НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
сентябрь–октябрь 2023 Том 23 № 5 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
September–October 2023 Vol. 23 No 5 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

doi: 10.17586/2226-1494-2023-23-5-967-979

Clustering in big data analytics: a systematic review and comparative analysis (review article)

Hechmi Shili✉

University of Tabuk, Tabuk, 47311, Arabie Saoudite

asuhaili@ut.edu.sa✉, https://orcid.org/0000-0001-5989-4457

Abstract

In the modern world, the widespread use of information and communication technology has led to the accumulation of vast and diverse quantities of data, commonly known as Big Data. This necessitates the need for novel concepts and analytical techniques to help individuals extract meaningful insights from rapidly increasing volumes of digital data. Clustering is a fundamental approach used in data mining to retrieve valuable information. Although a wide range of clustering methods have been described and implemented in various fields, the sheer variety complicates the task of keeping up with the latest advancements in the field. This research aims to provide a comprehensive evaluation of the clustering algorithms developed for Big Data highlighting their various features. The study also conducts empirical evaluations on six large datasets, using several validity metrics and computing time to assess the performance of the clustering methods under consideration.

Keywords

big data, clustering, data mining, empirical evaluations, performance metrics

For citation: Shili H. Clustering in big data analytics: a systematic review and comparative analysis (review article). *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2023, vol. 23, no. 5, pp. 967–979. doi: 10.17586/2226-1494-2023-23-5-967-979

УДК 004.65

Кластеризация в аналитике больших данных: системный обзор и сравнительный анализ (обзорная статья)

Шили Хечми✉

Университет Табука, Табук, 47311, Саудовская Аравия

asuhaili@ut.edu.sa✉, https://orcid.org/0000-0001-5989-4457

Аннотация

В современном мире широкое использование информационных и коммуникационных технологий привело к накоплению огромных и разнообразных объемов данных, широко известных как большие данные. Это обуславливает потребность в новых концепциях и аналитических методах, которые помогают извлекать важные идеи из быстро растущих объемов цифровых данных. Кластеризация — фундаментальный подход, используемый в интеллектуальном анализе данных для извлечения ценной информации. Несмотря на то, что в различных областях описано и реализовано множество методов кластеризации, данное разнообразие усложняет задачу отслеживания последних достижений в области больших данных. Работа направлена на всестороннюю оценку алгоритмов кластеризации, разработанных для больших данных, с выделением их различных функций. Выполнены эмпирические оценки шести больших наборов данных с использованием нескольких показателей достоверности и времени вычислений для оценки производительности рассматриваемых методов кластеризации.

Ключевые слова

большие данные, кластеризация, интеллектуальный анализ данных, эмпирические оценки, показатели производительности

Ссылка для цитирования: Хечми Шили. Кластеризация в аналитике больших данных: системный обзор и сравнительный анализ (обзорная статья) // Научно-технический вестник информационных технологий, механики и оптики. 2023. Т. 23, № 5. С. 967–979 (на англ. яз.). doi: 10.17586/2226-1494-2023-23-5-967-979

Introduction

The advent of Artificial Intelligence, Mobile Internet, Social Networks, and the Internet of Things has led to an explosion in data production, leading to the rapid growth of large data applications. However, conventional data processing approaches are insufficient to cope with the vast and complex nature of big data. As a result, analyzing this massive volume of data has become the greatest challenge for researchers and scientists, making big data one of the most relevant topics in computer science today. Furthermore, data volume is not the only challenge in big data analytics, as there are seven core problems, commonly known as the seven V's, in managing massive data [1]:

Volume: Big data refers to extremely large datasets which can range from terabytes to petabytes and beyond.

Velocity: Big data is generated and updated in real-time, often at high speed, which can pose challenges for storage, processing, and analysis.

Variety: Big data comes from a variety of sources, including structured, semi-structured, and unstructured data, such as social media, text, images, video, and sensor data.

Veracity: Big data can be noisy, uncertain, and inconsistent due to errors, biases, and data quality issues. Veracity refers to the accuracy and reliability of the data.

Value: The ultimate goal of big data is to extract insights and value from the data, such as identifying patterns, making predictions, and improving decision-making.

Variability: Big data is subject to changes over time, which can affect its meaning and relevance. Variability refers to the extent to which the data changes over time.

Visualization: Big data can be challenging to interpret and understand, which is why visualization tools and techniques are essential for making sense of the data and communicating insights to stakeholders.

Effective methods for information discovery are crucial to deal with the massive amount of data produced by the latest technological advancements. Data mining approaches, such as clustering, are well-suited for this task. Clustering is a fundamental process in data mining applications that groups together similar data points while separating unrelated ones, with the goal of optimizing similarities and reducing differences among different clusters [2]. Unlike classification, clustering is an unsupervised activity in machine learning, where data points with unknown class labels are used to learn a classification model. Due to its unsupervised nature, clustering is considered one of the most challenging tasks in machine learning, with different algorithms proposed by researchers leading to varying clustering outcomes. Additionally, the presentation sequence of data points and the selection of parameters can significantly impact the final partition using the same clustering algorithm [3]. Despite a wide range of reviews and comparative research on clustering techniques, the exploration of algorithms that can efficiently cluster large datasets remains an open challenge. Thus, the primary objective of this study is to provide a comprehensive review of clustering algorithms for massive datasets. To achieve this objective, we first analyze and evaluate five classified classes of clustering algorithms including model-based algorithms, partitioning, hierarchical, grid, and density. Second, we conduct accurate comparisons of the studied algorithms testing them on real datasets in terms of clustering criteria and big data characteristics.

Types of Clustering Algorithms

Due to the vast number of clustering algorithms available, this section presents a classification framework that categorizes the numerous clustering methods prevalent in the literature. As depicted in Fig. 1, clustering techniques can be classified into five types: Hierarchical, Partitioning, Model-based, Grid-based, and Density-based clustering.

In this study, we evaluated several clustering algorithms to determine their suitability for our dataset. Each algorithm has its own strengths and weaknesses, and it is important to understand these characteristics in order to select the most appropriate method for a given application. To summarize the benefits, drawbacks, and important aspects of different clustering algorithms, please refer to Tables 1 and 2 for a comprehensive overview of the clustering techniques and their corresponding characteristics. In Table 2, several symbols are used to represent key characteristics

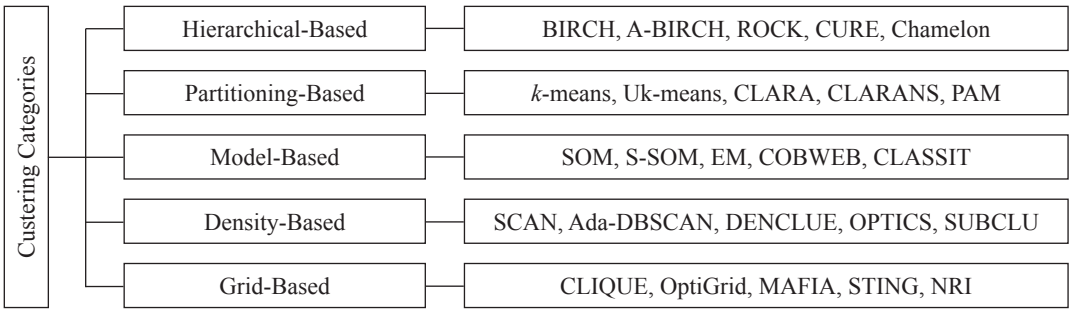

Fig. 1. Categorization of clustering techniques for big datasets

Table 1. The benefits and drawbacks of the different clustering types [2]

Categories	Benefits	Drawbacks
Hierarchical Algorithms	1) It can be useful for exploring the structure of the data and identifying patterns. 2) It does not require specifying the number of clusters beforehand.	1) It can be computationally expensive and slow for large datasets. 2) It may not work well with high-dimensional data.
Partitioning Algorithms	1) It is fast and efficient for large datasets. 2) It can handle high-dimensional data.	1) It may require specifying the number of clusters beforehand, which can be difficult. 2) It may not work well with non-linearly separable data.
Model-based Algorithms	1) It can handle complex data distributions. 2) It can estimate the number of clusters automatically.	1) It can be computationally expensive for large datasets. 2) It may be sensitive to initialization.
Density-based Algorithms	1) It can handle data with arbitrary shapes and sizes of clusters. 2) It can detect noise and outliers.	1) It can be computationally expensive for large datasets. 2) It may require setting a threshold for density, which can be difficult.
Grid-based Algorithms	1) It can handle large datasets efficiently. 2) It can identify clusters of varying sizes and shapes.	1) It may require specifying the size and shape of the grid. 2) It may not work well with high-dimensional data.

and complexities of different clustering algorithms. The symbol n refers to the input data size. The symbol k typically stands for the number of clusters, and d signifies the dimensionality of the data. Additionally, m represents a specific algorithm-related parameter, p denotes algorithmic parameters. For further details and references, you can refer to the sources mentioned in the table.

