RT&A, Special Issue No. 7 (83),
Volume 20, May 2025

EXPLORING BIG DATA CLUSTERING: APPROACHES, ALGORITHMS, AND PLATFORMS

Ramiz Aliguliyev[1], Tural Badalov[1]

•

[1]Institute of Information Technology, Baku, Azerbaijan
r.aliguliyev@gmail.com, bedelov.tural@gmail.com

Abstract

Clustering as the problem of discovering natural grouping in data has gotten a lot of attention due to its wide range of applications in health care, customer segmentation, image processing & transformation, market and recommendation systems, social network analysis, etc. It is an unsupervised learning task used to discover similar objects in a large dataset without relying on any prior information and gathering them into the same group. With the rapid growth of big data as result of data sets acquired by mobile devises, cameras, various sensors and other sources has necessitated research into extracting valuable information from enormous data sets. In this paper, we looked at different big data clustering approaches in the context of general clustering methods. In addition, we discussed several similarity measures as well as key clustering challenges such as cluster tendency assessment and cluster validity.

Keywords: big data, clustering, vertical scaling platforms, GPU, FPGA, MapReduce, Apache Spark.

I. Introduction

The amount of data generated by many sources is enormous, and this process is speeding up due to the high level of technology employed for various purposes. It is needed to deal with this huge data to extract useful information that benefits both businesses and individuals. So as a data mining tool cluster analysis or clustering is used to divide the data objects into subgroups or clusters so that objects within the same cluster resemble one another but differ greatly from those in other clusters. The fundamental advantage of clustering over classification is that class label for an object is not known in advance, allowing it to be used in a variety of fields such as, biology, business intelligence, image pattern recognition, Web search and etc. In Biology, It is typical to use cluster analysis to evaluate the gene expression data. For example, capturing and analyzing thousands of data points on gene expression level of cancer cells can be used to predict disease prognosis.

Clustering can be used in business intelligence to organize a large number of consumers into groups with strong common qualities. This makes it easier to come up with corporate strategies to improve customer relationship management. In Web search, clustering is used for organizing the search results according to the keyword into groups presenting them in a clear and accessible manner. Furthermore, approaches for grouping texts into subjects have been developed.

The rest of the paper is divided into three parts. Section 2 presents related work that provides a literature review of large data clustering approaches. Section 3 discusses the clustering of big data. The conclusion is made in Section 4.

II. Related Work

In [1] author gives a comparison of some existing algorithms with different attribute reduction techniques and address the challenges in IoT Big Data Clustering.

Researchers in [2] provides an overview of different clustering algorithms and make a taxonomy of different clustering techniques and analyze their suitability for clustering Big Data.

Authors in [3] evaluate clustering methods for big data. They enlist the criterion according to the properties of big data like volume, variety and velocity. By using results of experiments on real datasets it has been selected some algorithms over others according to which extent the listed criterion is satisfied.

In [4] Authors split big data clustering into two categories, single-machine and multi-machine clustering techniques. They give the characteristics of various clustering algorithms in each category and point out the some advantages of GPU based MapReduce technique over another algorithms.

Annad Nayyar's paper [5] represents a comprehensive analysis of different clustering algorithms classifying into the following groups: Partitioning-based, Hierarchical, Density-base, Grid-based and Model –based methods. He concludes his work in the summary table mentioning the competence of the algorithms for large data clustering.

III. Big data clustering algorithms and methods

Clustering big data is a hard task to do in data mining and most of the existing algorithms cannot deal with that. So, It requires modification of some appropriate algorithms or a totally new approach to cope with the problem. Generally, it is considered single- machine based techniques and multiple-machine based techniques.

A) Single-Machine Based Techniques

Single machine techniques use the resources of a single machine, implement on comparatively small data set, and lead to a conclusion for the entire data set. The idea of reducing the size of data distinguishes Sampling based techniques form dimension-reduction techniques in this category. There are different sampling methods involving probability and non-probability sampling and there have been conducted various studies on data sampling in the context of big data [21], [22], [23].

