Research Article

A Fast Multiscale Clustering Approach Based on DBSCAN

Runzi Chen ¹, Shuliang Zhao ^{2,3,4} and Meishe Liang ⁵

¹School of Mathematical Sciences, Hebei Normal University, Shijiazhuang 050024, China

²College of Computer and Cyber Security, Hebei Normal University, Shijiazhuang 050024, China

³Hebei Provincial Key Laboratory of Network and Information Security, Shijiazhuang 050024, China

⁴Hebei Provincial Engineering Research Center for Supply Chain Big Data Analytics & Data Security, Shijiazhuang 050024, China

⁵Department of Mathematics and Physics, Shijiazhuang Tiedao University, Shijiazhuang 050043, China

Correspondence should be addressed to Shuliang Zhao; 16120379@bjtu.edu.cn

Received 24 June 2021; Revised 6 July 2021; Accepted 17 July 2021; Published 1 August 2021

Academic Editor: Jian Su

Copyright © 2021 Runzi Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Multiscale brings great benefits for people to observe objects or problems from different perspectives. It has practical significance for clustering on multiscale data. At present, there is a lack of research on the clustering of large-scale data under the premise that clustering results of small-scale datasets have been obtained. If one does cluster on large-scale datasets by using traditional methods, two disadvantages are as follows: (1) Clustering results of small-scale datasets are not utilized. (2) Traditional method will cause more running overhead. Aims at these shortcomings, this paper proposes a multiscale clustering framework based on DBSCAN. This framework uses DBSCAN for clustering small-scale datasets, then introduces algorithm Scaling-Up Cluster Centers (SUCC) generating cluster centers of large-scale datasets by merging clustering results of small-scale datasets, not mining raw large-scale datasets. We show experimentally that, compared to traditional algorithm DBSCAN and leading algorithms DBSCAN++ and HDBSCAN, SUCC can provide not only competitive performance but reduce computational cost. In addition, under the guidance of experts, the performance of SUCC is more competitive in accuracy.

1. Introduction

Clustering is one of the vital data mining and machine learning techniques and that aims to group similar objects into the same cluster and separate dissimilar objects into different clusters [1]. It is so prominent and recently attracted significant attention of researchers and practitioners from different domains of science and engineering [2]. Thousands of papers have been published [3–6]. However, these investigations concentrate on clustering from a single perspective.

The scale can be equivalent to the following concept: generic concept, level of abstraction or perspective of observation, the same problem, or system can be perceived at different scales based on particular needs [7]. That is called multiscale phenomena and has been widely applied to the academic fields, such as geoscience [8, 9] and mathematics [10]. Multiscale clustering has gained popularity in recent years. However, these studies either focus on specific applica-

tion scenarios [11–13] or rely on specific clustering methods [14–16], and they perform clustering based on raw data, which is highly time consuming. Figure 1 shows two implementations of multiscale clustering, which can help users conduct clustering analysis on multiscale datasets. The straightforward way of obtaining the knowledge of every scale is to apply the traditional mining algorithms to each scale dataset, which is very inefficient. In this study, we concentrate on Scaling-Up Cluster Centers (SUCC) from small scale to large scale and avoid repetitive clustering on raw datasets, with competitive efficiency.

1.1. Motivations. The multiscale phenomenon is a widely accepted concept that helps people understand problems from different levels of abstraction. Clustering analysis on multiscale datasets considerably influences knowledge discovery when capturing the essence of the problem. For example, assume that a telecommunication supplier has many

FIGURE 1: Two implementing measures for multiscale data mining.

filiales, and each filiale accomplishes clustering analysis for its customers. After that, the controlling corporation also needs to conduct clustering analysis for all customers. It is inefficient that clustering at each scale data by using the traditional method, i.e., reclustering, while SUCC can improve efficiency by computing cluster centers belonged small-scale data and obtaining large scale's clusters. In addition, the concept of "fine-grained and more fine-grained" in the RFID domain [17–19] and the strategy of dividing and conquering, i.e., separating tags into many small subsets, then handling in sequence [20–23], can motivate our work.

1.2. Contributions. We provide a mathematical model (multi-scale clustering framework), then design a novel algorithm named Scaling-Up Cluster Centers (SUCC) from small scale to large scale, which avoids repetitive clustering on raw datasets.

We conducted extensive experiments to evaluate our work. Experimental results show that the SUCC is efficient and reduces runtime consumption with competitive accuracy compared to traditional methods and the leading algorithms, which need to deal with raw data that is much more than cluster centers belonged small-scale data in most instances.

1.3. Structure of the Paper. The rest of this paper is organized as follows. Section 2 discusses the related work. The problem is described detailedly in Section 3. The framework of multi-scale clustering and algorithm SUCC is designed in Section 4. Section 5 details the comparison experiments, and Section 6 concludes our work and provides some future research directions.

2. Related Work

Clustering has not been a precise definition; there are a lot of different clustering approaches towards various techniques.

In this section, we review the clustering approaches in two aspects.

2.1. Conventional Clustering Techniques. Han et al. suggested the following four categories for dividing clustering techniques: partitioning methods, hierarchical methods, density-based methods, and grid-based methods [24].

Partitioning method assigns a dataset into k clusters such that each cluster must contain at least one element. k -means algorithm proposed by MacQueen in 1967 is the most classical representation of Partitioning method [25], and that is one of the best-known and simplest clustering algorithms [26]. Dunn developed Fuzzy c -means algorithm (FCM) for clustering data in 1973 [27]. FCM [28] allows one element to belong to two or more clusters unlike k -means where only one cluster is assigned to each element and improved by Bezdek in 1981. However, the time complexity of k -means is much less than that of FCM; thus, k -means work faster than FCM [29]. The advantages of partitioning algorithms include that they are (1) relatively scalable and simple and (2) suitable for finding spherical-shaped clusters. However, disadvantages of these algorithms include (1) the necessity for the user to specify the number of clusters in advance; (2) high sensitivity to value of initialization, noise, and outliers; and (3) failure to deal with nonconvex clusters [30].

