JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 10, No. 1, March 2016, pp. 1-8

An Improved Hybrid Canopy-Fuzzy C-Means Clustering Algorithm Based on MapReduce Model

Wei Dai*
School of Economics and Management, Hubei Polytechnic University, Huangshi, China
dweisky@163.com

Changjun Yu and Zilong Jiang
School of Computer Science and Technology, Wuhan University of Technology, Wuhan, China
cjyuit@163.com, wuhanjzl@163.com

Abstract

The fuzzy c-means (FCM) is a frequently utilized algorithm at present. Yet, the clustering quality and convergence rate of FCM are determined by the initial cluster centers, and so an improved FCM algorithm based on canopy cluster concept to quickly analyze the dataset has been proposed. Taking advantage of the canopy algorithm for its rapid acquisition of cluster centers, this algorithm regards the cluster results of canopy as the input. In this way, the convergence rate of the FCM algorithm is accelerated. Meanwhile, the MapReduce scheme of the proposed FCM algorithm is designed in a cloud environment. Experimental results demonstrate the hybrid canopy-FCM clustering algorithm processed by MapReduce be endowed with better clustering quality and higher operation speed.

Category: Smart and intelligent computing

Keywords: FCM; Canopy; Clustering; MapReduce

I. INTRODUCTION

The fuzzy c-means (FCM) is one of the most popular ongoing areas of research in computer science, mathematics and other areas of engineering, and so on [1]. Several problems from various areas have been effectively solved by using FCM and variants. But, for efficient use of the algorithm in various diversified applications, some modifications or hybridization with other algorithms are needed.

At present, many researchers have proposed various innovative approaches related to the FCM clustering algorithm. The classification of application areas in this survey has been made in the following manner: neural network, clustering and classification, image analysis, structural analysis of algorithms together in various application domains. A comparative study between FCM and SVM has been performed by Hassen et al. [2] for classification of chest lesions and elaborates the possibility of increasing the interpretability of SVM classifier by hybridizing with FCM. Bharill and Tiwari [3] introduced a random sampling iterative optimization fuzzy c-means (RSIO-FCM) clustering algorithm which partitions large datasets into various subsets and results in formation of effective clusters for elimination of the problem of overlapping cluster centers. Kannan et al. [4] proposed an effective robust FCM by introducing a specialized center initialization method for execution of the proposed algo-

Open Access http://dx.doi.org/10.5626/JCSE.2016.10.1.1 http://jcse.kiise.org

Received 28 June 2015; **Revised** 18 December 2015; **Accepted** 05 February 2016

*Corresponding Author

pISSN: 1976-4677 eISSN: 2093-8020

rithm for a segmentation of breast and brain magnetic resonance images. Wang et al. [5] improved the performance of FCM by the appropriate selection of feature weight vectors and applying the gradient descent method. Esteves and Rong [6] compared k-means and FCM for clustering a noisy realistic and big dataset by using a free cloud computing solution Apache Mahout/Hadoop and Wikipedia's latest articles. Yu and Dai [7] proposed a parallel FCM algorithm based on the MapReduce to improve the increased amount of data the time complexity. Zhang et al. [8] proposed a solution of parallel FCM clustering algorithm in multi-core platform performance bottleneck with the help of the Intel Parallel Amplifier high-performance tool, to find hotspot and concurrency. In this paper, cloud computing is employed to further investigate the FCM algorithm. It is of great theoretical value and application significance to combine this algorithm with other algorithms to improve the execution speed of the FCM algorithm processed by MapReduce and enhance the clustering efficiency and effectiveness.

II. HYBIRD CANOPY-FCM CLUSTERING ALGORITHM DESIGN

The main idea of canopy-FCM algorithm is to use the canopy clustering algorithm to generate a cluster center, and then use the cluster center as the initial cluster center of FCM clustering algorithm [9].