The main advantages and disadvantages of single-machine-based clustering techniques are demonstrated in table1.

a) *Partitioning –Based Methods:* Mini-batch k-means (MBKM) [6] is an analog of K means for large data that achieves local optimum with minimal computation cost by using a randomly picked section of the whole data in each iteration.

ClusteringLargeApplications(CLARA) [7] is a variation of PAM, that was created to address PAM's drawbacks, as it employs a random subset of data rather than a complete dataset. The main problem with Clara is that it may produce incorrect clustering if one or more sampled medoids are far away from real medoids. To resolve this issue, it was introduced Clustering Large Applications based on Randomized Search(CLARANS) [8]. Instead of selecting a random subset, this algorithm picks sample of neighbors dynamically which is then specified as a parameter, to be examined for the best medoid in each iteration.

A single pass fuzzy-c-means algorithm was presented in [24] The algorithm doesn't require fitting the whole dataset in the memory. Here original dataset is divided into n equal parts and in each step only one part is loaded into the main memory and then by using Fuzzy C means clustering algorithm data is partitioned into c clusters. Then data in memory is compressed into c weighted points, and clustering is then continued with a newly loaded chunk. When the entire dataset has been processed, the procedure is finished.

b) *Hierarchical-Based Methods:* BIRCH [9] is a two-phase clustering, in the first phase it scans the entire data to build CF –tree, and then by using any known algorithm it clusters leaf nodes of CF-Tree. So It offers flexibility to be used in conjunction with other clustering methods.

CF (Clustering Feature) as a hierarchical data structure for the algorithm is a statistical summary of the data points consisting of three components : $CF = \left(N, \overrightarrow{LS}, SS\right)$. N –number of data points in the cluster, $\overrightarrow{LS}$ - is a linear sum of N data points, SS- is a square sum of N data points. This method has a linear scalability, with just one scan, it can identify a solid clustering. However, the algorithm uses radius or diameter as a hyper-parameter, that is why the clusters frequently have a spherical shape. All the mentioned algorithms above employ a single point as a cluster representative, so the clusters tend to be spherical. This problem is addressed in [10]. CURE algorithm uses multiple representative points to describe the cluster. First, it chooses data samples at random that will fit in main memory. The selected data is then divided into clusters by employing any clustering technique, most often a hierarchical approach. A few typical points for each cluster serve as an explanation. So that enables CURE to cluster of arbitrary shape.

c) Density-Based Methods: Algorithms in this category are generally not seen to be very effective in clustering large amounts of data. However, DENCLUE [11] is good at handling arbitrary shaped clusters in high dimensional dataset. This algorithm uses statistical density function to determine the influence of each data point. The total sum of influence functions applied to all data points constitute what is referred to as the data space's overall density. Density attractors, which are the local maxima of the overall density function, then identify clusters.

It has different variations. In order to improve time complexity, it was introduced DENCLUE-IM [12]. According to the experiments it has been observed that execution time of DENCLUE-IM is reduced multiple times compared to DENCLUE and its other variants.

d) Grid-Based Methods: Grid-Based Methods represent a distinctive approach to clustering, particularly well-suited for high-dimensional data and scenarios where data distribution exhibits complex shapes or patterns. Instead of directly working with individual data points, these methods partition the data space using a grid structure, and clustering is performed on these grid cells. One notable algorithm in this category is OptiGrid [13], which excels at clustering high-dimensional data with arbitrary shapes. Grid-Based Methods divide the feature space into a grid of cells. The size and granularity of the grid are crucial parameters that influence the clustering results. The grid cells serve as the basic units for clustering. Each grid cell often contains statistical information summarizing the data points within it. This information can include the mean, variance, or other relevant statistics. By summarizing the data in this way, grid-based methods reduce the dimensionality of the problem, making it more manageable for clustering algorithms. Grid-based clustering is not directly dependent on individual data points but rather on the characteristics of the grid cells. This can be advantageous in high-dimensional data scenarios where traditional clustering algorithms may struggle due to the curse of dimensionality. High-dimensional data often poses challenges for clustering algorithms, as the distance metrics become less effective in high-dimensional spaces. Grid-Based Methods can mitigate these challenges by focusing on the relationships between grid cells rather than individual data points. OptiGrid, in particular, is known for its ability to identify clusters of arbitrary shapes in high-dimensional data. This is a valuable feature, as many real-world datasets do not conform to simple geometric shapes. This algorithm can be highly scalable, making it suitable for large-scale and big data clustering tasks. The grid structure allows for efficient parallelization and distributed processing.

