Cogn Comput (2015) 7:161–182
DOI 10.1007/s12559-014-9281-0

A Cognitively Inspired Approach to Two-Way Cluster Extraction from One-Way Clustered Data

Ahsan Abdullah · Amir Hussain

Received: 23 August 2013 / Accepted: 2 June 2014 / Published online: 18 June 2014

Abstract Cluster extraction is a vital part of data mining; however, humans and computers perform it very differently. Humans tend to estimate, perceive or visualize clusters cognitively, while digital computers either perform an exact extraction, follow a fuzzy approach, or organize the clusters in a hierarchical tree. In real data sets, the clusters are not only of different densities, but have embedded noise and are nested, thus making their extraction more challenging. In this paper, we propose a density-based technique for extracting connected rectangular clusters that may go undetected by traditional cluster extraction techniques. The proposed technique is inspired by the human cognition approach of appropriately scaling the level of detail, by going from low level of detail, i.e., one-way clustering to high level of detail, i.e., biclustering, in the dimension of interest, as in online analytical processing. A number of experiments were performed using simulated and real data sets and comparison of the proposed technique made with four popular cluster extraction techniques (DBSCAN, CLIQUE, *k*-medoids and *k*-means) with promising results.

Keywords Biclustering · Cognition · Cluster extraction · Density · Noise · Similarity matrix

A. Abdullah (✉)
King Abdulaziz University, Jeddah, Saudi Arabia
e-mail: aabdullah1@kau.edu.sa

A. Hussain
University of Stirling, Stirling, Scotland, UK

Introduction

Data mining is an exploratory approach of knowledge discovery, where automatic browsing is performed using proven algorithmic techniques. Data mining can reveal previously unknown aspects that are non-trivial and valuable to the end user. Hence, in data mining, we expect the unexpected. Cognitive data mining can be considered to be knowledge discovery supported by cognitive sciences, i.e., adopting a multi-disciplinary approach. This includes usage of human pattern recognition skills along with their engineering implementations in computation. Information visualization is defined as the use of computer-supported, interactive, visual representations of abstract (non-physical) data in order to amplify cognition [1].

More formally, suppose we have n objects to be clustered and are denoted by Σ.

$\Sigma = \{o_1, \ldots o_i, \ldots o_n\}$ where o_i is the ith object. A partition δ breaks Σ into subsets or clusters $\{C_1, C_2 \ldots C_m\}$ satisfying the following "Eq. (1)":

$$C_i \cap C_j = \phi \quad \text{where} \quad i \neq j \,\text{and}\, C_1 \cup C_2 \cup \ldots C_m \tag{1}$$

Thus, the act of dividing meaningful groups of objects that share common properties is called clustering, such that the objects in a group (or cluster) are similar to each other (in some sense) and dissimilar to objects in other groups or clusters.

In multivariate statistics, a data matrix is a mathematical matrix of data of dimension n-by-m, where n is the number of samples drawn, and m is the number of variables in each sample [2]. In this representation, different rows represent different repetitions of an experiment, while columns represent different types of data (say, the results from particular probes). For example, suppose a survey is conducted where 15 people are questioned on the street and asked

eight questions. The data matrix M would be a 15×8 matrix (meaning 15 rows and 8 columns). The datum in row i and column j of this matrix would be the answer of the ith person to the jth question. Note that in a data matrix, correlation values are not calculated between instances.

The proximity (or similarity) among objects is described by a similarity matrix $S(n \times n)$ defined as in "Eq. (2)":

$$\mathbf{S} = \begin{pmatrix} s_{11} & \cdots & s_{1n} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nn} \end{pmatrix} \forall i = j,\ s_{ij} = 1 \tag{2}$$

Here, s_{ij} is the similarity between the ith row and the jth row of the data matrix, i.e., $s_{ij} = s_{ji}$.

The matrix $\mathbf{S}$ contains measures of similarity or dissimilarity among the n objects. If the values s_{ij} are distances, then they measure dissimilarity. The greater the distance, the less similar are the objects. If the values s_{ij} are proximity measures, then the opposite is true, i.e., the greater the proximity value, the more similar are the objects. Distance and similarity are of course dual. The nature of the observations plays an important role in the choice of proximity measure. Nominal values (like binary variables) lead in general to proximity values, whereas metric values lead (in general) to distance matrices. There can be different similarity measures, such as Jaccard coefficient and Tanimoto coefficient. However, in this paper, we consider the similarity measure based on Pearson's correlation as defined in "Eq. (3)":

$$\frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{3}$$

Additional details about similarity matrix are given in "Similarity matrix."

Consider the data stored in a table or a data matrix. There can be several ways of knowledge discovery in the data matrix. For example, clustering only the rows (or the columns) of a similarity matrix results in one-way clustering. For a detailed analysis or a local view, biclustering or co-clustering or two-way clustering is performed on the data matrix involving the direct and simultaneous clustering of the records and the attributes of the data matrix [3, 4].

If one-way clustering is performed by clustering the pair-wise correlation similarity matrix corresponding to the data matrix, the clusters would ideally be square. A typical data matrix has very large number of rows as compared to the number of columns; therefore, biclusters by definition are rectangular, but can be square. The proposed technique is unlike the generic two-way clustering that works by applying a one-way clustering method in a sequential manner [5]. Instead, the proposed technique is about starting with a global view of the data, i.e., one-way clustering, and then extracting the details thorough biclustering. The cognitive model of our approach can be best explained using the analogy of online analytical processing (OLAP), where drill-down is used to increase the level of detail. In this paper, we consider the problem of extracting connected rectangular clusters from one-way clustered data under non-ideal, i.e., noisy conditions. Connected clusters are defined similar to the definition given in WaveCluster [6], i.e., connected clusters are just groups of connected "pixels," i.e., pixels that are connected to one another horizontally or vertically; we do not consider diagonal connectivity, additional details in "Problem definition and our contribution."

There are different types of cluster extraction algorithms for different types of applications. The most common distinction is between partitioning, density-based and hierarchical cluster extraction algorithms [7], details in "Background." Density-based clusters are defined as clusters, which are differentiated from other clusters by variable densities, meaning a group which has dense region of objects may be surrounded by low-density regions. In this paper, we will consider DBSCAN [8] for comparison since DBSCAN is excellent for arbitrary-shaped clusters. Grid-based cluster extraction methods have been used in some data mining tasks for very large databases. In the grid-based cluster extraction, the feature space is divided into a finite number of rectangular cells forming a grid. All clustering operations are performed in this grid structure. Since clusters resulting from clustering of a similarity matrix can be rectangular, therefore, we will also consider CLIQUE (Clustering In QUEst)-based cluster extraction in our study [9] . We hypothesize that extraction of square or rectangular clusters or their combination (with an underlying grid) should not be a problem for CLIQUE. For the purpose of comparison, in this paper, we will also consider partition-based cluster extraction, i.e., k-means and k-medoids.

Clustering Versus Cluster Extraction

In the published literature, the words cluster and cluster extraction are sometimes used interchangeably. However, in this paper, clustering and cluster extraction (or cluster mining) are considered to be different and distinct. In cluster extraction, the order of the data items in the given data set is not changed, instead based on some similarity measure, similar data items are considered to belong to a certain cluster, e.g., k-means. In clustering, the order of the data items in the data or the correlation matrix is changed based on some similarity measure (e.g., one-way clustering), which is subsequently followed by cluster extraction. Both concepts are explained using Fig. 1. Note that the proposed technique works on both ordered and unordered data sets.

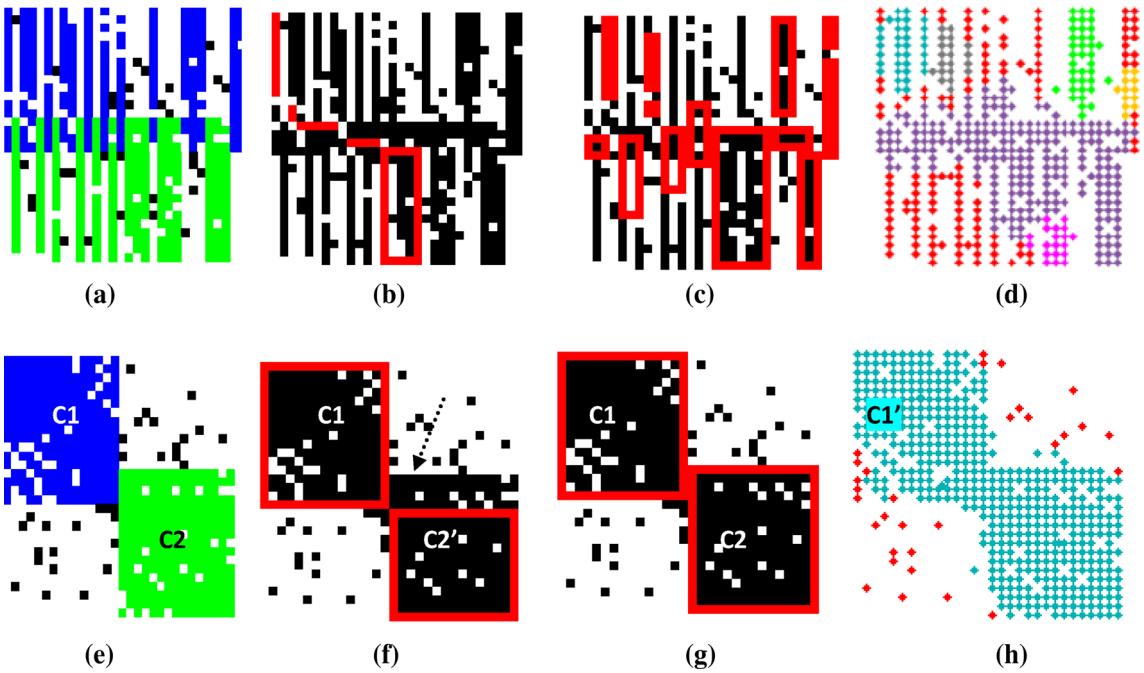

Fig. 1 Cluster extraction versus clustering of connected clusters. Clusters in **a**, **e** colored for ease of comprehension only. **a** Input noisy data set, **b** SCE clusters = 5, **c** CCE clusters = 13, **d** DBSCAN clusters = 6 (Eps = 10, MinPts = 9), **e** result of clustering Fig. 1a by crossing minimization [10], **f** two clusters extracted using SCE, one being partial, i.e., C_2', **g** two clusters completely extracted using CCE without noise, **h** DBSCAN (Eps = 10, MinPts = 9) two clusters and noise extracted as a single cluster.

Real data are always noisy, and it is very hard to differentiate noise from data, consider such a noisy data set with two clusters. We consider two types of noise, i.e., missing data within clusters and false relationships across clusters. Observe that if there are no clusters in the data set, then noise is likely to get clustered; therefore, the soundness of a cluster extraction technique is hard to verify. Thus, in Fig. 1, a data set with two clusters is used, which is the "ground truth" for verification purposes only. The clusters are color coded for ease of comprehension; however, the color information is neither used in clustering nor in cluster extraction, i.e., unsupervised learning. Figure 1a shows one such simulated data set ($n = 322$) with light random permutation to "hide" the two clusters. The two clusters are color coded (i.e., green and blue), missing data shown by white spots within the clusters and black spots corresponding to false relationships across clusters.

Now cluster extraction is performed on the data set of Fig. 1a using symmetric cluster extraction (SCE), cognitive cluster extraction (CCE), i.e., the proposed technique, and finally cluster extraction using DBSCAN, the results are shown in Fig. 1b–d. Note that for SCE and CCE (details in "Materials and methods"), the red boxes show the boundary of the clusters extracted. By comparing the results of Fig. 1b–d with Fig. 1a, it can be observed that since the data set was not clustered, therefore, the cluster extraction was partial and the "scattered pieces" of the two clusters were instead considered as multiple independent clusters. Observe the promising results of CCE.

Now the un-clustered data set of Fig. 1a is clustered using the indigenous clustering technique based on the crossing minimization paradigm [10], with the result shown in Fig. 1e. Note that the data set shown in Fig. 1a, e is identical except for reordering of columns. Although clusters C_1 and C_2 in Fig. 1e are visually well-defined to humans, but the computer cannot "see" them. To resolve this problem, we need to perform cluster extraction. Now cluster extraction is performed on the data set shown in Fig. 1e using SCE technique, CCE technique and finally using DBSCAN, the results shown in Fig. 1f–h. By comparing the results of Fig. 1f–h with Fig. 1e, it can be observed that SCE technique could not completely extract C_2 (shown by dotted arrow) while DBSCAN considered the two cluster and noise as a single cluster. The proposed CCE technique not only successfully extracted both clusters, but also differentiated noise from data, i.e., did not extracted noise with data.