The basic process of canopy-FCM algorithm is divided into two phases. The first phase is to divide the dataset into several canopy centers by using canopy clustering algorithm and remove the canopy center which is less than a certain threshold. The second phase is produced according to the first phase of the initial cluster center using FCM clustering algorithm for clustering.

A. Canopy Clustering Algorithm

The canopy clustering algorithm is an unsupervised pre-clustering algorithm introduced by McCallum et al. [10] and designed to speed up clustering operations on large data sets, where using another algorithm directly may be impractical due to the size of the data set. The algorithm proceeds as follows [10, 11]:

1) Put all records into a set S. Define two thresholds T_1 and T_2, where $T_1 > T_2$.

2) Remove any record r from S and create a canopy centered at r.

3) For each other record r_i, compute cheap distance d from r to r_i.

4) If $d < T_1$, place r_i in r's canopy.

5) If $d < T_2$, remove r_i from S.

6) Repeat from step 2 until there are no more data points in the set S to cluster.

The flow chart of canopy clustering algorithm as shown in Fig. 1.

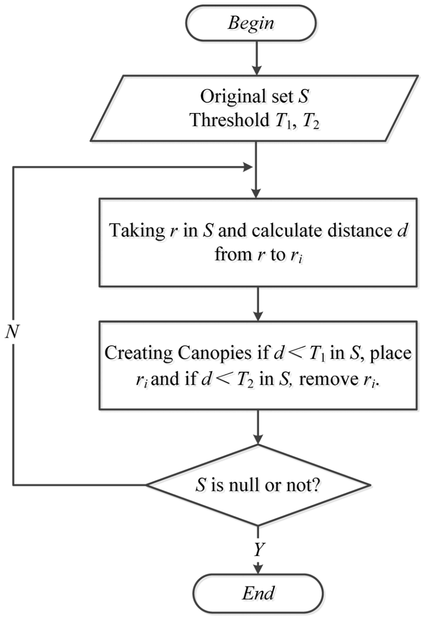

Begin
Original set S
Threshold T_1, T_2
Taking r in S and calculate distance d from r to r_i
Creating Canopies if $d < T_1$ in S, place r_i and if $d < T_2$ in S, remove r_i.
S is null or not?
N
Y
End

Fig. 1. Flow chart of canopy clustering algorithm.

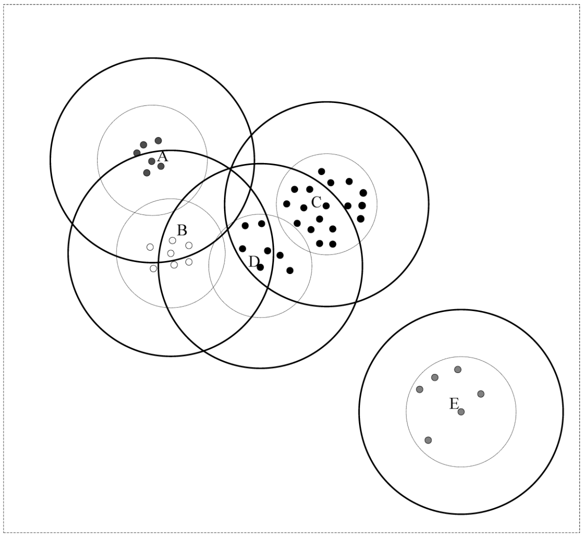


Fig. 2. Canopy division of data sets.

In Fig. 2, the cluster center vectors are randomly selected, and then a canopy based on data vector A is created. The canopy includes all the data vectors in the outer ring (solid circle), and the data vector of the inner ring (dotted circle) which is no longer a central vector of the candidate list.

B. FCM Clustering Algorithm

FCM is a method of clustering which allows one piece of data to belong to two or more clusters. This method (Dunn [12] and Bezdek [13]) is frequently used in pattern recognition and is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^{m}\|x_i - c_j\|^2,\ 1 \le m \le \infty \tag{1}$$

where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the ith of d-dimensional measured data, c_j is the d-dimension center of the cluster, and ||*|| is any norm expressing the similarity between any measured data and the center.

Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centers c_j [14] by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C}\left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|}\right)^{\frac{2}{m-1}}} \tag{2}$$

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} \cdot x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \tag{3}$$

This iteration will stop when $\max_{ij}\{|u_{ij}^{(k+1)} - u_{ij}^{(k)}|\} < \varepsilon$, where ε is a termination criterion between 0 and 1, whereas k are the iteration steps. This procedure converges to a local minimum or a saddle point of J_m [15]. The algorithm is composed of the following steps:

Algorithm 1. FCM clustering algorithm

```
1: Initialize $U=[u_{ij}]$ matrix, $U^{(0)}$
2: At $k$-step: calculate the centers vectors $C^{(k)}=[c_j]$
   with $U^{(k)}$
       $c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} \cdot x_i}{\sum_{i=1}^{N} u_{ij}^{m}}$
3: Update $U^{(k)}$, $U^{(k+1)}$
       $u_{ij} = \frac{1}{\sum_{k=1}^{C}\left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|}\right)^{\frac{2}{m-1}}}$
4: If $\left\|U^{(k+1)} - U^{(k)}\right\| < \varepsilon$ then Stop; otherwise
return to step 2.
```

III. PARALLELIZATION OF CANOPY-FCM CLUSTERING ALGORITHM

According to the concurrent thought of the canopy-FCM algorithm and the programming model of MapReduce, the MapReduce is performed on this algorithm under Hadoop platform. The parallelization of this algorithm can be divided into two phases: the MapReduce of canopy algorithm and FCM algorithm. The detailed framework is as shown in Fig. 3.

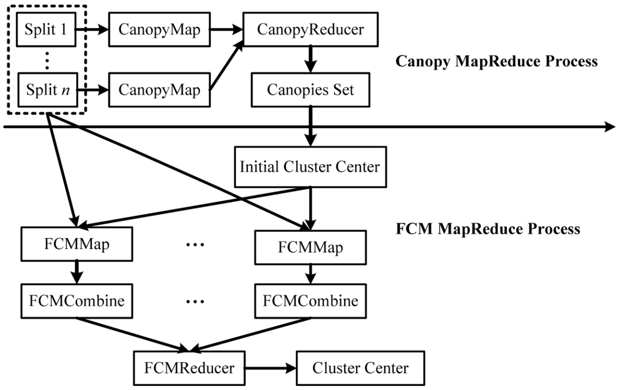


Fig. 3. Hybrid canopy-FCM clustering algorithm with MapReduce framework. FCM: fuzzy c-means.

A. The MapReduce of Canopy Clustering Algorithm

The objective of canopy algorithm is to provide FCM algorithm with initial cluster centers. The design of parallel canopy algorithm is comprised of two processes: Map and Reduce. The primary task of the Map process is to process the data objects in current node using the idea of parallel canopy algorithm. For each Map node, the threshold values (*T*1 and *T*2) are required to be input to further form the canopy center. As the canopy center generated in the Map process is partial, the number of generated canopy centers is far less than data objects on Map nodes. Thus, the number of the canopy centers on all Map nodes is significantly smaller than that of original datasets [16]. The threshold value (*T*3, *T*4) in the Reduce process must be slightly greater than that of the Map process. To reduce the times of running the Reduce process, only one Reduce node is required for merging the set of canopy centers obtained on all Map nodes.

Moreover, a threshold filter can be set to delete the canopy center containing a smaller number of data objects than filter. Owing to the data objects on each Map node in Map process being merely part of original data objects, presenting limitation, there is no need to set the value of filter in current stage. In contrast, the Reduce process merges all sets of global canopy centers, so a reasonable filter value needs to be set to delete the canopy center which contains fewer data objects than the filter. This process can remove outlier points and is in agreement with the specific and parallel flows of Map and Reduce processes in the canopy algorithm.