Table 1: *Main Advantages and Disadvantages of Single Machine Based Clustering Techniques*

Different Approaches to Big Data Clustering	**Advantages**	**Disadvantages**
Partitioning based methods	Scalability Flexibility in choosing the number of partitions Allowance for parallel processing	Sensitivity to Initial Partitioning Difficulty in handling high-dimensional data
Hierarchical-Based Methods	No prior knowledge of cluster number Interpretable results	Computational complexity Lack of support for distributed computing Difficulty in handling high-dimensional data
Density-Based Methods	Robustness to noise Flexibility in cluster shape and size	Sensitivity to parameter settings Lack of support for distributed processing computationally expensive, especially when dealing with large-scale datasets
Grid -based Methods	Scalability Parallel processing capabilities	Challenge in Grid cell size determination Grid shape constraints (assume that data distribution is uniform and etc.) Sensitivity to input order
Dimension Reduction Techniques	Making clustering algorithms more efficient and scalable for big data Improved Interpretability Handling Curse of Dimensionality	Information Loss Subjectivity in Feature Selection Increased Complexity- adds an extra step to the clustering pipeline
Vertical Scaling Platforms	Performance- fast processing Scalability- handle larger data volumes without the need for distributed systems Simplicity- more user-friendly compared to distributed systems	Limited scalability-physical limit to how much a single machine can be scaled up Cost- they require high-end hardware to support the increased resource requirements. Lack of fault tolerance

B) Dimension Reduction Techniques – One of the major problems with clustering, especially in image processing and text documentation is the high dimensionality of the dataset, commonly known as the "curse of dimensionality". In literature there have been suggested several dimension reduction techniques, basically they are divided into two types: feature selection and feature transformation.

The primary goal of feature selection is to identify and retain a subset of the most relevant and informative features from the original dataset while discarding the less important ones. This process aims to reduce the dimensionality of the data while preserving its essential characteristics, making it more suitable for clustering algorithms. It can be performed through manual selection by domain experts or through automated algorithms that evaluate the importance or relevance of each feature. For example: fast correlation based filter (FCBF) [14], fast clustering-based feature selection algorithm (FAST) [15], Markov Blanket Filter (MBF) [16]. FCBF measures feature relevance using correlation and conditional mutual information. It identifies and selects features that are highly correlated with the class labels. FAST employs a clustering technique to group similar features and selects representatives from each cluster. This reduces redundancy and retains essential information. MBF identifies features that are conditionally independent of the class label when other features are known. It aims to find a minimal set of features that predict the class label effectively. Even though both methods are utilized to lessen the number of attributes in a dataset, future selection extracts attributes without affecting them, but feature transformation generates new combinations of the original features to produce a reduced-dimensional representation of the data. These combinations are linear or nonlinear transformations of the original attributes. Principal component analysis (PCA) [17] Random projection (RP) [18] Linear Discriminant Analysis (LDA) [19] Canonical Correlation Analysis (CCA) [20] are some popular feature transformation algorithms. PCA is a widely-used linear feature transformation technique that projects the data onto a set of orthogonal axes (principal components) that capture the maximum variance. It reduces dimensionality while preserving as much variance as possible. RP is a dimensionality reduction technique that uses random projections to map high-dimensional data to lower-dimensional space while preserving pairwise distances to some extent. LDA is a supervised feature transformation technique that finds linear combinations of features that maximize the separation between classes, making it particularly useful for classification tasks. CCA is used for multivariate data analysis. It finds linear combinations of features from different datasets that maximize their correlation, which can be beneficial when dealing with data from multiple sources or domains.

C) Clustering Using Vertical Scaling Platforms- By adding more power, such as upgrading hardware on the existing system, we may increase the performance of the algorithms performed on a single machine. The most widely used vertical scale-up paradigms include multicore CPUs, Graphical Processing Units (GPU), and Field Programmable Gate Arrays (FPGA) [25].