Observe that for the data set of Fig. 1e, the data extraction by SCE and DBSCAN does not satisfy the clustering conditions of "Eq. (1)." This being also true for extraction using *k*-means, *k*-medoids and CLIQUE for which the results are not shown here.

Once the clusters are extracted, the obvious question is how to quantify cluster quality? There can be different

criterion for establishing the cluster quality, some of which are as follows [11]:

1. Distinguishing the existence of non-random structures in the data.
2. Comparing the results of cluster analysis to externally known results, i.e., with externally given class labels.
3. Evaluating the fitness of the clustering results with the data.
4. Different aspects of cluster validation without reference to external information and using only the data.
5. Comparing the results of two different sets of cluster analyses to determine which one is better.
6. Determining the "correct" number of clusters.
7. Establishing cluster quality by inspection of visualization.

In this paper, we will use the metric of normalized standard deviation (*Sn*) to evaluate cluster quality for real data and metric of *purity* to evaluate cluster quality for simulated data, details in Sects. 5.2.3 and 5.2.4, respectively.

The Challenges

Real-world data often contain noise. This poses two main challenges for clustering and cluster extraction, i.e., (1) how to extract correct clusters from noisy data and (2) how to extract correct clusters from noisy data and also remove the noise. While the former only considers clustering regardless of the noise, the latter performs both clustering and noise removal simultaneously. Obviously, the latter is more challenging and of more practical utility. Most of the clustering and cluster extraction algorithms including crossing minimization heuristics (CMH) [10] and CCE may even fail in the former problem, let alone the latter, because of masking and distortion of real data distribution due to noise. The results of noise tolerance of five cluster extraction techniques considered in this paper are discussed in Section 5.3

An important aspect of cognition is visualization. As per Card et al. [12], visualization is the use of computer-supported, interactive and visual representation of abstract data to amplify cognition. Visualization is considered one of the instinctive methods for cluster detection and validation, especially well suited for arbitrarily shaped clusters.

Several approaches have been proposed in the literature for visual cluster analysis [13, 14], but their arbitrary exploration of group information makes these techniques inefficient and time-consuming for cluster exploration. On the other hand, the impreciseness of visualization and limited screen resolution limits its utilization in quantitative verification and validation of clustering results. Thus, a color-coded visualization technique, focused toward cluster detection and precise contrast between clustering results, is also one of the motivations of this research.

Problem Definition and Our Contribution

The problem considered in this paper is contiguous rectangular clustering and extraction; this problem is a special case of the problem of connected cluster extraction discussed by Böhm et al. [15], i.e., in this paper, we only consider rectangular clusters. The problem of connected clusters is also encountered in case of spatial objects in image segmentation, e.g., for regional maps, road maps and also in wireless sensor networks (WSN), storage area networks (SAN) to name a few. Spatial objects have spatial location and distance properties. The problem of recognizing spatial continuity of some morphological elements in a map is reformulated as the problem of grouping adjacent cells resulting in a morphologically homogeneous area, i.e., a problem of clustering spatial objects according to the discrete spatial structure imposed by the relation of "adjacency" among cells [16]. Thus, spatial data mining means extracting implicit knowledge, spatial relations or other modes explicitly from the spatial database [17].

Figure 2 shows connected rectangular clusters for the block map of 2009 offshore petroleum acreage release area [18], while Fig. 2b shows connected rectangular clusters composed of horizontal and vertical rectangles that can be found on street maps.

Consider the 10 clusters shown in Fig. 2b, i.e., C_1–C_{10}. As a result of using DBSCAN with parameters used for better results, DBSCAN extracted C_1–C_4 and C_8 as a single cluster along with parts of C_7, C_9 and C_{10} and considered almost all of C_6 as noise. As can be observed, using the proposed CCE technique, each and every cluster was extracted without any overlap including the adjacent clusters with different densities, i.e., C_5 and C_6.

The main contributions of our work are as follows:

- A novel CCE technique based on backtracking with depth-limited search for extracting connected rectangular clusters or biclusters from a clustered or unclustered similarity matrix (i.e., global clustering).
- A CCE technique with high noise tolerance and limited sensitivity to clustering parameters as compared to the traditional cluster extraction techniques (DBSCAN, CLIQUE, *k*-medoids and *k*-means).
- A meta-heuristic technique based on using indigenous one-way clustering by crossing minimization, followed by two-way indigenous CCE.

Rest of the paper is organized as follows. In "Introduction" section, we provide the basic concepts, such as difference between similarity and data matrix, difference between clustering and cluster extraction and also provide problem definition. In "Background" section, we will provide the necessary background about OLAP, similarity matrices, compare biclustering with one-way clustering, briefly discuss

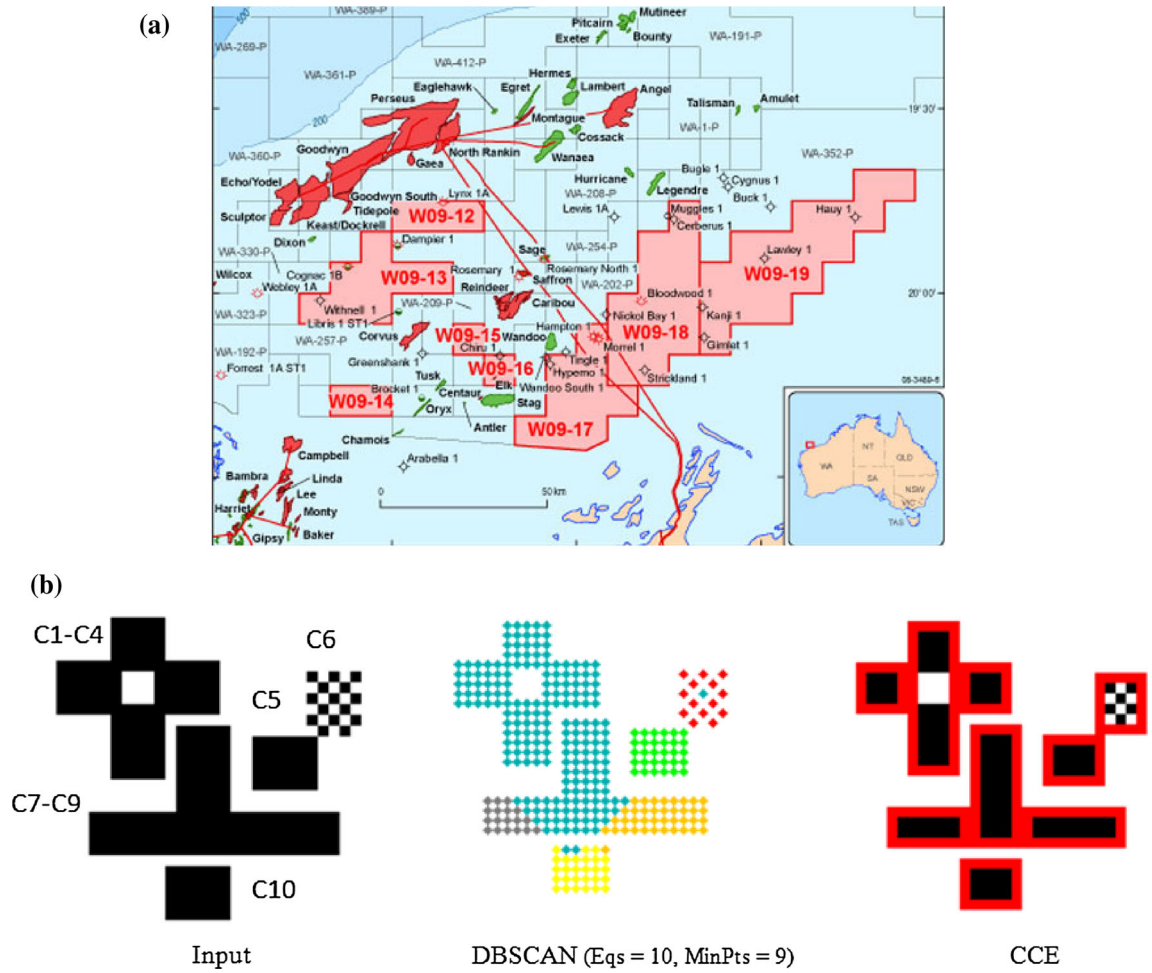

Fig. 2 Examples of connected rectangular clusters. **a** connected rectangular clusters of petroleum resources [15], **b** connected rectangular clusters extracted using DBSCAN and CCE

the traditional cluster extraction techniques, i.e., density-based, grid-based and partition-based techniques and the cognition part of the proposed approach. In "Related work" section, related work will be discussed and brief comparisons made. In "Materials and methods" section, we will describe the working of the CCE, and how data are prepared for clustering and subsequent cluster extraction. In "Results" section, we will present the results of different cluster extraction techniques using simulated as well as real data sets, i.e., Oyster data set and Breast cancer data set with comparisons done based on cluster quality, scalability and cluster discrimination; this will be followed by "Discussion" section, which consists of presentation of necessary theoretical and conceptual details explaining the effectiveness of our idea. In this section, we will also discuss the effect of noise and sensitivity to extraction parameters, and the potential strengths of the proposed technique. This will be followed by "Conclusions and future work" section, i.e., conclusions and future work.

Background

There are three basic approaches to cluster extraction, i.e., (1) partition-based, (2) density-based approach and (3) grid-based. In the partitioning approach, various partitions are constructed and then evaluated by some criterion, e.g., minimizing the sum of square errors, some of the typical methods are *k*-means, *k*-medoids, and CLARANS [19]. In the density-based approach, clusters are constructed based on connectivity and density functions; some of the typical methods are DBSCAN, OPTICS [20] and DENCLUE [21]. Finally, in the grid-based approach, clusters are constructed based on a multi-level granular structure; some of the typical methods are STING [22], WaveCluster [6] and CLIQUE. The proposed clustering approach CCE is a combination of density-based and grid-based approach. In this paper, we will consider all three approaches for the purpose of comparison.

Various biclustering techniques have been proposed in the literature, each having their own strengths and weaknesses. Discussion of these techniques is beyond the scope of this paper, as the paper deals with bicluster extraction instead of biclustering.

OnLine Analytical Processing (OLAP)

The term OLAP was coined by Codd [23]. OLAP exploits the exploration of multi-dimensional structured data, since this structure reflects the analyst's cognitive model of the data; therefore, the deduction process is less meticulous, and analysis is performed with ease [24]. The OLAP approach cognitively supports the iterative nature of the process of analysis, allowing the analyst to explore and navigate across different dimensions at different levels of aggregation detail, i.e., through drill-down and roll-up. *Drill-down* descends in the dimensional hierarchy of the OLAP cube by increasing the level of detail of the measure (and decreasing its level of abstraction), this being equivalent to bicluster extraction in our case. Note that analysis services can be utilized for both OLAP and data mining, as both are used to analyze data and find patterns. The big difference, however, is in performance, i.e., OLAP or more precisely MOLAP (multi-dimensional OLAP) is used interactively (almost in real time) due to memory-based usage of the pre-aggregates, while knowledge discovery using data mining can take from minutes to hours.

Similarity Matrix

Tucker [25] suggested a classification of data matrix based on two numerical variables, i.e., the number of ways and the number of modes. A formal description of that classification follows. Let I_1, I_2, …, In be sets of some entities, i.e., kinds of plant, kinds of bird, geographical sites and periods of times, observe that these entities are dimensions in the context of OLAP. It is permitted that some of the sets be coinciding; with k, i.e., number of different sets among the given n sets. An n-dimensional array of, usually numeric, code values $a(i_1, \ldots, i_n)$ given for any combination of $i_1 \in I_1$, $I_2 \in I_n$, …, $i_n \in I_n$, is referred to as a n-way k-mode table. Therefore, a similarity matrix a_{ij} between the entities $i, j \in I$ is a two-way one-mode table.

Biclustering Versus One-Way Clustering

In this paper, clustering of a similarity (or correlation) matrix is considered as one-way clustering, as either each record in the cluster is selected using all the attributes or each attribute in a cluster is selected using all the records. Thus one-way clustering methods give a global view or perspective of the relationships between all the records (or attributes), i.e., knowledge discovery at a lower level of detail. In one-way clustering, evaluating each record in a given cluster using all the attributes may actually distort the cluster, since all attributes may not be contributing toward that cluster. Similarly, each attribute in a one-way cluster is usually characterized by the effect of all the records, which may not always be true.

As opposed to one-way clustering, each record in a bicluster is selected using only a subset of the attributes, and each attribute in a bicluster is selected using a subset of the records [26], i.e., knowledge discovery at a higher level of detail. As per Maderia and Oliveria [27], there are four major classes of biclusters (details in next section). In this paper, we will consider the first class of biclusters, i.e., with constant values, as this goes well with the concept of discretized similarity matrix, i.e., a bicluster corresponding to a collection of contiguous 1's along the rows and columns.