B. The MapReduce of FCM Clustering Algorithm

From Fig. 3, the MapReduce of FCM algorithm comprises FCMMap (Map process), FCMCombine (Combine process) and FCMReduce (Reduce process). The Map calculates the membership degree of the data objects contained in Map nodes on initial cluster center; the Combine process calculates the Sum0 of the products for the data objects on Map nodes corresponding to the membership degree, and Sum1 for the *m*th power of all membership degrees corresponding to the center; according to the Combine process, the Reduce process is conducted to obtain Sum0, Sum1 and Sum0/Sum1 of all Map nodes, namely, a new cluster center. Since FCM algorithm cannot achieve a favorable cluster center until many iterations are run, the above mentioned processes of Map, Combine and Reduce need to be iterated. Consequently, the final cluster center is obtained through the iterations. However, the data objects in the data sets are not classified; a new MapReduce process is required for classification. The MapReduce of FCM algorithm includes five processes: Map, Combine, Reduce, iterations and the classification of data objects.

1) The Design of FCMMap

The Map process calculates the membership degree of the data objects in the current node on cluster center. Its input refers to the input of the data objects on current node and initial cluster center or the cluster center acquired in the iteration of last time; its output indicates the <key, value> form, where key is the index number of the cluster center, and value represents the membership degree of all data objects contained in Map nodes on the cluster center corresponding to the key as well as data objects. The data structure of <key, value> is represented as (center, (point, weight)).

2) The Design of FCMCombine

As the data produced in the Map process are saved in a local disk, the Combine process is operated on Map nodes after the Map process so as to reduce the communication cost of nodes and computing quantity of the Reduce process. The aim is to merge the data in current Map nodes.

The goal of the Combine process is to compute the above mentioned Sum0 and Sum1; the Combine process shows an output as <key, value>, where key represents the index of cluster center, while the data structure of the value is expressed as (Sum0, Sum1).

The output in the Map process represents the dataset comprised of the index of the cluster center, data objects, and membership degree. The dataset is applied as the input of the Combine process, with its input form to integrate the value of data set obtained in the Map process. The values containing the same key values are then combined into a data set <key, values>. Thus, one Combine process is to run <key, values> in fact. In the operation, each key value corresponds to the index of one cluster center, one Sum0 and Sum1 can be obtained after running one Combine process.

3) The Design of FCMReduce

Given both Map and Combine processes merely calculate the data objects on current Map nodes, the Reduce process was therefore designed. The objective of the Reduce process is to merge the data objects on all Map nodes and derive cluster center.

After the Combine process, the output of each Map node has become reduced. The output form is {center, Sum0, Sum1}, where center refers to the index of cluster center; Sum0 represents the sum for the product of *m*th power of all membership degrees corresponding to center in current Map node and the data objects corresponding to membership degrees. Sum1 is the sum of the *m*th power of all membership degrees corresponding to center in Map node.

All the same values for key and value are combined into a dataset when the data in the Map process is collected in the Reduce process. Thus, one Reduce process is the operation of acquiring the data set constituted by {Sum0, Sum1} set with the same center value. The operation process includes: 1) solving the sums of Sum0 and Sum1 using the data set formed by {Sum0, Sum1} set, respectively, 2) Sum0/Sum1 is set as the cluster center whose index is center, and 3) output of cluster center. All cluster centers are obtained as the whole Reduce process is finished.

As the running times of Reduce process equals the number of the cluster centers and one Reduce process can obtain one cluster center, and this process exerts little influence on other Reduce results. Hence, multiple Reduce nodes can be set to perform the Reduce process.

4) The Design of the Iterative Process

One important step of the FCM clustering algorithm is to judge whether or not the produced cluster centers are converged. Therefore, the aforementioned Map, Combine and Reduce processes require multiple iterations in the FCM clustering algorithm until cluster centers converge.

Converging of cluster centers is judged by comparing the cluster centers obtained last (or initial cluster centers) with those obtained most recently. If the variations of all cluster centers are lower than the given threshold value, the cluster centers considered to be converged, and thus the algorithm is stopped. Otherwise, the cluster centers obtained last are replaced by those acquired at this time to start the new round of the MapReduce process. To avoid the great time consumption caused by the excessive iterative times, the maximum iteration times should be set for the algorithm. When the iteration times in MapReduce process is greater than the maximum iteration times, this algorithm is expected to be stopped.