Hierarchical clustering

Hierarchical clustering [4] is a cluster analysis technique that utilizes either agglomerative or divisive algorithms to construct a cluster hierarchy. In the agglomerative algorithm, each item is treated as a cluster, and the clusters are eventually merged (bottom-up). Conversely, the divisive process starts with a single cluster containing all objects and separates it into individual clusters (top-down) by adding one object at a time. Generally, hierarchical algorithms have lower quality since they cannot adjust the clusters after merging in the merging method or dividing in the divisive method. To address this issue, hierarchical algorithms are often combined with another algorithm in a hybrid clustering technique. Popular algorithms of this clustering type include BIRCH [5], CURE [6] and ROCK [7]. A-BIRCH [8] is a recent method for the BIRCH clustering algorithm [5] that automates threshold estimates.

Partitioning-based clustering

A partitioning clustering method is designed to divide a set of n unlabeled objects into k groups, with at least one object in each group. *K*-means [9] is a commonly used algorithm for partitioning clustering, each cluster is represented by a prototype which is the average value of the elements in the cluster. The main idea behind the k-means algorithm is to minimize the cumulative distance between objects and their cluster prototypes. In the k-means procedure, the prototype is determined as the mean vector of the cluster elements. Another variant of the k-means algorithm is the Uk-means [10] which determines the optimal number of clusters without any initialization or parameter selection. The Partitioning around Medoids (PAM) algorithm [4] is another algorithm in the partitioning family where the cluster prototype is selected as one of the closest elements to the center of the cluster. Additionally, Clustering Large Applications (CLARA) [11] and Clustering Large Applications based upon Randomized Search (CLARANS) [12] are improved versions of the k-Medoid algorithm for spatial database mining. IF-CLARANS [13] aims to incorporate the concept of intuitionistic fuzzy sets into the CLARANS algorithm to handle ambiguity in large dataset mining.

Model based clustering

In these clustering approaches, the data set is segregated into theoretical models such as mathematics and statistical distribution. The probability distribution, as a mathematical framework, is used in this clustering algorithm for optimization purposes. Although this algorithm is suitable for creating a new model using an existing model to better represent the data, it is time-consuming, and the model parameters significantly impact the clustering outcome. Among the most renowned clustering algorithms in this category are EM [14], COBWEB [15], CLASSIT [16], Self-Organizing Map (SOM) [17]. Smoothed SOM (S-SOM) [18] is a novel SOM variant that improves the regular SOM vector quantification and grouping capabilities when dealing with outliers.

Density based clustering

Density-based clustering is founded upon the concept of density where clusters are formed in dense regions that are separated from other areas. The primary idea is to progressively introduce concepts until the neighboring cluster density surpasses a certain threshold or limit. DBSCAN [19] is one of the most well-known density-based algorithms which defines a cluster as a group of densely connected points. Ada-DBSCAN [20] is a novel adaptive density-based spatial clustering method that integrates a data block merger and division under control. OPTICS [21] is another density-based technique that addresses the challenge of identifying significant clusters in data with varying densities. DENCLUE [22] is a renowned

Table 2. The most important aspects of different clustering algorithms

Categories	Algorithm			Dataset				Execution/performance			
	Name	Author	Ref.	Volume	Type	Handle noise	Dimension	Inputs	Complexity	Cluster shape	Overlapped cluster
Hierarchical-Based	BIRCH	Zhang et al.	[5]	Big	Numerical	×	Low	2	$O(n)$	Non-convex	×
	A-BIRCH	Boris Lorbeer et al.	[8]	Big	Numerical	×	Low	1	$O(n)$	Non-convex	×
	CURE	Guha et al.	[6]	Big	Numerical	√	High	2	$O(n^2\log n)$	Arbitrary	√
	ROCK	Guha et al.	[7]	Big	Categorical/ Numerical	×	Low	1	$O(n^2 + \text{nmmma} + n^2\log n)$	Arbitrary	×
	Chamelon	Krpyis et al.	[28]	Big	All type	×	High	3	$O(n^2)$	Arbitrary	√
Partitioning-Based	PAM	R.T.Ng and J. Han	[4]	Little	Numerical	×	Low	1	$O(k-(n-k)^2)$	Non-convex	×
	CLARA	L. Kaufman and P. Rousseeuw	[11]	Big	Numerical	×	Low	1	$O(k(40+k)^2 + k(n-k))$	Non-convex	×
	CLARANS	R.T.Ng and J. Han	[12]	Big	Numerical	×	Low	2	$O(kn^2)$	Non-convex	×
	k-means	J MacQueen	[9]	Big	Numerical	×	Low	1	$O(nkd)$	Non-convex	×
	Uk-means	Kristina et al.	[10]	Big	Numerical	×	Low	No	$O(nkd)$	Non-convex	×
Model-Based	SOM	T. Kohonen	[15]	Little	Multivariate	×	High	2	$O(n^2m)$	Non-convex	√
	S-SOM	Pierpaolo et al.	[18]	Little	Multivariate	×	High	2	$O(n^2m)$	Non-convex	√
	EM	Dempster et al.	[14]	Big	Spatial	×	High	3	$O(knp)$	Non-convex	√
	COBWEB	Fisher et al.	[15]	Little	Numerical	×	Low	1	$O(n^2)$	Non-convex	×
	CLASSIT	Gennari JH. et al.	[16]	Little	Numerical	×	Low	1	$O(n^2)$	Non-convex	×
Density-Based	DBSCAN	Ester et al.	[19]	Big	Numerical	×	Low	2	$O(n\log n)$	Arbitrary	×
	Ada- DBSCAN	ZIHAO CAI et al.	[20]	Big	Numerical	×	Low	2	$O(n\log n)$	Arbitrary	×
	OPTICS	Ankerst et al.	[21]	Big	Numerical	√	Low	2	$O(en)$	Arbitrary	×
	DENCLUE	Hinneburg A. and Keim D. A.	[22]	Big	Numerical	√	High	2	$O(\log\|D\|)$	Arbitrary	√
	SUBCLU	Karin Kailling and Hans-Peter Kriege	[23]	Big	Numerical	√	High	2	no	Arbitrary	√
Grid-Based	CLIQUE	Agrawal et al.	[25]	Big	Numerical	×	High	2	$O(ck + mk)$	Arbitrary	√
	STING	Wang et al.	[24]	Big	Spatial	√	Low	1	$O(k)$	Arbitrary	×
	MAFIA	Goil et al.	[5]	Big	Numerical	×	High	2	$O(ck + mk)$	Arbitrary	×
	OptiGrid	Hinnebu rg and Keim	[26]	Big	Spatial	√	High	3	$O(nd)$; $O(nd\log n)$	Arbitrary	√
	NRI	CHENG et al.	[27]	Big	Numerical	×	High	2	$O(mk)$	Arbitrary	√

density-based clustering method that represents the overall density of a set of points as the sum of each point influence functions. SUBCLU [23] is another member of the density-based family which builds upon DBSCAN and adopts the notion of density-connected clusters.

Grid based clustering

Grid-based clustering methods involve dividing the data space into hierarchical structures composed of rectangular spaces. The fundamental idea behind this technique is to transform the original data space into a grid format that determines the clustering size. Although this algorithm is highly scalable and suitable for low time complexity parallel processing, it compromises the quality and accuracy of clusters and is vulnerable to high granularity. Common algorithms in this category include Statistic Information Grid Approach (STING) [24], CLIQUE [25], OptiGrid [26], and MAFIA [5]. NRI [27] is a novel grid-based clustering approach that employs an intuitive neighbor relationship to enhance data clustering efficiency.

Clustering Evaluation Indices

In the context of unsupervised learning methods, various performance measurements were required. This section focuses on elucidating the metrics employed for evaluating performance. Initially, in order to streamline and condense the section related to Cluster Validity Indices (CVIs), we establish the overarching notation employed throughout this research. Additionally, we introduce specific notations utilized to elucidate various indices [29].