Multicore CPUs- The goal of conventional CPU optimizations was to accelerate the serial execution of a single thread. A processor with several cores is referred to as a multicore CPU and parallel computation is accomplished using the multi-threading paradigm [26]. The task is divided into threads, each of which is executed simultaneously across many CPU cores. The primary disadvantage of CPUs is that system memory limits the amount of data that they can process. Some parallel clustering techniques based on multicore processor have been presented. For example, In [27] It is suggested modification of the FDBSCAN algorithm for multicore platforms. Authors named their algorithm M-FDBSCAN. Here the dataset is distributed equally among the cores, then FDBSCAN is applied to each subset. To obtain the end result, intermediate sub dataset pairs are merged according to ε neighborhood of the splitting line.

Graphical Processing Units- Originally GPU was designed to perform calculations at three-dimensional (3 D) graphics. The input primitives for the GPU are the vertices of the triangles that make up the three-dimensional representation of the data. Each vertex needs to be converted into pixels and shaded, then mapped onto the screen. The final image is created by combining those pixels. The same program is used by GPU to process several components (vertices) simultaneously. According to the SPMD (single-program multiple-data) programming paradigm, components are independent of one another and are unable to communicate [28]. That is why there is limited software integrated with GPUs. With the release of the CUDA framework, Nvidia opened up GPU programming to any programmers without requiring them to understand the specifics of the hardware. With this framework, authors in [29] represented implementation of Markov Clustering algorithm (MCL) on GPU. Two operations, expansion and inflation, which are sparse matrix-matrix multiplication and Markov matrix normalization, respectively, define the MCL's time complexity. The performance of the method MCL was enhanced by computing parallel tasks for both expansion and inflation procedures in this study. In [30] parallel implementation of K-means algorithm on GPU via CUDA platform is discussed. The suggested variant is compared with optimized sequential k-Means. Experiments on synthetic data showed that parallel K-Means with CUDA outperform by 4 to 43 times for large dataset.

Field Programmable Gate Arrays (FPGA)- FPGA is an integrated circuit device consisting of logic blocks, programmable routing and I/O blocks to make connections with external devices. Logic blocks perform basic computations and are used as a storage element. The programmable routing consists of wires and programmable switches that provides connection among logic blocks and I/O blocks [31]. On the programming side, hardware description language (HDL) is used to generate PFGA designs . FPGAs are usually compared with Application Specific Integrated Circuit (ASIC), where the latter is power-efficient by 12 times on average, faster by approximately 3.2 times, and requires less area by roughly 23 to 55 times [32].In spite of having these disadvantages, FPGAs are very suitable for a specific set of applications because of their flexible and adaptable nature. On FPGAs, parallelism is made possible by the programmable routing circuit design and architecture, while many applications rely on the parallel execution of the same tasks. There are several practical uses for FPGAs, including in medical electronics, video and image processing, search engine algorithms and other areas. In regard with clustering FPGA-based implementation of DBSCAN is considered in [33]. The authors of the study assert that since extended neighbourhood of point queries are independent of data and take up the majority of execution time, they can be carried out in parallel. The suggested technique is on average 32 times and 202 times quicker than established software algorithms, according to tests on two-dimensional real and synthetic datasets. Hardware implementation of KMeans on FPGA is represented in [34]. This approach considers a large number of clusters (up to 256) and is intended for color images. In order to accelerate the algorithm the author used simplified version of filtering and FEKM algorithms. The former technique is employed to attain high performance by restricting comparisons for the closet centers to 24. FEKM technique reduces the number of data points scanned during each iteration, allowing it to analyze large datasets effectively. Experiments on images of 512*512 and 640*480 display that the performance of the suggested implementation of K-means with one FPGA is more than 30 frames per second.

Clustering using vertical scaling platforms involves enhancing the computational power of a single machine to improve the performance of clustering algorithms. The choice of platform (e.g., multicore CPU, GPU, FPGA) depends on the specific clustering algorithm, dataset size, and available resources. It's essential to carefully evaluate the benefits and costs of vertical scaling to determine if it's the right approach for a particular clustering task. It's important to note that not all clustering algorithms can be easily parallelized or accelerated using these platforms. Some algorithms are inherently sequential and may not benefit significantly from vertical scaling platforms.