5) The Classification Design of Data Objects

The aforementioned iterative process can merely obtain the final cluster center, but fails to group all the data objects to the categories they belong to. To solve this problem of a MapReduce process, the Map process is required.

Generally speaking, data objects are grouped to the category which is corresponding to the cluster center with the maximum membership degree. Since calculating the membership degree of the data objects on cluster centers is an independent process, the MapReduce process can be designed. In this process, merely the Map stage is concluded, and the specific operation flow is given as follows: 1) the final cluster centers are obtained; 2) the membership degrees of the dada objects contained in Map nodes on all cluster centers are calculated; 3) the index of the cluster center corresponding to the maximum membership degree is obtained and regarded as the category of the corresponding data objects; 4) step 2 is repeated until all the data objects in Map nodes are processed.

IV. RESULTS AND DISCUSSION

A. Experimental Environment

In this section, we provide experiments to evaluate the performance of our MapReduce implementation of hybrid canopy-FCM clustering algorithm.

Hardware condition: a Hadoop cluster deployed on 5 PCs with 2.2 GHz dual-core CPU, 4 G RAM and 500 G hard disk. Each PC is a Hadoop node, and thus we have 5 nodes.

Software condition: Eclipse, Ubuntu12.10, HBase0.94, Hadoop0.20.3.

B. Experimental Results and Analysis

1) Data Sets Testing

In this paper, we use the following two kinds of data sets to test the algorithm performance which are obtained from the University of California at Irvine (UCI) machine learning library.

a) Car Evaluation data set: The Car Evaluation database contains examples with the structural information removed, categorical variable for assessment on car with four class values (unacc, unacceptable; acc, acceptable; good, good; vgood, very good), and six input attributes are respectively (vhigh, high, med, low), maint (vhigh, high, med, low), doors (2, 3, 4, 5, more), persons (2, 4, more), lug_boot (small, med, big), and safety (low, med, high).

b) Iris data set: The data set consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals.

2) Result Analysis

The precision ratios and recall ratios [17, 18] were used to evaluate the quality of clustering results in this paper.

- The Car Evaluation data set (1,728 records) is divided into three testsets: testset1 contains portion1; testset2 contains portion1 and portion2; and testset3 contains portion1, portion2 and portion3. Hadoop cluster deployed on 5 PCs, when the hybrid canopy-FCM algorithm and traditional FCM algorithms are all parallelized by the MapReduce based method, the precision ratios and recall ratios of the two algorithms under the data sets of different scales are shown in Table 1.

  As a general rule, the cluster number is required for the FCM algorithm. Since there are four types of data sets utilized in the research, *k* is set as 4;

  The thresholds of canopy-FCM algorithm are set as $T1$=0.23, $T2$=0.12, $T3$=1.5$T1$, $T4$=2$T1$, *filter* =10.
- Since the Iris data set is small (150 records), the algorithm is merely performed on the whole data set.

  As mentioned above, the cluster number should be given for FCM algorithm. Since there are three types of data sets utilized in the research, *k* is set as 3;

  The thresholds of canopy-FCM algorithm are set as $T1$=0.15, $T2$=0.08, $T3$=1.5$T1$, $T4$=2$T1$, *filter* =10.

The precision and recall of the two algorithms and clustering results are shown in Table 2.

By comparing Figs. 4 and 5, on the Iris data set, the intersection of all kinds of points in hybrid canopy-FCM algorithm is less than FCM algorithm, hybrid canopy-FCM algorithm clustering results are better than the FCM algorithm.