Hierarchical clustering can be divided into agglomerative method and divisive method [31]. The former follows the bottom-up strategy, which generates clusters starting with single element and then merging these small clusters into larger and larger clusters, until all of the elements are in a single cluster or otherwise until certain termination conditions are satisfied. The latter follows the top-down strategy, which breaks up cluster placed all objects into smaller and smaller clusters, until each cluster containing only one element or until certain termination conditions are satisfied. There are three forms of the agglomerative method, namely, single-

linkage clustering [32], complete-linkage clustering [33], and average-linkage clustering [34]. After that, some enhanced hierarchical clustering methods are introduced, such as BIRCH [35, 36], CURE [37], ROCK [38], and CHAMELEON [39].

Ester et al. demonstrated an algorithm called DBSCAN (density-based spatial clustering of applications with noise) [40], which discovers clusters of arbitrary shapes and is efficient for large spatial databases. Other well-known density-based techniques [41] are as follows: SNOB proposed by Wallace and Dowe in 1994 and MCLUST [42] introduced by Fraley and Raftery in 1998. Among these methods, the expectation-maximization (EM) algorithm is the most popular [41, 43].

These methods partition the space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. The main advantage of the approach is its fast processing time [24], no need of distance computations, and easy to determine which clusters are neighbouring. There are many other interesting grid-based techniques including STING (statistical information grid approach) [44] by Wang et al. in 1997, one of the highly scalable algorithm and has the ability to decompose the dataset into various levels of detail. CLIQUE [45] is developed by Agrawal et al. in 1998, which can be considered as both density-based and grid-based clustering methods. It automatically finds subspaces of high dimensional data space that allows better clustering than original space. The accuracy of the clustering result may be degraded at the expense of simplicity of the method CLIQUE.

2.2. Multilevel Clustering. In recent years, the researchers introduced the idea of stratification into clustering. Psomopoulos et al. presented a multiple level clustering algorithm for detecting inter- and intragenome gene clusters, by introducing the notion of a hierarchy. This method is experimentally proven to be consistent with the phylogenetic relation and position of the genes involved [46]. Vu et al. developed a new multithreaded tool, fMLC, addressed the problem of clustering large-scale DNA sequences [12]. A multilevel clustering for star/galaxy separation was designed in 2016, consisting of three phases: coarsening clustering, representative data clustering, and merging [47]. In 2019, Zunic et al. proposed a multilevel clustering algorithm that is used on Internal Banking Payment System in a bank of Bosnia and Herzegovina and explained how the parameter effects on results and execution time in algorithm [13]. [11] presented a multilevel clustering approach that addressed the issue that conventional algorithms spent too much execution time clustering physical activity data. [48] proposed an MCT to identify local communities termed microcosms this year. These algorithms all aim to a specific application and solve the corresponding problem. To settle the problem of clustering partial labeled data, Liu et al. proposed a feasible solution [49]. [16] applied multilevel to spectral clustering and proposed multilevel approximate spectral clustering method, while Nouredinov et al. applied multilevel to conformal clustering and proposed MLCC [15]. [14] improved the synchronization clustering by using an idea of “divide and collect” and

presented MLSynC. One of these algorithms improved a specific algorithm by introducing the idea of multilevel. This paper proposes a general framework that transfers cluster center from small scale to large scale, which is independent of the conventional algorithm.

3. Problem Description

To facilitate the discussion in the remainder of the paper, we outline some basic concepts in this section. Formally, let a dataset of n objects $X = \{x_1, x_2, \dots, x_n\}$, $x_i = \{I_{i1}, I_{i2}, \dots, I_{id}\} \in A$ be a data object, and each object has d attributes denoted by A . Clustering is the process of distributing the objects in X into k clusters, $c_1, c_2, \dots, c_k$, that is $c_i \cap D$ and $c_i \cap c_j = \emptyset$ for $(1 \leq i, j \leq k)$.

Definition 1. Scale partition: a set can be divided into one or more than one group by some scale, and each object has same scale value.

Example 1. There is “mod 2” scale on X , and it can divide X into $X_{\text{mod } 2=1} = \{x_1, x_3, \dots, x_{2m-1}\}$, and $X_{\text{mod } 2=0} = \{x_2, x_4, \dots, x_{2m}\}$, where $m = n/2$. So, scale partition of X is defined as follows: $X/\text{mod } 2 = \{X_{\text{mod } 2=1}, X_{\text{mod } 2=0}\}$.

Definition 2. Multiscale datasets: if there are multiple scales on a dataset, these scales have a partial order relation, denoted by “ $<$.” Multiscale datasets consist of multiple partitions set divided by multiple scales.

Example 2. Let π_1 be mod 4, π_2 be mod 2, and π_3 be mod 1; then, $\pi_1 < \pi_2 < \pi_3$, where π_1 is small scale, π_3 is large scale, and π_2 is small scale about to π_3 and large scale about to π_1 . Multiscale dataset on X is $\{X/\pi_1, X/\pi_2, X/\pi_3\}$, where $X/\pi_1 = \{\{x_1, x_5, \dots, x_{4m-3}\}, \{x_2, x_6, \dots, x_{4m-2}\}, \{x_3, x_7, \dots, x_{4m-1}\}, \text{ and } \{x_4, x_8, \dots, x_{4m}\}\}$; $X/\pi_2 = \{\{x_1, x_3, \dots, x_{2m-1}\}, \{x_2, x_4, \dots, x_{2m}\}\}$; and $X/\pi_3 = \{\{x_1, x_2, \dots, x_n\}\}$.