D) Multiple-Machine Based Techniques

The volume of big data is now measured in petabytes, and it is constantly growing. Even with the use of any dimension reduction techniques, processing this much data on a single machine would be challenging or impossible. The majority of conventional clustering algorithms, on the other hand, struggle because of their reliance on input parameters, data order, and computation costs. Due to these constraints, researchers have led to design different algorithms to perform in a parallel and distributed environment using MapReduce or Spark frameworks. Table 2 presents a comprehensive summary of clustering algorithms, highlighting their applicability based on data characteristics and providing an analysis of their scalability factor.

Table 2: *A brief overview of clustering methods*

Clustering Method	**Data Characteristics**	**Scalability**	**Examples**
Partitioning-Based	Low-dimensional, well-separated clusters	Medium scalability for small to moderate data	K-Means, Mini-batch K-Means, CLARA
Hierarchical-Based	Data without prior knowledge of clusters, interpretable structure	Limited scalability (high computational cost)	BIRCH, CURE
Density-Based	Arbitrary-shaped clusters, noisy data	Low scalability (parameter-sensitive)	DBSCAN, DENCLUE
Grid-Based	High-dimensional data, complex distributions	High scalability (supports parallel processing)	OptiGrid
Multi-Machine	Very large datasets, distributed computing needed	Very high scalability (e.g., petabyte-scale data)	MapReduce, Apache Spark
GPU/FPGA-Based	Real-time processing, iterative tasks	Medium scalability (requires specialized hardware)	CUDA K-Means, FPGA DBSCAN

MapReduce- MapReduce is highly scalable and fault-tolerant programming model that is used in Hadoop to process massive data in parallel fashion. There are two primitive functions, Map and Reduce functions and programmer defines their work on these two functions without taking care of parallel execution across nodes. Another advantage of MapReduce is its flexibility that offers to process structured or unstructured data. In spite of having many advantages, MapReduce inherits some disadvanges by its nature. Firstly, it does not support any high level language like SQL in DBMS, which would optimize coding, and programmers should write their operations with Map and Reduce only. So it is hard to implement most of complex algorithms in this framework. In addition to that , it takes much time for frequent I/O operations, since the intermediate results between queries are stored in local disks. That causes low efficiency.

In [35] it is represented Multiplex KMeans algorithm with MapReduce. The primary concept of the approach is that in order to increase clustering quality, they run multiple KMeans concurrently using various centroid groups and choose the best one among them. The proposed algorithm firstly runs KMeans processes, then it evaluates quality of clustering result by Total Within-Cluster Variation value (TWCV) for each centroid groups, at the end of each iteration. Those which have high TWCV values are pruned in the next step. In the third step, The Permute job is deployed to determine similar centroids and to provide the same location in each group for similar centroids by rearranging them. New centroid groups are formed in the last step. Experiments on real-world datasets show that Mux-KMeans outperform naive KMeans in terms of clustering quality.

Qing Liao et al. in [36] suggested another parallel implementation of Kmeans using MapReduce. They achieved an improvement in comparison to traditional parallel KMeans by adjusting distance measure and initial centroids. In this study, Euclidean distance is chosen as the default distance metric, while Manhattan distance is utilized conditionally. The Initial centroids are picked from the high density region that are the furthest apart.

The work proposed in [37] is a parallel implementation of well-known DBSCAN algorithm using MapReduce. In this work, genetic algorithm (GA) is utilized for adjusting hyper-parameters (minPts and Eps). In order to overcome the time consuming problem of GA-DBSCAN algorithm , it is deployed on MapReduce framework.

Spark- Spark is an open-source distributed computing engine for analyzing large data. It is designed as an alternative to Hadoop to overcome the I/O constrains and enhance performance. When compared to Hadoop MapReduce, they are both fault-tolerant, well scalable and very effective for processing massive data across clustered computers, but Apache Spark outperforms in terms of real-time and iterative processing. Spark employs the concept of RDD (Resilient Distributed Dataset), which makes it perform in-memory computation. Many studies have shown that Spark is up to 100 times quicker than Hadoop MapReduce when the data can fit in memory and up to 10 times faster for batch processing.

