Clustering Approximation via RSP Data Model

Journal:	*Transactions on Big Data*
Manuscript ID	TBD-2022-05-0242
Manuscript Types:	Regular Paper
Keywords:	Clustering Approximation, Distributed and Parallel Clustering, RSP Data Model, Incremental Clustering, I-niceDP

Clustering Approximation via RSP Data Model

Mohammad Sultan Mahmud, Joshua Zhexue Huang, Rukhsana Ruby, *Member, IEEE* and Kaishun Wu, *Senior Member, IEEE*

Abstract—Large-scale data clustering needs approximate and distributed clustering for a computationally intensive data analysis, along with efficient ensemble support of data subsets. In this paper, we propose a new distributed data Clustering Approximation method for large-scale datasets via the Random Sample Partition (RSP) technique, namely RSPCA. In this LMGI (load matching and grid interaction) computing framework, we first partition a big dataset into a set of disjoint data blocks, i.e., RSP data blocks, with the RSP distributed data model. We define the set of RSP data blocks as a RSP data model of objects. We then randomly select a few data blocks from the RSP data model and analyze the clustering properties in each RSP data block independently, and ensemble the obtained results of the selected subsets as the approximate result of the entire dataset. Furthermore, we can also improve the clustering results incrementally by combining new outputs from other RSP blocks until a stable result is obtained or all RSP blocks are used up. We use the improved I-nice algorithm (i.e., I-niceDP) to enable tracking of initial centroids such that the K-means sweep determines the most promising cluster centroids that are local patterns generated via individual RSP data blocks. Moreover, the innovation includes two different graph-based approaches to ensemble the resulting clusters of the subsets. Experimental test results on real and synthetic datasets demonstrate that the integration of clustering results on a few RSP data blocks is sufficient for global discovery from the perspective of local insights: (1) has good approximation guarantee, and (2) is efficient and scalable.

Index Terms—Clustering Approximation; Distributed and Parallel Clustering; RSP Data Model; Incremental Clustering; I-niceDP

1 Introduction

1.1 Motivation

We know that large-scale data clustering is a challenging task, and critically dependent on distributed and parallel processing, and scalable and approximate computing, unlike traditional computing. Due to the difficulties of designing parallel and scalable clustering algorithms and the inefficiency in processing large-scale datasets, challenges on applying clustering techniques in big data have arisen [1], [2]. Thus, the question is how to deploy clustering algorithms efficiently to unknown big data to obtain a "good" feasible solution within a reasonable time.

A key is divide-and-conquer, to divide the large dataset into subsets and process them individually to scale the computational efficiency of clustering algorithms on big data [3]. Thus, combining clustering solutions from distributed subsets is required. Combining a set of clustering results into a final clustering outcome with overall good quality is a growing research topic in the area of pattern recognition, data mining and machine learning. Conceptually, the problem is also known as clustering ensemble, clustering fusion and consensus clustering [4].

Traditional clustering ensemble procedure integrates multiple clustering solutions to obtain a robust result from different clustering algorithms or the same algorithm with various initial parameters over the same dataset. Compared with the classical paradigm, combining multiple clustering solutions of data-subsets in a scalable framework for large-scale data clustering requires efficiency in terms of time and memory complexity. A clustering ensemble solution of disjoint data-subsets within a scalable framework for a large dataset is addressed in this paper.

Incremental and approximate computations are increasingly adopted for large-scale data analysis with limited computing resources [1]. Both paradigms rely on computing over a subset of data instead of computing the entire dataset, and the random sampling technique is fundamentally used. Recently, many researchers suggested that block-level sampling is far more efficient compared to uniform-random sampling over a large dataset, but result in error-prone outcome if randomized testing is not applied [5]. Notably, data is rarely ordered randomly in the big data world [6]. To address the problem, it is required to enable high-performance sampling for big data processing. Therefore, how to effectively process and analyze big data under limited resources is both a theoretical and technical challenge in current big data clustering research [7].

In this paper, we focus on the computational, statistical and inferential aspects of large-scale data clustering. Indeed, clustering methods always come up with k groups whatever the dataset is. At this stage, some questions could be raised in our mind. Can we obtain an idea about the number of 'natural' clusters on partial data that are really present on a big dataset? On a subset of a given big dataset, does the clustering algorithm fit random noise, or does it discover the global structure from local insights? Do the results constructed on the partial data by a given clustering algorithm "converge" under increased ensemble size? If yes, how fast? These questions are demanding, and it is hoped that our introduced random sample partitioning (RSP)-based clustering solution will provide answers to these questions partially and guidance.

To enable approximate computing, the random sample partition (RSP) data model [8] combines the advantage of data partitioning and sampling and the load matching and

• *The authors are with National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, Shenzhen 518060, China, and Big Data Institute, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China. E-mail: {sultan, zx.huang, ruby, wu}@szu.edu.cn*

grid interaction (LMGI) computing framework for effective processing and analyzing big dataset. The LMGI is a non-MapReduce framework that allows the execution of serial algorithms independently on local nodes or virtual machines without data communication among the nodes. Therefore, we present a large-scale data clustering solution without memory limit, that executes serial algorithms directly in distributed computing and extends the scalability of data analysis.

1.2 Contributions

Given the statistical and computational advantages of the RSP distributed data model, we employ this approach to address the problem of large-scale data clustering. For clarity, the novelty and contributions of this work can be distinguished in terms of following aspects.

- We formulate the RSPCA, a new distributed clustering ensemble method for tractable approximation of large-scale datasets. The proposed method does not assume the number of clusters a priori, but it is decided by the underlying RSP data model.
- Consequently, RSPCA adopts two graph-based schemes, RSPCA-GM and RSPCA-MP, to sophisticatedly ensemble centroids (clusters) of disjoint subsets, in which each RSP data block can presume (identified) a different number of clusters.
- Our approach is scalable since it allows computing data subsets independently on serial clustering algorithms in parallel on a distributed system with available resources. Analyzing a subset of randomly selected RSP data blocks can approximate global clustering properties (i.e., number of clusters, centroids and cluster diameter) and can be incrementally maintained during the clustering process.
- Extensive experiments on synthetic and real-world datasets are conducted to demonstrate the effectiveness and stability of the RSPCA approach while comparing with several state-of-the-art methods.

The remaining paper is organized as follows. Section 2 presents the related work of distributed clustering of large-scale data. In Section 3, we demonstrate preliminaries for the RSP-based clustering approximation framework. In Section 4, we propose RSPCA with two graph-based ensemble schemes, RSPCA-GM and RSPCA-MP, for ensemble clustering of subsets. Finally, we report the experimental results and analysis in Section 5 and conclude this paper in Section 6.

2 Related Work

In this section, some of the state-of-the-art methods of distributed clustering ensemble for large-scale datasets are presented and several issues are pointed out to motivate the present study.

Clustering ensemble is a fundamental problem and has been extensively studied in past decades [9], [10], [11]. However, the classical clustering algorithms cannot be directly applied to large-scale datasets because of their high computation time. Combining distributed clustering is more challenging compared to traditional clustering ensembles (i.e., aggregating multiple clustering solutions on the same dataset using several algorithms or the same algorithm with different parameters) because of different results in terms of different disjoint subsets. The analysis of this large-scale data necessitates highly scalable clustering techniques [2], [12], [13]. There are two pervasive solutions for large-scale data clustering to overcome computational bottlenecks: parallel/distributed computation and data compression before clustering is applied. According to preprocessing approaches, the existing ensemble clustering algorithms can be mainly classified as follows.

Sampling-based methods: These techniques are efficient and most widely used cost-effective way of reducing the size while maintaining the essential properties of data. In general, the framework of sampling-based methods first chooses a subset of the given dataset using only the sampled data to find the clusters, and then assigns the remaining data points to the closest cluster. The success of these techniques depends on the premise that selected representative objects maintain the distribution characteristics of the dataset as much as possible. To achieve this goal, many clustering methods have been developed by combining traditional algorithms with sampling techniques, such as uniform random sampling, progressive sampling, stratified sampling and biased sampling. Conversely, sampling techniques for big data exploration are not always practical due to a poor match between the sampling design and structure of data (i.e., sample sizes are very small and require much more samples) [14]. Some research suggested the effective use of block-level sampling in statistical computing, and to simply use as much data as possible and run experiments considering the scalability issue [6], [7].

Divide-and-conquer methods: This methodology firstly splits the large-scale data into different subsets that can fit into the memory or be analyzed with available computing resources, and then the clustering algorithms are implemented on these subsets efficiently. The final clustering results are yielded by merging the partial clusters of different subsets, as typical algorithm include [3]. A similar practice can be found in the community under the name of parallel and distributed computing [15]. In practice, data partitions are obtained by projecting data onto different subspaces [16] by choosing different subsets of features or data sampling [17]. It is intuitively assumed that each clustering algorithm will provide different levels of performance for different subsets of a dataset [11]. Some researchers realized that not all the ensemble subsets (members) contribute to the final result, and investigate the selection of a suitable subset of members to obtain better results [16], [18].

Incremental methods: In order to cluster the data quickly, these algorithms only scan the data points once, such as modified global k-means [19] and multiple medoids based on fuzzy clustering [20]. Recently, incremental clustering approaches are gaining more and more attention [16], [21]. Meanwhile, addressing the problem of incremental overlapping clustering has been reported. Similarity histogram-based clustering [22] and dynamic hierarchical compact method [23] were proposed, but these methods are time consuming due to the framework of hierarchical clustering. An algorithm based on density and compactness for dynamic overlapping clustering [24] was developed, but it builds a

large number of small clusters. Dynamic split-and-merge operations for evolving cluster models [3] are proposed, which are learned incrementally but can only deal with crisp clustering. Moreover, online fuzzy medoid-based clustering algorithms [25] have been adapted to overlapping clusters, but the number of clusters needs to be defined in advance.

Condensation-based methods: These methods alleviate the computational burden by encapsulating the dataset into special data structures, such as trees, graphs or matrices. After enabling the data structures of a large dataset, the storage and time of computing operations are greatly reduced. GMC [26], BIRCH and pair-wise co-occurrence [9] are some examples of this category. Moreover, a multi-scale density estimation-based nonparametric data reduction scheme [27] and a fast algorithm to extract small "core-sets" from the input data based on $(1+\epsilon)$-approximation algorithms for the k-center clustering have been developed in [28]. Several $O(1)$-approximation or $(O(1), O(1))$-approximation algorithms have been proposed utilizing the linear programming approach [29], [30] with the complexity at least $\Omega(n^3)$, which is prohibitive on large-scale datasets. Recently, a local-search based $(O(1); O(klog(n))$-approximation algorithm for $(k;t)$-means method is proposed in [31], where n is the size of the dataset, k and t are centroids and outliers, respectively. The running time of the algorithm is $O(k^2n^2)$, which is again not quite scalable.

3 Preliminary Concepts

In this section, we first introduce the random sample partition (RSP) data model for big data analysis. Then, we discuss the approximate computing in big data analysis and the technical difficulties of sampling from a big dataset. Finally, we describe the density-peak-based I-nice clustering algorithm, namely I-niceDP.

3.1 RSP distributed data model

Let $\mathbb{D} = x_1, x_2, ..., x_N$ is a big dataset with N objects. To avoid the computational burden, a data partitioning process is applied on $\mathbb{D}$ for data analysis in an efficient manner. Assume that $\mathbb{F}(x)$ is the empirical distribution of $\mathbb{D}$. Let $\mathbb{D}$ is divided into m small disjoint subsets such that $D_i = D_1, D_2, ..., D_m$ holds, each containing n records. The data partitioning operation is then called Random Sample Partition (RSP) and D_i is ready-to-use RSP data block of $\mathbb{D}$ if the following relation holds.

1) $\bigcup_{i=1}^{m} D_i = \mathbb{D}$
2) $D_i \cap D_j = \emptyset$ for $\forall i \neq j$ and $i, j \in \{1, 2, ..., m\}$
3) $\mathbf{E}[F_i(x)] = \mathbb{F}(x)$,

where $F_i(x)$ and $\mathbb{F}(x)$ denote the sample distribution function (s.d.f.) of D_i and $\mathbb{D}$, respectively. $\mathbf{E}[F_i(x)]$ denotes the expectation of $F_i(x)$. In [32], it is theoretically proved that the expectation of the s.d.f. of RSP blocks equates to the s.d.f. of $\mathbb{D}$. Since D_i preserves statistical properties as $\mathbb{D}$ does and is decidedly smaller than $\mathbb{D}$ in size, it can be efficiently processed and analyzed with existing sequential algorithms as well as in a parallel fashion.

3.2 Approximate clustering of large-scale data

If we can obtain approximate clustering results of the entire dataset by analyzing only a few subsets in parallel to satisfy application requirements, we can reduce the computational costs and make the big data clustering with the limited available resources. We know that the precondition for approximate analysis of big data with a subset is that distributions of data in the subsets are required to be similar to the distribution of the original full dataset. However, the data subsets generated by the common data partitioning methods do not necessarily satisfy the precondition since they do not consider the properties of statistical distribution in the data. In fact, natural data are rarely randomized (ordered randomly) in the large-scale dataset [6], [7]. More precisely, statistical and inferential thinking is required for approximate clustering of large-scale datasets, and hence the concept of RSP data model offers a solution.