The test results of the aforementioned two data sets,

Table 1. Comparison on precision and recall of the two algorithms under the Car Evaluation data set

	FCM	Hybrid canopy-FCM
Testset1 precision	0.74	0.79
Testset1 recall	0.65	0.74
Testset2 precision	0.68	0.77
Testset2 recall	0.60	0.69
Testset3 precision	0.72	0.78
Testset3 recall	0.64	0.73

FCM: fuzzy c-means.

Table 2. Comparison on precision and recall of the two algorithms under the Iris data set

	FCM	Hybrid canopy-FCM
Precision	0.78	0.85
Recall	0.72	0.79

FCM: fuzzy c-means.

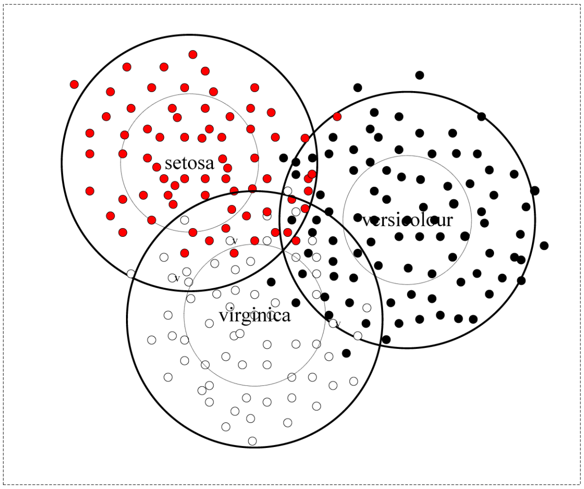


Fig. 4. Clustering results of the Iris data set using fuzzy c-means (FCM) algorithm.

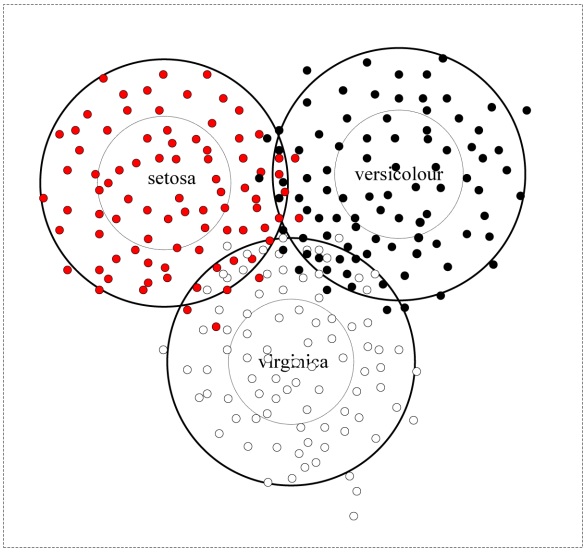


Fig. 5. Clustering results of the iris data set using hybrid canopy-FCM algorithm. FCM: fuzzy c-means.

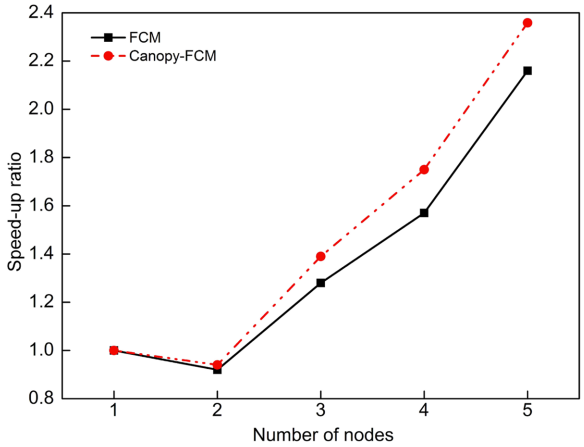


Fig. 6. Speed-up ratio with different number of nodes in Hadoop cluster. FCM: fuzzy c-means.

