Soft Clustering for Very Large Data Sets

Min Chen

State University of New York, New Paltz, NY, USA

Summary
Clustering is regarded as one of the significant task in data mining and has been widely used in very large data sets. Soft clustering is unlike the traditional hard clustering which allows one data belong to two or more clusters. Soft clustering such as fuzzy c-means and rough k-means have been proposed and successfully applied to deal with uncertainty and vagueness. However, the influx of very large amount of noisy and blur data increases difficulties of parallelization of the soft clustering techniques. The question is how to deploy clustering algorithms for this tremendous amount of data to get the clustering result within a reasonable time. This paper provides an overview of the mainstream clustering techniques proposed over the past decade and the trend and progress of clustering algorithms applied in big data. Moreover, the improvement of clustering algorithms in big data are introduced and analyzed. The possible future for more advanced clustering techniques are illuminated based on today's information era.

Key words:
Soft clustering, big data, parallel computing

1. Introduction

Massive volume of structured, unstructured or heterogeneous data have been agglomerated because of the growth of the web, the rise of social media, the use of mobile, and the information of Internet of Things (IoT) by and about people, things, and their interactions [1]. Due to the maturity of database technologies, how to store these massive amount of data is no longer a problem anymore. The problem is how to handle and hoard these very large data sets, as well as further find out solutions to understand or dig out useful information which can turn into data products is a major challenge.

Clustering [2] is one of the most fundamental tasks in exploratory data analysis that groups similar data points in an unsupervised process. Clustering techniques have been exploited in many fields including in many areas, such as data mining, pattern recognition, machine learning, biochemistry and bioinformatics [3]. The main process of clustering algorithms is to divide a set of unlabeled data objects into different groups. The cluster membership measure is based on a similarity measure. In order to obtain a high quality partition, the similarity measure between the data objects in the same group is to be maximized, and the similarity measure between the data objects from different groups is to be minimized. Most of the clustering task uses an iterative process to find locally or globally optimal solutions from a high-dimensional data set. Partitioned algorithms include two main clustering strategies [4]: the hard clustering and the soft clustering. The conventional hard clustering methods classify each object to only one cluster. As a consequence, the results are crisp. On the other hand, soft clustering allows the objects to belong to two or more clusters with varying degrees of membership. Soft clustering plays a significant role in various problems such as feature analysis, systems identification, and classification design [5]. Soft clustering is more realistic than hard clustering due to the ability of handling impreciseness, uncertainty, and vagueness for real-world problems.

In addition, tremendous amount of data are being accumulated at fast-speed at the beginning of this new century. This data is potentially contaminated with fuzziness due to the imprecision, uncertainty and vagueness. The problem becomes how we can analyze and reveal valuable knowledge that is hidden within the data in an efficient and effective way. With the high complexity and computational cost, traditional soft clustering techniques are however limited to handle very large volume of data with fuzziness.

Moreover, conventional clustering techniques cannot cope with this huge amount of data because of their high complexity and computational cost [6]. The question for big data clustering is how to scale up and speed up clustering algorithms with minimum sacrifice to the clustering quality. Therefore, an efficient processing model with a reasonable computational cost of this huge, complex, dynamic and heterogeneous data is needed in order to exploit this huge amount of data. There have already been some comparative studies on conventional soft clustering algorithms. However, a current comparison and survey of soft clustering algorithms for very large data sets is most desirable for current big data era.

The rest of the paper is organized as follows: an overview of soft clustering includes general information of soft clustering, main stream soft clustering algorithms and soft clustering validation index are introduced in Section 2. The key technologies using soft clustering in big data is

Manuscript received January 5, 2017
Manuscript revised January 20, 2017

illustrated in Section 3. A small selection of applications of soft clustering is discussed in Section 4. The paper is concluded in Section 5.