Given a RSP data model $\mathbb{D} = \{D_1, D_2, ..., D_m\}$, we can select a few subsets $\{D_1, D_2, ..., D_b\}(b < m)$ to analyze and use the clustering approximate results as the estimated results of $\mathbb{D}$. Using this approximated clustering solution, we can obtain an approximation of cluster characteristics (i.e., the number of clusters, centroids and cluster diameter) that makes clustering feasible. Since multiple data blocks of $\mathbb{D}$ are used, it is straightforward to use ensemble methods to improve the analysis results. The objective of the RSP data model is to enable the use of a few randomly selected RSP data blocks to produce approximate results without processing the entire dataset altogether or parallelizing data analysis. In the RSP-based method, we can update the previous output by appending new outputs from newly analyzed RSP blocks.

3.3 Improved I-nice for identifying the number of clusters and initial centroids

I-nice [33] is a parameter-free clustering algorithm, which identifies the number of clusters and initial centroids using observation points. Although I-nice achieves better clustering performance, there are two inherent limitations unfortunately that need to be improved to enhance its clustering capability. First, I-nice is sensitive to the position of the observation point and the number of observation points, and the other is that the number of nearest neighbours affecting the determination of high-density areas. Inspired by density peaks clustering, I-nice algorithm is improved to a density-peaks-based I-nice (I-niceDP) [34]. Instead of the k-nearest neighbors and a pre-defined threshold of I-nice, I-niceDP uses density peaks to determine the number of clusters and initial centroids in the components of the gamma mixture model. The comparative results with I-nice indicate that I-niceDP can more accurately identify the number of clusters and initial centroids for the datasets with a large number of clusters.

There are two evolved versions of I-nice algorithm, I-niceSO and I-niceMO. The I-niceMO combines the results of multiple random observation points to obtain the correct estimation of the distance distributions of objects, to detect a large number of clusters perfectly. An improper observation point generates inaccurate distances distributions (since it is random and probably in a weak position), and further

causes an incorrect estimation of the number of clusters. For multiple observation points $O_p = \{O_1, O_2, ..., O_d\} \in \mathbb{R}$ which are randomly generated, this calculates the Euclidean distances between O_p and $x_i, i = 1, 2, ..., N$. Finally, I-niceMO scheme obtains candidate cluster centroids corresponding to d observation points integrating the candidate cluster centroids with distance smaller than a given threshold, which increase computational cost.

Ideally, in the RSP data model, we need to deal with a small subset of $\mathbb{D}$ which has a similar clustering structure to $\mathbb{D}$. Each RSP data block produces local clustering outcome using the I-niceDP scheme with randomly generated single observation points and collaborative result dynamically constructed from multiple RSP data blocks. A subset of RSP data blocks portray multiple views (a set of RSP data blocks as multiple observations) of a dataset simultaneously, and thus it can capture the latent knowledge in data in a more comprehensive manner. Meanwhile, the RSP data model portrays that single observation point-based I-nice scheme (i.e., I-niceDP) is sufficient under the clear distinction of the high-density areas in the original data space. We consider that each RSP data block is a single-observation process to produce local clustering insights, and the collaborative step is to share information of their memberships among different observations views of RSP data blocks. These two steps are then continued in a collaborative aggregating manner to obtain a global outcome.

4 Approximation of Clustering via RSP Data Model

In this section, we first describe the basic idea and challenge of distributed clustering ensemble, and then present the key steps of the proposed RSPCA method in detail.

4.1 Distributed clustering ensemble problem definition

Assume that $\mathbb{D}$ is a dataset of N records, where N is very large, and hence $\mathbb{D}$ cannot be analyzed efficiently on a single machine. To process clustering analysis efficiently, $\mathbb{D}$ is divided into m non-overlapping data blocks of equal size (each with $n \ll N$ records), where the following relation holds.

1) $\bigcup_{i=1}^{m} D_i = \mathbb{D}$,
2) $D_i \cap D_j = \emptyset$ for $\forall i \neq j$ and $i, j \in \{1, 2, ..., m\}$.

Now, we assume that data blocks fit into memory and can be distributed to the computing nodes and analyzed independently on nodes sequentially in a parallel manner. Now, we can randomly select a few subsets $\mathcal{S} = \{D_1, D_2, ..., D_b\}, (b < m)$ to conduct clustering analysis, and use for approximations as the estimated results of $\mathbb{D}$. In general, a big dataset is unknown. It is important to note that no prior knowledge on the number of clusters in a dataset is available. Using classic and efficient algorithms, we can estimate the number of clusters and centroids of data subsets. b data blocks provide b sets of cluster centers individually as follows.

$$\pi(D_i) = \Phi(D_i), \quad \pi(D_i) = \{C_1^i, C_2^i, ..., C_K^i\},\ i = 1, 2, ..., b,$$

where $\Phi(.)$ is a clustering function and K is the estimated number of clusters locally in each data block and may not have the same number of clusters in the data subset. A unified clustering ensemble $\Pi^{\star}(\mathcal{S}) = \{\pi(D_1), \pi(D_2), ..., \pi(D_b)\}$ can then be built to combine b different clustering solutions to obtain results that reflect the entire dataset. A key challenge is how to combine the clusters that are generated by the individual clustering models (from data-subsets) in an ensemble, as this cannot be done through simple voting or averaging tool in the classical clustering process. To conquer this problem, we propose two strategies for the implementation of approximate analysis steps, which take two scenarios into consideration.

4.2 RSP-based clustering ensemble approach

In the following, we describe the key steps of our method in detail.

4.2.1 Generating RSP data model

Let $\mathbb{D}$ be a large dataset. We use RSP data model to convert $\mathbb{D}$ into a set of RSP data blocks, such as

$$\bigcup_{i=1}^{m} D_i = \mathbf{RSP}(\mathbb{D})$$

$$f(D_1) \approx f(D_2), ..., \approx f(D_m),$$

where $\mathbf{RSP}(.)$ is the conversion function to an RSP data model and $f(.)$ denotes statistical sample distribution. Since D_i is a random sample of $\mathbb{D}$ and has the statistical properties as $\mathbb{D}$ does, analyzing D_i can obtain approximate results of $\mathbb{D}$. The size of D_i is decidedly smaller than that of $\mathbb{D}$, and hence it can be efficiently processed in a traditional machine as well as in a parallel and distributed manner.

4.2.2 Sampling RSP data blocks

In this step, we select a few b data blocks randomly from $\bigcup_{i=1}^{m} D_i$ as

$$\mathcal{S}^i = \{D_1^i, D_2^i, ..., D_b^i\}, \mathcal{S}^i \subset \bigcup_{i=1}^{m} D_i,$$

where $b < m$ holds, and m is total number of RSP data blocks in $\mathbb{D}$. According to the RSP theory, each data block D_i is a random sample of $\mathbb{D}$, and therefore $\mathcal{S}^i$ is a subset of random samples in $\mathbb{D}$. A few RSP data blocks can provide sufficient clustering approximation with far less computational cost compared to analyzing the entire dataset directly.

4.2.3 Analyzing $\mathcal{S}$ for clustering properties

In this step, we use a density-peak-based improved I-nice approach (i.e., I-niceDP) for identifying the number of clusters and initial cluster centres to each independent RSP data block D_i in $\mathcal{S}$. The high-density points for the candidate cluster centroids are identified via I-niceDP clustering algorithm as

$$\pi(D_i) = \mathbf{I} - \mathbf{niceDP}(D_i), \pi(D_i) = \{c_1^i, c_2^i, ..., c_k^i\},$$

where c_i is the initial cluster centroids, and k is the desired number of clusters. I-niceDP provides preliminary insights of clusters, and K-means algorithm can refine it. The output from the I-niceDP scheme are applied as initial centroids

to the K-means clustering scheme to determine the final number of clusters and centres of RSP data blocks locally.

$$\Pi(D_i) = \mathbf{K} - \mathbf{means}(D_i, \pi), \Pi(D_i) = \{C_1^i, C_2^i, ..., C_K^i\},$$

where C_i is cluster centroid and K is the number of clusters. For b subsets, there will be b sets of centroids, i.e., $\{C_j^1\}_{j=1}^K, \{C_j^2\}_{j=1}^K, ..., \{C_j^b\}_{j=1}^K$. This set of centroids, the outputs from these RSP data blocks $\bigcup_{i=1}^m D_i = \mathcal{S}$, are combined to produce an approximate result for $\mathbb{D}$.

$$\Pi^\star(\mathcal{S}) = \{C_1^\star, C_2^\star, ..., C_K^\star\}$$

where $\Pi^\star(\mathcal{S})$ is ensembled into a single clustering and consolidated centroid $C_i^\star$.

4.2.4 Combining multiple clustering outcome

A measure is therefore needed to combine multiple clustering outcome. More specifically, in this case, we aggregate the "centroids" of selected data-subsets. In order to merge centorids of subsets (sub-centroids), two different schemes have been implemented.

Scheme 1: RSPCA using graph matching (RSPCA-GM) Here, we propose our RSPCA-GM algorithm for matching graphs with the following three procedures.

1) *Constructing a similarity matrix and a sparse graph:* We compute a similarity matrix $\mathbf{S}$ from $\mathcal{S}$. By associating each data point, we construct the sparse graph for the data via similarity matrix, where data points are the nodes and the distance between two objects as an edge value, with the edge weight indicates the strength of two objects (i.e., centroids) belonging to the same cluster.
2) *Splitting the sparse graph:* Now, we split the sparse graph into a large number of relatively sub-clusters by using the METIS algorithm [35]. Intuitively, METIS splits the whole graph into two sub-clusters and then selects the sub-cluster for the purpose of division in an iterative manner. In particular, if the number of sub-clusters is too large, the time complexity of splitting and merging increases and the accuracy of clustering result may not be improved in an obvious manner. On the contrary, if the number of sub-clusters is too small, the sub-clusters may contain more data samples belonging to other clusters. Therefore, it is important to choose an appropriate terminal condition.
   We decide the terminal condition as the number of sub-clusters, $s/\sqrt{s/v}$, where s represents the size of data (i.e., the number of vertices) and v denotes the kth smallest eigenvalue, which is a good indicator of the number of clusters. We estimate v in the data by using the eigenvalues of the Laplacian matrix $\mathbf{L}$. The number of zero eigenvalues (in magnitude) is a good indicator of the number of connected components in a similarity graph, and therefore we proceed with corresponding the number of eigenvalues that are approximately zero to the number of clusters. As we know, the number of sub-clusters is satisfied to 2^{itr}, where itr represents the number of iterations. When itr is satisfied with $2^{itr-1} < s/v < 2^{itr}$, the next iteration calculates the intra-connectivity of the present set of sub-clusters.
3) *Recombine sub-clusters:* After the sub-clusters are generated by step 2, we recombine together these small sub-clusters using the agglomerative hierarchical clustering approach that iteratively merges sub-clusters based on their similarity feature. The key of this step is to determine the most similar pair of sub-clusters by considering both their relative inter-connectivity and their relative closeness. Specifically, the relative inter-connectivity between a pair of clusters C_i and C_j is defined as

$$RI(C_i, C_j) = \frac{2 \cdot \left|EC_{\{C_i,C_j\}}\right|}{\left|EC_{C_i}\right| + \left|EC_{C_j}\right|}, \quad (1)$$

where $EC_{\{C_i,C_j\}}$ represents the edge-cut of the cluster containing both C_i and C_j. EC_{C_i} denotes the size of min-cut bisector of C_i (i.e., the weighted sum of edges that partition the graph into two roughly equal parts). The relative closeness between a pair of clusters C_i and C_j is defined as

$$RC(C_i, C_j) = \frac{\overline{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{EC_{C_i}} + \frac{C_j}{|C_i|+|C_j|}\overline{EC_{C_j}}}, \quad (2)$$

where $\overline{EC_{\{C_i,C_j\}}}$ is the average weight of the edges that connect vertices in C_i to vertices in C_j. $\overline{EC_{C_i}}$ and $\overline{EC_{C_j}}$ are the average weights of the edges that belong in the min-cut bisector of clusters C_i and C_j.

$$Similarity(C_i, C_j) = RI(C_i, C_j) \cdot RC(C_i, C_j)^\alpha, \quad (3)$$

where α is scale factor, which determines the importance of relative closeness and relative inter-connectivity on computing $Similarity(C_i, C_j)$. If $\alpha > 1$ holds, it gives a higher importance to the relative closeness $RC(C_i, C_j)$. On the other hand, if $\alpha < 1$ holds, it gives higher importance to relative inter-connectivity $RI(C_i, C_j)$. We set $\alpha = 1$, that gives the same weight to relative inter-connectivity and relative closeness features.

Scheme 2: RSPCA using message-passing (RSPCA-MP) The message-passing procedure outperforms several state-of-the-art clustering algorithms for small-to-moderate datasets [12], [36]. We also aim to introduce and analyze an affinity propagation-based ensemble algorithm to find K clusters via exchanging messages on graphs. Specifically, the graph allows exchange of information between nodes. Typically, this message (a similarity matrix) is derived from a set of pairwise similarities between the points (i.e., set of centroids) to be clustered. It requires exchange of two messages, responsibility and availability, between data points. The workflow of the message-passing-based clustering ensemble is specified as follows.