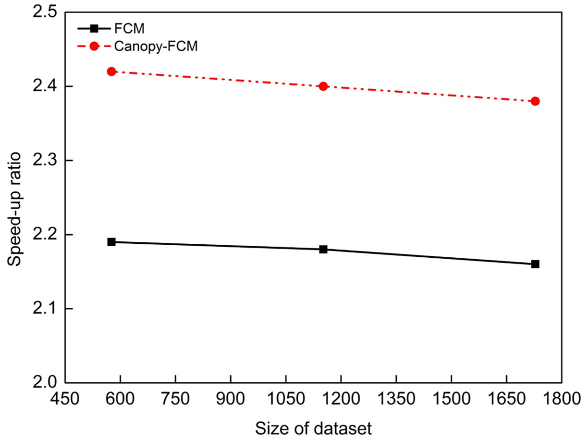


Fig. 7. Speed-up ratio with different size of dataset in Hadoop cluster. FCM: fuzzy c-means.

the hybrid algorithm is superior to the FCM algorithm because of the higher precision and recall ratios obtained during shorter operation time. The canopy algorithm can rapidly obtain favorable initial cluster centers, and thus accelerate the convergence rate.

C. Speed-up Ratio Analysis

Fig. 6 provides the speedup performance of various numbers of training instances as the number of nodes increases, where speedup is a popular measurement of parallel algorithm defined as the ratio of execution time of sequential algorithm to that of the parallel algorithm with specific numbers of processors.

As shown in Fig. 6, the same number of the nodes in Hadoop cloud platform, the execution efficiency of canopy-FCM algorithm is higher than that of the FCM algorithm. When there are two nodes in the cloud platform, the speed-up ratios of the both algorithms are smaller than that of a single PC, which is attributed to the fact that the data communication of nodes is time consuming during the executive processes of the two algorithms. Meanwhile, with the increase of the number of nodes, the execution time of the two algorithms is shortened gradually. This indicates that the communication time between data nodes is shortened, which gives rise to the gradually accelerated operation speed of the algorithms.

To compare the speed-up ratio with different size of dataset, the above mentioned dataset is used and divided into three portions of testset1, testset2, and testset3. Hadoop cluster deployed on 5 PCs, the speed-up ratio of two algorithms are shown in Fig. 7.

Fig. 7 illustrates the results, from which we have the following observations. First, the larger the dataset, the more time consuming the process. Second, the execution time of our MapReduce based algorithm is much less than that of the original FCM algorithm as the size of dataset increases. Therefore, the proposed method outperforms the traditional version.

V. CONCLUSION

Based on the canopy algorithm processed by the MapReduce process, the FCM algorithm is further explored in this work. The experimental results demonstrate the cluster centers obtained by utilizing the canopy algorithm can improve the clustering efficiency and effectiveness of the FCM algorithm.

However, with rising memory computing capacities, frameworks such as Spark have presented enormous advantages. Spark can solve two kinds of problems, including iterative calculation and interactive calculation that the Hadoop framework fails to effectively deal with.

The main principle of the Spark framework is the resilient distributed datasets (RDD), that is, all the calculated data and intermediate results are stored in distributed memory. The transformation of the operator sequences from RDD to RDD keeps occurring in the RDD space. Another important design of Spark is lazy evaluation, that is, there exists no actual calculation but continuous recording of the metadata. The number of metadata continues to increase until action operators appear, which can process all accumulated operators immediately. While operating the Spark program, developers merely need to submit tasks without concern over the allocation and scheduling of tasks and the transfer of the calculation results between nodes.

In the future, the hybrid canopy-FCM clustering algorithm is expected to be further improved by adopting the Spark technology so as to accelerate the execution speed of this algorithm.

ACKNOWLEDGMENTS

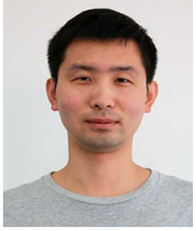
This work was supported by Humanities and Social Science Youth Fund Project of Ministry of Education of China (No. 13YJCZH028).