The complexity of the biclustering problem depends on its particular formulation. The problem of finding a minimum set of biclusters, either mutually exclusive or overlapping, covering all the elements in a data matrix is a generalization of the problem of covering a bipartite graph by a minimum set of bicliques, either mutually exclusive or overlapping. This problem has been shown to be NP-hard [28]. As a result, instead of devising exact algorithms, a pragmatic approach would be to come up with heuristics or approximation algorithms.

Major Classes of Biclusters

A useful criterion to evaluate a biclustering algorithm has been proposed [27] based on the identification of the type of biclusters the algorithm can discover. Based on this criterion, four major classes of biclusters are identified as follows:

- Biclusters with constant values.
- Biclusters with constant values on rows or columns.
- Biclusters with coherent values.
- Biclusters with coherent evolutions.

Patterns that rise and fall concordantly are considered to be coherent. Table 1 shows additive and multiplicative coherent biclusters.

The first three types of biclusters are based on the direct analysis of the numeric values in the data matrix; thus, the biclustering algorithm attempts to find subsets of columns and subsets of rows with similar properties. The similarity of these properties can be observed on the columns, on the rows, or on both columns and rows of the data matrix. For the fourth class, the biclustering algorithm attempts to find coherent properties regardless of the exact numeric values in the data matrix. Therefore, elements in the data matrix are considered as symbols for biclusters with coherent

Table 1 Additive and multiplicative coherent biclusters

Additive coherent bicluster			
1	2	5	0
2	3	6	1
4	5	8	3
5	6	9	4
Multiplicative coherent bicluster			
1	2	0.5	1.5
2	4	1	3
4	8	2	6
3	6	1.5	4.5

evolutions. These symbols can represent coherent positive and negative changes relative to a normal value, or could be purely nominal that may correspond to a given order.

Traditional Cluster Extraction Techniques

Density-Based Cluster Extraction

Density-based cluster extraction or mining has a broad range of applications ranging from monitoring moving objects to stock trade analysis. According to Yang et al. [29], although several methods for efficient extraction of density-based clusters have been studied in the literature, the problem of matching and summarizing of such clusters with complex cluster structures and arbitrary shapes remains unsolved.

DBSCAN [8] (Density-Based Spatial Clustering of Applications with Noise) is one of the most popular density-based algorithms where regions of high density are considered as a cluster and regions of low density are considered to be noise. Clusters are defined on some criteria, which is called *core*. Core points lie in the interior of density-based clusters and should lie within Eps (epsilon or radius or threshold value) and MinPts (minimum no of points), which are user-specified parameters.

Grid-Based Cluster Extraction

Recently, a number of algorithms have been proposed that segregate the data space into a finite number of cells, i.e., of required resolution and then perform all the operations on the quantized space. The main characteristic of these approaches is their fast processing time, which is usually independent of the number of data objects. The reason being this depends only on the number of cells in each dimension in the segregated space, i.e., high resolution, resulting in accurate results and correspondingly more processing time. Grid-based methods have been employed for extracting rectangular as well as non-rectangular clusters for hyper-rectangular cells as well as arbitrary-shaped polyhedral [30–32]. For d-dimensional space, hyper-rectangles (rectangular-shaped cube [33] of d-dimensions relate to the cells. In the hierarchical grid structure, the size of the grid cell can be decreased in order to achieve a more exact cell structure. The hierarchical structure can be split into several levels. Each cell at the higher level l is partitioned to form a number of cells at the next lower level $l + 1$. The cells at the level $l + 1$ are formed by splitting the cells at level l into smaller sub-cells.

CLIQUE is a hybrid of density-based and grid-based clustering method, as each dimension gets partitioned into the same number of equal length intervals; thus, an m-dimensional data space is partitioned into non-overlapping rectangular cells, i.e., biclusters. As per Agrawal et al. [9], a cell is dense if the fraction of total data points contained in that cell exceeds the input parameter; thus, a cluster is a maximal set of connected dense cells within a subspace. Two parameters are necessary, i.e., intervals (I) that is the number of units along one dimension and the threshold (T) that is the minimum points in one grid. CLIQUE automatically identifies subspaces of a high-dimensional data space allowing improved clustering than the original. CLIQUE-based cluster extraction is considered in this paper.

Partition-Based Cluster Extraction

In partitioned-based clustering data, space is divided into non-overlapping subsets or biclusters in such a way that no two subsets will share a common data object. The problem can be stated as, given n patterns in a d-dimensional space, determine the partitions of patterns into k biclusters, here k may or may not be defined. Thus, the partitioned biclustering solution to this problem is to choose a partitioning criterion and evaluate it for all partitions and pick the partition that best suits the criteria. Correct parameters for partitioning algorithms allow the algorithm to reveal the true structure of a data set. These parameters are in general difficult to determine, and they may not even exist.

k-Means is one of the simplest partitioning algorithms adopted for many problem domains. The algorithm follows simple and easy steps to extract k spherical clusters and possibly biclusters in the given data space, here k is fixed a priori. Sometimes k is just declared arbitrarily, sometimes the problem determines k and sometimes k is chosen naturally, but in general, this notion is not well defined. However, sometimes several iterations of k-means can reduce this effect. The main idea is to identify k centroids, i.e., one centroid assigned to each cluster. It can be proved that the k-means algorithm will always terminate; however, the algorithm (actually a heuristic) does not necessarily find the optimal clusters corresponding to the global objective function, i.e., minimum of the sum of the squared distance of each data point to its allocated mean.

Although *k*-means algorithm is one of the most popular, however, *k*-medoids-based algorithms have been shown to be robust, as they are less sensitive to the existence of outliers and are not limited in application to certain attribute types. Finally, the clusters found do not depend on the order of the input data set. The drawback of the *k*-medoids-based algorithms is that they do not scale well and therefore cannot be efficiently applied to large data sets. This has resulted in the development of several methods aimed at reducing the computational effort required for execution of these algorithms [8, 34, 35]. Both *k*-means and *k*-medoids-based cluster extraction are considered in this paper.

Related Work

The domain of density-based, grid-based and partition-based clustering is literature rich. However, in this section, we will consider some of the more popular and related techniques and do a comparison showing the utility of the proposed method/technique.

Böhm et al. [15] have considered the extraction of correlations between different features in a set of feature vectors, as correlation indicates a dependency between the features or some relationship of cause and effect. However, correlations are not global since the dependency between features can be different in different subgroups of the set. They [15] have used a method called 4C (Computing Correlation Connected Clusters) to identify local subgroups of the data objects sharing a uniform but arbitrarily complex correlation which we note to be a bicluster in a $n \times m$ data matrix, here n is the number of records or vectors and m number of attributes or dimensions. 4C is based on a combination of principal component analysis (PCA) that can find global linear correlations that are not embedded in noise and density-based clustering (DBSCAN). Subsequently, the results are compared with DBSCAN, CLIQUE and ORCLUS. Although we also extract biclusters and compare the results with DBSCAN and CLIQUE, but instead of extracting biclusters from correlations of feature vectors, we use CCE to extract biclusters from a clustered correlation matrix with noise. Note the prohibitively large time complexity of PCA, i.e., $O(nm^2)$ which forms the basis of 4C, while the proposed CCE techniques take only $O(e)$ time, where e is the number of 1's in the discretized similarity matrix. Due to discretization, the binary matrix is fairly sparse [10], resulting in small values for e. However, on the contrary, 4C relies on the assumption that the clustering structure is dense in the entire feature space, else 4C will fail to produce meaningful results [36].

Tan et al. [37] have analyzed some characteristics and weak points of traditional density-based clustering algorithms, followed by an improved cluster extraction technique based on density distribution function. *K* Nearest Neighbor (KNN) is used to measure the density of each point, and then, a local maximum density point is defined as the center point. By means of local scale, cluster extraction extends from the center point. This, however, is at the cost of additional time required for KNN, i.e., $O(n^2)$ for all pair nearest neighbor, and requirement of additional control parameters. Although Tan [37] do not use DBSCAN, instead use DENCLUE [15] with better time complexity, i.e., O(log n), but then there is the additional cost of creating and maintaining a $B+$ tree. In the proposed CCE technique, the cluster extraction exploits the property of the similarity matrix to be symmetric across the diagonal and does not employs the concept of radius thus achieving better noise tolerance with lower time complexity. The time complexity of the proposed technique is $O(e)$ where e is the number of 1's in the discretized similarity matrix and does not have the overhead of KNN and that of creating and maintaining a $B+$ tree.

Erten and Sözdinler [38] have used the idea of growing localized sub-matrices for bicluster extraction called as LEB (Localize-and-Extract Biclusters). Biclustering results in localization of correlated rows/columns, i.e., "localized" rows/columns exhibiting similar patterns placed in nearby locations within the given matrix. LEB starts by providing a constraint γ and dividing the biclustered data matrix into equal sized $\alpha \times \alpha$ grid called bags. The score of a sub-matrix corresponding to a possible bicluster is determined and grown (say) along x-direction until arrive at a bicluster with a desirable score, and then, the union is taken of adjacent bags meeting the criterion γ, this is repeated till all bags in the x-direction are exhausted. Subsequently, the process is repeated along y-direction. We observe the problem with this approach of forcing a regular structure on irregular biclusters; furthermore, wrong choice of α can split clusters thus reducing the accuracy of the results as in the case of CLIQUE. Although we also expand biclusters by traversing along the two directions, i.e., x-direction and y-direction, but unlike Erten and Sözdinler [38], we do so simultaneously along both directions and then backtrack, thus increasing the cluster quality. Furthermore, there is no user-specified criterion of γ i.e., a more unsupervised approach.

Zhou et al. [39] have used cluster extraction for anomaly detection using a distributed approach for data located at multiple local sites. Three cluster extraction techniques have been used for this purpose, i.e., *k*-means, SOM and DBSCAN. Although we have also considered *k*-means and DBSCAN for cluster extraction, but for different reasons. According to Zhou et al. [39] for *k*-means, an anomaly means its distance to any cluster centroid is higher than the given minimal distance threshold or the anomaly belongs to a cluster that has smaller number of instances than the

given threshold of minimal number of instances. For DBSCAN, data points are declared to be outliers, if there are few other points in the neighborhood. DBSCAN is used to identify the type of instances (normal or anomaly), so the type is taken as binary anomaly scores (1 for normal and 0 for anomaly). However, in our study, we are more interested in capturing the clusters instead of noise or anomalies. Furthermore, in this paper, DBSCAN is used to extract clusters instead of a binary identification of being normal or an anomaly.

As per Zhou et al. [40], DBSCAN is an outstanding representative of clustering algorithms that show good performance for spatial data clustering. However, for large spatial databases, DBSCAN requires large volume of memory support that could incur substantial I/O costs because it operates directly on the entire database. In their paper [40], several approaches are proposed to scale DBSCAN algorithm to large spatial databases, such as developing a fast DBSCAN algorithm, which considerably speeds up the original DBSCAN algorithm. They have also studied a sampling-based DBSCAN algorithm, a partitioning-based DBSCAN algorithm and a parallel DBSCAN algorithm in their work, and then, there is GDBSCAN [41]. Following that, based on the above-proposed algorithms, a synthetic algorithm is also given. Although in our work we also consider DBSCAN but not from the perspective of performance or speed, this is also evident from the size of data sets being considered. In our work, instead of looking at performance, we consider the accuracy of DBSCAN for cluster extracted in the presence of noise and also do an analysis of sensitivity to parameters.

As per Qian et al. [42], current clustering methods always have the problems of: (1) scanning the whole database leading to high I/O cost and expensive maintenance (e.g., R*-tree), (2) pre-specifying the uncertain parameter k, with which clustering can only be refined by trial and test many times and (3) lacking high efficiency in treating arbitrary shape under very large data set environment. In [42], first a new hybrid-clustering algorithm is presented to solve these problems. This new algorithm combines both distance and density strategies and can handle any arbitrary shape clusters effectively. Furthermore, it makes full use of statistical information in mining to reduce the time complexity while ensuring cluster quality. Although we also use DBSCAN for comparison in our work and use the global density to reduce time complexity, but one of the differences between our work and that of Qian et al. [42] is the definition of noise. In their work, the density of noise is considered to be well proportioned and noise is very sparse; however, in our work, noise is considered to be random and not very sparse. This can also be observed from the examples considered in their work and ours.