1) *Compute the similarity matrix:* To start, we compute a similarity matrix $\mathbf{S}$ for the $\mathcal{S}$ using the negative squared Euclidean similarity function, which takes a numerical value s_{ij} for each pair of points i, j. A standard similarity measure, negative squared distances is used. Affinity propagation clustering approach [36] seeks to group data points by exchanging messages between nodes of a graph representing the data. For each cluster, the algorithm identifies an 'exemplar', a member of the dataset that represents points in the cluster. Similarity between the nodes in the graph is specified using a

measure, such as the resulting clusters may be interpreted as subgraphs spanned by edges.

2) *Compute responsibility matrix:* Responsibilities r_{ij} are sent from data points to candidate exemplars to quantify the strength each data point serving the candidate exemplar over other candidate exemplars. We start off by constructing an availability matrix with all elements setting to zero. Then, we obtain the responsibility matrix $\mathbf{R}$ as

$$r_{ij} \leftarrow s_{ij} - \max_{j' \neq j}\{a_{ij'} + s_{ij'}\}, \tag{4}$$

where i and j refer to the row and the column of the associated matrix.

3) *Compute availability matrix:* Availability a_{ij} expresses the level of confidence that point i should choose j as an exemplar. These messages are derived from the max-sum algorithm, and the availability matrix $\mathbf{A}$ is computed as

$$a_{ij} \leftarrow \min\left[0, r_{jj} + \sum_{i' \notin \{i,j\}} \max\{0, r_{i'j}\}\right]. \tag{5}$$

4) *Compute criterion matrix:* In each iteration of the message-passing procedure, the criterion value, the sum of availability and responsibility is used to identify exemplars. After the messages have converged, two ways for data point i to be an exemplars as

$$c_{ij} = \begin{cases} 1, & \text{if } r_{ii} + a_{ii} > 0 \\ & \text{if } r_{ii} + a_{ii} > r_{ij} + a_{ij}, \forall i \neq j \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

As in the classical AP [36], binary variable c_{ij} indicates whether j is an exemplar of i or not, while c_{ij} takes on value 1 if j is the exemplar for i, and 0 otherwise. If j is an exemplar for any $i \neq j$, then j must also be an exemplar for itself. Thus, in the criterion matrix $\mathbf{C}$, the binary criterion value of each row is designated as the exemplar. The rows that share the same exemplar are in the same cluster. The entire procedure terminates once it reaches a predefined number of iterations or if the determined clusters remain constant for a certain number of iterations. The number of clusters is automatically inferred by the algorithm and can be tuned via self-similarity feature or preference of the data points.

4.3 A numerical example

To better illustrate the key steps of the proposed clustering approximation, we provide a small exaggerated numerical example as shown in Fig. 1. In this example, there are five random RSP data blocks (i.e., block #$\{11, 105, 138, 289, 365\}$) of two-dimensional 5,000 objects which belong to 10 clusters. Intuitively, using the I-niceDP cluster estimation algorithm in each selected RSP data block, the clusters correspond to $\{8, 7, 8, 9, 9\}$. In other words, the I-niceDP algorithm and Kmeans sweep calculate the centroids as illustrated in Fig. 1(a-e).

We aggregate the associated set of centroids in Fig. 1(f) that resolves a conclusion synthesized by an ensemble function. The collective centroids are compact and coreset representations of subsets. Consequently, to solve this ensemble problem, we propose two graph-based ensemble functions, RSPCA-GM and RSPCA-MP. According to both the RSPCA-GM and RSPCA-MP-based aggregation processes, the number of approximations is 10 clusters as shown in Fig. 1(g-h). Finally, Fig. 1(i-j) demonstrate the impact of ensemble results.

4.4 Computational complexity analysis

We finally analyze the computational complexity of the RSPCA scheme. It has three major parts that need to be calculated: (1) conducting an RSP operation to generate RSP data blocks and randomly select a subset of data blocks, (2) determining the initial centroids and number of clusters using the I-niceDP scheme and refining explored centroids via the Kmeans algorithm on each selected RSP data block, and (3) gathering the meta-data of intermediate results in RSP data blocks and performing an ensemble operation to obtain the final clustering results.

Suppose that given a dataset with N objects, we generate m RSP data blocks with n data points ($n \ll N$) and randomly select b subsets, where $b < m$ holds. The first part is to apply an RSP operation with $\mathcal{O}(n \log(N/n)$ complexity. The second part is that the I-niceDP scheme generates a one-dimensional data and density peaks from each RSP data block to define initial cluster centroids with $\mathcal{O}(nd)$ complexity, where d is the dimension of the data. To refine centroids via the K-means scheme, it requires $\mathcal{O}(ntdk)$ operations, where t is the maximum number of iterations and k is the number of clusters. Consequently, the I-niceDP algorithm requires $\mathcal{O}(ntdk)$ operations. The third part is to merge the individual results of RSP data blocks, and we propose two graph-based ensemble schemes, RSPCA-GM and RSPCA-MP, for this purpose.

The RSPCA-GM process needs to construct the $\tilde{n} \times \tilde{n}$ similarity matrix between each pair of objects, which is $\mathcal{O}(\tilde{n})$ (i.e., $\tilde{n}$ is a set of centroids and in reality $\tilde{n} \ll n \ll N$ holds). In particular, RSPCA-GM operates on the k-nearest neighbor graph and depends on the HMETIS scheme, and hence its overall computational complexity is $\mathcal{O}(\tilde{n} \log(\tilde{n}/k))$, which is bounded by $\mathcal{O}(\tilde{n} \log \tilde{n})$. In fact, the technique needs to compute the eigenvalues/eigenvectors of the Laplacian matrix with $\mathcal{O}(\tilde{n}^3)$ complexity. Letting the internal inter-connectivity and internal closeness for a particular cluster being computed by bisecting the corresponding k-nearest neighbor sub-graph of the cluster, it requires $\mathcal{O}(\tilde{n}k)$ operations. The overall calculation required to find the most similar pair of clusters is $\mathcal{O}(k^2 \log k)$. Thus, the overall complexity of the ensemble step (three-phases) of RSPCA-GM scheme is $\mathcal{O}(\tilde{n}k + \tilde{n} \log \tilde{n} + k^2 \log k)$.

Similar to RSPCA-GM, RSPCA-MP also computes a $\tilde{n} \times \tilde{n}$ similarity matrix among the objects (centroids), which incurs $\mathcal{O}(\tilde{n}^2)$ complexity. The complexity of performing T iteration that involves exchanging messages between the nodes is $\mathcal{O}(\tilde{n}^2 T)$. Overall, the computational complexity of the ensemble step in RSPC-MP scheme turns to $\mathcal{O}(\tilde{n}^2 T)$.

Note that the overall complexity of RSPCA is a linear function of the number of RSP data blocks b. Since the RSPCA is a distributed and parallel framework, we can proceed on a computing cluster with Q nodes (e.g., Apache Spark or Hadoop system). In summary, the total computational complexities of the two proposed clustering ensemble

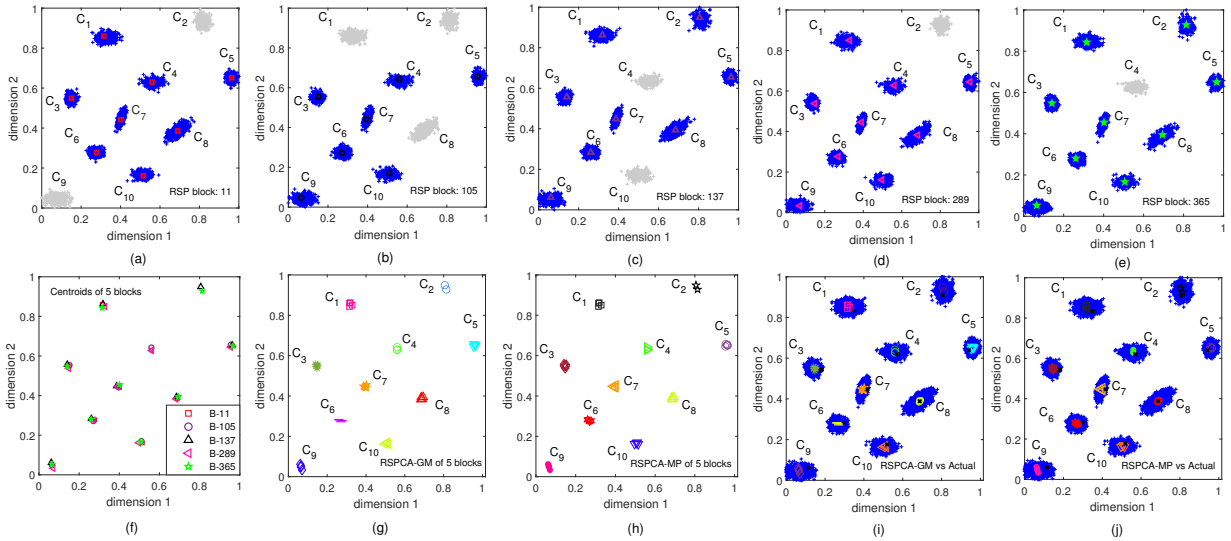


Fig. 1: An explanation of the RSPCA process using the 2M2D10C dataset. (a-e) Randomly selected five RSP data blocks of 5,000 points and identified I-niceDP's cluster centroids in individual data block. The positions of identified cluster centroids of five RSP data blocks in (f). The final positions of the merged cluster centriods using RSPCA-GM and RSPCA-MP are shown in (g) and (h). The associated clustering results when applying the RSPCA-GM and RSPCA-MP ensemble function are shown in (i) and (j), and actual centroids denoted by the symbol "×". Different colors and symbols represent individual cluster assignment of each observation.

schemes are in the order of $\mathcal{O}((n\log(N/n)+ntdk)/Q+\tilde{n}^3+\tilde{n}k+\tilde{n}\log\tilde{n}+k^2\log k)$ and $\mathcal{O}((n\log(N/n)+ntdk)/Q+\tilde{n}^2T)$, respectively. The individual result vectors are sent back to the master node with a communication complexity of $\mathcal{O}(1)$. Moreover, the RSP data model is a LMGI computing framework that allows execution of serial algorithms independently on local nodes without data communication among the nodes.

5 Empirical Analysis

In this section, we report on extensive experiments to evaluate the performance of the proposed RSPCA. First, we introduce datasets used including the experimental setting. Then, we illustrate the parameter sensitivity and convergence of RSPCA-GM and RSPCA-MP schemes. Finally, we present a performance comparison with a number of state-of-the-art algorithms.

5.1 Datasets

We tested the proposed clustering methods on several synthetic datasets as well as on real-world datasets. The characteristics of the used seven datasets are shown in Table 1. We start the evaluation with two-dimensional (2D) synthetic data to facilitate the presentation and demonstrate the benefits of RSPCA. We make use of the following datasets.

- *2M2D10C* and *2M2D20C*. We generate these two synthetic datasets from the multivariate Gaussian distribution models $\mathcal{N}(0,1)$, where observations have independent and identically distributed dimensions with different parameters. In particular, we focused on different sizes and the number of clusters.
- *3M2D5C*. This dataset is designed for clustering tasks and used in [37], [38]. This synthetic dataset consists of 3M objects drawn from a mixture of five bivariate normal distributions. Note that there are overlapped objects from components in the mixture.
- *Covertype*[1]. This dataset, containing 581,012 objects, describes seven forest cover types with 54 different geographic measurements. Note that there are 84% of objects that belong to 2 classes (type-1 36.5% and type-2 48.7%).
- *PokerHand*[2]. In this dataset, each record is an example of a hand consisting of five playing cards drawn from a standard deck of 52. There are about 1.2M data points with ten predictive features. There are ten classes, and two dominant classes account for over 90% of the samples (nothing in hand 49.9% and one pair 42.4%).
- *KDD'99*[3]. This dataset is from KDD Cup 1999 competition, and contains network intrusion detection data. There are about 4.9M objects with 42 features. There are 23 classes in this dataset, and 98.3% of the objects belong to 3 classes (normal 19.6%, neptune 21.6% and smurf 56.8%).
- *SUSY*[4]. This dataset has been produced using Monte Carlo simulations. There are 5M objects, and each object has 18 numerical features to help discriminate between two classes.

We hold all the features of real-world datasets and transform the categorical features into the numerical ones. We next construct the two-dimensional data using the truncated SVD (aka LSA) tool. For all datasets, we also normalize features using the max-min normalization technique.

5.2 Experiment setup and parameter analysis

Table 2 summarizes the experimental parameters of RSP-based clustering in order to fairly test the influence of

1. https://archive.ics.uci.edu/ml/datasets/covertype
2. https://archive.ics.uci.edu/ml/datasets/Poker+Hand
3. https://www.kdd.org/kdd-cup/view/kdd-cup-1999/Data
4. https://archive.ics.uci.edu/ml/datasets/SUSY

TABLE 1: Description of the datasets (d: dimensions, N: number of observations, K: number of clusters or classes).