Notation

Let X be a dataset comprising N objects represented as vectors in an F-dimensional space: $X = \{x_1, x_2, \ldots, x_N\} \subseteq \mathcal{R}^F$. A clustering or partition clustering in X refers to a collection of disjoint clusters that effectively divides X into K groups: $C = \{c_1, c, \ldots, c_k\}$ where the union of all c_k in C is equal to X, and c_k intersected with c_l is an empty set for any $k \neq l$. The centroid of a cluster c_k is determined by its mean vector, denoted as $\overline{c}_k = 1/|c_k| \sum_{x_i \in c_k} x_i$. Similarly, the centroid of the entire dataset X is calculated as the mean vector of all data points, represented by $\overline{X} = 1/N \sum_{x_i \in X} x_i$.

To quantify the distance between objects x_i and x_j, we will use the Euclidean distance, denoted as $d_e(x_i, x_j)$. In particular, the Point Symmetry-Distance [30] between the object x_i and the cluster c_k is defined as:

$$d^*_{ps}(x_i, c_k) = 1/2 \sum \min(2)_{x_i \in c_k} \{d_e(2\overline{c}_k - x_i, x_j)\}. \quad (1)$$

The point $2\overline{c}_k - x_i$ referred to as the symmetric point of x_i with respect to the centroid of c_k. The function $\sum\min$ can be understood as a modified version of the min function, where $\sum\min(n)$ calculates the sum of the n lowest values among its arguments. Similarly, we can define the $\sum\max$ function as a variation of the max function. Lastly, let's introduce the notation n_w, as it is utilized by several indices. n_w represents the count of object pairs within a partition that belong to the same cluster, given by $n_w = \sum c_k \in C \binom{c_k}{2}$.

Definitions of Evaluation Indices

In the context of clustering, it is important to have suitable internal evaluation indices for comparing different clustering algorithms. These internal indices are specifically designed to assess the quality of clustering solutions when the true clustering is unknown. They provide measures based on the structural characteristics of the clusters, allowing us to evaluate the consistency and effectiveness of the clustering algorithms. Therefore, in this context, we will present the definitions of several internal evaluation indices, including the Dunn (D) index, Calinski-Harabasz (CH) index, Davies-Bouldin (DB) index, Silhouette (Sil) index, and Negentropy Increment (NI) index [29]. These indices will be used to compare and assess the performance of different clustering approaches.

Dunn index. The D index evaluates both the compactness and separation of individual clusters [31]. It measures the inter-cluster distances (separation) relative to the intra-cluster distances (compactness). The index is calculated using the following equation

$$DVI = \frac{\min\limits_{c_k \in C} \left\{ \min\limits_{c_l \in C \setminus c_k} \{\delta(c_k, c_l)\} \right\}}{\max\limits_{c_k \in C} \{\Delta(c_k)\}}, \quad (2)$$

where

$$\delta(c_k, c_l) = \min_{x_i \in c_k} \min_{x_j \in c_l} \{d_e(x_i, x_j)\}, \quad (3)$$

$$\Delta(c_k) = \max_{x_i, x_j \in c_k} \{d_e(x_i, x_j)\}. \quad (4)$$

The variable c_l represents a specific cluster within the set C being evaluated. δ measures the minimum distance between points in different clusters c_k and c_l, reflecting their separation. Conversely, Δ computes the maximum distance between points within a single cluster c_k indicating its internal compactness. A higher D index value indicates better clustering, with more distinct and compact clusters.

Calinski–Harabasz index. The CH index is recognized as a variance ratio metric where the validity function of a cluster is based on the average of the sum of the squared distances between clusters and between elements within the cluster [32]. It aims to determine the dispersion of elements within their cluster and the distance from other clusters.

$$CH(C) = \frac{(N-K) \sum\limits_{c_k \in C} |c_k| d_e(\overline{c}_k, \overline{X})}{(K-1) \sum\limits_{c_k \in C} \sum\limits_{x_i \in c_k} d_e(x_i, \overline{c}_k)}.$$

A higher CH index value indicates better clustering, with more compact and well-separated clusters.

Davies-Bouldin index [33]. This metric measures the compactness and separation of clusters by calculating the average similarity between each cluster and its most similar neighboring cluster, while considering the distance between cluster centroids. The DB index aims to minimize intra-cluster dissimilarity and maximize inter-cluster dissimilarity.

$$DB(C) = \frac{1}{K} \sum_{c_k \in C} \max_{c_l \in C \setminus c_k} \left\{ \frac{S(c_k) + S(c_l)}{d_e(\overline{c}_k, \overline{c}_l)} \right\}.$$

The variable S denotes the average similarity of data points within a cluster to its centroid, and it's calculated as follows:

$$S(c_k) = 1/|c_k| \sum_{x_i \in c_k} d_e(x_i, \bar{c}_k).$$

A lower DB index value indicates better clustering, with more distinct and well-separated clusters.

The Silhouette index is a normalized summation-type index. It evaluates the cohesion of a cluster by measuring the distance between all the points within the cluster, and the separation is based on the nearest neighbor distance. The Sil index is defined as follows:

$$Sil(C) = 1/N \sum_{c_k \in C} \sum_{x_i \in c_k} \frac{b(x_i, c_k) - a(x_i, c_k)}{\max\{a(x_i, c_k), b(x_i, c_k)\}},$$

where a represents a data points average cohesion within its cluster, gauging its proximity to other cluster members. Meanwhile, b computes a data point average separation from the nearest neighboring cluster, highlighting its distinctiveness. These calculations are performed as follows:

$$a(x_i, c_k) = 1/|c_k| \sum_{x_j \in c_i} d_e(x_i, x_j),$$

$$b(x_i, c_k) = \min_{c_l \in C \setminus c_k} \left\{ 1/|c_i| \sum_{x_j \in c_l} d_e(x_i, x_j) \right\}.$$

The Sil index ranges from –1 to 1, with higher values indicating better clustering outcomes.

Negentropy increment index. This index is based on cluster normality estimation. It is not reliant on cohesion and separation measures. The index quantifies the improvement in cluster organization and randomness reduction achieved by a clustering algorithm. It is defined as:

$$NI(C) = 1/2 \sum_{c_k \in C} p(c_k) \log|\sum c_k| - 1/2 \log|\sum X| - \\ - \sum_{c_k \in C} p(c_k) \log p(p(c_k)),$$

where $p(c_k) = |c_k|/N$, $\sum c_k$ represents the covariance matrix of cluster c_k, $\sum X$ represents the covariance matrix of the whole dataset, and $|\sum\square|$ denotes the determinant of a covariance matrix. Although the authors initially proposed the index as defined above, they later suggested a correction due to unsatisfactory results obtained. However, for the purposes of this work, we will use the index in its original form since the correction does not meet the CVI selection criterion used. Lower NI values indicate superior clustering solutions.

Candidate Clustering Algorithms

The objective of this section is to identify successful candidate clustering algorithms for large datasets of each algorithm type listed in Table 1. Our selection process was based on the properties outlined in Table 2, with a focus on algorithms that satisfy most of the seven big data property criteria. In order to gauge the suitability of each algorithm, we referred to Table 2, which offers an outline of the evaluations conducted using diverse methodologies as specified earlier, and in accordance with the criteria provided. We also considered several studies that have effectively compared multiple modern clustering algorithms for large datasets [34, 35]. It is crucial to note that our aim is to compare algorithms of each type to determine their efficacy for clustering large datasets. Within the same type, we will only select the most appropriate algorithm based on the specified criteria. Consequently, we have selected the following algorithms: A-BIRCH [8], Ada-DBSCAN [20], S-SOM [18], NRI [27], and Uk-means [10]. In this section, we will discuss each chosen algorithm in detail, including its operation, advantages and disadvantages, and required input variables.

A-BIRCH Algorithm

The BIRCH algorithm utilizes a tree structure to create a dendrogram which is known as a clustering feature tree (CF tree) [28]. The CF tree is created by dynamically scanning the dataset and does not require the entire dataset in advance. It involves two key steps: first, scanning the database to create a memory tree, and second, grouping the leaf nodes using a specified threshold (T) and branching factor (B) criteria to create a high balanced CF tree. As the data is scanned, the CF tree is traversed and the nearest node is selected at each level. If the nearest leaf cluster is identified for the data point, a test is conducted to check if the data point belongs to the candidate cluster. If not, a new cluster is created with a diameter greater than the specified T. The BIRCH algorithm is efficient in handling noise and can find suitable clustering with a single dataset scan and further enhance the quality with a few more scans. However, to ensure satisfactory clustering performance, BIRCH requires both the cluster count and the threshold T to correctly measure the clusters.