As per Ester et al. [43], DBSCAN algorithm groups object to clusters based on the density of data and discover consistent arbitrary-shaped clusters along with detection of noisy outliers. The important issues of DBSCAN are its actual computational speed and efficiency for large data sets, and sensitivity to δ which we will discuss in Sect. 5.3. For DBSCAN, the time complexity is $O(n^2)$ where n is size of the data set [44]. The proposed technique is an order of magnitude faster as compared to DBSCAN and produces promising results for clustered similarity matrices with connected rectangular clusters. OPTICS (Ordering Points to Identify the Clustering Structure) [20] is an extension of DBSCAN and equivalent with DBSCAN in structure, both having the same time complexity, i.e., $O(n^2)$ or $O(n \log n)$ while adopting space index. Although OPTICS can realize auto and alternative clustering and is not sensitive to parameters, but its slow running speed is one of its drawbacks [37].

The strength of the CLIQUE algorithm is the unsupervised discovery of subspaces of high level of details when high-density clusters exist in those detailed subspaces. The technique is also *insensitive* to the order of records in input, and it rarely considers data distribution, which affect cluster quality to a certain extent. The time complexity of CLIQUE is $O(C^k + nk)$, where k is the highest dimensionality, n is the number of input points and C is the number of clusters [45]. Although CLIQUE scales *linearly* with the size of input as the number of dimensions increase, but its known major weakness is the accuracy, which may degrade due to simplicity of the method. Observe that extracting meaningful clusters using CLIQUE is dependent on proper selection of the grid size and the density threshold. This selection can be difficult in reality, since the grid size and density threshold are used across all combinations of dimensions in the data set, resulting in degradation of clustering results. In the proposed method, by virtue of one-way clustering, the effect of constant grid size and threshold is not there, and the localization of related data items minimizes the overlap of dense regions as in CLIQUE. Unlike CLIQUE, in the proposed technique, clusters are not "forced" into a rectangular grid, but are considered to be rectangular. Therefore, rectangular clusters are extracted with time complexity $O(e)$, where e is the number of 1's in the discretized similarity matrix. Thus, the proposed technique is not only fast, but also generates promising results (details in "Results").

Materials and Methods

The cognitive approach used in this paper proceeds as follows:

```
Inputs
B: Binary matrix (nxn)
s: minimum bicluster size
d: density of B
Steps
Step 1: Make an initial submatrix of size s x s having coordinates (x1, y1, x2, y2)
Step 2: Calculate density (d') of the submatrix
Step 3: if d' >= d
        Declare submatrix as bicluster 'B' and expand
        Continue expanding along x and y-direction til d' >= d
        Refine boundary of 'B'

Boundary refinement
Expansion along x-direction
        Expand 'B' along x-direction n*d units
        For each unit boundary expansion
                Calculate density d'' having coordinates (x1, y1, x2+s, y2)
                If d''>= d
                        Make expansion permanent
                Else
                        Backtrack and undo boundary expansion

Expansion along y-direction
Repeat process similar to x-direction
```

Fig. 3 Pseudo code for cognitive cluster extraction (CCE)

1. Use the given data matrix to create the similarity matrix (SM).
2. Perform one-way clustering of the SM.
3. Generate heat-maps for global results.
4. Perform cluster extraction using CCE for detailed results.

Biclusters provide a detailed view of the relationships between different data elements; this could have gone undetected by one-way cluster extraction. The sub-matrices corresponding to the clusters obtained by one-way clustering followed by two-way cluster extraction may not necessarily be square. As these sub-matrices are extracted from a similarity matrix instead of a data matrix, hence, they are not regarded as biclusters in the true formal sense. Hence, in this paper, the word cluster and bicluster will be used interchangeably wherever needed to enhance comprehension.

More formally, consider an $n \times m$ data matrix, where n is the number of rows and m is the number of columns. We proceed by creating a pair-wise $n \times n$ similarity matrix using Pearson's correlation. This is followed by knowledge discovery at low level of detail, i.e., one-way clustering (using crossing minimization) of the bipartite graph corresponding to the similarity matrix [10]. Subsequently, clusters are automatically extracted from the clustered matrix by traversing along the diagonal, i.e., symmetric cluster extraction (SCE) technique. Along with SCE technique, we implemented the more natural nonsymmetric and approximate CCE technique to achieve knowledge discovery at a higher level of detail. The proposed technique being inspired by the human cognition approach of appropriately scaling the level of detail, by going from low level of detail, i.e., one-way clustering to high level of detail, i.e., biclustering, in the dimension of interest, as in online analytical processing (OLAP). Pseudo code for the CCE extraction technique is shown in Fig. 3.

The CCE technique starts with a grid size of $s \times s$, where s is the default cluster size (usually 1×1) and then calculates the density of the cluster, i.e., d', if d' is more than the density of the similarity matrix, i.e., d (=number of 1's divided by matrix size), then $s \times s$ is considered to be a bicluster suitable for expansion. This expansion is simultaneously along x-direction and y-direction till d' falls below d. At this stage, instead of adopting a greedy approach and freezing the bicluster, the bicluster boundaries are refined, i.e., guided by d, expansion proceeds constant number of cells (say) along x-direction. If there is no increase in bicluster density, the expansion is undone and backtracked (details in "Effectiveness of CCE section") to the stage when the density fell below d for the first time. Subsequently, cluster expansion proceeds along y-direction in a similar manner, and when density falls below d the bicluster is frozen. For the next bicluster, the extraction subsequently starts from the cell at which it had stopped expansion, and the above process is repeated. Observe that SCE is a special case of CCE, i.e., once the density falls below d, boundary refinement is performed simultaneous along x-direction and y-direction, resulting in square clusters, i.e., one-way cluster extraction.

In real data sets, existence of clusters of very different point densities and in different regions of the data space is typical instead of an exception. However, these variable density clusters may also be nested, which makes cluster

extraction difficult using a traditional algorithm. This problem is due to the absence of global parameters, which may be required directly or indirectly by most partitioning algorithms that may characterize the clusters in the data space [46]. In the proposed CCE technique (Fig. 3), the global parameter d (density of the discretized similarity matrix) is first used to expand the biclusters and then to refine them; thus, the impact of global parameter is there. Median-based or mean-based discretization further contributes to the impact of global parameters.

Results

For the purpose of comparison, in addition to SCE, we will consider four other popular cluster extraction techniques, i.e., DBSCAN, CLIQUE, k-medoids and k-means, to extract clusters from the simulated as well as the real data sets used in this paper. For these four techniques, we will use their publically available online implementations: http://webdocs.cs.ualberta.ca/~yaling/Cluster/Applet/Code/Cluster.html.

Along with simulated data, real data sets will also be considered, i.e., the Oyster data set and the Breast cancer data set, corresponding details in "Oyster data set" and "Breast cancer data set." For cluster extraction using the Breast cancer data set, we will use our JAVA implementation of DBSCAN which unlike the online version does not limits the size of input data set.

Results Using Simulated Data

Consider the clustered 21 × 21 similarity matrix shown in Fig. 4a. For this rather small data set, we can identify the clusters of interest by inspection, for example, in Fig. 4a, there are eight clusters of size 3 × 3. However, the traditional symmetric cluster extraction (SCE) technique misses the clusters of interest altogether, instead extracts the less interesting 21 unit clusters (size 1 × 1) located along the diagonal as shown by red rectangles in Fig. 4a.

Note that in Fig. 4b–d, different sets of colored dots correspond to different clusters. Figure 4b shows the results of CLIQUE-based cluster extraction; recall that CLIQUE requires two parameters, i.e., intervals (I) that is the number of units along one dimension and threshold (T) that is the minimum points in one grid. For k-medoids and k-means, k is the number of clusters to be extracted based on some a priori knowledge.

From the results shown in Fig. 4b–d, we observe that although the three cluster extraction techniques considered performed better than SCE, i.e., by also clustering the non-trivial non-unit clusters, but were unable to correctly extract the useful/interesting clusters, i.e., these techniques incorrectly clustered together dissimilar clusters.

Now consider Fig. 5 where clusters are extracted using CCE and DBSCAN. The indigenous CCE technique (pseudo code given in Fig. 3) successfully extracted all 18 connected biclusters as shown by red rectangles in Fig. 5a, while six clusters were extracted by using DBSCAN as shown in Fig. 5b. Observe that in Fig. 5b DBSCAN fails to recognize C_i to C_{i+4} as four distinct clusters, instead DBSCAN treats them as a single cluster this being also true for other distinct clusters in the data space erroneously clustered together by DBSCAN. Quantitative comparison of bicluster extraction results of Figs 4 and 5 is discussed in Section 5.2.4

Cluster Quantification Using Simulated Data

For the simulated data with known number of clusters along with the corresponding details, we use the metric of *purity*. To compute *purity*, each cluster is assigned to the class that is most frequent in the cluster, and then, the accuracy of this assignment is measured by counting the number of correctly assigned data elements and dividing by N, i.e., the input size "Eq. (4)." Formally [47]:

$$\text{purity}(\Omega, C) = \frac{1}{N}\sum_{kj}^{\max} \left|\omega_k \cap c_j\right| \tag{4}$$

where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$ is the set of clusters and $C = \{c_1, c_2, c_3, \ldots, c_j\}$ is the set of classes. We interpret ω_k as the set of data elements in ω_k and c_j as the set of data elements in c_j.

So based on this metric, the *purity* of clusters extracted in Fig. 4a–d is 0, 0.24, 0.22 and 0.22, respectively. Note that increasing the value of k and subsequently performing several runs of k-means can improve the *purity* of the clusters extracted, but it is difficult to know a priori a good value of k and required number of runs? For the results shown in Fig. 4, the *purity* of clusters extracted via proposed CCE is 1, i.e., perfect, and for DBSCAN, the *purity* is 0.4. Note that unlike the other cluster extraction techniques discussed in this paper, CCE results were obtained without adjusting any parameter values.

Results Using Real Data

In this section, we will present the results of using two data sets, i.e., Oyster data set, which consist of a 30 × 5 data matrix, and the Breast cancer (BC) data set, which consist of a 699 × 11 data matrix available at the UCI Machine Learning repository http://archive.ics.uci.edu/ml/. The main purpose of using the Oyster data set is ease of comprehension of the presented concepts, while the BC data set

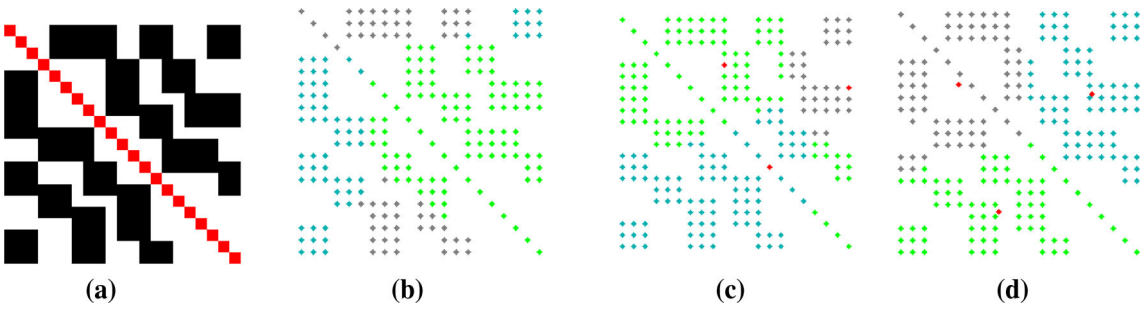

Fig. 4 Cognitive cluster extraction from simulated data using four different techniques. **a** SCE clusters = 21, **b** CLIQUE $I = 18$, $T = 3$, clusters = 3, **c** k-medoids $k = 3$, **d** k-means $k = 3$

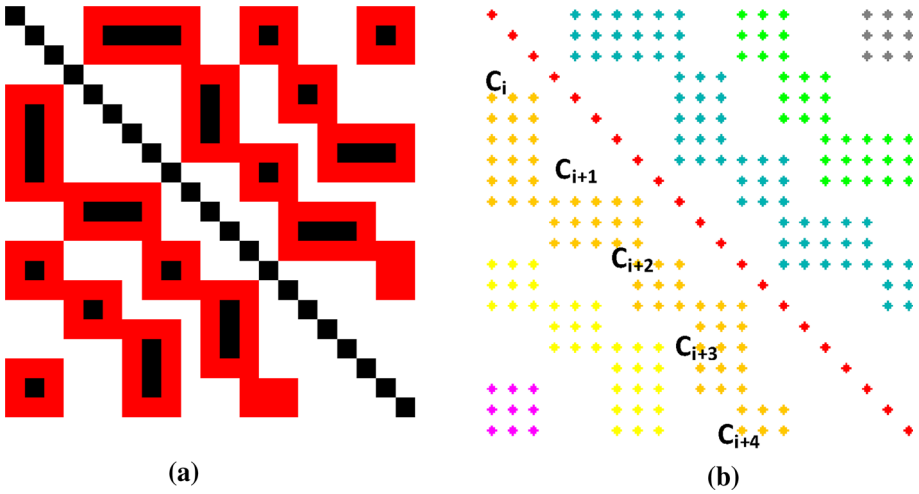

Fig. 5 Qualitative comparison of cluster extraction using CCE and DBSCAN. **a** CCE All 18 connected clusters extracted, **b** DBSCAN 6 clusters extracted

is used for comparative study w.r.t scalability and discrimination of the clusters extracted. Furthermore, clustering of BC results in spatially displaced clusters, thus making cluster extraction challenging.