Dataset	N	d	K	Cluster distributions
2M2D10C	2,000,000	2	10	100,000 (2: 3: 3: 1: 1: 2: 2: 3: 1: 2)
2M2D20C	2,000,000	2	20	10,000 (15: 5: 13: 7: 10: 12: 8: 10: 15: 5: 8 : 12 : 13 : 7: 10: 13: 7: 12: 8: 10)
3M2D5C	3,000,000	2	5	300,000 (2: 2: 3: 1: 2)
CT	581,012	2	7	211,840: 283,301: 35,754: 2,747: 9,493: 17,367: 20,510
PH	1,025,010	10	10	513,702: 433,097: 48,828: 3,978: 21,634: 2,050: 1,460: 236: 17: 8
KDD	4,898,431	42	23	972,781: 2,807,886: 1,072,017: 15,892: 12,481: 10,413: 2,316: 2,203: 1,020: 979: 264: 53: 30: 21: 20: 12: 10: 9: 8: 7: 4: 3: 2
SUSY	5,000,000	18	2	2,712,173 : 2,287,827

CT: Covertype, PH: Poker Hand, KDD: KDDCup'99

TABLE 2: Parameter Settings.

Parameters	Range
RSP data block size (n)	2,000; 5,000; 10,000
Ensemble data size (es)	5%; 10%; 15%; 20%
Number of observation points in I-niceDP	1; 2; 3

RSP data block size (n) and ensemble size (es). In the experiment, we investigate the performance of RSPCA-GM and RSPCA-MP schemes under various parameter settings. In order to explore the effect of the parameters, the value of each of them is trialed 25 times independently.

5.2.1 *Influence of the number of observation points of I-niceDP in RSP data blocks*

The number of random observation points is a common parameter in the I-nice algorithm to investigate the number of clusters and centroids (that is, density-peaks of distance distribution). Generally, multiple observation points (MO) lead to better performance but also bring in an increased computational cost. Instead, viewed through the individual RSP data block with I-niceDP-MO, multiple RSP data blocks with I-niceDP-SO are simple and indeed provide a generic solution (because of data-subset from the same domain). Broadly, I-niceDP-SO consistently requires a lower computational cost compared to I-niceDP-MO. Specifically, we decide to apply single-observation-point-based I-niceDP (i.e., I-niceDP-SO). Note that the RSP data model is required to analyze multiple data blocks, accomplishing a similar task to the multiple-observations-points-based I-niceDP in an investigation context. Fig. 2 offers a consolidated summary on the dependence of observation points achieved by I-niceDP.

5.2.2 *Effect of RSP data block and ensemble size*

To evaluate the effect of the RSP data block size (n), we generate RSP data blocks of sizes in $n \in \{2,000; 5,000; 10,000\}$. We then use the I-niceDP algorithm to solve the clustering problem on a set of randomly selected RSP data blocks. We use different data ensemble sizes, $es \in \{5; 10; 15; 20\}\%$ (i.e., RSP data blocks), as an approximation of the original dataset. To illustrate the effect of RSP data block cardinality and data ensemble cardinality in both RSPCA-GM

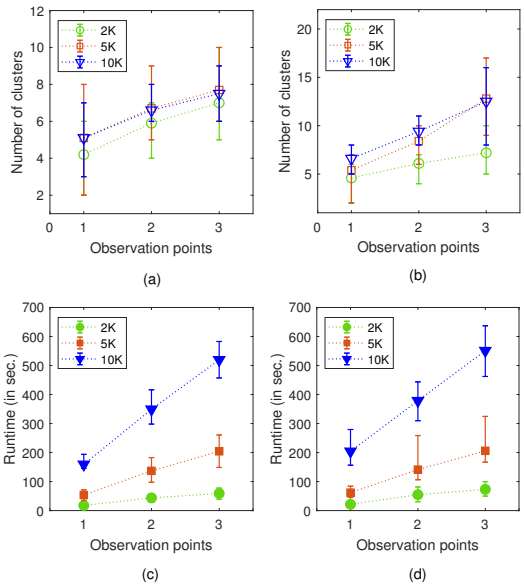


Fig. 2: Dependence of number of clusters on the number of observation points of I-niceDP and RSP data block size, $n = \{2000; 5000; 10,000\}$ (max-mean-min of 25 trails).

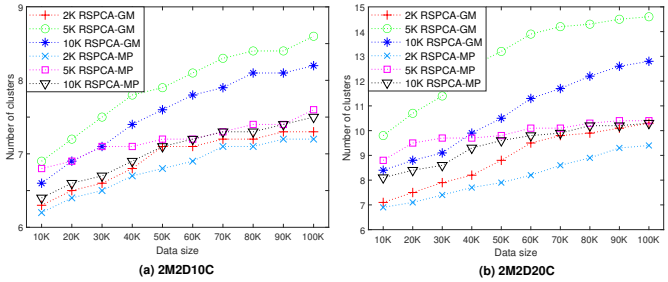


Fig. 3: Impact of RSP block and ensemble size in estimated number of clusters using I-niceDP-SO (average of 25 trails).

and RSPCA-MP approaches, two datasets are considered in Fig. 3. Similarly, we run the state-of-the-art algorithms on different data ensemble cardinalities, $es \in \{5; 10; 15; 20\}\%$ of the data subset in the entire dataset, and measure the time complexity and the solution quality as evaluated approximation on the full dataset.

5.3 Competitors

We compare RSPCA-GM and RSPCA-MP schemes with several state-of-the-art large-scale data clustering algorithms, including nbootstrap [17], kluster [39], CAMUSA [40], U-SENC [2], and SNN-DPC [41]. In our experiments, we comply with the following experimental settings.

nbootstrap [17] is the selection of the number of clusters via bootstrap. The number of resampling B runs two bootstraps drawn from the dataset and the number of clusters is chosen by optimising an instability estimation from these pairs. In the experiment, $B = 20$ is used.

kluster [39] is an efficient scalable procedure for approximating the number of clusters using the BIC Bayesian information criterion (BIC), partitioning around medoids (PAM), affinity propagation (AP), and Calinski and Harabasz index (CAL) approaches. It iteratively applies four statistical methods to small subsets of data and provides the most frequent and the mean approximated number of clusters.

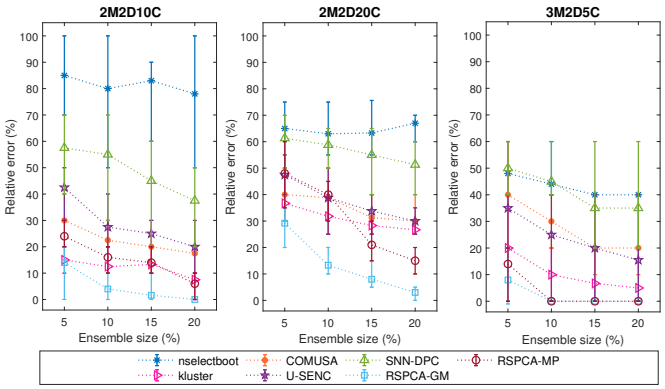


Fig. 4: Relative error in estimated number of clusters of applied methods. Error bars denote max-mean-min of 25 trails.

COMUSA [40] is a similarity graph-based clustering algorithm. A similarity graph is constructed by computing the co-associations of objects in the input. Then, COMUSA initiates a new cluster by picking a pivot vertex and extending the cluster with its neighbors if they are the most similar to the pivot.

U-SENC [2] is based on efficient affinity submatrix construction and bipartite graph formulation. The U-SENC method has a common parameter p. In the experiments, $p = 1000$ is used, and obtained k smallest eigenvectors close to zero is the number of clusters.

SNN-DPC [41] is a shared-nearest-neighbor-based clustering approach based on the fast search and density peaks technique. The parameter k is a significant argument, which controls granularity of clusters in SNN-DPC clustering. If k is small, it finds small and very tight clusters. On the other hand, if k is large, it finds big and well-separated clusters. In the experiment, we used $k = 20$.

5.4 Evaluation metrics

Effective comparison of different clustering algorithms is a non-trivial problem, especially if different algorithms produce results with different outcome. To provide an objective comparison of effectiveness, we report widely used two internal and three external evaluation measures, including Calinski-Harabasz index (CHI) [42], Davies-Bouldin index (DBI) [43], adjusted mutual information (AMI) [44], adjusted rand index (ARI) [44], and Fowlkes-Mallows index (FMI) [45].

5.5 Computation environment

All experiments are conducted on an Apache Spark cluster of 30 nodes equipped with Intel Xeon E5-2650 v2 2.60GHz, 128GB memory and 1.6TB of SSD. All algorithms are implemented in Python Anaconda platform. MATLAB is used to draw the summarized diagram.

5.6 Experimental result analysis

5.6.1 Comparison against state-of-the-art algorithms

The number of identified clusters with the experimented datasets and algorithms are shown in Table 3. The results show that the RSPCA-GM, RSPCA-MP, kluster, COMUSA, and U-SENC algorithms can correctly identify clusters, but the nselecboot and SNN-DPC algorithms fail to do so. However, for the three synthesis dataset, the number of clusters identified by U-SENC and SNN-DPC is much lower compared to the correct one, neither kluster and COMUSA nor U-SENC could well recognize the cluster centroids. Fig. 4 shows the percentage of relative error in estimated of number of clusters by the applied algorithms for three synthetic datasets. The nselecboot algorithm divides the candidate cluster into several sub-clusters, in which chosen centroids are inappropriate, leading to an unsatisfactory result. According to the visual judgment and results, we observe that SNN-DPC and U-SENC algorithms cannot identify small clusters.

TABLE 3: Results in approximating the number of clusters (mean±std of 25 trails).

Dataset	Method	$es = 5\%$	10%	15%	20%
2M2D10C	nselectboot	18.5±1.3	18.0±2.2	18.2±1.5	17.7±2.2
	kluster	8.5±0.6	8.7±0.5	8.6±0.6	9.2±0.5
	COMUSA	7.1±0.8	7.7±0.6	8.0±0.8	8.2±0.5
	U-SENC	5.7±1.0	7.2±0.9	7.5±0.5	8.1±0.8
	SNN-DPC	4.2±1.2	4.5±1.7	5.5±1.7	6.1±1.4
	RSPCA-GM	8.6±0.8	9.6±0.5	9.8±0.2	10.0±0.0
	RSPCA-MP	7.7±0.6	8.4±0.5	8.6±0.4	9.4±0.2
2M2D20C	nselectboot	33.2±2.2	32.6±2.1	32.7±2.5	33.3±1.1
	kluster	12.6±1.1	13.6±1.5	14.3±0.5	14.7±0.6
	COMUSA	12.0±1.6	12.2±2.1	13.7±1.2	14.0±1.4
	U-SENC	10.5±1.3	12.2±2.0	13.2±1.5	14.1±0.8
	SNN-DPC	7.7±1.8	8.2±1.2	9.1±2.2	9.7±1.7
	RSPCA-GM	14.2±1.2	17.3±1.0	18.4±0.9	19.7±0.5
	RSPCA-MP	10.4±1.8	12.5±1.4	15.8±1.2	16.9±0.8
3M2D5C	nselectboot	2.6±0.5	2.8±0.4	3.0±0.0	3.0±0.0
	kluster	4.0±0.8	4.5±0.6	4.6±0.5	4.8±0.5
	COMUSA	3.1±0.8	3.5±0.6	4.0±0.8	4.1±0.8
	U-SENC	3.2±0.5	3.7±0.9	4.0±0.8	4.2±0.9
	SNN-DPC	2.5±0.6	2.8±0.5	2.7±0.9	3.2±0.9
	RSPCA-GM	4.6±0.5	5.0±0.0	5.0±0.0	5.0±0.0
	RSPCA-MP	4.2±0.5	5.0±0.0	5.0±0.0	5.0±0.0
CT	nselectboot	2.2±0.5	2.3±0.6	2.2±0.5	2.1±0.4
	kluster	10.2±0.5	11.0±0.8	10.7±0.5	10.5±0.6
	COMUSA	8.2±1.2	10.2±1.5	10.1±1.4	10.3±1.8
	U-SENC	5.2±1.4	7.0±0.8	7.7±0.9	9.1±0.8
	SNN-DPC	3.3±0.8	3.5±1.2	4.1±0.8	4.7±1.3
	RSPCA-GM	6.2±0.8	6.8±0.8	7.8±0.9	9.1±0.5
	RSPCA-MP	5.6±0.6	7.4±0.5	9.2±0.9	9.4±0.3
PH	nselectboot	2.7±0.5	2.3±0.6	2.0±0.0	2.0±0.0
	kluster	3.5±0.6	3.6±0.5	4.0±0.0	4.0±0.0
	COMUSA	3.5±1.2	4.1±0.8	4.5±1.3	5.5±1.2
	U-SENC	3.5±0.5	4.0±0.8	4.2±0.5	4.5±0.6
	SNN-DPC	3.5±0.3	4.7±1.7	5.2±1.5	6.4±1.4
	RSPCA-GM	2.6±0.5	3.2±0.8	3.5±0.7	3.8±0.4
	RSPCA-MP	3.1±0.7	3.2±0.4	3.7±0.9	3.9±0.4
KDD	nselectboot	2.7±0.5	2.6±0.6	2.5±0.6	2.6±0.5
	kluster	9.7±0.9	10.5±0.5	10.6±0.5	9.8±1.7
	COMUSA	6.7±0.9	8.3±1.2	9.7±1.7	11.2±2.2
	U-SENC	8.7±1.7	9.5±1.2	10.1±1.4	10.7±1.2
	SNN-DPC	4.2±1.2	6.7±0.9	8.1±0.8	8.5±1.3
	RSPCA-GM	4.2±1.4	4.4±1.1	4.5±1.4	4.7±0.7
	RSPCA-MP	3.8±0.4	4.8±0.5	5.0±0.0	5.0±0.0
SUSY	nselectboot	2.6±0.7	2.5±0.5	2.3±0.5	2.2±0.5
	kluster	8.3±0.5	9.3±0.7	8.6±0.5	9.2±0.5
	COMUSA	5.2±0.9	6.7±1.2	6.5±1.3	7.7±1.5
	U-SENC	8.1±0.8	9.3±0.8	8.9±2.1	9.2±0.9
	SNN-DPC	4.5±1.2	5.0±1.6	5.2±1.2	6.1±0.8
	RSPCA-GM	2.6±0.6	2.4±0.5	3.3±0.4	3.7±0.5
	RSPCA-MP	3.2±0.4	3.4±0.5	3.8±0.8	3.9±0.5

Here RSP block size, $n = 5,000$.