Definition 3. Similarity: let x_i and x_j be two elements of X or the cluster centers; then, the similarity between x_i and x_j is defined as follows:

$$S(x_i, x_j) = \frac{1}{\left(1 + \sum_{t=1}^{|A|} w_t |x_{it} - x_{jt}|\right)}, \quad (1)$$

where $|A|$ represents for the number of attribute and is equal to d and w_t is the weight of t -th attribute.

Definition 4. The estimated clusters of large scale: assume that $c_1, c_2, \dots$, and c_p are p cluster centers of small scales, $1 \leq i, j \leq p$, and satisfy $S(c_i, c_j) \geq \delta$. The objects belonged these two clusters of small scale. The cluster centers of which are named c_i and c_j together constitute the estimated clusters of large scale.

From Figure 1, k_2 can be obtained by two mathematical models: (1) $k_2 = M(D_2)$, $D_2 = T(D_1)$, and (2) $k_2 = T(k_1)$, k_1

$= M(D_1)$, where $T(D_1)$ is scale partition, $T(k_1)$ is transferring knowledge from small scale, and M is mining knowledge from dataset. In multiscale data mining, $M(D_1)$ is assumed to be accomplished. So, we apply the second model to mine the knowledge on large-scale dataset.

4. Proposed Framework

In this section, we firstly propose the framework of multiscale clustering; then, the scaling-up algorithm is designed to implement our framework; an example is given to explain our work in the end.

4.1. Framework. In this paper, cluster centers are used as a medium for knowledge transformation from small scales to large scales. The framework of multiscale clustering is shown in Figure 2. It calculates cluster centers of large scale according to cluster centers of small scales rather than clustering on large-scale datasets repeatedly. The process is listed as follows: (1) construct multiscale datasets, (2) clustering on small-scale datasets using conventional method, (3) compute the similarity among cluster centers of small scale and construct relation matrix of cluster centers, (4) found relation the graph of cluster centers, and (5) produce connected component of the graph. In particular, it applies formula (2) to found the graph of cluster centers,

$$m_{ij} = \begin{cases} 1 & \text{if } S(x_i, x_j) \geq \delta \\ 0 & \text{otherwise} \end{cases}, \quad (2)$$

where $S(x_i, x_j)$ is computing according (1).

4.2. Design. Multiscale clustering uses three algorithms to discuss its workflow. Algorithm 1 and Algorithm 2 are invoked sequentially, where algorithm 2 uses Algorithm 3 to obtain the connected component. In Algorithm 1, variable lengthD and label_s represent the length of original dataset D and the predicted label of small scale, where its length is lengthD (line 1). Dataset is divided into multiple scales, and the smallest scale datasets will be cluster (line 2). Then, it clusters on every partition of the scale by using conventional method; the result is cluster center list and their corresponding cluster label list (lines 3-6); in particular, the label value of list_{i+1} starts from the label value of list_i + cluster's number of D_{ssi} .

Algorithm 2 produces clusters of large scale. The p -centers is the combination of all cluster centers belonged small scale, and center_dic is dictionary and used to record cluster center and its corresponding cluster label list (line 1). The mapping relationship between cluster center and its clusters' object index list is established (lines 2-9). Then, the relation matrix is computed (lines 10-12). Finally, it returns the clustering result (cluster label) of large scale, where getConnectCompont is invoked to produce new cluster by merging the close cluster of small scale (lines 13-22).

Algorithm 3 applies the mind of deep first search to relation matrix M and generates the cluster center list of which

have close relation, and these indexes form a new cluster center of large scale.

4.3. Analysis. In this section, we analyze execution time and memory consumption in detail. It should be noticed that the characteristic of our framework is that clustering of large scale is based on cluster center of small scale not primitive large-scale dataset. So, we need to consider Algorithm 2 that needs invoke Algorithm 3. Algorithm 3 computes one point's connected component using recursion; let m be the number of small-scale cluster center, then both the cost of its time and memory are $O(m)$. There are three main works in Algorithm 2 as follows: (1) construct the dictionary that is the relation between cluster center and the index list of data elements belonged that cluster, (2) compute the relation matrix M of small-scale cluster center, and (3) generate the clustering result of large scale. Let the number of D be n , their time complexity is $O(nm)$, $O(m^2)$, and $O(nm)$, respectively, and their space complexity is $O(n)$, $O(m^2)$, and $O(m)$. In general, m is far less than n , so time complexity of Scaling-Up Cluster Centers is $O(nm)$, and space complexity is $O(n)$ that is used to construct the dictionary.

4.4. Case Study. This section describes the framework using the dataset D that has two scale, where $D = [(0, 2), (0, 1), (1, 2), (1, 1), (2, 3), (8, 9), (7, 8), (9, 9), (8, 8), (7, 9), (4, 4), (4, 5), (5, 6), (6, 5)]$. The small scale divides D into two partitions denoted by D_{s1} and D_{s2} , and the large scale divides D into only one partition denoted by D_l , where $D_{s1} = [(0, 2), (0, 1), (1, 2), (1, 1), (2, 3), (8, 9), (7, 8)]$, $D_{s2} = [(9, 9), (8, 8), (7, 9), (4, 4), (4, 5), (5, 6), (6, 5)]$, and $D_l = D$. Let parameter δ be 0.6. The remainder of process performs as follows:

- (1) We apply DBSCAN algorithm to D_{s1} and D_{s2} , where parameters eps and min_samples both are 2. The result of clustering is as follows: $\text{label}_{s1} = [0\ 0\ 0\ 0\ 1\ 1]$, $\text{label}_{s2} = [0\ 0\ 0\ 1\ 1\ 1]$, and $\text{label}_s = [0\ 0\ 0\ 0\ 1\ 1\ 2\ 2\ 2\ 3\ 3\ 3]$. There are four clusters denoted by $\{[0.8, 1.8]: [0, 1, 2, 3, 4], [7.5, 8.5]: [5, 6], [8.0, 8.667]: [7-9], [4.75, 5.0]: [10-13]]\}$.
- (2) The similarity of four cluster centers is computed as follows:

$$\begin{bmatrix} 1.0 & 0.095 & 0.091 & 0.164 \\ 0.095 & 1.0 & 0.655 & 0.183 \\ 0.091 & 0.655 & 1.0 & 0.170 \\ 0.164 & 0.183 & 0.170 & 1.0 \end{bmatrix}. \quad (3)$$

- (3) After setting $\delta = 0.6$, the cluster centers' relation graph is constructed. Then, we combine the second and third cluster

FIGURE 2: Overview of multiscale clustering.

Input: Original dataset D

Output: Cluster center list $(cs_1, cs_2, \dots, cs_p)$, clusters list of small scale $(label_s)$

1. $lengthD \leftarrow \text{len}(D)$, $label_s \leftarrow NULL$
2. divide data set into multiple scales, smallest scale data is represented as D_{ss} , P is the number of data set on D_{ss}
3. for each dataset D_{ssi} do
4. $c_p, list_i \leftarrow \text{clustering on } D_{ssi}$ // the label value of $list_{i+1}$ starts from the label value of $list_i$ + cluster's number of D_{ssi}
5. $label_s \leftarrow label_s + list_i$
6. end for

ALGORITHM 1: Clustering on small-scale dataset.

Input: Cluster center list $(cs_1, cs_2, \dots, cs_p)$, clusters list of small scale $(label_s)$, parameter δ

Output: Clusters list of large scale $(label_l)$

1. $p_centers \leftarrow cs_1 + cs_2 + \dots + cs_p$, $center_dic \leftarrow NULL$
2. for each i in $\text{len}(p_centers)$ do
3. $index \leftarrow NULL$
4. for each j in $\text{len}(label_s)$ do
5. if $label_s[j] = i$ then
6. $index.add(j)$
7. end if
8. $center_dic[p_centers[i]] \leftarrow index$
9. end for
10. for each i, j in $\text{len}(p_centers)$ do
11. compute M_{ij} according to (2)
12. end for
13. $k \leftarrow 0$, $visitlabel[] \leftarrow 0$, $label_l[] \leftarrow NULL$
14. for each i in $\text{len}(label_s)$ do
15. if $visitlabel[i] = 0$ then
16. $temp \leftarrow \text{getConnectCompont}(M, i)$
17. for each x in $temp$ do
18. $label_l[x] \leftarrow k$
19. end for
20. $k \leftarrow k + 1$
21. end if
22. end for

ALGORITHM 2: Scaling-Up Cluster Centers.

Input: M, i : M is graph of cluster center relation, i is the point of beginning

Output: $list_i$: list of point connected to i

1. $list_i \leftarrow i$
2. for each j in $\text{len}(M)$ do
3. if $M_{ij} = 1$ and j not be handled then
4. $list_i \leftarrow j + list_i$
5. $list_i \leftarrow list_i + \text{getConnectCompont}(M, j)$
6. end for

ALGORITHM 3: getConnectCompont.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4)$$

- (4) The ultimate clustering result shows the label of large scale is [0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2, 2, 2], which is equivalent to the result that directly applies DBSCAN algorithm to the dataset of large scale and obtains, with more efficiency

5. Performance Evaluations

In this section, firstly we perform conventional algorithm DBSCAN on small-scale dataset and get clustering result, then run SUCC and obtain the clustering result belonged large-scale dataset. Secondly, we evaluate the accuracy and efficiency of SUCC by comparing it with DBSCAN [40] that is classical clustering algorithm, HDBSCAN [50] and DBSCAN++ [51], which are leading algorithms in the field of clustering and performed on large-scale dataset. HDBSCAN and SUCC have been implemented with python. DBSCAN comes from sklearn library. We directly use the relevant result of DBSCAN++ but proportionally calculate the run time by DBSCAN. These programs have been performed on a computer with a Inter(R) Core (TM) i7 Dual-Core 2.8GHz CPU, 16GB RAM, and the windows 10 × 64 Home operating system.

5.1. Data Preprocessing. There are 10 real datasets in this article, as shown in Table 1, where C is the number of clusters, A is the number of features, n is the number of transactions in D , and size represents dataset size. In this section, we firstly divide each dataset into two parts, then divide them into three parts. These two partitions are used two methods: the first is sequence, and the second is interval. That is to say, the first partition is dividing the data into two equal parts; one method (2A) is that the data is divide into former one second and latter one second. Another method (2B) is odd rows as the first partition and even rows as the second partition. The second method has also similar two methods (3A, 3B). It is straightforward that original data is large-scale data, and its partitions are small-scale data.

5.2. Performance Comparison. We evaluate the performance via two widely used clustering scores: the adjusted RAND index (ARI) [52] and adjusted mutual information score (AMI) [53], which are computed against the ground truth. We let min_samples be 10 for all procedures throughout experiments.

Comparison result of ARI values is shown in Table 2. SUCC (3A or 3B) surpasses the compared algorithms on 6 datasets, and the average ARI value of SUCC (3A or 3B) in 10 datasets is also the highest. We analyze that the distribution of 6 datasets (such as *Iris* and *Wine*) is relatively uni-

TABLE 1: Properties of the datasets used for experimental analysis.