Oyster Data Set

Shucking and grading of oysters is a labor intensive and tedious task. Therefore, a 3D computer vision system for oyster classification was developed by Lee et al. [48] to improve on the existing 2D system for oyster grading. Oysters were defined to be small, medium or large if their volume fell in (0, 10], [10, 13), [13, inf), respectively. Data on 30 oysters (10 small, 10 medium and 10 large) were collected in a calibration experiment in which simple linear regression models were used to predict observed oyster volume, one using digital image area (2D) in pixels as a predictor, and the other using digital image volume (3D).

For this study, we consider the Oyster volume estimation data set [48] consisting of 30 observations and 5 variables. The data were collected in 2001 in a research and development lab at AGRI-TECH Inc., Woodstock, VA. Details of how the data were collected can be found in [48].

Table 2 Column variables for Oyster data set

Columns	Description of attributes
1–2	Oyster ID
11–15	Oyster weight (g)
27–31	Oyster volume (cc)
44–50	Oyster size information from the 3D imaging system (in volume pixels)
54–58	Oyster size information from the 2D imaging system (in pixels)

The variable description of the oyster data set is given in Table 2; note that there were no missing values.

Cluster Extraction Oyster Data Table 2 shows the attribute description of the data matrix. It can be observed from Table 2 that there is large variation in the range of attribute values, for example, the oyster weight and volume being four-digit numbers, while the pixels for 2D image are six digit and that for 3D image this being seven digits. To have a common basis for visual comparison of attributes, last four attributes of Table 2 were discretized using average attribute value of that attribute, resulting in the discretized data matrix

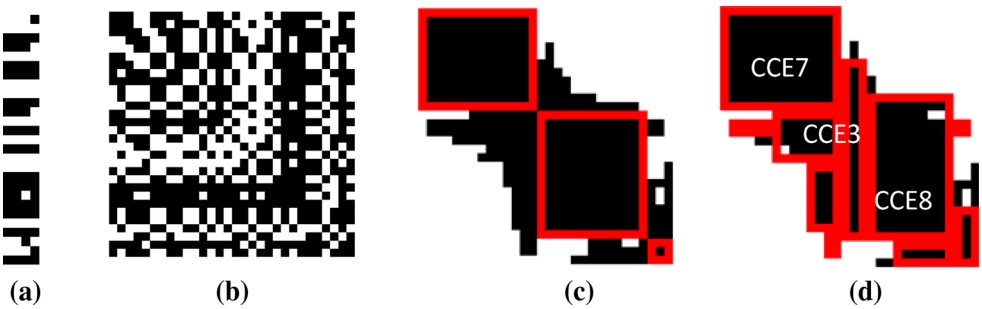

Fig. 6 Cluster extraction using SCE and CCE from clustered Oyster data set. **a** Discretized data matrix, **b** un-clustered similarity matrix resulting from 6(a), **c** 3 clusters extracted using SCE, **d** 10 clusters extracted using CCE

as shown in Fig. 6a. Using Pearson's correlation (Eq. 3), a 30×30 similarity matrix was created using the actual values of the last four attribute of Table 2. Due to very fine variation in Pearson correlation values, the color-coded similarity matrix using RGB color model resulted in single color assigned to each pixel. Therefore, the similarity matrix was discretized using average attribute values; the corresponding discretized similarity matrix is shown in Fig. 6b.

As expected, hardly any clustering can be observed in Fig. 6b, because without clustering there is no ordering of dimensions of the similarity matrix, the ordering of the dimensions of the similarity matrix is as per the ordering of the rows of the data matrix used to create that similarity matrix. Using the CMH [10], clustering is performed, resulting in the clustered similarity matrix as shown in Fig. 6c, d, observe that clustering of the similarity matrix changes the ordering of the dimensions, resulting in enhanced cluster extraction. Figure 6c shows the results of one-way cluster extraction (global view) using the SCE technique, resulting in three clusters, while Fig. 6d shows the results of bicluster extraction (detailed view) using the CCE technique for the same clustered matrix shown in Fig. 6c with 10 clusters extracted. Thus, scaling the level of detail, i.e., going from a low level of detail to a higher level of detail in the dimension of interest results in extraction of these biclusters that were missed by SCE.

Figure 7 shows the cluster extraction results using the four cluster extraction techniques for the clustered similarity matrix shown in Fig. 7b. Figure 7 is based on those extraction parameters that generated better results. Note that unlike CCE and SCE where the cluster boundaries are generated automatically, for ease of comprehension, the dashed cluster boundaries have been drawn manually in Fig. 7. From Fig. 7, it is obvious that that the four techniques have been unable to interpret the symmetric nature and structure of the similarity matrix and the corresponding rectangular clusters. *k*-Medoids (Fig. 7c) seems to have demonstrated better results; however, the results could have been bad, if a "lucky" guess had not been made about the number of clusters present. Note that in Fig. 7d, the cluster boundary is drawn square because by definition CLIQUE works with square grid cells.

Cluster Quantification Oyster Data To quantify the clustering results, we consider the metric of normalized standard deviation denoted by *Sn*. *Sn* is the normalized measure of spread obtained by dividing a measure of spread (except the variance as it has squared units) by a measure of location "Eq. (4)." Thus, normalized standard deviation (or coefficient of variance) is simply the standard deviation (S) divided by the mean ($\bar{\pi}$), i.e.,:

$$Sn = \frac{S}{\bar{\pi}} \tag{5}$$

Note that the smaller the value of Sn, the better. This metric allows the comparison of the spread of the distribution of a variable with large standard deviation and a correspondingly large mean more appropriately with the spread of the distribution of another variable, with smaller standard deviation and correspondingly smaller mean.

For each of the 10 clusters extracted using CCE, i.e., CCE1 through CCE10 and for each of the three clusters extracted using SCE and *k*-medoids (KMD), we calculate the normalized standard deviation using the Oyster weight and subsequently arrange the clusters in ascending value of *Sn*. The corresponding results are shown in Fig. 8; here, the number in the parenthesis is the cluster size, i.e., the number of rows.

From Fig. 8, it can be observed that the proposed CCE technique was able to extract at least three significant additional biclusters (with rows >3), i.e., CCE8, CCE7 and CCE17 (identified by arrows) and that too with *Sn* better than SCE2 and KMD2. These biclusters could not be extracted using the traditional SCE technique and the *k*-medoids technique. Note that CCE1, SCE1 and KMD1 are statistically identical. Thus, as compared to the traditional cluster extraction techniques, the proposed CCE technique is able to perform knowledge discovery from within the discovered knowledge.

Breast Cancer Data Set

In this section, we will compare CCE with DBSCAN w.r.t scalability and cluster discrimination, i.e., ability to cluster dense region into multiple clusters. DBSCAN is considered for further analysis because:

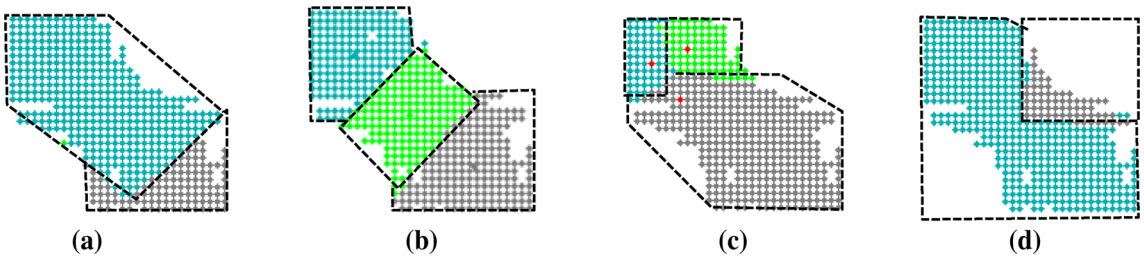

Fig. 7 Cluster extraction using DBSCAN, CLIQUE, *k*-medoids and *k*-means from clustered Oyster data set. **a** DBSCAN (Eps = 10, MinPts = 3), **b** *k*-means ($k = 3$), **c** *k*-medoids ($k = 3$), **d** CLIQUE ($I = 12$, $T = 2$)

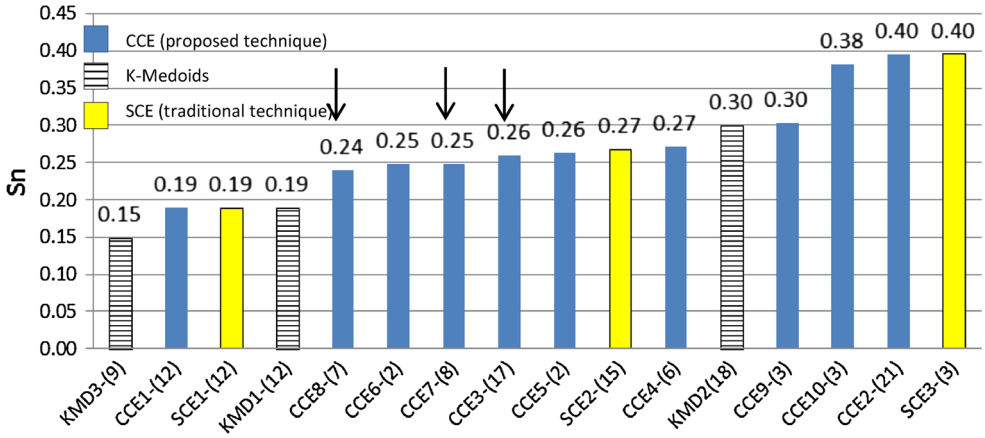

Fig. 8 Comparison of three cluster extraction techniques based on normalized standard deviation

- The technique is considered popular with diverse applications as compared to the other three techniques considered.
- The reported scalability issues of DBSCAN in terms of execution time [49, 50].

The BC data set consists of 699 rows and 11 columns/attributes, excluding the patient ID and the resultant class, i.e., Malignant or Benin, we are left with nine attributes that will be used in this paper; the same are briefly described here. A1 Clump thickness: Benign cells tend to be grouped in monolayers, while cancerous cells are often grouped in multi-layer. A2 Uniformity of cell size: Cancer cells tend to vary in size, that is why these parameters are valuable in determining whether the cells are cancerous or not. A3 Uniformity of cell shape: Cancer cells tend to vary in shape, that is why these parameters are valuable in determining whether the cells are cancerous or not. A4 Marginal adhesion: Normal cells tend to stick together, cancer cells tend to lose this ability. So loss of adhesion is a sign of malignancy. A5 Single epithelial cell size: Is related to the uniformity mentioned above. Epithelial cells that are significantly enlarged may be a malignant cell. A6 Bare nuclei: This is a term used for nuclei that is not surrounded by cytoplasm (the rest of the cell). Those are typically seen in benign tumors. A7 Bland Chromatin: Describes a uniform "texture" of the nucleus seen in benign cells. In cancer cells, the chromatin tends to be coarser. A8 Normal nucleoli: Nucleoli are small structures seen in the nucleus. In normal cells, the nucleolus is usually very small if visible at all. In cancer cells, the nucleoli become more prominent, and sometimes there are more of them. A9 Mitoses: The act of dividing cells. All attribute values on a scale of 0–10.

Although the online clustering tool used in this study is excellent for basic analysis, however, the restriction of working with very small data sets stifles its practical utility. Therefore, we coded DBSCAN in JAVA without a restriction on the size of data set and with additional features, such as using image as input data, writing cluster extraction results in a text file such as number of clusters extracted, number of noise points and cluster details. Our application also supports standard inputs such as using x,y coordinates for reading each data point, or reading CSV or space delimited files.

Scalability For the scalability study, we proceed by creating a pair-wise similarity matrix (details in "Materials and methods") of size 699 × 699 and after discretization had 245,329 data points. The proposed CCE technique performed clustering in 5 s on a core-5 machine with 1.6-GHz clock and 3.7 GB RAM running windows 8; subsequently, cluster extraction was performed in less than this time. However, for the same data set, cluster extraction

using DBSCAN took 30 min. Our implementation of DBSCAN may not be optimal, i.e., having time complexity $O(n^2)$, to speed-up DBSCAN if we would have used a tree structure for indexing spatial information, such as R*-trees [49], the time complexity would have reduced to $O(n \log n)$. However, considering the scale factor and the constants, it is unlikely that the clocked time of fast DBSCAN, i.e., $O(n \log n)$ would have reduced from 30 min to (say) under 5 min on the same machine for the data set being considered. The large running time of DBSCAN has been reported [50, 51]; actually, DBSCAN is reported to be impractically slow due to the huge I/O cost [51]. Parallelization can also be used to speedup DBSCAN, such as for 1 million point data set, Eps = 0.00015 and MinPts = 5 with 2,048 nodes DBSCAN took about 280 s [52]; however, parallelization is beyond the scope of this paper.