TABLE 4: Clustering performance evaluation using internal validation measures (mean±std of 25 runs).

Dataset	Method	CHI (higher values better) $es=5\%$	$es=10\%$	$es=15\%$	$es=20\%$	DBI (lower values better) $es=5\%$	$es=10\%$	$es=15\%$	$es=20\%$
2M2D10C	nselectboot	**2.0**$\times10^6$ $\pm5.8\times10^4$	**4.2**$\times10^6$ $\pm9.2\times10^4$	**6.4**$\times10^6$ $\pm1.4\times10^5$	8.4$\times10^6$ $\pm2.3\times10^5$	0.789 ±0.056	0.758 ±0.112	0.817 ±0.075	0.773 ±0.088
	kluster	4.4$\times10^5$ $\pm1.3\times10^5$	1.0$\times10^6$ $\pm2.2\times10^5$	1.4$\times10^6$ $\pm3.8\times10^5$	3.9$\times10^6$ $\pm3.3\times10^6$	0.371 ±0.053	0.349 ±0.046	0.356 ±0.051	0.285 ±0.083
	COMUSA	2.4$\times10^5$ $\pm6.8\times10^4$	6.0$\times10^5$ $\pm1.1\times10^5$	1.1$\times10^6$ $\pm4.1\times10^5$	1.5$\times10^6$ $\pm4.5\times10^5$	0.479 ±0.057	0.437 ±0.036	0.410 ±0.063	0.393 ±0.045
	U-SENC	1.7$\times10^5$ $\pm4.3\times10^4$	5.4$\times10^5$ $\pm1.5\times10^5$	8.5$\times10^5$ $\pm1.7\times10^5$	1.4$\times10^6$ $\pm5.5\times10^5$	0.583 ±0.086	0.465 ±0.065	0.448 ±0.037	0.409 ±0.064
	SNN-DPC	1.3$\times10^5$ $\pm3.2\times10^4$	2.9$\times10^5$ $\pm1.2\times10^5$	6.2$\times10^5$ $\pm2.9\times10^5$	8.0$\times10^5$ $\pm3.6\times10^5$	0.748 ±0.146	0.731 ±0.182	0.574 ±0.157	0.567 ±0.113
	RSPCA-GM	5.4$\times10^5$ $\pm3.6\times10^3$	1.1$\times10^6$ $\pm7.6\times10^3$	4.1$\times10^6$ $\pm3.5\times10^5$	**8.9**$\times10^6$ $\pm1.2\times10^3$	**0.350** ±0.114	**0.226** ±0.090	**0.216** ±0.134	**0.162** ±0.002
	RSPCA-MP	2.9$\times10^5$ $\pm7.1\times10^4$	8.9$\times10^5$ $\pm3.1\times10^5$	1.3$\times10^6$ $\pm4.6\times10^5$	5.5$\times10^6$ $\pm4.7\times10^5$	0.440 ±0.036	0.375 ±0.062	0.369 ±0.048	0.243 ±0.089
2M2D20C	nselectboot	**8.2**$\times10^6$ $\pm7.8\times10^4$	**1.6**$\times10^7$ $\pm1.0\times10^5$	**2.5**$\times10^7$ $\pm3.1\times10^5$	**3.3**$\times10^7$ $\pm5.6\times10^5$	0.712 ±0.088	0.687 ±0.056	0.701 ±0.062	0.702 ±0.065
	kluster	2.5$\times10^5$ $\pm5.0\times10^4$	6.1$\times10^5$ $\pm1.5\times10^5$	9.5$\times10^5$ $\pm1.5\times10^5$	1.3$\times10^6$ $\pm1.6\times10^5$	0.489 ±0.074	0.434 ±0.085	0.393 ±0.029	0.392 ±0.037
	COMUSA	2.3$\times10^5$ $\pm5.5\times10^4$	5.1$\times10^5$ $\pm1.6\times10^5$	8.8$\times10^5$ $\pm1.5\times10^5$	1.2$\times10^6$ $\pm2.6\times10^5$	0.517 ±0.081	0.521 ±0.108	0.443 ±0.108	0.521 ±0.259
	U-SENC	1.9$\times10^5$ $\pm2.3\times10^4$	5.1$\times10^5$ $\pm1.6\times10^5$	8.2$\times10^5$ $\pm1.7\times10^5$	1.2$\times10^6$ $\pm1.8\times10^5$	0.601 ±0.074	0.521 ±0.108	0.490 ±0.113	0.422 ±0.047
	SNN-DPC	1.5$\times10^5$ $\pm2.4\times10^4$	3.3$\times10^5$ $\pm3.1\times10^5$	5.3$\times10^5$ $\pm1.1\times10^5$	7.5$\times10^5$ $\pm1.1\times10^5$	0.647 ±0.035	0.651 ±0.029	0.636 ±0.036	0.716 ±0.130
	RSPCA-GM	2.9$\times10^5$ $\pm3.1\times10^4$	1.0$\times10^6$ $\pm1.7\times10^5$	3.8$\times10^6$ $\pm1.1\times10^6$	2.1$\times10^7$ $\pm1.9\times10^6$	**0.419** ±0.068	**0.285** ±0.060	**0.212** ±0.063	**0.109** ±0.076
	RSPCA-MP	1.9$\times10^5$ $\pm4.1\times10^3$	4.8$\times10^5$ $\pm3.3\times10^4$	1.2$\times10^6$ $\pm1.3\times10^5$	2.0$\times10^6$ $\pm3.5\times10^5$	0.552 ±0.074	0.489 ±0.101	0.343 ±0.044	0.314 ±0.027
3M2D5C	nselectboot	2.1$\times10^5$ $\pm5.0\times10^4$	4.4$\times10^5$ $\pm1.1\times10^5$	7.5$\times10^5$ $\pm2.5\times10^4$	1.0$\times10^6$ $\pm2.5\times10^4$	0.723 ±0.146	0.687 ±0.161	0.596 ±0.003	0.596 ±0.002
	kluster	3.9$\times10^5$ $\pm1.7\times10^5$	8.8$\times10^5$ $\pm2.8\times10^5$	1.4$\times10^6$ $\pm4.7\times10^5$	2.0$\times10^6$ $\pm5.2\times10^5$	0.512 ±0.076	0.475 ±0.030	0.466 ±0.037	0.460 ±0.027
	COMUSA	2.5$\times10^5$ $\pm6.0\times10^4$	5.7$\times10^5$ $\pm6.3\times10^4$	1.1$\times10^6$ $\pm4.3\times10^5$	1.4$\times10^6$ $\pm5.9\times10^5$	0.635 ±0.149	0.550 ±0.051	0.512 ±0.061	0.512 ±0.062
	U-SENC	2.7$\times10^5$ $\pm3.2\times10^4$	6.9$\times10^5$ $\pm2.8\times10^5$	1.1$\times10^6$ $\pm4.3\times10^5$	1.7$\times10^6$ $\pm7.0\times10^5$	0.571 ±0.048	0.536 ±0.072	0.512 ±0.061	0.497 ±0.072
	SNN-DPC	2.1$\times10^5$ $\pm5.6\times10^4$	4.5$\times10^5$ $\pm9.5\times10^4$	6.6$\times10^5$ $\pm2.4\times10^5$	1.0$\times10^6$ $\pm2.8\times10^5$	0.728 ±0.155	0.665 ±0.139	0.715 ±0.199	0.613 ±0.165
	RSPCA-GM	**4.0**$\times10^5$ $\pm1.5\times10^5$	**1.1**$\times10^6$ $\pm1.5\times10^4$	**1.7**$\times10^6$ $\pm4.0\times10^4$	**2.3**$\times10^6$ $\pm4.0\times10^4$	**0.504** ±0.099	0.447 ±0.003	**0.445** ±0.002	**0.445** ±0.002
	RSPCA-MP	3.9$\times10^5$ $\pm1.3\times10^5$	**1.1**$\times10^6$ $\pm3.3\times10^4$	**1.7**$\times10^6$ $\pm4.6\times10^4$	**2.3**$\times10^6$ $\pm1.3\times10^4$	0.594 ±0.111	**0.446** ±0.002	0.446 ±0.006	0.446 ±0.002
CT	nselectboot	**3.1**$\times10^4$ $\pm3.2\times10^3$	**6.5**$\times10^4$ $\pm2.9\times10^3$	**9.8**$\times10^4$ $\pm1.2\times10^3$	**1.3**$\times10^5$ $\pm3.6\times10^4$	0.879 ±0.087	0.837 ±0.001	0.837 ±0.001	0.837 ±0.002
	kluster	2.7$\times10^4$ ±679.5	5.4$\times10^4$ ±561.7	8.1$\times10^4$ ±460.0	1.0$\times10^5$ ±682.8	**0.806** ±0.008	**0.810** ±0.012	**0.811** ±0.006	**0.804** ±0.010
	COMUSA	2.7$\times10^4$ ±92.6	5.4$\times10^4$ ±589.9	8.0$\times10^4$ ±888.7	1.0$\times10^5$ ±902.1	0.828 ±0.036	0.828 ±0.017	0.828 ±0.017	0.827 ±0.012
	U-SENC	2.8$\times10^4$ $\pm1.4\times10^3$	5.4$\times10^4$ ±655.3	8.0$\times10^4$ ±572.8	1.0$\times10^5$ ±574.4	0.862 ±0.024	0.849 ±0.017	0.848 ±0.018	0.826 ±0.020
	SNN-DPC	2.8$\times10^4$ $\pm3.3\times10^3$	5.7$\times10^4$ $\pm5.7\times10^3$	8.2$\times10^4$ $\pm3.1\times10^3$	1.0$\times10^5$ $\pm4.4\times10^3$	0.936 ±0.093	0.895 ±0.080	0.912 ±0.069	0.905 ±0.073
	RSPCA-GM	2.8$\times10^4$ ±386.1	5.4$\times10^4$ ±740.9	8.1$\times10^4$ ±819.3	1.1$\times10^5$ ±211.6	0.869 ±0.025	0.854 ±0.012	0.852 ±0.023	0.830 ±0.026
	RSPCA-MP	2.9$\times10^4$ $\pm1.0\times10^3$	5.4$\times10^4$ ±655.1	8.0$\times10^4$ ±279.0	1.1$\times10^5$ $\pm1.1\times10^3$	0.856 ±0.011	0.870 ±0.023	0.828 ±0.024	0.825 ±0.013
PH	nselectboot	2.9$\times10^4$ $\pm3.0\times10^3$	5.9$\times10^4$ $\pm4.8\times10^3$	1.1$\times10^5$ $\pm1.5\times10^4$	2.1$\times10^5$ ±136.5	1.123 ±0.156	1.108 ±0.145	1.103 ±0.139	1.192 ±0.088
	kluster	3.3$\times10^4$ $\pm1.1\times10^3$	6.8$\times10^4$ $\pm2.3\times10^3$	2.4$\times10^5$ ±576.3	1.3$\times10^5$ ±326.4	0.921 ±0.020	0.917 ±0.021	0.904 ±0.003	0.907 ±0.003
	COMUSA	3.2$\times10^4$ $\pm3.1\times10^3$	6.6$\times10^4$ $\pm1.8\times10^3$	1.3$\times10^5$ $\pm2.5\times10^3$	2.3$\times10^5$ $\pm6.5\times10^3$	0.993 ±0.148	0.917 ±0.017	0.901 ±0.040	0.875 ±0.041
	U-SENC	3.3$\times10^4$ $\pm1.1\times10^3$	6.7$\times10^4$ $\pm2.1\times10^3$	2.4$\times10^5$ $\pm5.9\times10^3$	1.3$\times10^5$ $\pm3.7\times10^3$	0.921 ±0.020	0.917 ±0.017	0.907 ±0.003	0.906 ±0.001
	SNN-DPC	**6.1**$\times10^6$ $\pm4.7\times10^6$	**1.8**$\times10^7$ $\pm8.5\times10^6$	**3.2**$\times10^7$ $\pm9.8\times10^6$	**4.9**$\times10^7$ $\pm1.1\times10^7$	**0.348** ±0.095	**0.353** ±0.099	**0.403** ±0.083	**0.561** ±0.267
	RSPCA-GM	3.0$\times10^4$ $\pm3.7\times10^3$	6.7$\times10^4$ $\pm2.8\times10^3$	1.4$\times10^5$ $\pm4.7\times10^3$	2.4$\times10^5$ $\pm4.2\times10^3$	0.994 ±0.138	0.987 ±0.144	0.981 ±0.147	0.907 ±0.013
	RSPCA-MP	3.4$\times10^4$ $\pm1.1\times10^3$	6.6$\times10^4$ $\pm2.3\times10^3$	1.3$\times10^5$ ±326.4	2.4$\times10^5$ ±576.3	0.915 ±0.019	0.905 ±0.042	0.837 ±0.009	0.845 ±0.008
KDD	nselectboot	3.2$\times10^6$ $\pm1.7\times10^6$	5.9$\times10^6$ $\pm4.0\times10^6$	7.1$\times10^6$ $\pm6.1\times10^6$	1.1$\times10^7$ $\pm8.0\times10^6$	**0.283** ±0.113	**0.318** ±0.128	0.425 ±0.269	0.403 ±0.128
	kluster	1.3$\times10^7$ $\pm1.8\times10^5$	2.6$\times10^7$ $\pm1.7\times10^5$	4.1$\times10^7$ $\pm5.7\times10^5$	**5.4**$\times10^7$ $\pm2.0\times10^6$	0.578 ±0.048	0.602 ±0.040	0.627 ±0.009	0.566 ±0.080
	COMUSA	1.3$\times10^7$ $\pm2.9\times10^5$	**2.7**$\times10^7$ $\pm5.4\times10^5$	**4.2**$\times10^7$ $\pm7.5\times10^5$	5.3$\times10^7$ $\pm1.5\times10^6$	0.482 ±0.017	0.496 ±0.044	0.562 ±0.065	0.600 ±0.027
	U-SENC	**1.4**$\times10^7$ $\pm2.8\times10^5$	2.6$\times10^7$ $\pm5.6\times10^5$	4.1$\times10^7$ $\pm6.8\times10^5$	5.3$\times10^7$ $\pm1.8\times10^6$	0.524 ±0.073	0.554 ±0.065	0.591 ±0.073	0.619 ±0.058
	SNN-DPC	3.4$\times10^4$ ±836.4	6.9$\times10^4$ ±650.0	1.3$\times10^5$ $\pm1.3\times10^3$	2.3$\times10^5$ $\pm3.0\times10^3$	0.890 ±0.030	0.843 ±0.008	0.853 ±0.006	0.848 ±0.012
	RSPCA-GM	5.4$\times10^6$ $\pm2.1\times10^6$	1.8$\times10^7$ $\pm3.3\times10^6$	2.7$\times10^7$ $\pm5.2\times10^6$	4.1$\times10^7$ $\pm7.5\times10^6$	0.373 ±0.092	0.360 ±0.085	0.401 ±0.088	0.396 ±0.057
	RSPCA-MP	6.8$\times10^6$ $\pm2.1\times10^6$	2.0$\times10^7$ $\pm3.6\times10^6$	3.4$\times10^7$ $\pm3.1\times10^5$	4.6$\times10^7$ $\pm1.0\times10^6$	0.310 ±0.045	0.367 ±0.026	**0.384** ±0.017	**0.368** ±0.012
SUSY	nselectboot	1.7$\times10^5$ $\pm7.0\times10^3$	**3.9**$\times10^5$ $\pm1.9\times10^4$	**6.1**$\times10^5$ $\pm5.7\times10^4$	**8.0**$\times10^5$ $\pm5.8\times10^4$	0.943 ±0.070	0.952 ±0.030	0.924 ±0.021	0.937 ±0.023
	kluster	1.6$\times10^5$ $\pm3.1\times10^3$	3.7$\times10^5$ $\pm2.2\times10^4$	5.8$\times10^5$ $\pm2.1\times10^4$	7.6$\times10^5$ $\pm2.7\times10^4$	0.838 ±0.019	0.831 ±0.007	0.837 ±0.020	0.844 ±0.017
	COMUSA	1.6$\times10^5$ $\pm7.9\times10^3$	3.7$\times10^5$ $\pm1.2\times10^4$	5.9$\times10^5$ $\pm1.8\times10^4$	7.7$\times10^5$ $\pm2.1\times10^4$	**0.827** ±0.020	0.834 ±0.009	0.841 ±0.015	0.845 ±0.024
	U-SENC	1.7$\times10^5$ $\pm7.2\times10^3$	3.7$\times10^5$ $\pm1.7\times10^4$	5.8$\times10^5$ $\pm2.5\times10^4$	7.7$\times10^5$ $\pm2.7\times10^4$	0.835 ±0.024	0.839 ±0.009	**0.832** ±0.028	0.841 ±0.021
	SNN-DPC	**1.8**$\times10^5$ $\pm4.1\times10^3$	3.8$\times10^5$ $\pm1.9\times10^4$	5.8$\times10^5$ $\pm2.3\times10^4$	7.9$\times10^5$ $\pm1.4\times10^4$	0.880 ±0.067	0.835 ±0.046	0.839 ±0.041	0.836 ±0.021
	RSPCA-GM	**1.8**$\times10^5$ $\pm5.4\times10^3$	3.6$\times10^5$ ±334.8	5.8$\times10^5$ $\pm2.4\times10^4$	7.5$\times10^5$ $\pm1.3\times10^4$	0.914 ±0.073	0.922 ±0.085	0.901 ±0.075	0.824 ±0.013
	RSPCA-MP	**1.8**$\times10^5$ $\pm3.5\times10^3$	3.8$\times10^5$ $\pm2.4\times10^4$	5.6$\times10^5$ $\pm1.1\times10^4$	7.5$\times10^5$ $\pm1.5\times10^4$	0.852 ±0.031	**0.820** ±0.022	0.835 ±0.023	**0.809** ±0.030