NO	Name	C	A	n	Size (KB)
1	Iris	3	4	150	5
2	Wine	3	13	178	11
3	Spam	2	57	4601	687
4	Images	7	19	210	35
5	PCA_MNIST	10	20	60000	25803
6	Libras	15	90	360	251
7	Mobile	4	20	2000	120
8	Zoo	7	16	101	5
9	Seeds	7	19	210	10
10	Letters	26	16	20000	696

form, and the cluster densities among different divisions are similar, which is conducive to our algorithm. The results in Table 2 show that DBSCAN++ gives better performance compared to HDBSCAN and DBSCAN with respect to optimal parameter ϵ . In contrast, SUCC (2B or 3B) suffers more on *PCA_MNIST*, *Mobile*, *Zoo*, and *Letters*. The reason can be explained by that the distribution of the data divided by interval partition is quite different from that of the raw data, which causes SUCC to underperform. Therefore, our algorithm works better under the guidance of professional background.

The AMI values of four algorithms are shown in Table 3; the highest score of the row is in italic. We see that SUCC performs better than DBSCAN++ on 7 out of 10 datasets, while SUCC performs better than HDBSCAN on 9 out of 10 datasets, and SUCC performs better than DBSCAN on all datasets. It is obvious that the index AMI of algorithms is basically consistent with index ARI, except for on dataset *Letters*, where algorithm HDBSCAN has the best performance. An explanation of HDBSCAN has the highest score on *Letters*. It is not optimal that the parameter min_samples = 10 in algorithms SUCC, DBSCAN, and DBSCAN++ with respect to letters, but algorithm HDBSCAN does not need this parameter.

Table 4 shows the runtime (milliseconds) and standard errors for each algorithm and data. The runtime of SUCC (2A) is similar to that of SUCC (2B), while the runtime of SUCC (3A) is similar to that of SUCC (3B), so we omit the runtime of algorithm SUCC (2B) and SUCC (3B). The conclusions can be found from Table 4 as follows. At first, the runtime of SUCC on the three partitions is more than that on the two partitions with respect to every dataset. This is intuitive, because SUCC needs to merge more cluster centers belonged each partition on 3 partitions than that on 2 partitions. Then, an explanation of why SUCC spends shorter runtime clustering than compared algorithms did. SUCC deals with cluster cores belonged each division, while the compared algorithms run on raw dataset. In addition, there is no doubt that DBSCAN++ runs faster than DBSCAN and HDBSCAN on every dataset, because DBSCAN++ only computes the density estimates for a subset of the original dataset.

TABLE 2: ARI of SUCC vs. compared algorithms.

Dataset	SUCC				DBSCAN	DBSCAN++	HDBSCAN
	2A	2B	3A	3B			
Iris	0.796	0.532	<i>1</i>	0.431	0.568	0.663	0.568
Wine	0.804	0.447	<i>0.95</i>	0.509	0.285	0.369	0.242
Spambase	0.609	0.591	<i>0.457</i>	<i>0.73</i>	0.285	0.369	0.021
Images	0.239	0.239	<i>0.396</i>	<i>0.396</i>	0.292	0.385	0.082
PCA_MNIST	0.111	0.0000134	0.096	0.0000172	0.084	<i>0.142</i>	0.026
Libras	0.12	0.12	<i>0.243</i>	<i>0.243</i>	0.094	0.21	0.042
Mobile	0.064	0.003	0.089	0.002	0.055	<i>0.174</i>	0.012
Zoo	0.416	0.011	0.122	0.021	0.685	<i>0.734</i>	0.636
Seeds	0.651	0.535	<i>1</i>	<i>1</i>	0.404	0.499	0.303
Letters	0.047	0.00001424	0.019	0.0000967	0.062	<i>0.09</i>	0.022
Average	0.386	0.248	<i>0.437</i>	0.333	0.281	0.364	0.195

TABLE 3: AMI of SUCC vs. compared algorithms.

Dataset	SUCC				DBSCAN	DBSCAN++	HDBSCAN
	2A	2B	3A	3B			
Iris	0.774	0.644	<i>1</i>	0.559	0.577	0.73	0.732
Wine	0.766	0.526	<i>0.928</i>	0.63	0.359	0.415	0.372
Spambase	0.573	0.546	0.532	<i>0.641</i>	0.359	0.415	0.057
Images	0.444	0.444	<i>0.594</i>	<i>0.594</i>	0.494	0.56	0.264
Fashion MNIST	0.309	1.33E-04	0.294	1.33E-04	0.174	<i>0.377</i>	0.075
Libras	0.373	0.373	<i>0.568</i>	<i>0.568</i>	0.217	0.466	0.145
Mobile	0.209	0.004	0.22	0.006	0.212	<i>0.259</i>	0.017
Zoo	0.411	0.017	0.23	0.01	0.635	<i>0.746</i>	0.633
Seeds	0.69	0.598	<i>1</i>	<i>1</i>	0.44	0.506	0.283
Letters	0.401	2.36E-04	0.35	0.001	0.382	0.396	<i>0.423</i>
Average	0.495	0.315	<i>0.572</i>	0.401	0.384	0.487	0.3

TABLE 4: Runtime of SUCC vs. compared algorithms.