To quickly identify suitable input parameter values of Eps (Epsilon) and MinPts (Minimum Points), we adopted the downscaling approach, i.e., creating two versions of the input image (of the clustered similarity matrix). We scaled down from 699 × 699 pixels to medium size (300 × 300 pixels) and from 699 × 699 pixels to small size (150 × 150 pixels) and then running DBSCAN on the scaled-down versions of input image data. We found the medium-sized version to give comparable results to that obtained by using the actual data set. Image downscaling was performed using standard image editing tools such as *PhotoShop*. Therefore, first cluster extraction was performed on the medium-sized image in order to identify good values for DBSCAN parameters of Eps and MinPts; subsequently, using these parameter values performed cluster extraction using the actual clustered 699 × 699 similarity matrix. Note that for some parameter values, nothing was clustered, such as Eps = 1 and MinPts = 10, but some combinations gave fairly good results. Thus, because of downscaling, we got to know the appropriate values of Eps and MinPts in a couple of minutes instead of discovering in half an hour!

Cluster Discrimination We experimented with different values of Eps and MinPts for DBSCAN, but found that the data points located at the bottom-right corner (BR) of the clustered similarity matrix were always clustered as a single cluster, i.e., low cluster discrimination. Apparently, DBSCAN got "stuck" in the BR region of the similarity matrix most likely due to its fixed "density-based" approach. Some of the results of running DBSCAN on the complete BC data set (699 × 699) with different values of Eps and MinPts are shown in Fig. 9 with random color assignment to each cluster. For ease of comprehension, the noise points, i.e., data points, not assigned to any cluster are not shown in Fig. 9.

Figure 10 shows the cluster extraction results for the 699 × 699 clustered similarity matrix using the proposed CCE technique (Fig. 10a, b) and the traditional SCE technique (Fig. 10c). In Fig. 10 cluster boundaries are shown by red rectangles. Here, min size is the minimum size of a group of data points to be considered a cluster.

From Fig. 10a, high cluster discrimination of the proposed CCE technique can be observed, i.e., 13 clusters extracted in the BR region with minimum cluster size set to 10, while DBSCAN extracted 1,699 clusters with low cluster discrimination in BR region with MinPoints = 10 and Eps = 2 (Fig. 9a). For the proposed CCE technique, when minimum cluster size was set to five 269 biclusters extracted (Fig. 10b) with high cluster discrimination in the BR region, while DBSCAN extracted too many, i.e., 4,942 clusters for MinPts = 5 with some cluster discrimination. For CCE, when minimum cluster size was set to seven, 112 biclusters were extracted (not shown here) as opposed to 1,117 clusters extracted by DBSCAN with MinPts = 7 as shown in Fig. 9c. The traditional SCE extracted three clusters in the BR region of the clustered similarity matrix (Fig. 10c), but could not extract spatially displaced clusters w.r.t the diagonal.

Since most of the extraction takes place in the BC region for the proposed CCE technique, therefore, we do a comparison of the modes of the nine attributes (A1–A9) for the 13 clusters extracted by CCE (Fig. 10a) from the BR region with results shown in Fig. 11. Observe the spike in the value of A6 (Bare Nuclei) for clusters CCE15, CCE21 and CCE23, the region corresponding to these three clusters in the clustered similarity matrix is indicated by yellow-dotted circle (Fig. 10a).

Knowledge Discovery In order to analyze the spike in the value of A6 (Fig. 11), we revert back to the complete BC data set. In the given BC data set, Benin records are 458 (65.5 %) and malignant records are 241 (34.5 %). Summary of bicluster and cluster sizes and percentage of malignant results for the complete BC data set, BR, CCE15, CCE21 and CCE23 are shown in Table 3.

Thus, we have an interesting knowledge discovery here, i.e., although BR mainly consists of Benin records, but embedded within BR, there are three biclusters, i.e., CCE15, CCE21 and CCE23, that mostly consist of malignant records, actually the percentage malignancy of each cluster being up to twice that of the entire BC data set. On further exploration, it was found that out of 699 records, 129 records were of malignant cases having attribute A6 = 10, and out of this, 121 records were clustered and captured by the proposed CCE paradigm in few seconds.

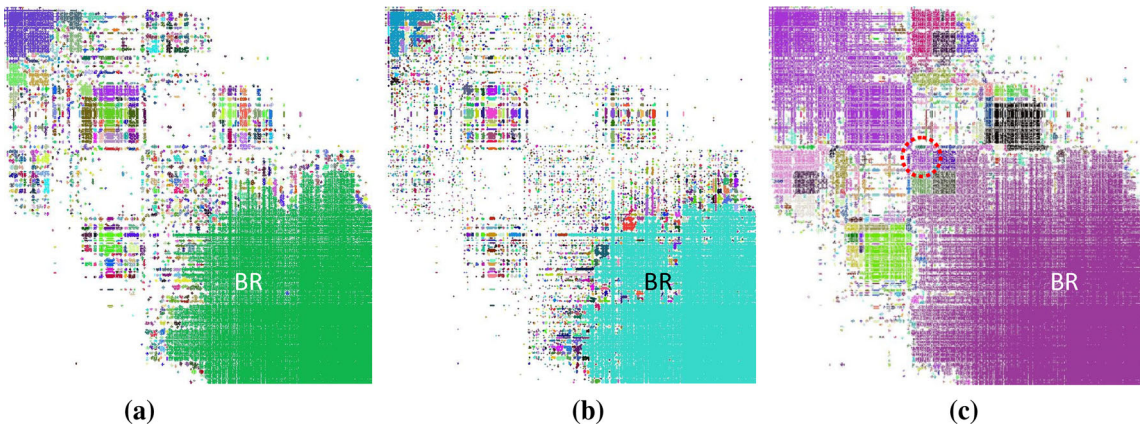

Fig. 9 Clustering results of DBSCAN using different values of Eps and MinPts for the BC data set. **a** Eps = 2, MinPts = 10, clusters = 1,699, **b** Eps = 1, MinPts = 5, clusters = 4,942, **c**: Eps = 2, MinPts = 7, clusters = 1,117

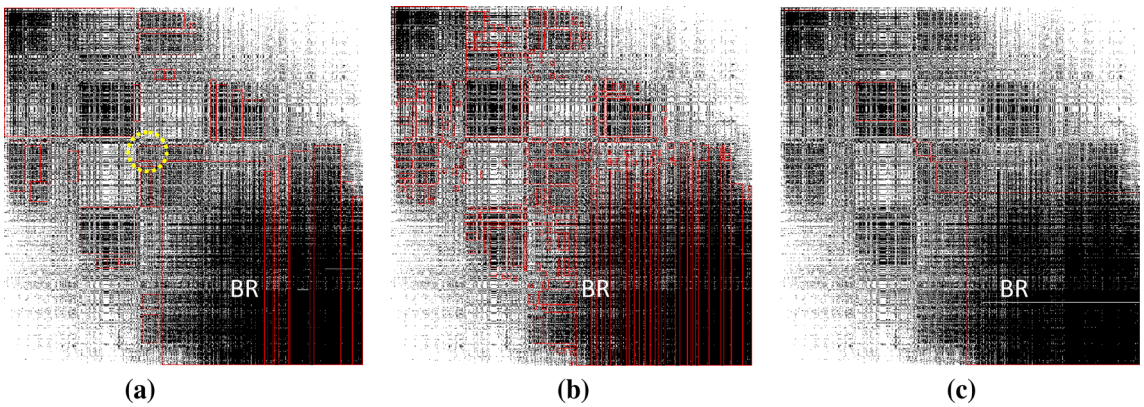

Fig. 10 Comparison of CCE and SCE results for the BC data set with high cluster discrimination in BR region. **a** CCE technique, min size = 10, clusters = 34, **b** CCE technique, min size = 5, clusters = 269, **c** SCE technique, min size = 5, clusters = 7

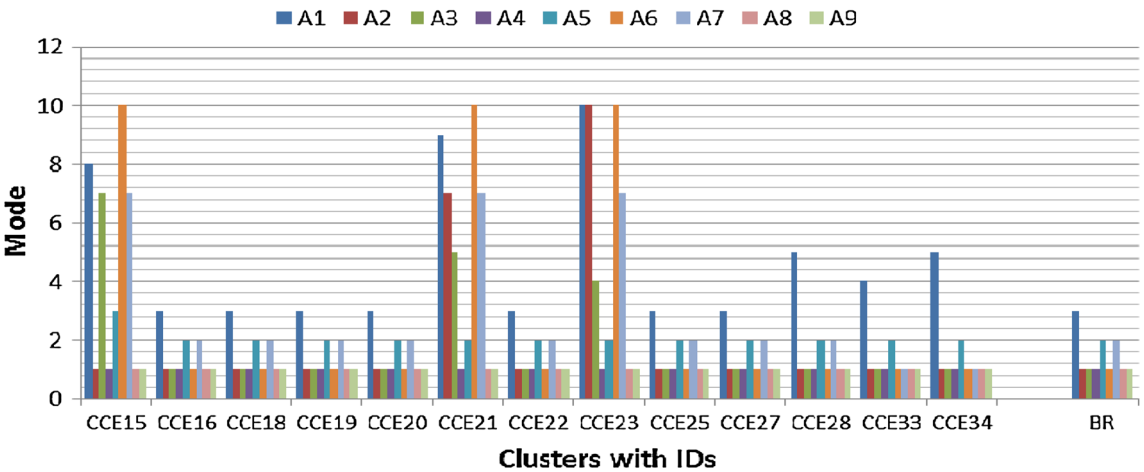

Fig. 11 Comparison of nine attributes of BC data set for the 13 CCE clusters in the BR region

Analysis of Results By visually comparing the clustering results of DBSCAN in Fig. 9 with the malignant clustering region identified by CCE as shown in Fig. 10a, one could argue that DBSCAN also seem to have clustered the malignant region too, however, note that:

Table 3 Summary of malignant results for different bi/clusters in the BR region

Bi/cluster	Rows	Columns	% Malignant
CCE15	30	13	66.66
CCE21	11	34	54.54
CCE23	80	22	77.5
BR region	439	439	29.38
BC complete	699	699	34.5

- DBSCAN performs cluster extraction by consuming CPU time orders of magnitude more than CCE.
- Identifying outlier (malignant) regions from thousands of DBSCAN clusters is much more complex as compared to performing the same task with a few dozen CCE clusters and
- Due to the inherent density connectivity property of DBSCAN, such outliers (small clusters) may be "annexed" by their large dense neighbors as can be observed in Fig. 9c shown by dotted red circle.

At the detailed data point level, since DBSCAN requires dense neighborhood for core points (such as BR in our case), therefore, with increase in density, fewer core points are obtained. However, depending on the density of the points just outside the Eps neighborhood, a core point x loses its status from being a core point, resulting in three outcomes as follows:

- x is still density reachable from the core points of its former cluster, and the remaining core points are able to hold the cluster together. Then, the number of clusters remains unchanged.
- x is still density reachable from at least two core points, but no longer acts as a density-connecting "bridge" between the core points, causing them to form separate clusters. The number of clusters increases and x is assigned to one of those clusters.
- Neither x, nor its neighboring points are able to sustain their former cluster and disappear; therefore, x becomes noise and the number of clusters decreases.

The impact of the points discussed above becomes profound, as a typical similarity matrix is unlikely to have constant density regions because of inherent noise in the data and unpredictable relationships among the records. Therefore, under such conditions, DBSCAN is unlikely to produce better results as compared to the proposed CCE technique that benefits from backtracking ("Effectiveness of CCE section") and may escape the local optima.

Discussion

In this section, we will discuss the results of sensitivity of the parameters to noise and the extraction parameters. Finally, we will discuss why the proposed technique demonstrated promising results as compared to the traditional techniques in the context of backtracking.

Sensitivity to Noise

To study the sensitivity to noise, two experiments will be performed using a 30×30 similarity matrix. In the first experiment, global random noise (i.e., 0's) will be added incrementally only in the clustered region of the clustered matrix and 0 % noise added in the non-clustered region and cluster extraction performed. This process to be repeated by adding noise to the point, beyond which CCE fails to extract clusters, the limit was found to be 40 % global noise as shown in Fig. 12a. Observe that here the global noise means the noise with reference to the entire 30×30 matrix, while local noise means noise is with reference to the 15×15 sub-matrix corresponding to the clustered region. After trying out different combinations of clustering parameters, both DBSCAN and CLIQUE successfully extracted both clusters. For k-means and k-medoids, to get the right value of k, multiple tries were made to successfully extract the two clusters. The proposed CCE technique, however, extracted the two clusters automatically, i.e., without adjusting any parameters.