A-BIRCH is a newer version of the BIRCH clustering algorithm that automatically estimates the threshold value required for clustering. This approach calculates the optimum threshold variable from the data, enabling A-BIRCH to perform clustering without the global clustering phase which is typically the final phase of BIRCH. A-BIRCH can determine if the data satisfies the necessary constraints and will provide an alert if the constraints are not met. This approach eliminates the need to know the number of clusters ahead of time and can improve the efficiency of the BIRCH algorithm by avoiding the computationally expensive final clustering phase (see A-BIRCH pseudocode).

A-BIRCH pseudocode

Input: N two-dimensional data points $\{X_i\}$, $k_{\max}$, branching factor Br, number of Monte Carlo iterations B, distance metric D.

Output: CF-tree

Begin

$k^* \leftarrow$ parallel Gap Statistic (subsample ($\{X_i\}$), $k_{\max}$, B)

labels $\leftarrow$ k-means (subsample($\{X_i\}$), k^*)

Calculate the minimum value of $D_{\min}$ and the shared radius R based on the clustered data

if ($D_{\min} < 6 \times R$) then

Warning: The result of BIRCH may be imprecise — clusters are too close

end_if

$T \leftarrow (1/4)\, D_{\min} + 0.7 \times R$
CF-tree ← tree-BIRCH ($\{X_i\}$, *Br*, distance metric *D*, *T*).
End.

Ada-DBSCAN Algorithm

Adaptive Density-based Spatial Clustering of Applications with Noise (Ada-DBSCAN) addresses the issue of linear connections in DBSCAN and other density-based clustering algorithms, while also optimizing the parameter settings and performance of DBSCAN when applied to large datasets. Ada-DBSCAN achieves this by first using the data block splitter to divide the data points into a collection of data blocks in a top-down approach, followed by performing local clustering within each data block. To obtain the final clustering results, Ada-DBSCAN employs a bottom-up global clustering procedure, where the data block merger plays a critical sub-component role. The Ada-DBSCAN method is comprised of four primary modules, as demonstrated in the Ada-DBSCAN pseudocode: (1) data block splitter, (2) local clustering, (3) global clustering, and (4) data block merger. For further information on each proposed module in Ada-DBSCAN, refer to [20].

Ada-DBSCAN pseudocode
Input: a dataset *D*
Output: a set of clusters *C*

Step 1. Hierarchically divide the dataset *D* into a set of data blocks that ensure uniform distribution of the data points within each block.
Step 2. Group the data points into blocks based on the concept of uniform data distribution.
Step 3. Apply local density-based clustering to each block.
Step 4. Merge the resulting local clusters to obtain the final clustering results. This is also known as the global clustering process.
Step 5. Merging data block (used during global clustering for merging each two data blocks)

Smoothed self-organizing map (S-SOM)

The SOM operates as a network comprising *P* neurons arranged in a one- or two-dimensional layout. This network is designed to map a collection of *I* input vectors, each possessing a dimensionality of *J*. Every neuron within the SOM possesses a scalar vector r_p, representing its location on the map. Additionally, each neuron is equipped with an initial *J*-dimensional codebook denoted as μ_p which aligns in size with the input vectors. The process of SOM learning occurs across *S* phases during which a randomly selected input vector $\xi_i(s)$ is compared to the codebooks using a designated metric. The neuron that emerges as the winner labeled as $c = c(i)$ or the best matching unit *bmu* is chosen from the i^{th} input vector. This selection is based on the proximity of the neuron weight vector μ_p to the input vector $\xi_i(s)$.

The S-SOM [17] is an improved version of the SOM that enhances input density mapping, vector quantization, and clustering properties in the presence of outliers. The S-SOM achieves this by upgrading the learning rule to smooth the representation of outlying input vectors onto the map. This is done by taking into account the additional exponential distance between the input vector and its closest codebook during the SOM learning rule. The smoothing factor is then obtained for all neurons in the neighborhood of the best matching unit, using the additional exponential distance $\underline{d}_{\exp}(x_i, x_l)$. This approach makes the S-SOM more robust in the presence of outliers.

$$\underline{d}_{\exp}(x_i, x_l) = 1 - (1 - \exp\{-\beta \|x_i - x_l\|^2\})^{1/2},$$

where $\beta = \left(\sum_{i=1}^{l} \|x_i, \underline{x}_q\|^2 / l \right)^{-1}$ is calculated as the inverse of the overall variance of the data available.

The upgraded learning rule of S-SOM is:

$$\mu_p(s+1) = \begin{cases} \tilde{\alpha}(s) h_{p,i}(s) \xi_i(s) + [1 - \tilde{\alpha}(s) h_{p,i}(s)] \mu_p(s) & if\ p \in N_c \\ \mu_p(s) & otherwise \end{cases}, \quad (5)$$

where $\xi_i(s)$ is the i^{th} input vector and the smoothed learning rate $\tilde{\alpha}(s) = \alpha(s)(\underline{d}_{\exp}(\mu_c(s), \xi_i(s)))$ satisfies $0 < \tilde{\alpha}(s) < 1$ since 1 restricts the exponential distance. For further details about this clustering method, please refer to [18]. S-SOM pseudocode illustrates the detailed steps of S-SOM approach.

S-SOM pseudocode
Step 0. Fix the topology of the map to one or two dimensions (linear) (rectangular), the number of neurons *P* (map size), the neighborhood function $h_{p,i}(s)$, the learning rate $\tilde{\alpha}(s)$ and the number of iterations limit (*S*). Randomly generate the weights $\mu_p(0)$, $p = 1, \ldots, P$.
Step 1. Choose an input vector that will update the map at step *s*, $\xi_i(s)$. Using the euclidean distance, determine *c*, the nearest neuron to $\xi_i(s)$. Update the weights μ_p using equation (5).
Step 2. Update $\tilde{\alpha}(s)$, and $h_{p,i}(s)$.
Step 3. If the number of iteration $s = S$ stops the algorithm, otherwise go to **Step 1**.

NRI Algorithm

The NRI algorithm [27] is a grid-based clustering technique that improves upon traditional approaches. The algorithm partitions space into grids, labeling each as 'valid' or 'invalid' based on whether it contains a minimum number of parameter points. The continuous 'valid' grids are then merged into a single cluster. The algorithm also examines adjacent grids to determine whether they should be grouped into the same cluster or classified as a new one. If the neighbor has multiple clusters, they are merged into a single cluster. While the NRI algorithm follows a traditional grid-based clustering strategy, it includes a simple yet unique extension scheme. To better understand the NRI algorithm steps, refer to its pseudocode.

NRI pseudocode
Step 0. Enter datasets and initialize all arguments.
Step 1. Utilize the DataSize parameter to partition the data space into grid-like structures. ||grids||=Grids*Grids.

Step 2. Find all the grids sequentially.
Step 3. Grids are categorized as "valid" or "invalid" depending on whether they contain at least the specified minimum number of parameter points.
Step 4. Grids that are continuously "valid" are grouped into one cluster.
Step 5. If adjacent grids are successfully clustered together, they are classified as belonging to the same cluster. If not, they may be designated as a new cluster.
Step 6. If there are multiple clusters in the neighborhood, merge them into a single cluster.

Uk-means Algorithm

Uk-means [10] is a new unsupervised classification version for the *k*-means process that requires no initializations and the optimum number of clusters can also be selected simultaneously using the entropy concept. Let $X = \{x_1, \ldots, x_n\}$ be a dataset in a *d*-dimensional Euclidian space $\mathbb{R}^d$. Let $A = \{a_1, \ldots, a_c\}$ be the *c* cluster. Let $Z = [z_{ik}]_{n\times c}$, where z_{ik} is a binary variable (i.e $z_{ik} \in \{0, 1\}$) showing if x_i relates to k^{th} cluster, $k = 1, \ldots, c$. The unsupervised *k*-means algorithm (Uk-means) suggest a modified objective function as seen below:

$$J_{UKM_2}(Z, A, \alpha) = \sum_{i=1}^{n}\sum_{k=1}^{c} z_{ik}\|x_i - a_k\|^2 - \beta_n \sum_{k=1}^{c} \alpha_k \mathrm{In}\alpha_k - \\ - \gamma \sum_{i=1}^{n}\sum_{k=1}^{c} z_{ik}\mathrm{In}\alpha_k,$$

where:

$$\gamma^{(t)} = e^{-c^{(t)}/250}. \quad (6)$$

And

$$\beta^{(t+1)} = \left(\frac{\sum_{k=1}^{c} \exp(-\eta n|\alpha_k^{(t+1)} - \alpha_k^{(t)}|)}{c}, \frac{1 - \max\limits_{1\le k\le c}\left(\frac{1}{n}\sum_{i=1}^{n} z_{ik}\right)}{-\max\limits_{1\le k\le c}\alpha_k^{(t)}\left(\sum_{k=1}^{c} In\alpha_k^{(t)}\right)} \right). \quad (7)$$

The variable η controls cluster center adjustments based on α_k updates, with larger values indicating more significant changes. γ regulates entropy impact via a computed coefficient, reducing as iterations increase, thus modulating the influence of entropy on the algorithm objective function.