Dataset	SUCC		DBSCAN	DBSCAN++		HDBSCAN
	3A	2A		Uniform	K-center	
Iris	1.05 (± 0.02)	1.01 (± 0.03)	2.25 (± 0.07)	1.15 (± 0.08)	1.87 (± 0.01)	3.22 (± 0.47)
Wine	1.07 (± 0.03)	1.03 (± 0.11)	1.87 (± 0.08)	1.2 (± 0.07)	0.72 (± 0.02)	5.46 (± 0.18)
Spambase	25.2 (± 1.8)	22.5 (± 1.31)	375 (± 25.8)	25.6 (± 0.93)	50.2 (± 0.13)	360 (± 25.8)
Images	0.45 (± 0.01)	0.43 (± 0.02)	1.97 (± 0.08)	0.61 (± 0.04)	0.53 (± 0.03)	1.57 (± 0.01)
Fashion MNIST	46.7 (± 2.23)	41.2 (± 2.31)	2346 (± 57.3)	54.74 (± 5.71)	110 (± 5.5)	12346 (± 99.3)
Libras	6.58 (± 0.19)	5.66 (± 0.48)	36.58 (± 0.48)	7.97 (± 0.17)	9.6 (± 0.21)	42.4 (± 6.48)
Mobile	7.11 (± 0.07)	5.89 (± 0.12)	48.11 (± 0.37)	8.19 (± 0.11)	10.93 (± 0.11)	200.7 (± 3.7)
Zoo	0.89 (± 0.03)	0.88 (± 0.02)	1.41 (± 0.08)	1.03 (± 0.03)	1.07 (± 0.01)	4.88 (± 0.18)
Seeds	0.78 (± 0.04)	0.68 (± 0.04)	1.56 (± 0.07)	0.81 (± 0.02)	0.86 (± 0.02)	5.73 (± 0.07)
Letters	26.8 (± 3.55)	23.5 (± 1.15)	1296 (± 37.5)	67.39 (± 6.97)	19.5 (± 0.71)	14711 (± 49.5)

5.3. Robustness to Hyperparameter p . In this section, clustering on small-scale dataset is performed firstly by DBSCAN in 6 datasets in which SUCC is dominant, where parameter `min_samples` is fixed 10 and `eps` is between 10 and 50, with interval of 4. Then, we adjust the parameter p from 0.01 to 1 and generate the clustering results of large-scale data by

using the cluster cores belonged small-scale datasets and p . The clustering indexes are shown in Figures 3–8 on 6 datasets. On the whole, the clustering results of large-scale data are correlated with parameter p , except for Wine and Images. This explains the reason that the distribution of these two datasets is particularly uniform, or the data distribution is

FIGURE 3: ARI and AMI on Iris.

FIGURE 6: ARI and AMI on Images.

FIGURE 4: ARI and AMI on Wine.

FIGURE 7: ARI and AMI on Libras.

FIGURE 5: ARI and AMI on Spasebase.

FIGURE 8: ARI and AMI on Seeds.

consistent with data partition method. In addition, the clustering result is greatly affected by the value of p on some datasets, such as Libras. Therefore, we declare that the result of SUCC is more guaranteed under the guidance of experts.

6. Conclusions

In this study, we focus on multiscale clustering and propose an approach SUCC that generates clusters belonged large-scale data, which just needs to merge the cluster cores belonged small-scale data, not the raw dataset. In general, the size of cluster cores belonged small-scale data is much smaller than that of the original data, so SUCC needs smaller time overhead. The experimental results show that algorithm SUCC is efficient and effective on the dataset whose distribution is uniform. SUCC can perform better if that is under expert guidance in practice.

In future work, we mainly focus on the practical application of our study; we will consider applying multiscale clustering to radio-frequency identification [54, 55], collision detection [56–59], time series analysis [60–62], brain-computer interface [63–67], and scheduling [68, 69] based on previous researches.

Data Availability

The data underlying the results presented in the study are available within the manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is partially supported by the National Social Science Fund of China (No. 18ZDA200, No. 13&ZD091), the Hebei Provincial Key Research and Development Project of China (No. 20370301D), and the Key Technology Development Project of Hebei Normal University (No. L2020K01).