2. Overview of Soft Clustering

2.1 General information of soft clustering

Soft clustering is one of the most fundamental tasks in exploratory data analysis that groups similar data points in an unsupervised process. The main process of clustering algorithms is to divide a set of unlabeled data objects into different groups. The cluster membership measure is based on a similarity measure. In order to obtain a high quality partition, the similarity measure between the data objects in the same group is to be maximized, and the similarity measure between the data objects from different groups is to be minimized [7]. Most of the clustering task uses an iterative process to find locally or globally optimal solutions from a high-dimensional data sets. In addition, there is no unique clustering solution for real-life data and it is also hard to interpret the 'cluster' representations [8]. Therefore, the clustering task requires much experimentation with different algorithms or with different features of the same data set. Hence, how to save computational complexity is a significant issue for the clustering algorithms. Moreover, clustering very large data sets that contain large numbers of records with high dimensions is considered a very important issue nowadays. Most conventional clustering algorithms suffer from the problem that they do not scale with larger sizes of data sets, and most of them are computationally expensive with regards to memory space and time complexities. For these reasons, the parallelization of clustering algorithms is a solution to overcome the aforementioned problems, and the parallel implementation of clustering algorithms is inevitable.

More importantly, clustering analysis is unsupervised 'nonpredictive' learning. It divides the data sets into several clusters based on a subjective measurement. Clustering analysis is unlike supervised learning and it is not based on a 'trained characterization'. In general, there is a set of desirable features for a clustering algorithm [9]: scalability, robustness, order insensitivity, minimum user-specified input, arbitrary-shaped clusters, and point proportion admissibility. Thus, a clustering algorithm should be chosen such that duplicating the data set and the re-clustering task should not change the clustering results.

Depending on how the membership of an instance to a cluster is define, two groups of soft clustering algorithms are identified: discrete and continuous methods. Specifically, fuzzy clustering and rough clustering. Hard clustering can be considered as a special case of soft clustering which membership values are discrete and restricted to either 0 or 1 (see Fig. 1). Fuzzy clustering provides continuous membership degrees which range from 0 to 1. The objective of fuzzy clustering is to minimize the weighted sum of Euclidean distance between the objects. Fuzzy clustering is a method of clustering that allows one piece of data to belong to two or more clusters (see Fig. 2). The Fuzzy C-Means (FCM) algorithm is an iterative partition clustering technique that was first introduced by Dunn [10], and was then extended by Bezdek [11]. FCM uses a standard least squared error model that generalizes an earlier and very popular non-fuzzy c-means model that produces hard clusters of the data.

Rough clustering extends the theory of rough or approximation sets. Rough k-means is first introduced by Lingras [12]. Each cluster has a lower and an upper approximation. The lower approximation is a subset of the upper approximation (see Fig. 3). In other words, the upper approximation is a boundary region. The members of the lower approximation belong to any other cluster. The data objects in an upper approximation may belong to the cluster. Since their membership is uncertain they must be member of an upper approximation of at least another cluster. Hence, an object to a cluster has two membership degrees. One for its lower approximation and one for its upper approximation.

2.2 Fuzzy Clustering

Fuzzy clustering is a method of clustering which allows one piece of data to belong to two or more clusters. The fuzzy c-means algorithm is a pretty standard least squared error model that generalizes an earlier and very popular non-fuzzy c-means model that produces hard clusters of the data. An optimal c partition is produced iteratively by minimizing the weighted within group sum of squared error objective function [13]:

$$J_m = \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}^{m} d^2(y_i, c_j) \quad (1)$$

where $Y = [y_1, y_2, ..., y_n]$ is the data set in a d-dimensional vector space. n is the number of data items. c is the number of clusters which is defined by the user where $2 \leq c \leq n$. u_{ij} is the degree of membership of y_i in the j^{th} cluster. m is a weighted exponent on each fuzzy membership. c_j is the center of cluster j. $d^2(x_i, c_j)$ is a square distance measure between object y_i and cluster c_j . An optimal solution with c partitions can be obtained via an iterative process which is as follows:

1. Input(c, m, ϵ, data)
2. Initialize the fuzzy partition matrix $U = [u_{ij}]$
3. Iteration starts and set t=1
4. Calculate the c cluster centers with U^t:

$$c_i = \frac{\sum_{i=1}^{n} u_{ij}^m y_i}{\sum_{i=1}^{n} u_{ij}^m} \quad (2)$$

5. Calculate the membership U^{t+1} using:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} (\frac{d_{ij}}{d_{kj}})^{\frac{2}{m-1}}} \quad (3)$$