The next experiment is a repetition of the first experiment, but instead of adding noise in the clustered region, the noise is incrementally added in the non-clustered region in addition to the noise already present (i.e., 0, s) in the clustered region, as shown in Fig. 12b. This process repeated till CCE failed to extract clusters; the limit was found to be 5 % noise. Despite trying several combinations of values of control parameters, neither DBSCAN nor CLIQUE could extract the two clusters cleanly; actually, noise and data were not differentiated and were instead extracted together. Figure 12c shows the results of using the proposed CCE technique by using default density value but dropping clusters of sizes 2×2. Observe clean extraction of the first cluster while most of the second cluster was also extracted cleanly. The cluster extraction results of k-means with $k = 3$ are shown in Fig. 12d, observe k-means technique could not differentiate between noise and data; similar results were observed that for k-medoids (not shown here). Note that although clustering was performed perfectly, i.e., by clustering the randomly permuted matrix that resulted in Fig. 12b, but cluster extraction of Fig. 12c was less than perfect.

Sensitivity to Extraction Parameters

In this section, we will study the effect of cluster extraction parameters for the four cluster extraction techniques discussed in this paper with summary of results shown in

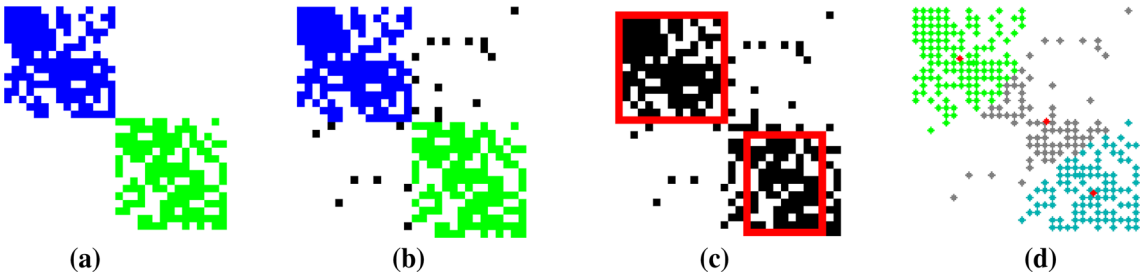

Fig. 12 Cluster extraction under noise using the proposed technique and *k*-medoids. **a** Noise added only in the clustered regions, **b** noise in the clustered and non-clustered regions, **c** cluster extraction using the proposed CCE technique, **d** cluster extraction using *k*-means with $k = 3$

Table 2. Using all combinations of parameter values for the four techniques considered would result in 40+ combinations, with some combinations being of academic nature. Therefore, a manageable subset of those combinations will be considered. Thus, for DBSCAN and CLIQUE, we will use five combinations of parameter values, i.e., (1) small values for the two control parameters, (2) large values for the two control parameters (3) large value for the first control parameter and small value for the second control parameter, (4) small value for the first control parameter and large value for the second control parameter and finally (5) the default values. For *k*-means and *k*-medoids, *k* is set to four values by extending the reasoning as in the case of DBSCAN and CLIQUE. The metric for cluster quality will be purity (*P*), and we will consider the data space of Fig. 4a for comparison. Note that using CCE, all 18 clusters were extracted with purity 1 as shown in Fig. 5a.

From Table 4, it can be observed that except for the case with parameter values Eps = 10 and MinPts = 3, using the other four combinations of extraction parameters, DBSCAN more or less considered the entire data space to be a single cluster, similar results for Eps = 10 and MinPts = 9. Although DBSCAN has been noted to be very sensitive to parameter values, with tiny changes producing clustering results with significant differences [37], however, we observe different behavior for the data sets being considered in this paper.

From Table 2, it can be observed that except for the case with parameter values $I = 16$ and $T = 3$, using the other four combinations of control parameters, CLIQUE more or less considered the entire data space to consist of two clusters instead of 18. Note that for $I = 12$ and $T = 2$, CLIQUE considered the entire data space to be a single cluster; therefore, the results are not presented here. Tuning the CLIQUE clustering parameters for a specific data set can be difficult. The reason being, both grid size and the density threshold are the parameters that greatly affect the quality of the clustering results. Even with proper tuning, the pruning stage of CLIQUE can possibly eliminate small, yet important clusters from consideration see Parson et al. [53].

Table 2 shows the results of sensitivity to parameters for *k*-means and *k*-medoids for four different values of k. It can be observed that the two partition-based cluster extraction techniques gave better results as compared to CLIQUE, but it is rather difficult to have a priori knowledge about a suitable value for k.

Effectiveness of CCE

Apparently, most of the combinatorial computing applications can be handled only by what amounts to be the brute force approach, i.e., an exhaustive search through all possible solutions. Such searches can readily be performed by using a well-known "depth-first" procedure which Walker [54] has called backtracking.

Backtracking is a modification of the brute force approach, which methodically searches for a solution for a problem among all available solutions. The algorithm proceeds by assuming that the solutions are represented by vectors (v_1, …, v_m) of values (i.e., '*B*' as per our case, Fig. 3)and by traversing the domains of the vectors in a depth-first manner, until the solutions are found. However, in the proposed CCE technique, the search is abandoned after a constant number of searches, i.e., *d*, and the best solution discovered is used.

Backtracking starts with an empty vector. Subsequently, at each stage, the algorithm extends the partial vector with a new value. Upon reaching a partial vector (v_1, …, v_i) that cannot represent a partial solution, i.e., no improvement in density after constant number of searches, i.e., $n*d$ in case of CCE, backtracking is performed by removing the values from the vector to the point density decreased, and subsequently, the algorithm proceeds by extending the vector with alternate values. Figure 13 shows a recursive version of the backtracking cluster extraction algorithm.

If S_i is the domain of v_i, then $S_1 \times \ldots \times S_m$ is considered to be solution space of the problem. The authenticity criteria used to check for acceptable vectors determines what portion of the solution space needs to be searched and that also determines the resources required by the algorithm.

Table 4 Summary of extraction results for the four techniques

	MinPts	Purity	Clusters
DBSCAN			
10	3	0.4	6
25	9	0.128	9
25	3	0.176	3
20	5	0.174	3
10	9	0.085	1
1	*T*	Purity	Clusters
CLIQUE			
IS	3	0.34	6
13	5	0.2	5
12	5	0.266	3
IS	2	0.174	2
12	2	0.085	1
k	Purity		Clusters
k-Means			
2	0.171		2
3	0.323		4
4	0.342		4
5	0.37		4
k	Purity		Clusters
k-Medoids			
2	0.171		2
3	0.257		3
4	0.314		4
5	0.35		4

The solution space can be considered as a tree, of which a depth-first traversal (shown by dashed line) is performed. The tree itself is seldom stored entirely by the algorithm; instead, just a path is stored toward a root, thus enabling backtracking. Figure 14 shows the tree corresponding to the search space used by backtracking with depth set to 2 with two levels of search spaces, i.e., S_1 and S_2, shown by dotted boundaries.

Now we do a time complexity analysis. The initialization part of DFS has time complexity $O(n)$, as every vertex must be visited once so as to mark it as "visited." The main (recursive) part of the algorithm has time complexity $O(m)$, as every edge must be crossed (twice) during the examination of the adjacent vertices of every vertex. In total, the algorithm's time complexity is $O(m + n)$. In the case of CCE, the solution space is limited to constant probing depth; therefore, the time complexity is fairly low, actually $O(1)$. The proposed CCE techniques takes $O(e)$ time, where e is the number of 1's in the discretized similarity matrix,

```
ALGORITHM extract(v1,...,vi)
  IF (v1,...,vi) is a solution THEN RETURN (v1,...,vi)
  FOR each v DO
    IF (v1,...,vi,v) is acceptable vector  THEN
      sol = extract(v1,...,vi,v)
      IF sol ≠ () THEN RETURN sol
    END
  END
  RETURN ()
```

Fig. 13 Cluster extraction by backtracking

thus performing DFS for each "1" results in an overall time complexity of O(e). However, due to discretization, the binary matrix is fairly sparse, resulting in small values for e [10].

CCE and Cognition

Comparative psychologists commonly attribute eight properties to cognitive processes in their experimental practice. These properties being (1) context sensitivity, (2) speed, (3) class formation, (4) higher order and abstract learning, (5) multi-modality, (6) inhibition, (7) monotonic integration and (8) expectation generation and monitoring [55]. The first property, i.e., context sensitivity, is one of the most important ways that a flexible strategy can differ from a stimulus-bound approach. In contemporary psychology, context sensitivity is a prerequisite for many of the most-studied cognitive capacities. Episodic memory is precisely a matter of putting experienced events in a what-when-where, autobiographical context which is simulated in "depth-limited" backtracking of the proposed technique as discussed in the last section. In our case, cognitive mapping requires the boundary refinement algorithm to "know" where it is going (search space in case of our work), where it has been (solutions evaluated in case of our work), and to be able to place its location in the context of landmarks, i.e., solution space vertices visited during the depth-first search. The third property attributed to cognitive processing, i.e., class formation commonly attributed to cognition, is the ability to group objects and situations into classes (clustering in case of our work) governed by a firm member/non-member distinction (i.e., Eq. 1) rather than by superficial perceptual similarity (similarity matrices in our case). Thus, we observe that the two major properties of cognitive processing are closely related with the proposed cluster extraction technique.

The goal of clustering is either to identify groups in a set of entities or to group the entities in some way. In the case of humans, we may associate such a process with cognition. Cognition is attributed to a whole bunch of high-level or mental functions that are concerned with attention, memory, learning, perception and planning [53]. In the

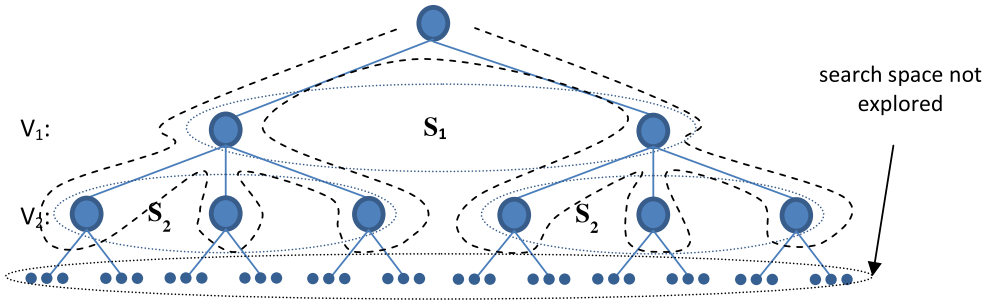

Fig. 14 Depth-limited solution search space for backtracking (*dashed line* depth-first search and *dotted line* search space)

case of computers, we can argue more technically that clustering is a process that is associated with a mapping fc between a set of N different data items and a set of k data labels.

Potential Strengths of the Proposed Technique

As per Hartigan et al. [56] and Zhang et al. [57], traditional cluster extraction methods such as k-mean style clustering, treat clusters as a statistical entity. Consequently, many vital features of the clusters, such as their densities and shapes, are aggregated using a rather simplistic approach. More specifically, first these techniques assume clusters to be spherical shaped, which may not be true in all cases, such as rectangular clusters embedded in similarity matrices. The shape of a cluster is typically described using a simple "centroid + radius" formula. Secondly, these techniques do not capture the internal features of the clusters, such as distribution of density. Therefore, the density of a cluster is either assumed uniform or varying along the radius only. Obviously, such simplistic formula-based approaches cannot adequately describe the complex cluster structure of density-based clusters. In real data, both the shapes and density distributions of density-based clusters can be arbitrary, not to mention the complex sub-region connections within each cluster.

Although DBSCAN is known to extract arbitrarily shaped clusters, but it is also well known that DBSCAN cannot handle data sets consisting of clusters of varying densities. The reason being DBSCAN's density-based definition of core points cannot categorize the core points with varying densities. For example, if the user defines the neighborhood of a point by specifying a particular radius, then DBSCAN only looks for core points that have pre-defined number of points within that radius. Subsequently, either the dense cluster is extracted as one cluster and the rest will be marked as noise, or else every data point (including noise) is extracted as one cluster.

k-Means is considerably vulnerable to the initial random position of the cluster centroids; thus, choosing *k* is an irritating problem of cluster analysis with no agreed upon solution. Thus, the strengths of *k*-means are its simplicity, yet fast for low-dimensional data and ability to find pure sub-clusters if large value of *k* is specified. The major weaknesses of *k*-means are the inability to handle non-globular data of different sizes and densities and to identify outliers; furthermore, *k*-means is restricted to data that has the notion of a center (centroid).