References

- [1] A. Gupta, S. Datta, and S. Das, “Fuzzy clustering to identify clusters at different levels of fuzziness: an evolutionary multi-objective optimization approach,” *IEEE Transactions on Cybernetics*, vol. 51, no. 5, pp. 2601–2611, 2021.
- [2] T. Singh, “A novel data clustering approach based on whale optimization algorithm,” *Expert Systems*, vol. 38, no. 3, article e12657, 2021.
- [3] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
- [4] A. Nagpal, A. Jatain, and D. Gaur, “Review based on data clustering algorithms,” in *2013 IEEE Conference on Information And Communication Technologies*, Thuckalay, India, April 2013.
- [5] R. Xu and D. C. Wunsch, “Clustering algorithms in biomedical research: a review,” *IEEE Reviews in Biomedical Engineering*, vol. 3, pp. 120–154, 2010.
- [6] A. Saxena, M. Prasad, A. Gupta et al., “A review of clustering techniques and developments,” *Neurocomputing*, vol. 267, pp. 664–681, 2017.
- [7] R. Chen, S. Zhao, and M. Liu, “A fast approach for up-scaling frequent itemsets,” *IEEE Access*, vol. 8, pp. 97141–97151, 2020.
- [8] H. Guo, G. Liu, Y. Ding, Y. Zou, and H. Ye, “Moon-based earth observation for large scale geoscience phenomena,” in *Presented at the IGARSS 2016, 2016 IEEE International Geoscience and Remote Sensing Symposium*, Beijing, China, 2016.
- [9] Z. Zhang, G. Wang, E. Carranza, J. J. Zhang, and Z. Pang, “Metallogenic model of the Wulong gold district, China, and associated assessment of exploration criteria based on multi-scale geoscience datasets,” *Ore Geology Reviews*, vol. 114, p. 103138, 2019.
- [10] J. Zhang, J. Du, and L. Dai, “Multi-scale attention with dense encoder for handwritten mathematical expression recognition,” in *Presented at the 24th International Conference on Pattern Recognition (ICPR)*, Beijing, China, 2018.
- [11] P. Parameshwarappa, Z. Chen, and G. Koru, “A multi-level clustering approach for anonymizing large-scale physical activity data,” CoRR, 2019, <https://arxiv.org/abs/1908.07976>.
- [12] D. Vu, S. Georgievskia, S. Szoke, A. Kuzniar, and V. Robert, “fMLC: fast multi-level clustering and visualization of large molecular datasets,” *Bioinformatics*, no. 9, pp. 1577–1579, 2018.
- [13] E. Zunic, Z. Tucaković, K. Hodžić, and S. Delalić, “Multi-level generalized clustering approach and algorithm for anomaly detection in internal banking payment systems,” in *Proceedings of 18th International Conference on Smart Technologies*, Chiang Mai, Thailand, 2019.
- [14] X. Chen and Y. Qiu, “An effective multi-level synchronization clustering method based on a linear weighted Vicsek model,” *Applied Intelligence*, vol. 50, no. 11, pp. 4063–4080, 2020.
- [15] I. Nourtdinov, J. Gammerman, M. Fontana, and D. Rehal, “Multi-level conformal clustering: a distribution-free technique for clustering and anomaly detection,” *Neurocomputing*, pp. 1–29, 2019.
- [16] L. Wang, D. Ming, and A. Kotov, “Multi-level approximate spectral clustering,” in *IEEE International Conference on Data Mining*, Atlantic City, NJ, USA, 2015.
- [17] J. SU, X. ZHAO, D. HONG, Z. LUO, and H. CHEN, “Q-value fine-grained adjustment based RFID anti-collision algorithm,” *IEICE Transactions on Communications*, vol. E99.B, no. 7, pp. 1593–1598, 2016.
- [18] J. Su, Z. Sheng, A. X. Liu, Z. Fu, and Y. Chen, “A time and energy saving based frame adjustment strategy (TES-FAS) tag identification algorithm for UHF RFID systems,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 2974–2986, 2020.
- [19] X. Liu, J. Zhang, S. Jiang et al., “Accurate localization of tagged objects using mobile RFID-augmented robots,” *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1273–1284, 2021.
- [20] J. Su, A. X. Liu, Z. Sheng, and Y. Chen, “A partitioning approach to RFID identification,” *IEEE/ACM Transactions on Networking*, vol. 28, no. 5, pp. 2160–2173, 2020.
- [21] J. Su, Y. Chen, Z. Sheng, and L. Sun, “Bit query based M-ary tree protocol for RFID tags identification,” in *IEEE Global Communications Conference*, pp. 1–7, Singapore, 2017.
- [22] J. Su, Y. Chen, Z. Sheng, Z. Huang, and A. X. Liu, “From M-ary query to bit query: a new strategy for efficient large-scale RFID