6. If the stopping criteria is not met, $t = t + 1$ and go to Step 4.

2.3 Rough Clustering

Rough clustering is a partitioning algorithm which divides a set of objects into several rough clusters. A rough cluster is described by a lower approximation and an upper approximation. Data points in the lower approximation belong to the corresponding cluster only. Data points in the upper approximation can be members of upper approximations of other clusters. Hence, a data point i has two membership degrees to a cluster k, one for its lower approximation and one for its upper approximation [14]:

$$u_{ik}^{Lower\ Approx.} = \overline{u_{ik}} \in \{0,1\} \quad (4)$$

$$u_{ik}^{Upper\ Approx.} = \widehat{u_{ik}} \in \{0,1\} \quad (5)$$

Rough k-means clustering uses the squared Euclidean distance to measure the dissimilarity between a vector and cluster centroids.

$$min \left\{ J = \sum_{k=1}^{n} \left(\frac{w^L}{|L_k|} \sum_{y_i \in L_k} d^2(y_i, c_j) + \frac{w^U}{|U_k|} \sum_{y_i \in U_k} d^2(y_i, c_j) \right) \right\} \quad (6)$$

where w^L is the weight for the lower approximation and $w^U = 1 - w^L$ is the weight for the upper approximation. $|L_k|$ and $|U_k|$ are number of objects in lower approximation and upper approximation, respectively.

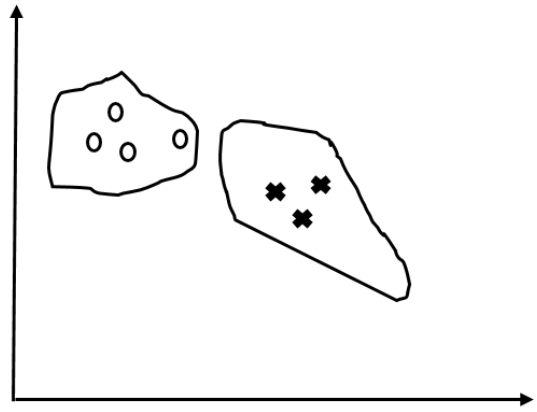

Fig. 1 Hard clustering with crisp membership

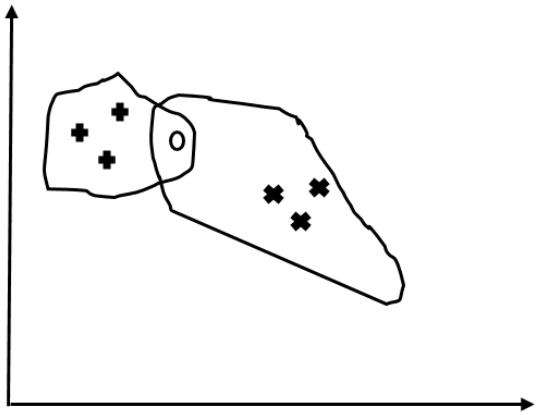

Fig. 2 Fuzzy clustering with overlapping region.

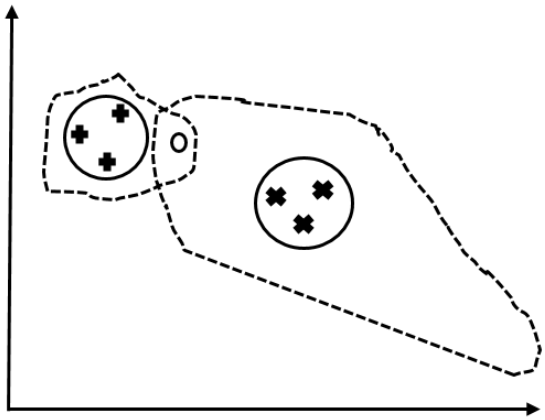

Fig. 3 Rough clustering with a lower and an upper approximation.

Let c_j be the centroid of j^{th} cluster and $d^2(y_i, c_j)$ be the squared Euclidean distance between the data point y_i and c_j. The new centroids are updated as follows:

$$c_j = \begin{cases} w^L \sum_{y_i \in L_k} y_i/|L_k| + w^U \sum_{y_i \in U_k} x_i/|U_k|, & U_k \neq \emptyset \\ \sum_{y_i \in L_k} x_i/|L_k|, & otherwise \end{cases} \quad (7)$$

The procedure of rough k-means algorithm is as follows:

1. Randomly assign each data point of the data set S to the lower approximation and upper approximation with a predefined *k* clusters.
2. Update the cluster centroid c_j using Eq. (7).
3. For each remaining data object
    a. Find its nearest cluster centroid and update upper approximation.

b. Check if further centroids are not significantly farther away than the closest one and update lower approximation.