Apache Spark provides four higher-level libraries: Spark SQL and Data Frames, Spark Streaming, Spark's Machine Learning Library (MLlib), and GraphX. MLlib offers more than 55 common algorithms for distributed data modeling, such as classification, regression, clustering, feature transformations and etc. [38]. Few clustering algorithms that have been implemented using Spark are included in MLlib as an open-source package. Several studies have been undertaken in order to solve specific shortcomings of the clustering algorithms using Spark or to give completely new techniques. Majority of the recent work on clustering with Spark is related to two prominent algorithms: KMeans and DBSCAN.

One of the first implementation of KMeans based on Spark is suggested in [39]. The proposed algorithm works in two phases: In the first phase, the Basic KMeans algorithm runs for the large number of k. Instead of random selection of initial centroids the algorithm utilize probability sampling, which is aimed to cut off number of iterations to converge. In the second phase, resulting centroids are merged according to certain criteria. Experiments on synthetic large scale datasets display that the proposed method solves the problem of over-resolution without degrading clustering performance.

S_DBSCAN [40] is a performance-oriented extension of DBSCAN that makes use of SPARK. The proposed algorithm consists of three steps: In the first step, data partition is performed by using a random sampling method according to the number of worker nodes. The local DBSCAN algorithm then runs in parallel to build intermediate clusters and each resulting cluster is saved to HDFS as a new RDD. In the last stage, the algorithm combines the partial clusters to generate global clustering results based on the proximity of centroids. The experiments demonstrate that the suggested method is as accurate as regular DBSCAN but more efficient when dealing with large amounts of data.

There are various other articles related to the Spark-based implementation of KMeans, such as [41],[42] and the Spark-based implementation of DBSCAN, such as [43], [44].

IV. Conclusion

Which algorithm and which platform to choose to cluster big data? There is no straight answer to the question and it usually depends on application. Time complexity and scalability are generally traded off. In other words, if it is required to get result in real-time processing, then it is most suitable to run the modification of the algorithms on GPU or FPGA platforms. If it is needed to deal with huge amount of data like hundreds of gigabytes or even petabytes , then it is compulsory to choose multiple machine techniques, like MapReduce or Apache Spark. It is worth noting that Spark would be the best option of these two for implementing suitable clustering algorithms, because of its fast and easy big data processing characteristics. In the future, a hybrid method to clustering massive data is more suited, taking advantage of Spark's huge scalability and GPU's processing speed.

This work was supported by the Azerbaijan Science Foundation - Grant No. AEF-MCG-2023-1(43)-13/04/1-M-04.

References

[1] Nweso Emmanuel Nwogbaga, "A Review of Big Data Clustering Methods and Research Issues", International Journal of Science and Research (IJSR) vol. 9 no. 5, pp. 253-264, 2020.

[2] K. Djouzi and K. Beghdad-Bey, "A Review of Clustering Algorithms for Big Data," 2019 International Conference on Networking and Advanced Systems (ICNAS), pp. 1-6, 2019.

[3] A. Fahad et al., "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis," in IEEE Transactions on Emerging Topics in Computing, vol. 2, no.3, pp. 267-279, 2014.

[4] Shirkhorshidi, A.S., Aghabozorgi, S., Wah, T.Y., Herawan, T. Big Data Clustering: A Review. In: , et al. Computational Science and Its Applications – ICCSA 2014. ICCSA 2014. Lecture Notes in Computer Science, vol 8583. Springer, (2014).

[5] Anand Nayyar, Vikram Puri, "COMPREHENSIVE ANALYSIS & PERFORMANCE COMPARISON OF CLUSTERING ALGORITHMS FOR BIG DATA", Review of Computer Engineering Research, vol 4, no. 2, 2017

[6] D. Sculley, "Web-scale k-means clustering," in Proceedings of the 19th international conference on World wide web. ACM, pp. 1177–1178, 2010.