Here RSP block size, $n = 5000$. The best and second best results are shown in bold and underline, respectively.

The RSPCA-GM and RSPCA-MP schemes recognize both the clusters and the corresponding centers perfectly. It is observed that the approximation of clusters of RSPCA-GM and RSPCA-MP schemes gradually confer to ensemble sizes. We were curious about the quantity of data required for the RSPCA-GM and RSPCA-MP schemes to obtain the best and stable solution. Based on the obtained results, RSPCA-GM and RSPCA-MP schemes achieve stability in $\approx 15-20\%$ of ensemble data of the dataset.

Table 4 and Table 5 show the clustering quality in terms of the internal and external validation measure, respectively. As shown in the tables, RSPCA-GM and RSPCA-MP schemes consistently outperform other clustering methods w.r.t. CHI, DBI, AMI, ARI, and FMI on the synthetic and real-word datasets under the varying ensemble cardinality. It is also noted that the clustering performance of the methods improves with the increasing percentage of clustering ensemble size.

To show the performance of the RSP data model, we also conduct experiments with all the clustering algorithms on the RSP data blocks. We find that the candidate clustering algorithms achieve an outstanding improvement in estimating the number of clusters while considering RSP data model. In Fig. 5, we show the estimated number of clusters of the three synthesis datasets. The corresponding averages of 25 trails are reported. Notice that the RSP data model leads to a higher accuracy regarding the estimation on the number of clusters compared to the typical one without RSP settings.

Running Time: Fig. 6 shows the running time of our RSP-based method and other state-of-the-art clustering algorithms on the experimented datasets. The U-SPEC and kluster require a lower computational cost compared to the other methods on most of the datasets. The CPU time re-

TABLE 5: Clustering performance evaluation using external validation measures (mean±std of 25 runs).