The cluster center α_k updating equation is as follows:

$$\alpha_k = \sum_{i=1}^{n} z_{ik}x_{ij} \Big/ \sum_{i=1}^{n} z_{ik}.$$

The updating equation for the variables membership z_{ik} is as follows:

$$z_{ik} = \begin{cases} 1 & if \|x_i - a_k\|^2 - \gamma \mathrm{In}\alpha_k \\ \|x_i - a_k\|^2 - \gamma \mathrm{In}\alpha_k \, 0 & otherwise \end{cases}. \quad (8)$$

The cluster number is updated automatically as described below:

$$c^{(t+1)} = c^{(t)} - \left| \left\{ \alpha_k^{(t+1)} < \frac{1}{n}, k = 1, \ldots, c^{(t)} \right\} \right|, \quad (9)$$

where:

$$\alpha_k^{(t+1)} = \sum_{i=1}^{n} z_{ik}x_{ij} \sum_{i=1}^{n} z_{ik}. \quad (10)$$

Uk-means pseudocode

Step 0. Fix $\sum > 0$. $c(0) = n$, $\alpha_k^{(0)} = 1/n$, $a_k^{(0)} = x_i$, $\gamma(0) = \beta(0) = 1$, Set $t = 0$.
Step 1. Calculate $z_{ik}^{(t+1)}$ employing $a_k^{(t)}$, $\alpha_k^{(t)}$, $\gamma^{(t)}$, $\beta^{(t)}$ using equation (9).
Step 2. Calculate $\gamma^{(t+1)}$ using equation (6).
Step 3. Update $\alpha_k^{(t+1)}$ with $\mu_{ik}^{(t+1)}$ and $\alpha_k^{(t)}$ using (10).
Step 4. Calculate $\beta^{(t+1)}$ with $\alpha^{(t+1)}$ and $\alpha^{(t)}$ using equation (7).
Step 5. Update $c^{(t+1)}$ to $c^{(t)}$ by discard those clusters with $\alpha^{(t+1)} \le 1/n$ and adjust $\alpha^{(t+1)}$ and $\mu_{ik}^{(t+1)}$.
IF $t \ge 60$ and $c^{(t-60)} - c^{(t)} = 0$. THEN let $\beta^{(t+1)} = 0$.
Step 6. Update $a_k^{(t+1)}$ with $c^{(t+1)}$ and $z_{ik}^{(t+1)}$ by equation (8).
Step 7. Compare $a_k^{(t+1)}$ and $a_k^{(t)}$.
IF $\|a_k^{(t+1)} - a_k^{(t)}\| < \varepsilon$. THEN Stop.
ELSE $t = t + 1$ and return to **Step 2**.

The variable *c* signifies the current count of clusters during the algorithm's iteration. *n* represents the total number of data points within the dataset. *k* stands as the cluster index, taking values from 1 to *c*. The term x_i denotes a specific data point within the dataset. *t* symbolizes the current iteration or step of the algorithm. Lastly, ε is a small threshold value that plays a role in determining the stopping condition of the algorithm.

Experimental Evaluation

In Types of Clustering Algorithms, we analyzed the practical aspects of well-known algorithms used for clustering big data. In this section, we aim to evaluate their performance on a clustering benchmark dataset to provide empirical evidence. For this evaluation, we have selected the most prominent and modern algorithms from each category, including A-BIRCH [8], Ada-DBSCAN [20], S-SOM [18], NRI [27], and Uk-means [10].

Experimental environments

We employed the MATLAB application as the chosen tool to generate clusters for all experiments. The machine used for these tests possesses the following technical specifications: an AMD Ryzen 9 processor, 32 GB of memory, and a 1 TB hard disk, running on Windows 10. To ensure reliable results, each experiment was conducted 10 times, and the average values from the experimental findings were utilized.

Experimental datasets

We utilized six popular datasets, sourced from the OpenML [36] and UCI [37] Machine Learning Repository, to investigate the performance of different clustering techniques. The datasets include Baseball, Anuran Calls, Japanese Vowel, Digits, Codon usage, and Fashion MNIST. For effective evaluation, all variables/attributes from these datasets were used simultaneously. Table 3 provides key details of the five real datasets utilized in our study, and for further information about the datasets, please refer to [36, 37].

Results and discussions

Firstly, in this paragraph, we compare the clustering results using well-known internal evaluation indices (Fig. 2). Subsequently, we evaluate the selected clustering algorithms in terms of their runtime performance.

Table 3. Summary of Datasets

Dataset	Dataset size	# dimensions	# clusters
Baseball	1340	18	3
Anuran Calls	7195	22	4
Japanese Vowel	9960	15	9
Digits	10,992	16	10
Codon usage	13,028	69	3
Fashion MNIST	70,000	784	10

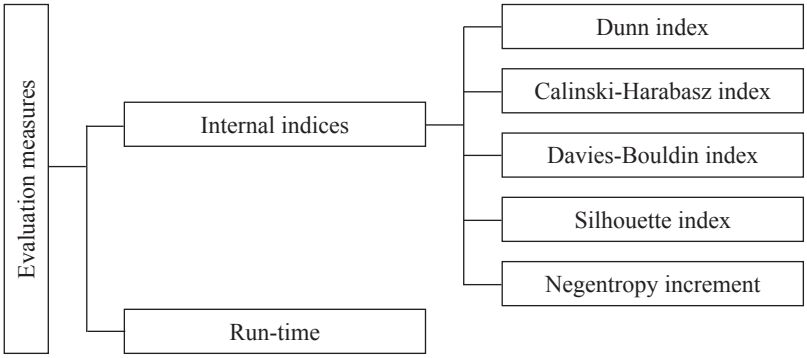

Fig. 2. Evaluation measures used in the current comparison

Table 4 displays the results of the D index for different clustering algorithms across various datasets. Analyzing the results, it can be observed that Ada-DBSCAN consistently achieved higher scores compared to the other algorithms across most datasets. For example, in the "Baseball" dataset, Ada-DBSCAN obtained a D score of 0.623, while the other algorithms ranged from 0.394 to 0.597. This suggests that Ada-DBSCAN is better at creating well-separated clusters based on the Dunn metric. Similarly, in the "Digits" dataset, Ada-DBSCAN obtained the highest D score of 0.741, indicating its ability to generate distinct and well-separated clusters. The other algorithms scored lower ranging from 0.471 to 0.643. On the other hand, NRI consistently achieved lower D scores compared to the other algorithms across multiple datasets. For instance, in the "Fashion MNIST" dataset, NRI scored 0.449 while the other algorithms ranged from 0.322 to 0.439. This suggests that NRI may struggle in creating highly separated clusters according to the Dunn metric. Overall, the results indicate that Ada-DBSCAN shows better performance in terms of creating well-separated clusters based on the D metric, while NRI performs relatively poorer. However, it's important to consider additional evaluation metrics and conduct further analysis to gain a comprehensive understanding of algorithm performance.