4. Recalculate the cluster centroid using Eq. (7).

5. Repeat step 3 to step 5 until stop criteria have been met.

2.4 Mainstream Soft Clustering Algorithms

Table 1 list some extensions and derivatives of fuzzy clustering and rough clustering algorithms. Rough clustering is relatively new. The number of its extensions and derivatives is smaller than that of the fuzzy clustering.

2.5 Soft Clustering Validation Index

Clustering validation is aimed to evaluate the clustering results to find the best partition that fits the underlying data. Thus, cluster validity is used to quantitatively evaluate the result of clustering algorithms. Compactness and separation are considered as two widely used criteria in measuring the quality of partitions. Traditional approaches run the algorithm iteratively using different input values and select the best validity measure to determine the "optimum" number of clusters. A collection of validity indices in fuzzy clustering is listed below [24].

Table 1: Extensions and derivatives of fuzzy clustering and rough clustering.

List of soft clustering algorithms	***Authors***
Fuzzy C-Means [10, 11]	Bezdek, et al.
Possibilistic c-means[15]	Raghuram, et
Possibilistic fuzzy c-means[16]	Pal et al.
Gustafson-Kessel(GK) algorithm[17]	Guerrero-Bote, et al.
Fuzzy C-Varieties[18]	Łeski, et al.
Fuzzy C-Regression[19]	Hathaway, et
Evidential c-means[20]	Dubois, et al.
Rough k-means[12]	Lingras, et al.
Rough-fuzzy clustering[21]	Dubois, et al.
Rough-fuzzy possibilistic clustering[22]	Maji, et al.
Shadowed set clustering[23]	Mitra, et al.

1) Least Squared Error (SE) Index: The weighted within cluster sum of squared error function is used:

$$J_m = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^m d^2(y_i, c_j) \quad (8)$$

where y_i is the i^{th} data point with d dimensions. cj is the value of the jth cluster, and $||y_i - c_j||$ is the Euclidean distance between y_i and c_j. J_m takes its minimum value when the cluster structure is best.

2) Partition Coefficient (PC) Index: The partition coefficient (PC) is defined as:

$$PC = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^2 \quad (9)$$

PC obtains its maximum value when the cluster structure is optimal.

3) Partition Entropy (PE) Index: The partition entropy was defined as:

$$PE = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}\log_b u_{ij} \quad (10)$$

where b is the logarithmic base. PE gets its minimum value when the cluster structure is optimal.

4) Modified Partition Coefficient (MPC) Index: Modification of the PC index, which can reduce the monotonic tendency, is proposed by Dave in 1996:

$$MPC = 1 - \frac{c}{c-1}(1 - PC) \quad (11)$$

where c is the number of cluster. An optimal cluster number is found by maximizing MPC to produce a best clustering performance for a data set.

5) Fukuyama and Sugeno (FS) Index: Fukuyama and Sugeno proposed a validity function in 1989. It is defined as:

$$FS = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^m \|y_i - c_i\| - \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^m \left\|c_j - \bar{c}_i\right\| \quad (12)$$

where $\bar{c}_i = \sum_{j=1}^{c} c_j/c$. FS measures the separation. The first term equals to J_m which is the least squared error. It measures the compactness. The best clustering performance for a data set is found by maximizing the value of FS.

6) Xie-Beni (XB) Index: Xie and Beni proposed a validity function in 1991, and later it was modified by Bezdek in 1995:

$$XB = \frac{J_m}{n \times min_{i\neq j}\left\|y_i - c_i\right\|^2} \quad (13)$$

XB reaches its minimum value when the cluster structure is optimal.

7) Partition Coefficient and Exponential Separation (PCAES) Index: The partition coefficient and exponential separation (PCAES) index is defined as:

$$PCAES = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^2/u_M - \sum_{j=1}^{c}\exp(-\min_{i\neq j}\|z_i - z_j\|^2/\beta_T) \quad (14)$$

where $u_M = \min_{1<j<c}\{\sum_{i=1}^{n} u_{ij}^2\}$ and $\beta_T = \sum_{j=1}^{c}\|z_j - \bar{z}\|^2/c$. PCAES takes its maximum value when the cluster structure is optimal.