Dataset	Method	AMI (higher values better)				ARI (higher values better)				FMI (higher values better)			
		$es=5\%$	$es=10\%$	$es=15\%$	$es=20\%$	$es=5\%$	$es=10\%$	$es=15\%$	$es=20\%$	$es=5\%$	$es=10\%$	$es=15\%$	$es=20\%$
2M2D10C	nselectboot	0.878±0.012	0.886±0.018	0.883±0.013	0.884±0.023	0.673±0.032	0.703±0.039	0.695±0.030	0.691±0.066	0.734±0.025	0.778±0.046	0.754±0.024	0.748±0.051
	kluster	0.960±0.012	0.965±0.010	0.963±0.012	0.975±0.010	0.900±0.035	0.912±0.030	0.910±0.035	0.948±0.035	0.902±0.029	0.915±0.025	0.923±0.029	0.955±0.030
	COMUSA	0.903±0.040	0.940±0.017	0.948±0.021	0.955±0.010	0.753±0.114	0.857±0.023	0.875±0.041	0.885±0.030	0.808±0.079	0.880±0.017	0.895±0.033	0.903±0.025
	U-SENC	0.848±0.054	0.919±0.044	0.933±0.014	0.947±0.022	0.655±0.123	0.804±0.108	0.849±0.022	0.874±0.042	0.741±0.083	0.843±0.075	0.876±0.017	0.896±0.033
	SNN-DPC	0.745±0.088	0.760±0.117	0.852±0.098	0.854±0.066	0.508±0.100	0.555±0.188	0.685±0.191	0.664±0.141	0.646±0.063	0.678±0.125	0.765±0.130	0.748±0.098
	RSPCA-GM	**0.974**±0.024	**0.990**±0.014	**0.995**±0.032	**0.998**±0.005	**0.939**±0.058	**0.972**±0.038	**0.972**±0.069	**1.000**±0.000	**0.939**±0.048	**0.976**±0.032	**0.977**±0.056	**1.000**±0.000
	RSPCA-MP	0.946±0.013	0.957±0.015	0.966±0.015	0.985±0.014	0.852±0.021	0.894±0.031	0.915±0.032	0.958±0.039	0.889±0.016	0.922±0.025	0.930±0.026	0.964±0.033
2M2D20C	nselectboot	0.916±0.013	0.926±0.006	0.926±0.012	0.923±0.006	0.762±0.048	0.770±0.027	0.770±0.040	0.767±0.015	0.796±0.039	0.797±0.023	0.800±0.030	0.797±0.015
	kluster	0.913±0.015	0.927±0.015	0.940±0.014	0.940±0.010	0.753±0.042	0.778±0.042	0.805±0.035	0.813±0.021	0.790±0.036	0.817±0.032	0.830±0.028	0.837±0.015
	COMUSA	0.903±0.025	0.908±0.029	0.930±0.016	0.933±0.017	0.725±0.066	0.733±0.074	0.790±0.034	0.795±0.040	0.768±0.054	0.775±0.053	0.818±0.025	0.760±0.154
	U-SENC	0.876±0.026	0.906±0.031	0.923±0.019	0.932±0.009	0.665±0.059	0.731±0.072	0.773±0.044	0.794±0.019	0.726±0.043	0.775±0.054	0.806±0.035	0.823±0.015
	SNN-DPC	0.804±0.051	0.824±0.034	0.833±0.053	0.860±0.038	0.521±0.095	0.558±0.065	0.579±0.118	0.633±0.082	0.623±0.066	0.649±0.046	0.665±0.085	0.639±0.081
	RSPCA-GM	**0.933**±0.015	**0.973**±0.010	**0.984**±0.009	**0.994**±0.006	**0.807**±0.033	**0.917**±0.037	**0.952**±0.022	**0.980**±0.019	**0.832**±0.028	**0.925**±0.035	**0.958**±0.022	**0.982**±0.016
	RSPCA-MP	0.882±0.036	0.934±0.020	0.952±0.013	0.968±0.008	0.692±0.079	0.754±0.042	0.852±0.036	0.895±0.033	0.724±0.079	0.798±0.042	0.870±0.036	0.905±0.033
3M2D5C	nselectboot	0.618±0.127	0.655±0.126	0.725±0.006	0.726±0.005	0.448±0.136	0.528±0.136	0.605±0.005	0.605±0.004	0.674±0.072	0.695±0.071	0.735±0.004	0.735±0.003
	kluster	0.844±0.121	0.900±0.081	0.912±0.085	0.935±0.074	0.782±0.188	0.867±0.131	0.904±0.134	0.924±0.114	0.850±0.126	0.904±0.093	0.930±0.095	0.945±0.081
	COMUSA	0.699±0.136	0.777±0.056	0.838±0.099	0.839±0.099	0.585±0.158	0.679±0.083	0.773±0.154	0.773±0.155	0.728±0.087	0.776±0.052	0.842±0.103	0.842±0.104
	U-SENC	0.754±0.051	0.813±0.114	0.838±0.099	0.872±0.121	0.644±0.074	0.737±0.176	0.773±0.154	0.829±0.186	0.759±0.044	0.820±0.117	0.842±0.104	0.881±0.125
	SNN-DPC	0.619±0.128	0.670±0.108	0.645±0.161	0.723±0.150	0.489±0.138	0.545±0.116	0.526±0.188	0.620±0.180	0.675±0.073	0.702±0.059	0.697±0.103	0.749±0.100
	RSPCA-GM	**0.920**±0.064	**0.968**±0.005	**0.968**±0.005	**0.968**±0.005	**0.910**±0.099	**0.978**±0.005	**0.980**±0.000	**0.980**±0.000	**0.930**±0.071	**0.982**±0.005	**0.980**±0.000	**0.980**±0.000
	RSPCA-MP	0.878±0.052	**0.968**±0.005	**0.968**±0.005	0.967±0.002	0.856±0.076	0.978±0.002	0.979±0.003	0.978±0.002	0.892±0.055	0.983±0.001	0.984±0.002	0.983±0.001
CT	nselectboot	0.057±0.011	0.052±0.001	0.052±0.002	0.052±0.001	-0.009±0.013	-0.015±0.001	-0.016±0.002	0.001±0.004	**0.410**±0.038	**0.429**±0.001	**0.429**±0.001	**0.428**±0.001
	kluster	**0.089**±0.002	**0.091**±0.003	0.091±0.001	**0.094**±0.001	**0.012**±0.001	**0.012**±0.001	**0.014**±0.000	0.014±0.000	0.225±0.006	0.220±0.005	0.220±0.004	0.223±0.005
	COMUSA	0.086±0.005	**0.091**±0.001	**0.093**±0.001	0.093±0.002	**0.012**±0.007	**0.012**±0.001	**0.014**±0.001	0.013±0.002	0.244±0.012	0.228±0.014	0.229±0.014	0.230±0.015
	U-SENC	0.076±0.009	0.085±0.006	0.089±0.005	0.093±0.001	0.000±0.004	0.007±0.005	0.010±0.005	0.013±0.001	0.298±0.033	0.258±0.009	0.251±0.010	0.237±0.007
	SNN-DPC	0.065±0.011	0.067±0.012	0.070±0.007	0.075±0.008	-0.001±0.013	-0.002±0.011	0.001±0.007	0.003±0.007	0.362±0.046	0.355±0.065	0.327±0.031	0.314±0.040
	RSPCA-GM	0.081±0.006	0.084±0.006	0.088±0.010	**0.094**±0.007	0.006±0.002	0.007±0.006	0.012±0.007	**0.015**±0.003	0.354±0.002	0.322±0.006	0.325±0.007	0.297±0.003
	RSPCA-MP	0.082±0.006	0.088±0.007	0.091±0.006	0.093±0.006	0.007±0.004	0.009±0.007	**0.014**±0.003	**0.015**±0.003	0.322±0.029	0.287±0.013	0.287±0.027	0.258±0.007
PH	nselectboot	0.001±0.002	0.001±0.002	0.000±0.000	0.000±0.000	0.001±0.002	0.001±0.002	0.000±0.000	0.000±0.000	**0.438**±0.048	**0.436**±0.048	**0.437**±0.047	**0.455**±0.001
	kluster	0.005±0.003	0.006±0.003	0.007±0.001	0.007±0.001	0.003±0.003	0.003±0.002	0.003±0.001	0.003±0.001	0.357±0.030	0.349±0.029	0.331±0.001	0.332±0.001
	COMUSA	0.006±0.005	**0.008**±0.004	0.007±0.004	0.009±0.002	**0.004**±0.003	0.005±0.003	0.005±0.002	**0.006**±0.002	0.371±0.072	0.329±0.038	0.323±0.045	0.292±0.031
	U-SENC	0.005±0.003	0.007±0.003	0.007±0.001	0.008±0.002	0.003±0.001	0.004±0.002	0.004±0.003	**0.006**±0.003	0.357±0.030	0.337±0.034	0.324±0.015	0.317±0.018
	SNN-DPC	**0.007**±0.001	**0.008**±0.001	**0.010**±0.001	**0.010**±0.001	**0.004**±0.002	**0.006**±0.000	0.006±0.000	**0.006**±0.000	0.310±0.027	0.264±0.016	0.245±0.012	0.239±0.017
	RSPCA-GM	0.002±0.001	0.005±0.002	0.005±0.004	0.006±0.003	0.003±0.002	0.004±0.002	0.005±0.003	0.005±0.001	0.414±0.045	0.379±0.054	0.369±0.058	0.347±0.021
	RSPCA-MP	0.006±0.004	0.006±0.003	0.009±0.002	0.009±0.002	**0.004**±0.003	0.005±0.003	**0.007**±0.001	**0.006**±0.001	0.339±0.041	0.295±0.011	0.264±0.006	0.256±0.008
KDD	nselectboot	0.804±0.104	0.791±0.118	0.756±0.119	0.791±0.118	0.831±0.188	0.800±0.217	0.799±0.177	0.800±0.217	0.917±0.080	0.903±0.092	0.877±0.092	0.904±0.091
	kluster	0.794±0.026	0.784±0.023	0.769±0.007	0.795±0.030	0.891±0.021	0.887±0.019	0.872±0.002	0.891±0.022	0.935±0.013	0.933±0.012	0.924±0.002	0.935±0.013
	COMUSA	0.805±0.004	0.817±0.009	0.794±0.029	0.780±0.026	0.902±0.001	0.908±0.003	0.891±0.021	0.881±0.018	0.937±0.001	0.946±0.002	0.935±0.013	0.929±0.011
	U-SENC	0.800±0.027	0.802±0.026	0.782±0.027	0.779±0.026	0.900±0.020	0.898±0.017	0.882±0.018	0.881±0.018	0.931±0.012	0.940±0.011	0.930±0.011	0.928±0.011
	SNN-DPC	0.801±0.105	0.843±0.025	0.812±0.032	0.839±0.029	0.838±0.190	0.906±0.017	0.903±0.023	0.920±0.020	0.920±0.081	0.944±0.010	0.939±0.014	0.953±0.012
	RSPCA-GM	0.806±0.092	**0.857**±0.017	0.815±0.110	0.848±0.017	0.848±0.159	**0.929**±0.015	0.860±0.166	0.923±0.014	0.925±0.068	**0.958**±0.009	0.928±0.074	0.955±0.008
	RSPCA-MP	**0.877**±0.014	0.856±0.016	**0.850**±0.006	**0.854**±0.006	**0.943**±0.014	0.928±0.011	**0.923**±0.002	**0.925**±0.002	**0.966**±0.008	0.957±0.006	**0.954**±0.001	**0.959**±0.001
SUSY	nselectboot	0.065±0.002	0.077±0.009	0.081±0.014	0.081±0.010	**0.078**±0.014	**0.091**±0.022	**0.100**±0.030	**0.103**±0.024	**0.535**±0.034	**0.527**±0.044	**0.544**±0.046	**0.549**±0.039
	kluster	0.082±0.002	0.083±0.002	0.083±0.001	0.082±0.002	0.058±0.004	0.058±0.004	0.059±0.001	0.057±0.003	0.345±0.012	0.335±0.008	0.340±0.008	0.334±0.007
	COMUSA	**0.083**±0.003	**0.084**±0.003	**0.084**±0.003	**0.083**±0.002	0.066±0.002	0.070±0.011	0.070±0.010	0.062±0.005	0.404±0.035	0.384±0.037	0.389±0.037	0.360±0.028
	U-SENC	0.081±0.002	0.080±0.001	0.082±0.003	0.080±0.002	0.062±0.005	0.059±0.002	0.059±0.008	0.058±0.003	0.354±0.017	0.338±0.009	0.340±0.031	0.338±0.011
	SNN-DPC	0.079±0.004	0.082±0.011	0.087±0.003	0.086±0.004	0.067±0.001	0.076±0.008	0.077±0.009	0.071±0.009	0.453±0.047	0.442±0.059	0.429±0.037	0.399±0.026
	RSPCA-GM	0.058±0.013	0.066±0.020	0.067±0.012	0.072±0.011	0.079±0.007	0.070±0.018	0.078±0.007	0.084±0.008	0.534±0.021	0.518±0.050	0.524±0.019	0.497±0.038
	RSPCA-MP	0.065±0.007	0.071±0.016	0.072±0.008	0.078±0.012	0.072±0.007	0.069±0.006	0.069±0.014	0.069±0.013	0.505±0.023	0.487±0.033	0.468±0.047	0.440±0.012

Here RSP block size $n = 5,000$. The best and second best results are shown in bold and underline, respectively.

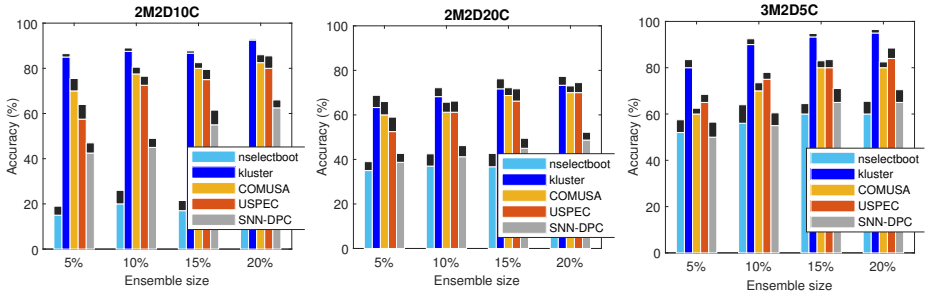


Fig. 5: Cluster number approximation accuracy of compared methods in RSP data model for three synthesis datasets (averages of 25 trails). The bars display the average accuracy in typical settings, and the black shade (on the top) denotes improved accuracy in the RSP data model.

quired by RSPCA-GM, RSPCA-MP, nselectboot, COMUSA and SNN-DPC algorithms grows linearly with the ensemble size (es), where the CPU time needed by these methods is mainly independent of K. We were curious about the required computation duration of RSPCA-GM and RSPCA-MP schemes to obtain the best possible solution scale with ensemble cardinality. Both the RSPCA-GM and RSPCA-MP schemes provide better scalability and perform substantially better in the same duration of time.

5.6.2 *Statistical analysis*

To evaluate the estimated centroids of different algorithms, we have also considered the two-samples Kolmogorov-Smirnov (K-S) test to the results in Table 6. The explanatory K-S test is offered to determine whether two distributions (i.e., centroids distance distributions of estimated and actual) are different or the same. We have analyzed cumulative distribution functions (CDF) of the centroids distance distributions that we compare in Fig. 7c under three synthesis datasets with different ensemble sizes, $es = \{10\%, 20\%\}$.

In order to check whether the algorithm is significantly

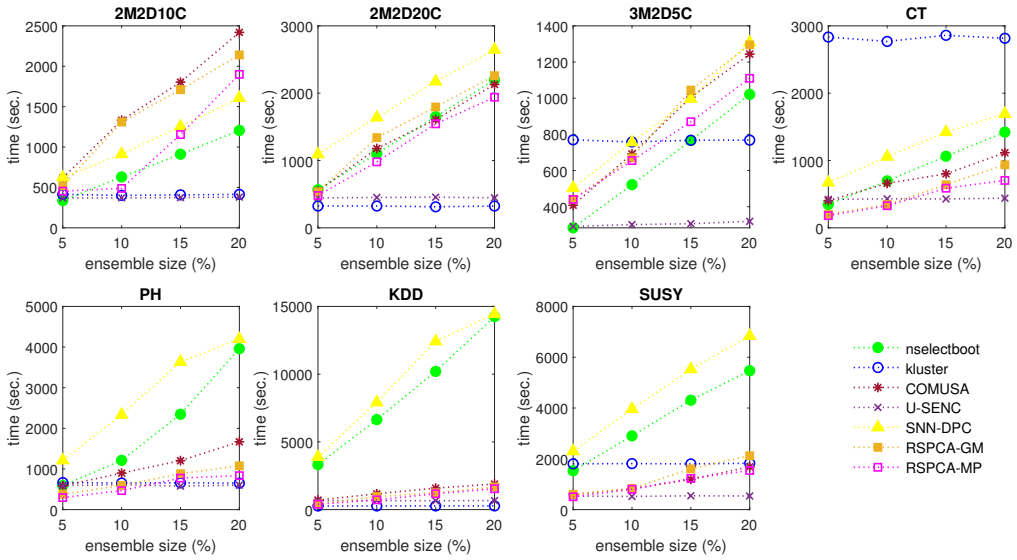


Fig. 6: Execution time (average of 25 runs) of our methods and the compared clustering methods.

compared to the others, the k and p-value of the K-S test are computed in practice, which represents the level of significance of a hypothesis. The K-S statistic (k) quantifies a distance between two distributions. According to the K-S test, a p is the probability of observing a test statistic (k) as extreme as the observed value under the null hypothesis. The null hypothesis for this test assumes that the distance distribution between the actual and estimated centroids (i.e., CDF) of the algorithm is same, whereas the alternative hypothesis that the CDF of the estimated one is larger (different) compared to the actual one. We can observe that the proposed RSPCA-GM and RSPCA-MP are statistically better compared to five competitors at the 95% confidence level (i.e., Alpha). In other words, the proposed RSPCA-based schemes are the "winner" from this perspective.