The comparison of candidate clustering algorithms based on the CH index in Table 5 reveals distinct patterns. The S-SOM algorithm consistently achieved the highest scores across most datasets, such as scoring 25.634 in the "Japanese Vowel" dataset compared to the range of 7.395 to 17.342 for other algorithms. This highlights S-SOM ability to generate clusters with superior inter-cluster separation and reduced intra-cluster dispersion. Ada-DBSCAN also performed well in several datasets, particularly achieving a score of 22.248 in the "Codon usage" dataset compared to the range of 13.445 to 31.624 for other algorithms. This showcases Ada-DBSCAN effectiveness in producing clusters with significant separation and compactness. On the other hand, the NRI algorithm consistently obtained lower CH scores across most datasets, scoring 1.651

Table 4. Candidate algorithms compared in terms of D index

Datasets	Clustering Algorithms				
	Ada-DBSCAN	NRI	Uk-means	S-SOM	A-BIRCH
Baseball	0.623	0.597	0.394	0.527	0.502
Anuran Calls	0.389	0.351	0.384	0.411	0.364
Japanese Vowel	0.617	0.521	0.459	0.582	0.510
Digits	0.741	0.588	0.471	0.643	0.527
Codon usage	0.556	0.463	0.394	0.571	0.514
Fashion MNIST	0.412	0.449	0.322	0.439	0.401
Average	0.556	0.494	0.404	0.528	0.469

Table 5. Candidate algorithms compared in terms of CH index

Datasets	Clustering Algorithms				
	Ada-DBSCAN	NRI	Uk-means	S-SOM	A-BIRCH
Baseball	18.364	9.328	17.366	21.450	13.624
Anuran Calls	1.260	1.651	2.514	2.315	1.110
Japanese Vowel	16.927	8.442	17.342	25.634	7.395
Digits	12.388	5.327	8.369	18.941	4.521
Codon usage	22.248	13.445	19.301	31.624	16.692
Fashion MNIST	17.392	8.661	12.847	19.214	11.363
Average	14.763	7.809	12.956	19.863	9.117

in the "Anuran Calls" dataset compared to the range of 1.110 to 2.514 for other algorithms indicating challenges in generating well-separated and compact clusters. Overall, S-SOM and Ada-DBSCAN demonstrated strong performance based on the CH index, while NRI exhibited relatively lower performance in terms of clustering quality.

Table 6 compares candidate clustering algorithms based on the DB index. The results reveal that Ada-DBSCAN achieved the lowest DB scores in the Digits (0.835) and Anuran Calls (4.985) datasets, indicating superior cluster quality. However, all algorithms struggled with the Fashion MNIST dataset which had the highest DB scores. On average, Ada-DBSCAN and NRI obtained lower DB scores (3.374 and 4.992, respectively) suggesting better clustering performance. In contrast, Uk-means and A-BIRCH had higher DB scores (5.876 and 5.402, respectively) indicating room for improvement. Consideration of dataset characteristics is crucial for selecting the most suitable algorithm for optimal clustering outcomes.

Table 7 presents the results of the candidate algorithms compared using the Sil index for various datasets. The scores indicate the clustering quality with higher values indicating better cluster cohesion and separation. Among the algorithms, Ada-DBSCAN consistently obtained high scores across most datasets, ranging from 0.556 to 0.741. NRI generally had lower scores, ranging from 0.351 to 0.597, suggesting suboptimal cluster formation. Uk-means, S-SOM, and A-BIRCH achieved moderate scores across the datasets. The average Sil index values for the algorithms were approximately 0.553 for Ada-DBSCAN, 0.494 for NRI, 0.413 for Uk-means, 0.529 for S-SOM, and 0.457 for A-BIRCH. These results indicate that Ada-DBSCAN and S-SOM exhibited relatively better clustering performance compared to NRI, Uk-means, and A-BIRCH.

Table 6. Candidate algorithms compared in terms of DB index

Datasets	Clustering Algorithms				
	Ada-DBSCAN	NRI	Uk-means	S-SOM	A-BIRCH
Baseball	2.671	3.981	5.218	3.312	4.356
Anuran Calls	4.985	6.517	7.324	4.327	6.558
Japanese Vowel	1.364	3.951	5.327	3.248	4.582
Digits	0.835	3.827	4.637	2.214	4.139
Codon usage	3.976	4.296	4.397	3.384	5.327
Fashion MNIST	6.415	7.384	8.358	6.971	7.452
Average	3.374	4.992	5.876	3.909	5.402

Table 7. Candidate algorithms compared in terms of Sil index

Datasets	Clustering Algorithms				
	Ada-DBSCAN	NRI	Uk-means	S-SOM	A-BIRCH
Baseball	0.623	0.597	0.394	0.527	0.502
Anuran Calls	0.389	0.351	0.384	0.411	0.364
Japanese Vowel	0.617	0.521	0.459	0.582	0.510
Digits	0.741	0.588	0.471	0.643	0.527
Codon usage	0.556	0.463	0.394	0.571	0.415
Fashion MNIST	0.412	0.449	0.322	0.439	0.401
Average	0.553	0.494	0.413	0.529	0.457

The analysis of the candidate clustering algorithms based on the NI in Table 8 reveals distinct trends: across most datasets, the S-SOM algorithm consistently achieved the highest NI scores, ranging from 0.582 to 0.521 for the other algorithms. This highlights S-SOM exceptional ability to reduce entropy and significantly improve the quality of clustering results. Similarly, Ada-DBSCAN demonstrated notable performance, obtaining relatively high NI scores in multiple datasets ranging from 0.741 to 0.471 for the other algorithms. This signifies Ada-DBSCAN effectiveness in reducing entropy and enhancing the purity of clusters. Conversely, the NRI algorithm consistently obtained lower NI scores compared to the other algorithms with a range of 0.457 to 0.623 for the alternative methods. This indicates challenges in achieving substantial NI and improving clustering performance using NRI. In summary, the results clearly indicate that S-SOM and Ada-DBSCAN exhibit impressive performance in terms of increasing Negentropy and improving clustering quality, while NRI lags behind in terms of NI.

The analysis of the runtime results presented in Table 9 comparing candidate clustering algorithms and revealing the following insights: S-SOM consistently recorded the highest runtime across all datasets with durations ranging from 14.854 to 656.362. This suggests that S-SOM requires more computational time to perform clustering compared to other algorithms. Uk-means and A-BIRCH exhibited relatively lower runtimes compared to S-SOM, with durations ranging from 0.964 to 82.145 for Uk-means and from 0.715 to 65.238 for A-BIRCH. This indicates that these algorithms are generally faster in executing the clustering process. NRI and Ada-DBSCAN demonstrated the shortest runtimes across all datasets, with durations ranging from 0.031 to 9.627 for NRI and from 0.015 to 2.130 for Ada-DBSCAN. These algorithms exhibit superior efficiency in terms of runtime requiring significantly less computational time compared to the other algorithms. Overall, the results suggest that S-SOM is the slowest algorithm in terms of runtime, while NRI and Ada-DBSCAN are the fastest.

The analysis of the different clustering algorithms based on multiple evaluation metrics provides valuable insights into their performance. Ada-DBSCAN consistently outperformed other algorithms in terms of the D index, indicating its ability to create well-separated clusters. Similarly, Ada-DBSCAN and S-SOM showed strong performance in terms of the CH index indicating their effectiveness in producing clusters with high inter-cluster separation and compactness. Ada-DBSCAN also demonstrated superior performance based on the DB index, achieving lower scores and indicating better cluster quality. Moreover, Ada-DBSCAN consistently obtained high scores in the Sil index, indicating its ability to generate cohesive and well-separated clusters. In terms of the NI, S-SOM and Ada-DBSCAN stood out by achieving the highest scores and improving clustering quality. Finally, NRI and Ada-DBSCAN exhibited the shortest runtimes making them more efficient in terms of computational time compared to other algorithms. These findings suggest that Ada-DBSCAN consistently performs well across multiple evaluation metrics, highlighting its potential as a strong clustering algorithm. However, further analysis and consideration of additional factors are necessary for a comprehensive understanding and selection of the most suitable algorithm for specific clustering tasks.

Table 8. Candidate algorithms compared in terms of Negentropy increment

Datasets	Clustering Algorithms				
	Ada-DBSCAN	NRI	Uk-means	S-SOM	A-BIRCH
Baseball	0.623	0.457	0.394	0.527	0.502
Anuran Calls	0.389	0.351	0.384	0.411	0.364
Japanese Vowel	0.617	0.521	0.459	0.582	0.510
Digits	0.741	0.588	0.471	0.643	0.527
Codon usage	0.556	0.463	0.394	0.571	0.415
Fashion MNIST	0.412	0.449	0.322	0.439	0.401
Average	0.553	0.494	0.413	0.529	0.457

Table 9. Candidate algorithms compared in terms of runtime

Datasets	Clustering Algorithms				
	S-SOM	Uk-means	A-BIRCH	NRI	Ada-DBSCAN
Baseball	14.854	0.964	0.715	0.031	0.015
Anuran Calls	42.265	3.562	2.149	0.781	0.520
Japanese Vowel	242.347	21.354	15.378	3.154	1.214
Digits	313.326	29.348	23.641	4.261	1.541
Codon usage	335.197	38.241	29.286	4.157	1.631
Fashion MNIST	656.362	82.145	65.238	9.627	2.130
Average	267.391	29.269	22.734	3.668	1.175

Conclusion

This paper provides a comprehensive evaluation of clustering algorithms for Big Data. The study compares various methods and assesses their performance on large datasets using validity metrics and computing time. The findings assist researchers and practitioners in selecting appropriate clustering techniques for their specific needs. Future work could focus on developing advanced algorithms exploring hybrid approaches, addressing scalability and interpretability challenges. This research contributes to the field and lays the groundwork for further advancements in clustering for Big Data analytics.