3. Key Technologies Using Soft Clustering in Big Data

3.1 Big data problem for soft clustering

Conventional soft clustering algorithms only deal with structural data with limited size. However, due to growth of the web, the rise of social media, the use of mobile, and the information of Internet of Things (IoT) by and about people, things, and their interactions, huge volume of structured, unstructured or heterogeneous data have been agglomerated. Substantial changes in the architecture of storage system because of the large volume of data is necessary for soft clustering.

Thus, the issue of soft clustering for very large data sets is how to speed up and scale up the clustering algorithms with the minimum sacrifice to the clustering quality. Generally, there are three approaches to speed up and scale up soft clustering techniques [25]. The most basic way to address big data is to use sampling-based techniques to reduce the iterative process. A sample of the datasets instead of using on the whole dataset is used to perform the clustering. CLARA, CURE and the core set algorithms are popular sampling-based techniques [26]. Another approach uses randomized algorithms to reduce the data dimension. The data sets are projected from a high dimensional space to a lower dimensional space. BIRCH and CLARANS are two well-known algorithms of this type. The most common approach is to apply parallel and distributed algorithms use multiple machines to speed up the computation in order to increase the scalability.

Parallel processing is essential to processing a huge volume of data in a timely manner in the big data era. It uses a divide and conquer approach to divide the huge amount of data into small data chunks. These small data chunks can be handled and loaded on different machines and the solutions are combined to solve the huge problem. A general framework for both parallel and MapReduce clustering algorithms is illustrated in Fig 4. The most common parallel processing models for computing data-intensive applications are OpenMP, MPI [27], and MapReduce are common parallel. In here, we only discuss the conventional parallel and MapReduce clustering algorithms.

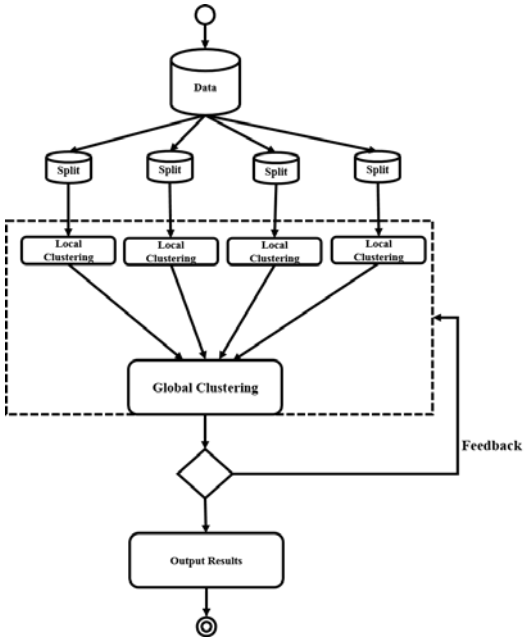

Fig. 4 A general framework of soft clustering for very large data sets.

3.2 Parallel Soft Clustering Algorithms

Fuzzy c-means is the most popular algorithm for fuzzy clustering and a parallel fuzzy c-means is proposed in [28]. The data is partitioned equally among the available processors. The initial centers is set and broadcasts them to all the processors. Each processor compute the geometrical center of its local data and communicate its centers to other processors in order to compute the global centers. The procedure is repeated as many times as convergence is achieved. Bit-reduced fuzzy c-means is designed to handle large images clustering [29]. Moreover, kernel fuzzy c-means algorithm is another approach to address very large data. One high-speed rough clustering is proposed to deal with very large document collections [34].

3.3 MapReduce based Soft Clustering

Due to the virtue of simplicity, scalability and fault-tolerance, MapReduce [30] is one of the most efficient big data solutions which enables to process a massive volume of data in parallel with many low-end computing nodes. This programming paradigm is a scalable and fault-tolerant data processing technique that was developed to provide significant improvements in large-scale data-intensive applications in clusters.

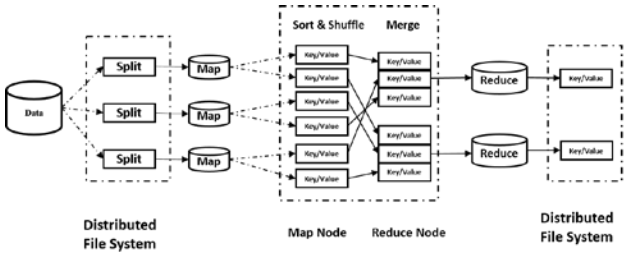

Fig. 5 The procedure of a MapReduce system.