[7] P. J. Rousseeuw and L. Kaufman, Finding Groups in Data. Wiley Online Library, 1990.

[8] R.T. Ng and J. Han, "Clarans: A method for clustering objects for spatial data mining," IEEE transactions on knowledge and data engineering, vol. 14, no. 5, pp. 1003–1016, 2002.

[9] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: an efficient data clustering method for very large databases," in ACM Sigmod Record, vol. 25, no. 2. ACM, pp. 103–114, 1996.

[10] S. Guha, R. Rastogi, and K. Shim, "Cure: an efficient clustering algorithm for large databases," Information Systems, vol. 26, no. 1, pp. 35–58, 2001.

[11] A. Hinneburg, D. A. Keim et al., "An efficient approach to clustering in large multimedia databases with noise," in KDD, vol. 98, pp. 58–65, 1998.

[12] H. Rehioui, A. Idrissi, M. Abourezq, F. Zegrari "DENCLUE-IM: A New Approach for Big Data Clustering", Procedia Computer Science, vol. 83, pp. 560 – 567, 2016.

[13] A. Hinneburg and D. A. Keim, ''Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering,'' in Proc. 25th Int. Conf. Very Large Data Bases (VLDB), pp. 506–517, 1999.

[14] Yu. Lei, Huan Liu, Efficient feature selection via analysis of relevance and redundancy, J. Mach. Learn. Res. 5, 1205–1224, (Oct. 2004)

[15] Qinbao Song, Jingjie Ni, Guangtao Wang, A fast clustering-based feature subset selection algorithm for high-dimensional data, IEEE Trans. Knowl. Data Eng. 25 (1), 1–14, (2013).

[16] Y. Wang, J. Wang, H. Liao, and H. Chen, "An efficient semi-supervised representatives feature selection algorithm based on information theory," Pattern Recognition, vol. 61, pp. 511–523, 2017.

[17] X. Kong, C. Hu, and Z. Duan, "Generalized principal component analysis," in Principal Component Analysis Networks and Algorithms. Springer,pp. 185–233, 2017.

[18] S. Tasoulis, L. Cheng, N. Valim ¨ aki, N. J. Croucher, S. R. Harris, ¨ W. P. Hanage, T. Roos, and J. Corander, "Random projection based clustering for population genomics," in Big Data (Big Data), 2014 IEEE International Conference on. IEEE, pp. 675–682,2014.

[19] D. Chu, L.-Z. Liao, M. K.-P. Ng, and X. Wang, "Incremental linear discriminant analysis: a fast algorithm and comparisons," IEEE transactions on neural networks and learning systems, vol. 26, no. 11, pp. 2716–2735, 2015.

[20] Sakar, C.O., Kursun, O. & Gurgen, F. "Ensemble canonical correlation analysis". Applied Intelligence, vol. 40, pp. 291–304 (2014).

[21] Ying Yan, Liang Jeff Chen, and Zheng Zhang. Error-bounded sampling for analytics on big sparse data. PVLDB, 7(13):1508–1519, 2014

[22] Z. Liu and A. Zhang, "Sampling for Big Data Profiling: A Survey," in IEEE Access, vol. 8, pp. 72713-72726, 2020.

[23] Julian Ramos Rojas, Mary Beth Kery, Stephanie Rosenthal, and Anind K. Dey. Sampling techniques to improve big data exploration. In 7th IEEE Symposium on Large Data Analysis and Visualization, LDAV 2017, Phoenix, AZ, USA, October 2, 2017, pages 26–35, 2017

[24] P. Hore, L. O. Hall, and D. B. Goldgof, "Single pass fuzzy c means," in 2007 IEEE International Fuzzy Systems Conference. IEEE, pp.1–7, 2007.

[25] Singh, Dilpreet, and Chandan K Reddy. "A Survey on Platforms for Big Data Analytics." Journal of Big Data 2, no. 1 (2014).

[26] Sodan, Angela & Machina, Jacob & Deshmeh, Arash & Macnaughton, Kevin & Esbaugh, Bryan." Parallelism via Multithreaded and Multicore CPUs". Computer. 43. 24 – 32, (2010).