References

1. Hinneburg A., Keim D.A. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. *Proc. of the 25th International Conference on Very Large Data Bases*, 1999, pp. 506–517.
2. Hinneburg A., Keim D.A. An efficient approach to clustering in large multimedia databases with noise. *Proc. of the 4th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1998, pp. 58–65.
3. Guha S., Rastogi R., Shim K. ROCK: a robust clustering algorithm for categorical attributes. *Proc. of the 15th International Conference on Data Engineering*, 1999, pp. 512–521. https://doi.org/10.1109/icde.1999.754967
4. Gennari J.H., Langley P., Fisher D. Models of incremental concept formation. *Artificial Intelligence*, 1989, vol. 40, no. 1-3, pp. 11–61. https://doi.org/10.1016/0004-3702(89)90046-5
5. Kaufman L., Rousseeuw P.J. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, 1990, 342 p.
6. Arbelaitz O., Gurrutxaga I., Muguerza J., Pérez J.M., Perona I. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 2013, vol. 46, no. 1, pp. 243–256. https://doi.org/10.1016/j.patcog.2012.07.021
7. Xu D., Tian Y. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2015, vol. 2, no. 2, pp. 165–193. https://doi.org/10.1007/s40745-015-0040-1
8. Sinaga K.P., Yang M. Unsupervised k-means clustering algorithm. *IEEE Access*, 2020, vol. 8, pp. 80716–80727. https://doi.org/10.1109/ACCESS.2020.2988796
9. Shili H., Romdhane L.B. IF-CLARANS: intuitionistic fuzzy algorithm for big data clustering. *Communications in Computer and Information Science*, 2018, vol. 854, pp. 39–50. https://doi.org/10.1007/978-3-319-91476-3_4
10. Karypis G., Han E.H., Kumar V. Chameleon: hierarchical clustering using dynamic modeling. *Computer*, 1999, vol. 32, no. 8, pp. 68–75. https://doi.org/10.1109/2.781637
11. Davies D.L., Bouldin D.W. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1979, vol. PAMI-1, no. 2, pp. 224–227. https://doi.org/10.1109/TPAMI.1979.4766909
12. Ankerst M., Breunig M., Kriegel H., Sander J. OPTICS: Ordering points to identify the clustering structure. *ACM SIGMOD Record*, 1999, vol. 28, no. 2, pp. 49–60. https://doi.org/10.1145/304181.304187
13. Cai Z., Wang J., He K. Adaptive density-based spatial clustering for massive data analysis. *IEEE Access*, 2020, vol. 8, pp. 23346–23358. https://doi.org/10.1109/ACCESS.2020.2969440
14. Wang W., Yang J., Muntz R. STING: a statistical information grid approach to spatial data mining. *Proc. of the 23th International Conference on Very Large Data Bases*, 1997, pp. 186–195.
15. Vanschoren J., van Rijn J.N., Bischl B., Torgo L. OpenML: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 2013, vol. 15, no. 2, pp. 49–60. https://doi.org/10.1145/2641190.2641198
16. Goil S., Nagesh H., Choudhary A. *MAFIA: Efficient and scalable subspace clustering for very large data sets*. Technical Report CPDC-TR-9906-010, 1999.
17. Agrawal R., Gehrke J., Gunopulos D., Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications. *ACM SIGMOD Record*, 1998, vol. 27, no. 2, pp. 94–105. https://doi.org/10.1145/276305.276314
18. Dempster P., Laird N.M., Rubin D.B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 1977, vol. 39, no. 1, pp. 1–38. https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

Литература

1. Hinneburg A., Keim D.A. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering // Proc. of the 25th International Conference on Very Large Data Bases. 1999. P. 506–517.
2. Hinneburg A., Keim D.A. An efficient approach to clustering in large multimedia databases with noise // Proc. of the 4th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 1998. P. 58–65.
3. Guha S., Rastogi R., Shim K. ROCK: a robust clustering algorithm for categorical attributes // Proc. of the 15th International Conference on Data Engineering. 1999. P. 512–521. https://doi.org/10.1109/icde.1999.754967
4. Gennari J.H., Langley P., Fisher D. Models of incremental concept formation // Artificial Intelligence. 1989. V. 40. N 1-3. P. 11–61. https://doi.org/10.1016/0004-3702(89)90046-5
5. Kaufman L., Rousseeuw P.J. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990. 342 p.
6. Arbelaitz O., Gurrutxaga I., Muguerza J., Pérez J.M., Perona I. An extensive comparative study of cluster validity indices // Pattern Recognition. 2013. V. 46. N 1. P. 243–256. https://doi.org/10.1016/j.patcog.2012.07.021
7. Xu D., Tian Y. A comprehensive survey of clustering algorithms // Annals of Data Science. 2015. V. 2. N 2. P. 165–193. https://doi.org/10.1007/s40745-015-0040-1
8. Sinaga K.P., Yang M. Unsupervised k-means clustering algorithm // IEEE Access. 2020. V. 8. P. 80716–80727. https://doi.org/10.1109/ACCESS.2020.2988796
9. Shili H., Romdhane L.B. IF-CLARANS: intuitionistic fuzzy algorithm for big data clustering // Communications in Computer and Information Science. 2018. V. 854. P. 39–50. https://doi.org/10.1007/978-3-319-91476-3_4
10. Karypis G., Han E.H., Kumar V. Chameleon: hierarchical clustering using dynamic modeling // Computer. 1999. V. 32. N 8. P. 68–75. https://doi.org/10.1109/2.781637
11. Davies D.L., Bouldin D.W. A cluster separation measure // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1979. V. PAMI-1. N 2. P. 224–227. https://doi.org/10.1109/TPAMI.1979.4766909
12. Ankerst M., Breunig M., Kriegel H., Sander J. OPTICS: Ordering points to identify the clustering structure // ACM SIGMOD Record. 1999. V. 28. N 2. P. 49–60. https://doi.org/10.1145/304181.304187
13. Cai Z., Wang J., He K. Adaptive density-based spatial clustering for massive data analysis // IEEE Access. 2020. V. 8. P. 23346–23358. https://doi.org/10.1109/ACCESS.2020.2969440
14. Wang W., Yang J., Muntz R. STING: a statistical information grid approach to spatial data mining // Proc. of the 23th International Conference on Very Large Data Bases. 1997. P. 186–195.
15. Vanschoren J., van Rijn J.N., Bischl B., Torgo L. OpenML: Networked science in machine learning // ACM SIGKDD Explorations Newsletter. 2013. V. 15. N 2. P. 49–60. https://doi.org/10.1145/2641190.2641198
16. Goil S., Nagesh H., Choudhary A. MAFIA: Efficient and scalable subspace clustering for very large data sets: Technical Report CPDC-TR-9906-010. 1999.
17. Agrawal R., Gehrke J., Gunopulos D., Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications // ACM SIGMOD Record. 1998. V. 27. N 2. P. 94–105. https://doi.org/10.1145/276305.276314
18. Dempster P., Laird N.M., Rubin D.B. Maximum likelihood from incomplete data via the em algorithm // Journal of the Royal Statistical Society: Series B (Methodological). 1977. V. 39. N 1. P. 1–38. https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