The MapReduce model hides the details of the parallel execution, which allows users to focus only on data processing strategies. The MapReduce model consists of mappers and reducers. The main aspect of the MapReduce algorithm is that if every map and reduce is independent of all other ongoing maps and reduces, then the operations can be run in parallel on different keys and lists of data.

The process of the MapReduce approach can be decomposed as follows (see Fig. 5): (1) data preparation: an underlying storage layer to read input and store output is a Distributed File System (DFS) [31]. GFS and HDFS are most common systems which are a chunk-based distributed file system that supports fault-tolerance by data partitioning and replication. The input data is divided into small chunks on different slave machines. (2) Map step: the map function of each node is applied to local data and the output is written to a temporary storage space. (3) Sort and combine step: the output from step (2) is sorted and shuffled with key such that all data belonging to one key are located on the same node. The sorted results are emitted to the reducers. (4) reduce step: each group of output data (per key) is processed in parallel on each reduce node. The user-provided reduce function is executed once for each key value produced by the map step. (5) Final output: The final output is produced by the reducer of the MapReduce system and is stored in the DFS.

A MapReduced-based fuzzy c-means clustering algorithm is implemented to explore the parallelization and scalability [32]. A parallel method for computing rough set approximations is proposed. This parallel method based on the MapReduce technique are put forward to deal with the massive data [33].

4. Applications

Soft clustering has been proved to perform better for noisy data. Soft clustering has been used in a numerous number of real life applications. The sources of big data are mainly social media, mobile and internet of things. Soft clustering is well for this huge volume of structured, unstructured or heterogeneous data due to the capability of handling uncertainty and vagueness. A small selection of applications of soft clustering for very large data sets has provided (see Table 2) [14, 26].

Table 2: Selections of applications of soft clustering.

Applications of Fuzzy Clustering	***Applications of Rough Clustering***
Community detection	Patterns of gene
Image segmentation	Speech recognition
Pattern recognition	Retail data clustering
Metabolomics in bioinformatics	Path profiles on a website
Market segmentation	Traffic monitor

5. Concluding Remarks

Soft clustering for very large data and the corresponding soft clustering algorithms are reviewed. Due to the ability of handling impreciseness, uncertainty, and vagueness for real-world problems, soft clustering is more realistic than hard clustering. Soft clustering as a partitioning algorithm is well for big data due to the heterogeneous structure of very large data. Parallelism in soft clustering is potentially useful for big data. MapReduce has gained significant momentum from industry and academia in recent years.

References

[1] B. Mirkin, “Clustering: A Data Recovery Approach,” Second Edition (Chapman & Hall/CRC Computer Science & Data Analysis).

[2] H. A. Edelstein. Introduction to data mining and knowledge discovery (3rd ed). Potomac, MD: Two Crows Corp. 1999

[3] Chen, Min, and Simone A. Ludwig. "Fuzzy clustering using automatic particle swarm optimization." 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). IEEE, 2014.

[4] V. P. Guerrero-Bote, et al., “Comparison of neural models for document clustering," Int. Journal of Approximate Reasoning, vol. 34, pp.287-305, 2003.