REFERENCES

1. J. Nayak, B. Naik, and H. S. Behera, “Fuzzy C-means (FCM) clustering algorithm: a decade review from 2000 to 2014,” in *Computational Intelligence in Data Mining-Volume* 2, New Delhi: Springer India, pp. 133-149, 2015.
2. D. B. Hassen, H. Taleb, I. B. Yaacoub, and N. Mnif, “Classification of chest lesions with using fuzzy c-means algorithm and support vector machines,” in *International Joint Conference SOCO’13-CISIS’13-ICEUTE’13*, Cham: Springer International Publishing, pp. 319-328, 2014.
3. N. Bharill and A. Tiwari, “Handling big data with fuzzy based classification approach,” in *Advance Trends in Soft Computing*, Cham: Springer International Publishing, pp. 219-227, 2014.
4. S. R. Kannan, S. Ramathilagam, A. Sathya, and R. Pandiyarajan, “Effective fuzzy c-means based kernel function in segmenting medical images,” *Computers in Biology and Medicine*, vol. 40, no. 6, pp. 572-579, 2010.
5. X. Wang, Y. Wang, and L. Wang, “Improving fuzzy c-means clustering based on feature-weight learning,” *Pattern Recognition Letters*, vol. 25, no. 10, pp. 1123-1132, 2004.
6. R. M. Esteves and C. Rong, “Using Mahout for clustering Wikipedia’s latest articles: a comparison between k-means and fuzzy c-means in the cloud,” in *Proceedings of IEEE 3rd International Conference on Cloud Computing Technology and Science (CloudCom)*, Athens, Greece, 2011, pp. 565-569.
7. Q. Yu and Y. Dai, “Parallel fuzzy C-means algorithm based on MapReduce,” *Computer Engineering and Applications*, vol. 49, no. 14, pp. 133-137, 2013.
8. J. Q. Zhang, X. W. Zheng, and H. P. Wu, “Research on fuzzy C-means clustering algorithm parallel,” *Microcomputer & Its Applications*, vol. 29, no. 23, pp. 8-18, 2010.
9. D. Irfan, X. Xu, S. Deng, and Z. He, “S-Canopy: a feature-based clustering algorithm for supplier categorization,” in *Proceedings of IEEE 4th Conference on Industrial Electronics and Applications*, Xi’an, China, 2009, pp. 677-681.
10. A. McCallum, K. Nigam, and L. H. Ungar, “Efficient clustering of high-dimensional data sets with application to reference matching,” in *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, MA, 2000, pp. 169-178.
11. Y. Li, “Research on parallelization of clustering algorithm based on MapReduce,” Sun Yat-Sen University, Guangzhou, China, 2010.
12. J. C. Dunn, “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters,” *Journal of Cybernetics*, vol. 3, no. 3, pp. 32-57, 1973.
13. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algoritms*, New York: Plenum Press, 1981.
14. J. V. de Oliveira and W. Pedrycz, *Advances in Fuzzy Clustering and Its Applications*, New York: Wiley, 2007.
15. M. Meloun, and J. Militky, *Kompendium statistického zpracování dat*, Praha: Academia, 2006.
16. E. H. Ruspini, “Numerical methods for fuzzy clustering,” *Information Sciences*, vol. 2, no. 3, pp. 319-350, 1970.
17. A. Al-Dallal and R. S. Abdulwahab, “Achieving high recall and precision with HTLM documents: an innovation approach in information retrieval,” in *Proceedings of the World Congress on Engineering*, London, 2011, pp. 1883-1888.
18. J. Euzenat, “Semantic precision and recall for ontology alignment evaluation,” in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India, 2007, pp. 348-353.

Wei Dai

Wei Dai is Associate Professor in School of Economics and Management at Hubei Polytechnic University, China. His current research interests include intelligence computing, cloud computing and information management.



Changjun Yu

Changjun Yu is a postgraduate student in School of Computer Science and Technology at Wuhan University of Technology, China. His research areas include parallel computing and computer simulation.

Zilong Jiang

Zilong Jiang is a Ph.D. candidate in School of Computer Science and Technology at Wuhan University of Technology, China. His research focuses on machine learning, internet advertising and HPC.