19. Calinski T., Harabasz J. A dendrite method for cluster analysis. *Communications in Statistics — Simulation and Computation*, 1974, vol. 3, no. 1, pp. 1–27. https://doi.org/10.1080/03610917408548446
20. Dunn J. Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 1974, vol. 4, no. 1, pp. 95–104. https://doi.org/10.1080/01969727408546059
21. Canbay Y., Sağıroğlu S. Big data anonymization with spark. *Proc. of the 2017 International Conference on Computer Science and Engineering (UBMK)*, 2017, pp. 833–838. https://doi.org/10.1109/UBMK.2017.8093543
22. Lorbeer B., Kosareva A., Deva B., Softić D., Ruppel P., Küpper A. Variations on the Clustering Algorithm BIRCH. *Big Data Research*, 2018, vol. 11, pp. 44–53. https://doi.org/10.1016/j.bdr.2017.09.002
23. Tsai C., Huang S. An effective and efficient grid-based data clustering algorithm using intuitive neighbor relationship for data mining. *Proc. of the 2015 International Conference on Machine Learning and Cybernetics (ICMLC)*, 2015, pp. 478–483. https://doi.org/10.1109/ICMLC.2015.7340603
24. Kailing K., Kriegel H., Kröger P. Density-connected subspace clustering for high-dimensional data. *Proc. of the 2014 SIAM International Conference on Data Mining*, 2004, pp. 246–257. https://doi.org/10.1137/1.9781611972740.23
25. Kohonen T. The self-organizing map. *Proceedings of the IEEE*, 1990, vol. 78, no. 9, pp. 1464–1480. https://doi.org/10.1109/5.58325
26. Bandyopadhyay S., Saha S. A point symmetry-based clustering technique for automatic evolution of clusters. *IEEE Transactions on Knowledge and Data Engineering*, 2008, vol. 20, no. 11, pp. 1441–1457. https://doi.org/10.1109/tkde.2008.79
27. Guha S., Rastogi R., Shim K. CURE: an efficient clustering algorithm for large databases. *ACM SIGMOD Record*, 1998, vol. 27, no. 2, pp. 73–84. https://doi.org/10.1145/276305.276312
28. Mahmud M.S., Huang J.Z., Salloum S., Emara T.Z., Sadatdiynov K. A survey of data partitioning and sampling methods to support big data analysis. *Big Data Mining and Analytics*, 2020, vol. 3, no. 2, pp. 85–101. https://doi.org/10.26599/BDMA.2019.9020015
29. Djouzi K., Beghdad-Bey K. A review of clustering algorithms for big data. *Proc. of the International Conference on Networking and Advanced Systems (ICNAS)*, 2019, pp. 1–6. https://doi.org/10.1109/ICNAS.2019.8807822
30. Fahad A., Alshatri N., Tari Z., Alamri A., Khalil I., Zomaya A.Y., Foufou S., Bouras A. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, 2014, vol. 2, no. 3, pp. 267–279. https://doi.org/10.1109/TETC.2014.2330519
31. D'Urso P., De Giovanni L., Massari R. Smoothed self-organizing map for robust clustering. *Information Sciences*, 2020, vol. 512, pp. 381–401. https://doi.org/10.1016/j.ins.2019.06.038
32. Asuncion A., Newman D.J. *UCI Machine Learning Repository*. University of California, School of Information and Computer Science, Irvine, CA, 2007.
33. Zhang T., Ramakrishnan R., Livny M. BIRCH: a new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1997, vol. 1, no. 2, pp. 141–182. https://doi.org/10.1023/A:1009783824328
34. MacQueen J. Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley Symposium Mathematical Statist. Probability. V. 1*, 1967, pp. 281–297.
35. Ng R.T., Han J. Efficient and effective clustering methods for spatial data mining. *VLDB '94: Proc. of the 20th International Conference on Very Large Data Bases*, 1994, pp. 144–144.
36. Fisher D.H. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 1987, vol. 2, no. 2, pp. 139–172. https://doi.org/10.1007/bf00114265
37. Ester M., Kriegel H.P., Sander J., Xu X. A density-based algorithm for discovering clusters in large spatial data bases with noise. *KDD'96: Proc. of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231.

19. Calinski T., Harabasz J. A dendrite method for cluster analysis // Communications in Statistics — Simulation and Computation. 1974. V. 3. N 1. P. 1–27. https://doi.org/10.1080/03610917408548446
20. Dunn J. Well-separated clusters and optimal fuzzy partitions // Journal of Cybernetics. 1974. V. 4. N 1. P. 95–104. https://doi.org/10.1080/01969727408546059
21. Canbay Y., Sağıroğlu S. Big data anonymization with spark // Proc. of the 2017 International Conference on Computer Science and Engineering (UBMK). 2017. P. 833–838. https://doi.org/10.1109/UBMK.2017.8093543
22. Lorbeer B., Kosareva A., Deva B., Softić D., Ruppel P., Küpper A. Variations on the Clustering Algorithm BIRCH // Big Data Research. 2018. V. 11. P. 44–53. https://doi.org/10.1016/j.bdr.2017.09.002
23. Tsai C., Huang S. An effective and efficient grid-based data clustering algorithm using intuitive neighbor relationship for data mining // Proc. of the 2015 International Conference on Machine Learning and Cybernetics (ICMLC). 2015. P. 478–483. https://doi.org/10.1109/ICMLC.2015.7340603
24. Kailing K., Kriegel H., Kröger P. Density-connected subspace clustering for high-dimensional data // Proc. of the 2014 SIAM International Conference on Data Mining. 2004. P. 246–257. https://doi.org/10.1137/1.9781611972740.23
25. Kohonen T. The self-organizing map // Proceedings of the IEEE. 1990. V. 78. N 9. P. 1464–1480. https://doi.org/10.1109/5.58325
26. Bandyopadhyay S., Saha S. A point symmetry-based clustering technique for automatic evolution of clusters // IEEE Transactions on Knowledge and Data Engineering. 2008. V. 20. N 11. P. 1441–1457. https://doi.org/10.1109/tkde.2008.79
27. Guha S., Rastogi R., Shim K. CURE: an efficient clustering algorithm for large databases // ACM SIGMOD Record. 1998. V. 27. N 2. P. 73–84. https://doi.org/10.1145/276305.276312
28. Mahmud M.S., Huang J.Z., Salloum S., Emara T.Z., Sadatdiynov K. A survey of data partitioning and sampling methods to support big data analysis // Big Data Mining and Analytics. 2020. V. 3. N 2. P. 85–101. https://doi.org/10.26599/BDMA.2019.9020015
29. Djouzi K., Beghdad-Bey K. A review of clustering algorithms for big data // Proc. of the International Conference on Networking and Advanced Systems (ICNAS). 2019. P. 1–6. https://doi.org/10.1109/ICNAS.2019.8807822
30. Fahad A., Alshatri N., Tari Z., Alamri A., Khalil I., Zomaya A.Y., Foufou S., Bouras A. A survey of clustering algorithms for big data: Taxonomy and empirical analysis // IEEE Transactions on Emerging Topics in Computing. 2014. V. 2. N 3. P. 267–279. https://doi.org/10.1109/TETC.2014.2330519
31. D'Urso P., De Giovanni L., Massari R. Smoothed self-organizing map for robust clustering // Information Sciences. 2020. V. 512. P. 381–401. https://doi.org/10.1016/j.ins.2019.06.038
32. Asuncion A., Newman D.J. UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA, 2007.
33. Zhang T., Ramakrishnan R., Livny M. BIRCH: a new data clustering algorithm and its applications // Data Mining and Knowledge Discovery. 1997. V. 1. N 2. P. 141–182. https://doi.org/10.1023/A:1009783824328
34. MacQueen J. Some methods for classification and analysis of multivariate observations // Proceedings of the fifth Berkeley Symposium Mathematical Statist. Probability. V. 1. 1967. P. 281–297.
35. Ng R.T., Han J. Efficient and effective clustering methods for spatial data mining // VLDB '94: Proc. of the 20th International Conference on Very Large Data Bases. 1994. P. 144–144.
36. Fisher D.H. Knowledge acquisition via incremental conceptual clustering // Machine Learning. 1987. V. 2. N 2. P. 139–172. https://doi.org/10.1007/bf00114265
37. Ester M., Kriegel H.P., Sander J., Xu X. A density-based algorithm for discovering clusters in large spatial data bases with noise // KDD'96: Proc. of the Second International Conference on Knowledge Discovery and Data Mining. 1996. P. 226–231.

Author

Hechmi Shili — PhD, Assistant Professor, University of Tabuk, Tabuk, 47311, Arabie Saoudite, SC 26027243600, https://orcid.org/0000-0001-5989-4457, asuhaili@ut.edu.sa

Автор

Шили Хечми — PhD, доцент, Университет Табука, Табук, 47311, Саудовская Аравия, SC 26027243600, https://orcid.org/0000-0001-5989-4457, asuhaili@ut.edu.sa

Received 18.01.2023
Approved after reviewing 13.07.2023
Accepted 17.09.2023

Статья поступила в редакцию 18.01.2023
Одобрена после рецензирования 13.07.2023
Принята к печати 17.09.2023