TABLE 6: Kolmogorov-Smirnov (K-S) test results. k: test statistic and p: probability of observing a test statistic. (higher p value is better)

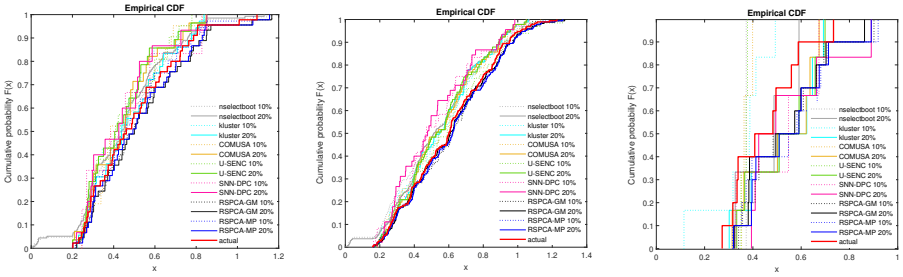
Dataset	Method	$es = 10\%$		$es = 20\%$	
		k	p	k	p
2M2D10C	nselectboot	0.102	0.884	0.114	0.747
	kluster	0.168	0.671	0.163	0.705
	COMUSA	0.168	0.773	0.167	0.677
	U-SENC	0.152	0.865	0.186	0.546
	SNN-DPC	0.244	0.859	0.333	0.263
	RSPCA-GM	0.111	0.930	0.111	0.929
	RSPCA-MP	0.117	0.961	0.083	0.998
2M2D20C	nselectboot	0.118	0.086	0.122	0.043
	kluster	0.111	0.553	0.117	0.347
	COMUSA	0.094	0.754	0.110	0.426
	U-SENC	0.088	0.827	0.093	0.709
	SNN-DPC	0.176	0.275	0.213	0.063
	RSPCA-GM	0.066	0.894	0.053	0.952
	RSPCA-MP	0.061	0.900	0.068	0.751
3M2D5C	nselectboot	0.600	0.689	0.267	0.785
	kluster	0.333	0.703	0.400	0.472
	COMUSA	0.600	0.242	0.400	0.473
	U-SENC	0.600	0.242	0.400	0.472
	SNN-DPC	0.366	0.725	0.400	0.473
	RSPCA-GM	0.400	0.313	0.300	0.675
	RSPCA-MP	0.400	0.312	0.300	0.671

5.6.3 *Discussions*

Although the distributed and parallel data clustering model is used in the current mainstream big data paradigms, computing an entire dataset is inefficient due to the high computational cost. However, as we have shown in this paper, if a big dataset is represented as a RSP data model, the clustering analysis of the dataset is turned to the analysis of a few randomly selected RSP blocks. After the conversion of a big dataset into the RSP data model, the entire dataset is no longer needed to be analyzed as a whole to estimate clustering properties. Consequently, we ensemble estimations and models computed from a few RSP blocks via a serial clustering algorithm or in a parallel manner. The expensive record-level sample data clustering process of taking random samples from a big dataset can also be replaced by an efficient block-level sample clustering process. As we see in the experiment results, a few RSP blocks are sufficient to obtain a good clustering approximation.

6 Conclusions

In this paper, we presented a distributed, randomized and incremental clustering approximation method namely, RSPCA, for clustering large-scale data. In consequence, for RSPCA, two graph-based ensemble schemes RSPCA-GM and RSPCA-MP were designed. Specifically, we introduced a RSP-based LMGI framework that allows execution of serial algorithms independently and uses multiple random RSP data blocks to enable clustering approximation of a large-scale dataset. Extensive experiments had been conducted on seven large-scale datasets, which demonstrate the scalability and approximation robustness of our proposed algorithm. RSPCA yields a more accurate and interpretable solution compared to the existing clustering methods. Therefore, the purported effective processing and analyzing advantage of RSPCA, and capability to identify the number of clusters and tracking cluster without addi-



(a) 2M2D10C (b) 2M2D20C (c) 3M2D5C

Fig. 7: Cumulative distribution functions of the centroids distance distributions of three synthesis datasets. For each method, two distributions are drawn from ensemble sizes 10% and 20%.

tional priori knowledge, makes it a desirable algorithm for the exploration of big data.

Acknowledgments

This research has been supported by the National Natural Science Foundation of China Grant-61473194.

References

[1] S. Ma and J. Huai, "Approximate computation for big data analytics," *SIGWEB Newsl.*, 2021.
[2] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai, and C.-K. Kwoh, "Ultra-scalable spectral clustering and ensemble clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 6, pp. 1212–1226, 2020.
[3] E. Lughofer, "A dynamic split-and-merge approach for evolving cluster models," *Evolving Systems*, vol. 3, pp. 135–151, 2012.
[4] S. Mimaroglu and E. Erdil, "An efficient and scalable family of algorithms for combining clusterings," *Eng. Appl. Artif. Intell.*, vol. 26, no. 10, p. 2525–2539, 2013.
[5] E. S. Edgington, *Randomization Tests*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1182–1183.
[6] J. A. Ramos Rojas, M. Beth Kery, S. Rosenthal, and A. Dey, "Sampling techniques to improve big data exploration," in *2017 IEEE 7th Symposium on Large Data Analysis and Visualization (LDAV)*, 2017, pp. 26–35.
[7] M. S. Mahmud, J. Z. Huang, S. Salloum, T. Z. Emara, and K. Sadatdiynov, "A survey of data partitioning and sampling methods to support big data analysis," *Big Data Mining and Analytics*, vol. 3, no. 2, pp. 85–101, 2020.
[8] S. Salloum, J. Z. Huang, and Y. He, "Random sample partition: A distributed data model for big data analysis," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 11, pp. 5846–5854, 2019.
[9] N. Iam-On, T. Boongoen, S. Garrett, and C. Price, "A link-based approach to the cluster ensemble problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2396–2409, 2011.
[10] D. Huang, J. Lai, and C.-D. Wang, "Ensemble clustering using factor graph," *Pattern Recognition*, vol. 50, pp. 131–142, 2016.
[11] C. Domeniconi and M. Al-Razgan, "Weighted cluster ensembles: Methods and analysis," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 4, 2009.
[12] Y. Wang and L. Chen, "K-meap: Multiple exemplars affinity propagation with specified k clusters," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2670–2682, 2016.
[13] J.-X. Chen, Y.-J. Gong, W.-N. Chen, M. Li, and J. Zhang, "Elastic differential evolution for automatics data clustering," *IEEE Transactions on Cybernetics*, vol. 51, no. 8, pp. 4134–4147, 2021.
[14] G. Kollios, D. Gunopulos, N. Koudas, and S. Berchtold, "Efficient biased sampling for approximate clustering and outlier detection in large data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 5, pp. 1170–1187, 2003.
[15] G. R. Andrews, *Foundations of Parallel and Distributed Programming*, 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc., 2002.
[16] Z. Yu, P. Luo, J. You, H.-S. Wong, H. Leung, S. Wu, J. Zhang, and G. Han, "Incremental semi-supervised clustering ensemble for high dimensional data clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 3, pp. 701–714, 2016.
[17] Y. Fang and J. Wang, "Selection of the number of clusters via the bootstrap method," *Computational Statistics & Data Analysis*, vol. 56, no. 3, pp. 468–477, 2012.
[18] F. Yang, T. Li, Q. Zhou, and H. Xiao, "Cluster ensemble selection with constraints," *Neurocomput.*, vol. 235, no. C, p. 59–70, 2017.
[19] A. M. Bagirov, J. Ugon, and D. Webb, "Fast modified global k-means algorithm for incremental cluster construction," *Pattern Recogn.*, vol. 44, no. 4, pp. 866–876, 2011.
[20] Y. Wang, L. Chen, and J.-P. Mei, "Incremental fuzzy clustering with multiple medoids for large data," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 6, pp. 1557–1568, 2014.
[21] J. Hu, T. Li, C. Luo, H. Fujita, and Y. Yang, "Incremental fuzzy cluster ensemble learning based on rough set theory," *Knowledge-Based Systems*, vol. 132, pp. 144–155, 2017.
[22] K. Hammouda and M. Kamel, "Efficient phrase-based document indexing for web document clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 10, pp. 1279–1296, 2004.
[23] R. Gil-García and A. Pons-Porrata, "Dynamic hierarchical algorithms for document clustering," *Pattern Recognition Letters*, vol. 31, no. 6, pp. 469–477, 2010.
[24] A. Pérez-Suárez, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and J. E. Medina-Pagola, "An algorithm based on density and compactness for dynamic overlapping clustering," *Pattern Recognition*, vol. 46, no. 11, pp. 3040–3055, 2013.
[25] N. Labroche, "Online fuzzy medoid based clustering algorithms," *Neurocomputing*, vol. 126, pp. 141–150, 2014.
[26] H. Wang, Y. Yang, and B. Liu, "Gmc: Graph-based multi-view clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 6, pp. 1116–1129, 2020.
[27] P. Mitra, C. Murthy, and S. Pal, "Density-based multiscale data condensation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 734–747, 2002.
[28] M. Bundefineddoiu, S. Har-Peled, and P. Indyk, "Approximate clustering via core-sets," in *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, ser. STOC'02, New York, NY, USA, 2002, p. 250–257.
[29] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, "Algorithms for facility location problems with outliers," in *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '01, USA, 2001, p. 642–651.
[30] K. Chen, "On coresets for k-median and k-means clustering in metric and euclidean spaces and their applications," *SIAM J. Comput.*, vol. 39, no. 3, p. 923–947, 2009.
[31] S. Gupta, R. Kumar, K. Lu, B. Moseley, and S. Vassilvitskii, "Local search methods for k-means with outliers," *Proc. VLDB Endow.*, vol. 10, no. 7, p. 757–768, 2017.
[32] C. Wei, S. Salloum, T. Z. Emara, X. Zhang, J. Z. Huang, and Y. He, "A two-stage data processing algorithm to generate random sample partitions for big data analysis," in *Cloud Computing - CLOUD 2018 - 11th International Conferenceon Cloud Computing*, ser. Lecture Notes in Computer Science, M. Luo and L. Zhang, Eds., vol. 10967. Springer, 2018, pp. 347–364.
[33] M. A. Masud, J. Z. Huang, C. Wei, J. Wang, I. Khan, and M. Zhong, "I-nice: A new approach for identifying the number of clusters and initial cluster centres," *Information Sciences*, vol. 466, pp. 129–151, 2018.
[34] Y. He, Y. Wu, H. Qin, J. Z. Huang, and Y. Jin, "Improved i-nice clustering algorithm based on density peaks mechanism," *Information Sciences*, vol. 548, pp. 177–190, 2021.

[35] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Sci. Comput.*, vol. 20, no. 1, p. 359–392, 1998.

[36] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[37] X. Zhao, J. Liang, and C. Dang, "A stratified sampling based clustering algorithm for large-scale data," *Knowledge-Based Systems*, vol. 163, pp. 416–428, 2019.

[38] L. Wang, J. C. Bezdek, C. Leckie, and K. Ramamohanarao, "Selective sampling for approximate clustering of very large data sets," *International Journal of Intelligent Systems*, vol. 23, no. 3, pp. 313–331, 2008.

[39] H. Estiri, B. A. Omran, and S. N. Murphy, "kluster: An efficient scalable procedure for approximating the number of clusters in unsupervised learning," *Big Data Res.*, vol. 13, pp. 38–51, 2018.

[40] S. Mimaroglu and E. Erdil, "Combining multiple clusterings using similarity graph," *Pattern Recogn.*, vol. 44, no. 3, p. 694–703, 2011.

[41] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Information Sciences*, vol. 450, pp. 200–226, 2018.

[42] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, vol. 3, no. 1, pp. 1–27, 1974.

[43] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224–227, 1979.

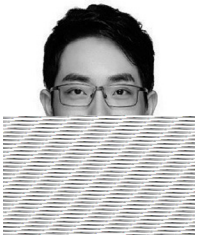
[44] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, 2010.

[45] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings," *Journal of the American Statistical Association*, vol. 78, no. 383, pp. 553–569, 1983.

Mohammad Sultan Mahmud is currently a PhD candidate at Shenzhen University, China. He received the master degree from King Mongkut's University of Technology North Bangkok, Thailand, in 2014 and bachelor degree from BGC Trust University Bangladesh, in 2009. Mr. Mahmud was awarded the Outstanding Doctoral Student of Shenzhen University in 2017 and Shenzhen Universiade International Scholarship in 2018. His current research focuses on big data mining and distributed and parallel computing.

Joshua Zhexue Huang received the Ph.D. degree from the Royal Institute of Technology, Sweden, in 1993. He is a distinguished professor at Shenzhen University, China. Also, he is the founding director of Big Data Institute and the deputy director of the National Engineering Laboratory for Big Data System Computing Technology. Prof. Huang is known for his contributions to the development of a series of k-means type clustering algorithms in data mining, such as k-modes, fuzzy k-modes, k-prototypes, and w-k-means, that are widely cited and used, and some of which have been included in commercial software. He has extensive industry expertise in business intelligence and data mining, and has been involved in numerous consulting projects in Australia, China and Hong Kong. Prof. Huang has published over 220 research papers in conferences and journals. In 2006, he received the most influential paper award in the First PAKDD. He is recognized as a scientist of Career Scientific Impact in Stanford University Worlf's top 2% scientists list. His main research interests include big data technology and applications.

Rukhsana Ruby received the master's degree from the University of Victoria, Canada, in 2009, and the Ph.D. degree from The University of British Columbia, Canada, in 2015. She has authored nearly 60 technical papers of well-recognized journals and conferences. Her research interest includes the management and optimization of next-generation wireless networks. She was a recipient of several awards or honors, notable among which are the Wait-listed for Canadian NSERC Postdoctoral Fellowship, the IEEE Exemplary Certificate (IEEE Communications Letters in 2018 and IEEE Wireless Communications Letters in 2018).



Kaishun Wu received the Ph.D. degree from the Hong Kong University of Science and Technology, in 2011. He is a distinguish professor at Shenzhen University, China. He has co-authored 2 books and published over 100 research papers in journals and conferences, such as the IEEE Transactions on Mobile Computing, the IEEE Transactions on Parallel and Distributed Systems, ACM MobiCom, and IEEE INFOCOM. He holds 6 U.S. patents and over 90 Chinese pending patents. Prof. Wu was a recipient of the 2012 Hong Kong Young Scientist Award, 2014 Hong Kong ICT awards and the 2014 IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award. He is a Fellow of the IET and IEEE Senior Member.