge criterion can significantly improve the solution accuracy of the K-means algorithm. CDKM model is fundamentally different from the K-means model, direct application of split-merge criterion is not feasible. Therefore, this paper explores the integration of the split-merge criterion into the CDKM clustering model to enhance its solution accuracy. We aim to apply the split-merge criterion, traditionally used in the K-means model, to the CDKM model. In traditional K-means, there are several methods for split-merge operations and subsequent improvements, such as random swap clustering [24], which effectively enhance the solution accuracy of the original K-means model. These represent further advancements built upon the foundation of split-merge. Some effective indexes can measure clustering performance. In our future work, we plan to explore the integration of these advanced methods and indexes into our research. Our goal is to demonstrate the successful application of these methods and to explore the potential of incorporating such traditional methods into the CDKM model in future research.

Acknowledgement: The authors would like to acknowledge the valuable feedback provided by the reviewers.

Funding Statement: This research was funded by National Defense Basic Research Program, grant number JCKY2019411B001. This research was funded by National Key Research and Development

Program, grant number 2022YFC3601305. This research was funded by Key R&D Projects of Jilin Provincial Science and Technology Department, grant number 20210203218SF.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Fuheng Qu, Yuhang Shi; data collection: Yuhang Shi; analysis and interpretation of results: Yuhang Shi, Yong Yang, Yating Hu; draft manuscript preparation: Yuhang Shi, Yuyao Liu. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: All data used in this study are freely available and accessible. The sources of the data utilized in this research are thoroughly explained in the main manuscript.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] S. Jha, G. P. Joshi, L. Nkenyereya, D. W. Kim, and F. Smarandache, “A direct data-cluster analysis method based on neutrosophic set implication,” *Comput. Mater. Contin.*, vol. 65, no. 2, pp. 1203–1220, 2020. doi: 10.32604/cmc.2020.011618.

[2] G. J. Oyewole and G. A. Thopil, “Data clustering: Application and trends,” *Artif. Intell. Rev.*, vol. 56, no. 7, pp. 6439–6475, 2023. doi: 10.1007/s10462-022-10325-y.

[3] S. K. Dubey, S. Vijay, and A. Pratibha, “A review of image segmentation using clustering methods,” *Int. J. Appl. Eng. Res.*, vol. 13, no. 5, pp. 2484–2489, 2018.

[4] M. Jamjoom, A. Elhadad, H. Abulkasim, and S. Abbas, “Plant leaf diseases classification using improved K-means clustering and SVM algorithm for segmentation,” *Comput. Mater. Contin.*, vol. 76, no. 1, pp. 367–382, 2023. doi: 10.32604/cmc.2023.037310.

[5] R. Aarthi, K. Divya, N. Komala, and S. Kavitha, “Application of feature extraction and clustering in mammogram classification using support vector machine,” in *2011 Third Int. Conf. Adv. Comput.*, IEEE, 2011, pp. 62–67.

[6] M. Ahmed, R. Seraj, and S. M. S. Islam, “The *k-means* algorithm: A comprehensive survey and performance evaluation,” *Electronics*, vol. 9, no. 8, 2020, Art. no. 1295. doi: 10.3390/electronics9081295.

[7] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, “K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data,” *Inf. Sci.*, vol. 622, pp. 178–210, 2023. doi: 10.1016/j.ins.2022.11.139.

[8] A. K. Jain, “Data clustering: 50 years beyond K-means,” *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010. doi: 10.1016/j.patrec.2009.09.011.

[9] S. S. Khan and A. Ahmad, “Cluster center initialization algorithm for K-means clustering,” *Pattern Recognit. Lett.*, vol. 25, no. 11, pp. 1293–1302, 2004. doi: 10.1016/j.patrec.2004.04.007.

[10] M. E. Celebi, H. A. Kingravi, and P. A. Vela, “A comparative study of efficient initialization methods for the k-means clustering algorithm,” *Expert Syst. Appl.*, vol. 40, no. 1, pp. 200–210, 2013. doi: 10.1016/j.eswa.2012.07.021.

[11] M. Gul and M. A. Rehman, “Big data: An optimized approach for cluster initialization,” *J. Big Data*, vol. 10, no. 1, pp. 120, 2023. doi: 10.1186/s40537-023-00798-1.

[12] T. K. Biswas, K. Giri, and S. Roy, “ECKM: An improved K-means clustering based on computational geometry,” *Expert Syst. Appl.*, vol. 212, 2023, Art. no. 118862. doi: 10.1016/j.eswa.2022.118862.

[13] A. Layeb, “*ck*-means and *fck*-means: Two deterministic initialization procedures for K-means algorithm using a modified crowding distance,” *Acta Inform. Prag.*, vol. 12, no. 2, pp. 379–399, 2023. doi: 10.18267/j.aip.223.

[14] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," *Soda*, vol. 7, pp. 1027–1035, 2007.

[15] S. Lattanzi and C. Sohler, "A better k-means++ algorithm via local search," in *Int. Conf. Mach. Learn.*, PMLR, 2019, pp. 3662–3671.

[16] A. Şenol, "ImpKmeans: An improved version of the K-means algorithm, by determining optimum initial centroids, based on multivariate kernel density estimation and Kd-tree," *Acta Polytechnica Hungarica*, vol. 21, no. 2, pp. 111–131, 2024. doi: 10.12700/APH.21.2.2024.2.6.

[17] D. Reddy, P. K. Jana, and I. S. Member, "Initialization for K-means clustering using Voronoi diagram," *Procedia Technol.*, vol. 4, pp. 395–400, 2012. doi: 10.1016/j.protcy.2012.05.061.

[18] E. Baratalipour, S. J. Kabudian, and Z. Fathi, "A new initialization method for k-means clustering," in *2024 20th CSI Int. Symp. Artif. Intell. Signal Process. (AISP)*, IEEE, 2024, pp. 1–5.

[19] S. J. Wright, "Coordinate descent algorithms," *Math. Program.*, vol. 151, no. 1, pp. 3–34, 2015. doi: 10.1007/s10107-015-0892-3.

[20] H. -J. M. Shi, S. Tu, Y. Xu, and W. Yin, "A primer on coordinate descent algorithms," 2016, *arXiv:1610.00040*.

[21] F. Nie, J. Xue, D. Wu, R. Wang, H. Li and X. Li, "Coordinate descent method for k-means," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 5, pp. 2371–2385, 2021. doi: 10.1109/TPAMI.2021.3085739.

[22] T. Kaukoranta, P. Fränti, and O. Nevalainen, "Iterative split-and-merge algorithm for VQ codebook generation," *Opt. Eng.*, vol. 37, no. 10, pp. 2726–2732, Oct. 1998.

[23] P. Fränti, T. Kaukoranta, and O. Nevalainen, "On the splitting method for VQ codebook generation," *Opt. Eng.*, vol. 36, no. 11, pp. 3043–3051, Nov. 1997.

[24] P. Fränti, "Efficiency of random swap clustering," *J. Big Data*, vol. 5, no. 13, pp. 1–29, 2018.

[25] H. Ismkhan, "Ik-means−+: An iterative clustering algorithm based on an enhanced version of the *k*-means," *Pattern Recognit.*, vol. 79, pp. 402–413, 2018.

[26] F. Nie, Z. Li, R. Wang, and X. Li, "An effective and efficient algorithm for k-means clustering with new formulation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3433–3443, 2022. doi: 10.1109/TKDE.2022.3155450.