- identification,” *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2381–2393, 2020.
- [23] J. Su, Z. Sheng, A. X. Liu, Y. Han, and Y. Chen, “A group-based binary splitting algorithm for UHF RFID anti-collision systems,” *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 998–1012, 2020.
 - [24] J. Han, *Data Mining: Concepts and Techniques*, vol. 5 of Data Mining Concepts Models Methods & Algorithms Second Edition, 2004.
 - [25] J. B. Macqueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, USA, 1967.
 - [26] L. Dao, [Academic Press Library in Signal Processing] *Academic Press Library in Signal Processing: Volume 1 - Signal Processing Theory and Machine Learning Volume 1 || Clustering*, pp. 1115–1149, 2014.
 - [27] J. C. Dunn, “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters,” *Journal of Cybernetics*, vol. 3, no. 3, pp. 32–57, 1973.
 - [28] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, 1981, Pattern Recognition with Fuzzy Objective Function Algorithms.
 - [29] S. Ghosh and S. Kumar, “Comparative analysis of K-means and fuzzy C-means algorithms,” *International Journal of Advanced Computer Science and Applications*, vol. 4, no. 4, pp. 35–39, 2013.
 - [30] D. Sisodia, L. Singh, and S. Sisodia, “Clustering techniques: a brief survey of different clustering algorithms,” vol. 1, pp. 82–87, 2012.
 - [31] F. Murtagh, “A survey of recent advances in hierarchical clustering algorithms,” *Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.
 - [32] P. Sneath and R. R. Sokal, *Numerical Taxonomy: The Principles and Practice of Numerical Classification*, W.H. Freeman Co, San Francisco, CA, 1973.
 - [33] B. King, “Step-wise clustering procedures,” *Journal of the American Statistical Association*, vol. 62, no. 317, pp. 86–101, 1967.
 - [34] J. H. Ward Jr., “Hierarchical grouping to optimize an objective function,” *Publications of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.
 - [35] P. Andritsos, *Data Clustering Techniques*, University of Toronto, 2002.
 - [36] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH : an efficient data clustering method for very large data bases,” in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, New York, USA, 1996.
 - [37] S. Guha, R. Rastogi, and K. Shim, “Cure: an efficient clustering algorithm for large databases,” *Information Systems*, vol. 26, no. 1, pp. 35–58, 2001.
 - [38] S. Guha, R. Rastogi, and K. Shim, “A clustering algorithm for categorical attributes,” *Information Systems*, vol. 25, no. 5, pp. 345–366, 1999.
 - [39] G. Karypis, Eui-Hong Han, and V. Kumar, “Chameleon: hierarchical clustering using dynamic modeling,” *Computer*, vol. 32, no. 8, pp. 68–75, 1999.
 - [40] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” AAAI Press, 1996.
 - [41] C. S. Wallace and D. L. Dowe, “Intrinsic classification by MML - the Snob program,” in *Proceedings of seventh australian joint conf.artificial intelligence*, 1994.
 - [42] C. Fraley and A. E. Raftery, “How many clusters? Which clustering method? Answers via model-based cluster analysis,” *Computer Journal*, vol. 41, no. 8, pp. 578–588, 1998.
 - [43] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions: Second Edition*, The EM Algorithm and Extensions, Second edition, 2007.
 - [44] W. Wang, J. Yang, and R. Muntz, “STING: a statistical information grid approach to spatial data mining,” in *VLDB’97, Proceedings of 23rd International Conference on Very Large Data Bases*, Athens, Greece, August 1997.
 - [45] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, “Automatic subspace clustering of high dimensional data for data mining applications,” *Data Mining and Knowledge Discovery*, vol. 11, no. 1, pp. 5–33, 1998.
 - [46] F. E. Psomopoulos and P. A. Mitkas, “Multi-level clustering of phylogenetic profiles,” *International Journal on Artificial Intelligence Tools*, vol. 21, no. 5, p. 1240023, 2012.
 - [47] H. Du, Y. Wang, C. Ren, J. Zhong, and X. Gao, “Multi-level clustering algorithm for star/galaxy separation,” in *2016 12th International Conference on Computational Intelligence and Security (CIS)*, Wuxi, China, December 2016.
 - [48] I. Inuwa-Dutse, M. Liptrott, and I. Korkontzelos, “A multilevel clustering technique for community detection,” *Neurocomputing*, vol. 441, pp. 64–78, 2021.
 - [49] H. Liu, Z. Tao, and Y. Fu, “Partition level constrained clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 10, pp. 2469–2483, 2018.
 - [50] L. McInnes and J. Healy, “Accelerated hierarchical density based clustering,” in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, New Orleans, LA, USA, November 2017.
 - [51] J. Jang and H. Jiang, “DBSCAN++: towards fast and scalable density clustering,” in *Presented at the ICML*, Long Beach, USA, 2019.
 - [52] L. Hubert and P. Arabie, “Comparing partitions,” *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.
 - [53] N. X. Vinh, J. Epps, and J. Bailey, “Information theoretic measures for clusterings comparison,” in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML ’09*, pp. 2837–2854, Montreal, Quebec, Canada, 2009.
 - [54] J. Su, Z. Sheng, A. X. Liu, Y. Han, and Y. Chen, “Capture-aware identification of mobile RFID tags with unreliable channels,” *IEEE Transactions on Mobile Computing*, pp. 1–14, 2020.
 - [55] J. Su, Z. Sheng, V. C. M. Leung, and Y. Chen, “Energy efficient tag identification algorithms for RFID: survey, motivation and new design,” *IEEE Wireless Communications*, vol. 26, no. 3, pp. 118–124, 2019.
 - [56] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, “Idle slots skipped mechanism based tag identification algorithm with enhanced collision detection,” *KSII Transactions on Internet and Information Systems*, vol. 14, no. 5, pp. 2294–2309, 2020.
 - [57] J. Su, Z. Sheng, A. X. Liu, Z. Fu, and C. Huang, “An efficient missing tag identification approach in RFID collisions,” *IEEE Transactions on Mobile Computing*, pp. 1–12, 2021.
 - [58] J. Su, Z. Sheng, L. Xie, G. Li, and A. X. Liu, “Fast splitting-based tag identification algorithm for anti-collision in UHF RFID system,” *IEEE Transactions on Communications*, vol. 67, no. 3, pp. 2527–2538, 2019.

- [59] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, "Redundant rule detection for software-defined networking," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2735–2751, 2020.
- [60] Z. Jia, Y. Lin, Y. Liu, Z. Jiao, and J. Wang, "Refined nonuniform embedding for coupling detection in multivariate time series," *Physical Review E*, vol. 101, no. 6, p. 062113, 2020.
- [61] Z. Jia, Y. Lin, Z. Jiao, Y. Ma, and J. Wang, "Detecting causality in multivariate time series via non-uniform embedding," *Entropy*, vol. 21, no. 12, p. 1233, 2019.
- [62] Z. Jia, Y. Lin, J. Wang et al., "GraphSleepNet: adaptive spatial-temporal graph convolutional networks for sleep stage classification," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 1324–1330, Yokohama, Japan, July 2020.
- [63] Z. Jia, Y. Lin, X. Cai, H. Chen, H. Gou, and J. Wang, "Sst-emotionnet: spatial-spectral-temporal based attention 3D dense network for EEG emotion recognition," in *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2909–2917, October 2020.
- [64] X. Cai, Z. Jia, M. Tang, and G. Zheng, "Brainsleepnet: learning multivariate EEG representation for automatic sleep staging," in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 976–979, Seoul, Korea (South), December 2020.
- [65] Z. Jia, Y. Lin, J. Wang, X. Wang, P. Xie, and Y. Zhang, "Salient-SleepNet: multimodal salient wave detection network for sleep staging," 2021, <https://arxiv.org/abs/2105.13864>.
- [66] Z. Jia, Y. Lin, T. Liu et al., "Motor imagery classification based on multiscale feature extraction and squeeze-excitation model," *Journal of Computer Research and Development*, vol. 57, no. 12, p. 2481, 2020.
- [67] Z. Jia, X. Cai, G. Zheng, J. Wang, and Y. Lin, "SleepPrintNet: a multivariate multimodal neural network based on physiological time-series for automatic sleep staging," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 3, pp. 248–257, 2020.
- [68] Z. Kuang, L. Li, J. Gao, L. Zhao, and A. Liu, "Partial offloading scheduling and power allocation for mobile edge computing systems," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6774–6785, 2019.
- [69] Z. Kuang, L. Zhang, and L. Zhao, "Energy- and spectral-efficiency tradeoff with α -fairness in energy harvesting D2D communication," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9972–9983, 2020.