[5] B. Mirkin, “Clustering: A Data Recovery Approach,” Second Edition (Chapman & Hall/CRC Computer Science & Data Analysis).
[6] Z. Xu and Y. Shi, Exploring Big Data Analysis: Fundamental Scientific Problems. Annals of Data Science, 2(4), 363-372, 2015.
[7] A. S. Shirkhorshidi, S. Aghabozorgi, T. Y. Wah, and T. Herawan, Big data clustering: a review. In International Conference on Computational Science and Its Applications (pp. 707-720). Springer International Publishing, 2014.
[8] B. L. Kaufman, P. J. Rousseeuw, Finding groups in data: an introduction to cluster analysis (Vol. 344). John Wiley & Sons, 2009.
[9] G. L. Carl, “A fuzzy clustering and fuzzy merging algorithm, Technical Report,” CS-UNR-101, 1999.
[10] J. C. Dunn, ``A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," Journal of Cybernetics 3: 32-57, 1973.
[11] J. C. Bezdek, ``Pattern Recognition with Fuzzy Objective Function Algorithms". ISBN 0-306-40671-3, 1981.
[12] Lingras, Pawan, and Georg Peters. "Applying rough set concepts to clustering." Rough Sets: Selected Methods and Applications in Management and Engineering. Springer London, 2012. 23-37.
[13] J. C. Bezdek. “Cluster validity with fuzzy sets.” (1973): 58-73.
[14] Peters, Georg, et al. "Soft clustering–fuzzy and rough approaches and their extensions and derivatives." International Journal of Approximate Reasoning 54.2 (2013): 307-322.
[15] Krishnapuram, Raghuram, and James M. Keller. "The possibilistic c-means algorithm: insights and recommendations." IEEE transactions on Fuzzy Systems 4.3 (1996): 385-393.
[16] Pal, Nikhil R., et al. "A possibilistic fuzzy c-means clustering algorithm." IEEE transactions on fuzzy systems 13.4 (2005): 517-530.
[17] V. P. Guerrero-Bote, et al., Comparison of neural moddls for document clustering, Int. Journal of Approximate Reasoning, vol. 34, pp.287-305, 2003.
[18] Łęski, Jacek M. "Fuzzy c-varieties/elliptotypes clustering in reproducing kernel hilbert space." Fuzzy Sets and Systems 141.2 (2004): 259-280.
[19] Hathaway, Richard J., and James C. Bezdek. "Switching regression models and fuzzy clustering." IEEE Transactions on fuzzy systems 1.3 (1993): 195-204.
[20] Masson, Marie-Hélène, and Thierry Denoeux. "ECM: An evidential version of the fuzzy c-means algorithm." Pattern Recognition 41.4 (2008): 1384-1397.
[21] Dubois, Didier, and Henri Prade. "Rough fuzzy sets and fuzzy rough sets*." International Journal of General System 17.2-3 (1990): 191-209.
[22] P. Maji, S.K. Pal, Rough set based generalized fuzzy c-means algorithm and quantitative indices, IEEE Transactions on Systems,Man, and Cybernetics – Part B: Cybernetics 37 (6) (2007) 1529–1540.
[23] S. Mitra,W. Pedrycz, B. Barman, Shadowed c-means: Integrating fuzzy and rough clustering, Pattern Recognition 43 (2010) 1282–1291.
[24] Chen, Min, and Simone A. Ludwig. "Particle swarm optimization based fuzzy clustering approach to identify optimal number of clusters." Journal of Artificial Intelligence and Soft Computing Research 4.1 (2014): 43-56.
[25] C. C.Aggarwal, C. K. Reddy, Data clustering: algorithms and applications. CRC Press, 2013.
[26] Min Chen, Simone A. Ludwig and Keqin Li, “Clustering in Big Data,” Big Data: Management, Architecture, and Processing, Ch 16: p.g. 331-246. CRC Press, Taylor & Francis Group, 2017.
[27] J. A. Zhang, Parallel Clustering Algorithm with MPI–Kmeans. Journal of computers 8.1 (2013): 10-17.
[28] Kwok, Terence, et al. "Parallel fuzzy c-means clustering for large data sets." European Conference on Parallel Processing. Springer Berlin Heidelberg, 2002.
[29] Havens, Timothy C., et al. "Fuzzy c-means algorithms for very large data." IEEE Transactions on Fuzzy Systems 20.6 (2012): 1130-1146.
[30] X.Wu, X. Zhu, G. Q. Wu, and W. Ding, Data mining with big data. IEEE transactions on knowledge and data engineering, 26(1), 97-107, 2014.
[31] K. Shim, MapReduce algorithms for big data analysis. Proceedings of the VLDB Endowment, 5(12), 2016-2017, 2012.
[32] Ludwig, Simone A. "MapReduce-based fuzzy c-means clustering algorithm: implementation and scalability." International Journal of Machine Learning and Cybernetics 6.6 (2015): 923-934.
[33] Zhang, Junbo, Tianrui Li, and Yi Pan. "Parallel rough set based knowledge acquisition using MapReduce from big data." Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications. ACM, 2012.
[34] Kishida, Kazuaki. "High-speed rough clustering for very large document collections." Journal of the American Society for Information Science and Technology 61.6 (2010): 1092-1104.

Min Chen is now an Assistant Professor at State University of New York at New Paltz. She received her bachelor‘s degree in mathematics and physics from College of St. Benedict in 2009, and earned her master’s degree in computer science and doctoral degree in software engineering at North Dakota State University in 2011 and 2015, respectively. Her current research interests include artificial intelligence, machine learning and big data computing.