[27] Erdem, Atakan & Gündem, Taflan. "M-FDBSCAN: A multicore density-based uncertain data clustering algorithm". TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES. 22. 143-154, (2014).

[28] Owens, J.D., M. Houston, D. Luebke, S. Green, J.E. Stone, and J.C. Phillips. "GPU Computing." Proceedings of the IEEE 96, no. 5 , pg.879–899, (2008).

[29] A. Bustamam, K. Burrage and N. A. Hamilton, "Fast Parallel Markov Clustering in Bioinformatics Using Massively Parallel Computing on GPU with CUDA and ELLPACK-R Sparse Format," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 9, no. 3, pp. 679-692, May-June 2012.

[30] Zechner, Mario & Granitzer, Michael. (2009). "Accelerating K-Means on the Graphics Processor via CUDA". Proceedings of the 1st International Conference on Intensive Applications and Services, INTENSIVE, 7-15, 2009.

[31] Ian Kuon, Russell Tessier and Jonathan Rose, "FPGA Architecture: Survey and Challenges", Foundations and Trends® in Electronic Design Automation: Vol. 2: No. 2, pp 135-253, (2008).

[32] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 26, no. 2, pp. 203–215, (2007).

[33] Scicluna, Neil & Bouganis, Christos.A Multidimensional FPGA-Based Parallel DBSCAN Architecture. ACM Transactions on Reconfigurable Technology and Systems. 9. 1-15, (2015).

[34] Maruyama, Tsutomu." Real-time K-Means Clustering for Color Images on Reconfigurable Hardware". Proceedings - International Conference on Pattern Recognition. 2. 816-819, (2006)

[35] Chen Li, Yanfeng Zhang, Minghai Jiao, and Ge Yu. 2014." Mux-Kmeans: multiplex kmeans for clustering large-scale data set". In Proceedings of the 5th ACM workshop on Scientific cloud computing (ScienceCloud '14). Association for Computing Machinery, New York, NY, USA, 25–32. 2014

[36] Liao, Qing & Yang, Fan & Zhao, Jingming." An improved parallel K-means clustering algorithm with MapReduce", 15th IEEE International Conference on Communication Technology (ICCT), 764-768, (2013)

[37] Hu, Xiaojuan & Liu, Lei & Qiu, Ningjia & Yang, Di & Li, Meng. A MapReduce-based improvement algorithm for DBSCAN. Journal of Algorithms & Computational Technology. 12.(2017).

[38] Assefi, Mehdi & Behravesh, Ehsun & Liu, Guangchi & P. Tafti, Ahmad. "Big Data Machine Learning using Apache Spark MLlib". IEEE Big Data , 2017.

[39] Sinha, Ankita & Jana, Prasanta." A novel K-means based clustering algorithm for big data". International Conference on Advances in Computing, Communications and Informatics (ICACCI), 1875-1879.(2016)

[40] G. Luo, X. Luo, T. F. Gooch, L. Tian and K. Qin, "A Parallel DBSCAN Algorithm Based on Spark," 2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom), pp. 548-553, 2016.

[41] Wang B, Yin J, Hua Q, Wu Z, Cao J (2016) Parallelizing k-means-based clustering on spark. In: International conference on advanced cloud and Big Data (CBD). IEEE, Chengdu, pp 31–36.

[42] Zayani A, Ben N'Cir CE, Essoussi. N Parallel clustering method for non-disjoint partitioning of largescale data based on spark framework. In: IEEE international conference on big data (Big Data). IEEE, Washington, DC, pp 1064–1069. (2016)

[43] D. Han, A. Agrawal, W. -k. Liao and A. Choudhary, "Parallel DBSCAN Algorithm Using a Data Partitioning Strategy with Spark Implementation," 2018 IEEE International Conference on Big Data (Big Data), pp. 305-312, 2018.

[44] Gong, Y., Sinnott, R.O., Rimba, P. RT-DBSCAN: Real-Time Parallel Clustering of Spatio-Temporal Data Using Spark-Streaming. In: , et al. Computational Science – ICCS 2018. ICCS 2018. Lecture Notes in Computer Science(), vol 10860. Springer, Cham.(2018).