The proposed technique, however, appears to be impervious to the problems identified in this section, as the proposed technique performs backtracking, thus allowing it to escape the local optima; though global optima is not guaranteed. Furthermore, the technique assumes the clusters to be rectangular, which is always the case for one-way and two-way clustering, which effectively extracts clusters.

Conclusions and Future Work

In this paper, we have proposed a cognitively inspired approach of appropriately scaling level of detail, specifically by going from low level of detail, i.e., one-way clustering, to high level of detail, i.e., biclustering, in the dimension of interest—as in OLAP. The proposed technique is impervious to the ordering of the instances (rows) and the ordering of the features (columns). Thus, the proposed technique can be used for both unordered instances and features and ordered instances and features. The novelty of the proposed technique is in utilizing the benefits of global clustering, while capitalizing on the details of the biclusters extracted, i.e., knowledge discovery within knowledge. By using global parameter and backtracking with depth-limited search, the proposed technique efficiently and fairly accurately extracts connected clusters quickly. Although classical density-based clustering techniques have been reported to successfully extract arbitrary-shaped clusters from noise, however, we note the converse to be true for the data set considered in this paper. The proposed technique has also demonstrated promising noise

immunity as compared to the classical extraction techniques and unlike DBSCAN is not limited by fixed density.

We intend to conduct further studies to provide more extensive results on much larger data sets from different scientific domains. CCE algorithm here considers only rectangular objects, but it could be extended to other spatial objects like polygons. Applications of CCE to high-dimensional feature spaces should be investigated and adjacency check for high-dimensional data need to be explored. The technique showed promising results with fixed density need to explore how to dynamically adjust the density for possibly better results.

Acknowledgments This project was supported by the NSTIP strategic technologies program in the Kingdom of Saudi Arabia—Project No. (12-AGR2709-3). The authors also, acknowledge with thanks Science and Technology Unit, King Abdulaziz University for technical support.

References

1. Card SK, Mackinlay JD, Shneiderman B. Readings in information visualization—using vision to think. San Francisco: Morgan Kaufmann Publishers; 1999.
2. Ravindra K, Naik D. Multivariate data reduction and discrimination with SAS software. Cary: SAS Institute; 2000. p. 2.
3. Hartigan JA. Direct clustering of a data matrix. J Am Stat Assoc. 1972;67(337):123–9.
4. Mirkin B. Mathematical classification and clustering. Norwell: Kluwer Academic Publishers; 1996.
5. Heather T. www.heatherturner.net/turnerchapter3.pdf; 2013.
6. Sheikholeslami G et al. WaveCluster: a multi-resolution clustering approach for very large spatial databases. In: Gupta A, Shmueli O, Widom J, editors. In: Proceedings of 24th international conference very large data bases. New York City, Morgan Kaufmann; 1998, p. 428–438.
7. Kaufman L, Rousseeuw PJ. Finding groups in data: an introduction to cluster analysis, vol. 344. New York: Wiley-Interscience; 2009.
8. Ester M, Kriegel HP, Xu X. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In: Egenhofer M, Herring J, editors. Advances in spatial databases. Berlin: Springer; 1995. p. 67–82.
9. Agrawal R, Gehrke J, Gunopulos D, Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of the 1998 ACM SIGMOD international conference on management of data, vol 27; 1998. p. 94–105.
10. Ahsan A, Amir H. A new biclustering technique based on crossing minimization. Neurocomputing. 2006;69(16):1882–96.
11. Pang-Ning T, Steinbach M, Kumar V. Introduction to data mining. Boston: Addison-Wesley Publishers; 2006.
12. Card SK, Mackinlay JD, Shneiderman B, editors. Readings in information visualization: using vision to think. San Francisco: Morgan Kaufmann; 1999.
13. Chen K, Liu L. VISTA: validating and refining clusters via visualization. J Inf Visual. 2004;3(4):257–70.
14. Kaski S, Sinkkonen J, Peltonen J. Data visualization and analysis with self-organizing maps in learning metrics. DaWaK 2001, LNCS 2114, (2001), p. 162–173.
15. Böhm C, Kailing K, Kröger P, Zimek A. Computing clusters of correlation connected objects. In: Proceedings of the 2004 ACM SIGMOD international conference on management of data; 2004, p. 455–466.
16. Appice A, Lanza A, Varlaro A. Spatial clustering of related structured objects for topographic map interpretation. In: Proceedings of the workshop on mining spatio-temporal data (MSTD) in conjunction with ECML/PKDD 2005, 9–21, Porto, Portugal.
17. Lu W, Han J et al. Discovery of general knowledge in large spatial databases, 2005, In: Proceedings of far east workshop on geographic information systems. Singapore; 1993. p. 275–289.
18. http://www.ret.gov.au/resources/upstream_petroleum.
19. Ng RT, Han J. Efficient and effective clustering methods for spatial data mining. In: Bocca JB, Jarke M, Zaniolo C, editors. In: Proceedings of the 20th international conference very large data bases (VLDB'94). Santiago de Chile: Morgan Kaufmann; 1994. p. 144–155.
20. Ankerst M et al. OPTICS: ordering points to identify the clustering structure. In Delis A, Faloutsos C, Ghandeharizadeh S, editors. In: Proceedings 1999 ACM SIGMOD international conference on management of data Philadelphia: ACM Press; 1999. p. 49–60.
21. Hinneburg A, Keim DA. An efficient approach to clustering in large multimedia databases with noise. In: Proceedings of international conference on knowledge discovery and data mining, KDD-98, New York: AAAI Press; 1998. p. 58–65.
22. Gibson D, Kleinberg JM, Raghavan P. Clustering categorical data: an approach based on dynamical systems. In: Gupta A, Shmueli O, Widom J, editors. In: Proceedings of 24th international conference on very large data bases. New York City: Morgan Kaufmann; 1998. p. 311–322.
23. Codd EF, Codd SB, Salley CT. Providing OLAP to user-analysts: an IT mandate. Technical report. E. F. Codd & Associates; 1993.
24. Rivest Sonia, Bedard Yvan, Marchand Pierre. Toward better support for spatial decision making: defining the characteristics of spatial on-line analytical processing (SOLAP). GEOMATICA-OTTAWA. 2001;55(4):539–55.
25. Tucker LR. The extension of factor analysis to three-dimensional matrices. In: Frederiksen N, Gulliksen H, editors. Contributions to mathematical psychology. New York: Holt, Rinehart, and Winston; 1964. p. 109–27.
26. Cheng Y, Church GM. Biclustering of expression data. In: Proceedings of the eighth international conference on intelligent systems for molecular biology, vol 8; 2000, p. 93–103.
27. Madeira SC, Oliveira AL. Biclustering algorithms for biological data analysis: a survey. Comput Biol Bioinform IEEE/ACM Trans. 2004;1(1):24–45.
28. Orlin J. Containment in graph theory: covering graphs with cliques. Nederl Akad Wetensch Indag Math. 1977;39:211–8.
29. Yang D, Rundensteiner EA, Ward MO. Summarization and matching of complex patterns in streaming environment. In: Proceedings of the VLDB endowment, vol 5; 2011. p. 121–132. http://vldb.org/pvldb/vol5/p121_diyang_vldb2012.pdf.
30. Qiu BZ, Zhang L. An effective nonparametric grid-based clustering algorithm. J Inform Comput Sci. 2008;5(1):1-6.
31. Xiaoyun C, Yi C, Xiaoli Q, Min Y, Yanshan H. PGMCLU: a novel parallel grid-based clustering algorithm for multi-density datasets. In Web Society, SWS'09. 1st IEEE Symposium on, 2009, p. 166–171.
32. Akodjènou-Jeannin MI, Salamatian K, Gallinari P. Flexible grid-based clustering. In: Kok JN, Koronacki J, Lopez de Mantaras R, Matwin S, Mladenič D, Skowron A, editors. Knowledge discovery in databases: PKDD. Berlin: Springer; 2007. p. 350–357.
33. Schikuta E. Grid-clustering: an efficient hierarchical clustering method for very large data sets. In: Proceedings of the 13th international conference on pattern recognition, vol. 2; 1996. p. 101–105.

34. Chu SC, Roddick JF, Pan JS. An efficient k-medoids-based algorithm using previous medoid index, triangular inequality elimination criteria, and partial distance search. In Data warehousing and knowledge discovery, Berlin: Springer; 2002, p. 63–72.
35. Zhang Q, Couloigner I. A new and efficient k-medoid algorithm for spatial clustering. In: Computational science and its applications—ICCSA. Berlin: Springer; 2005, p. 181–189.
36. Achtert E, Böhm C, David J, Kröger P, Zimek A. Robust clustering in arbitrarily oriented subspaces. In: Proceedings of SDM. 2008.
37. Tan J, Zhang J, Li W. An improved clustering algorithm based on density distribution function. Comput Inf Sci. 2010;3(3):23.
38. Erten C, Sözdinler M. Biclustering expression data based on expanding localized substructures. In: Bioinformatics and computational biology. Berlin: Springer; 2009. p. 224–235.
39. Zhou J, Lazarevic A, Hsu KW, Srivastava J, Fu Y, Wu Y. Unsupervised learning based distributed detection of global anomalies. Int J Inf Technol Decis Mak. 2010;9(06):935–57.
40. Zhou A, Zhou S, Cao J, Fan Y, Hu Y. Approaches for scaling DBSCAN algorithm to large spatial databases. J Comput Sci Technol. 2000;15(6):509–26.
41. Sander J, Ester M, Kriegel H-P, Xu X. Density-based clustering in spatial databases: the algorithm gdbscan and its applications. Data Min Knowl Disc. 1998;2(2):169–94.
42. Qian Weining, Gong XueQing, Zhou AoYing. Clustering in very large databases based on distance and density. J Comput Sci Technol. 2003;18(1):67–76.
43. Ester M, Kriegel HP, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd international conference on knowledge discovery and data mining. AAAI Press; 1996, p. 226–231.
44. Viswanath P, Pinkesh R. l-dbscan: a fast hybrid density based clustering method. In: Pattern recognition, 2006. ICPR 2006. 18th international conference, vol 1, 2006. p. 912–915.
45. Ilango MR, Mohan V. A survey of grid based clustering algorithms. Int J Eng Sci Technol. 2010;2(8):3441–6.
46. Sander J, Qin X, Lu Z, Niu N, Kovarsky A. Automatic extraction of clusters from hierarchical clustering representations. In: Kyu-Young W, Jongwoo J, Kyuseok S, Jaideep S, editors. Advances in knowledge discovery and data mining. Berlin: Springer; 2003, p. 75–87.
47. Manning CD, Raghavan P, Schütze H. Introduction to information retrieval. New York: Cambridge University Press; 2008.
48. Lee DJ, Lane RM, Chang GH. Three-dimensional reconstruction for high-speed volume measurement. In: Proceedings of the international society for optical engineering, machine vision and three-dimensional imaging systems for inspection and metrology, vol 4189; 2001. p. 258–267.
49. Liu B. A fast density-based clustering algorithm for large databases. In: Proceedings of International Conference on Machine Learning and Cybernetics; 2006, p. 996–1000.
50. Dash M, Liu H, Xu X. '$1 + 1 > 2$': merging distance and density based clustering. In: Proceedings of seventh international conference on database systems for advanced applications; 2001, p. 32–39.
51. Bach JR, Horowitz B. Indexing method for image search engine. U.S. Patent No. 6,084,595. 4 Jul. 2000.
52. Welton B, Samanas E, Miller BP. Mr. scan: extreme scale density-based clustering using a tree-based network of gpgpu nodes. In Proceedings of SC13: international conference for high performance computing, networking, storage and analysis. 2013; 13:84.
53. Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: a review. ACM SIGKDD Explor Newsl. 2004;6(1):90–105.
54. Walker RJ. An enumerative technique for a class of combinatorial problems. In: Bellman R, Hall M Jr, editors. Combinatorial analysis. In: Proceedings of symposium applied mathematics 10. Providence, Rhode Island: Ame. Math. Society; 1960. p. 91–94.
55. Buckner C. A property cluster theory of cognition. Philos Psychol (ahead-of-print); 2013;1–30.
56. Hartigan JA, Wong MA. Algorithm AS 136: a k-means clustering algorithm. J R Stat Soc Ser C. 1979;28(1):100–8.
57. Zhang T, Ramakrishnan R, Livny M. BIRCH: an efficient data clustering method for very large databases. In: ACM SIGMOD Record. 25(2):103–114; 1996.