International Journal of Innovative Research in Advanced Engineering (IJIRAE)
Volume 1 Issue 1 (March 2014)

Extended Fuzzy C-Means with Random Sampling Techniques for Clustering Large Data

Dhanesh Kothari
Department of Information Technology
Sri Venkateswara College of Engineering
dhaneshkanak@gmail.com

S. Thavasi Narayanan
Department of Information Technology
Sri Venkateswara College of Engineering
thavasinarayanan@gmail.com

K. Kiruthika Devi
Department of Information Technology
Sri Venkateswara College of Engineering
kiruthika@svce.ac.in

Abstract – Big data are any data that you cannot load into your computer's primary memory. Clustering is a primary task in pattern recognition and data mining. We need algorithms that scale well with the data size. The former implementation, literal Fuzzy C-Means is linear or serialized. FCM algorithm attempts to partition a finite collection of n elements into collection of c fuzzy clusters. So, given a finite set of data, this algorithm returns a list of c cluster centers. However it doesn't scale well and slows down with increase in the size of data and is thus impractical and sometimes undesirable. In this paper, we propose an extended version of fuzzy c-means clustering algorithm by means of various random sampling techniques to study which method scales well for large or very large data.

Keywords - Big data, very large data, fuzzy c-means (FCM) clustering, sampling, probability

I. INTRODUCTION

CLUSTERING [1] is a form of data analysis in which data are separated into groups such that data with similar properties are in same group. Clustering is used in pattern recognition, compression, image processing[9] and many other fields as mentioned in references [6] and [10]. IoT or Internet of Things, personal computing has produced an incredible amount of data ranging from few terabytes to Exabyte of data. These big data are still unloadable in the modern day computers. There are many approaches applied for clustering the big data. One technique is distributed clustering, where the data is distributed to various systems and the clustering is done independently on each system. Then, the average cluster centers are evaluated based on the cluster centers from all systems. This method is however quite expensive and result data may not be accurate.

There are three main types of partitions: crisp, fuzzy and possibilistic. Crisp or hard partitioning group data into mutually disjoint subsets of points and the partition element, uki = 1 if oi is labelled k, else 0. Fuzzy partitions are more flexible than crisp portioning in that each object can belong to more than one cluster group. This is particularly useful in identify the online behavior of person for advertisements. The set of all fuzzy c-partitions is

$$M_1 fcn = \{ (U \in R^{\dagger}(c \times n) \mid u_1 ij \in [0,1] \forall j, i \; @ 0 < \Sigma_1 (j=1)^{\dagger} n \cong u_1 ij < n, \forall i; \; \Sigma_1 (i=1)^{\dagger} c \cong u_1 ij = 1, \forall j) \} \quad (1)$$

Each column of the fuzzy partition U must sum to 1, thus ensuring that every object has unit total membership in a partition. Many algorithms have been proposed to cluster the big data but only few of them use fuzzy partitions. One such is fuzzy c-means clustering. However, its literal schemes simply cluster the entire dataset. In contrast, extended clustering [2] schemes apply clustering algorithm to subset (or representative) of this dataset and non-iteratively extend the sample data result to the full dataset. One such algorithm is random sampled extended fuzzy c-means algorithm. In this paper, we compare three different sampling techniques to compute fuzzy partitions of the vector data: simple random sampling, systematic random sampling and multistage random sampling. Section II describes the simple random sampling based extended fuzzy c-means, and Section III proposes systematic and multistage sampling for extending fuzzy c-means algorithm. Section IV presents the results. Section V contains our conclusions and some ideas for future enhancements.

II. SIMPLE RANDOM SAMPLED FUZZY C-MEANS ALGORITHM

The FCM algorithm is defined as the optimization of the squared-error distortion

$$J_m(U,V) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} \left\| x_j - v_i \right\|_A^2 \quad (2)$$

Where U is (c x n) partition matrix, V={v1,v2,…., vc} is the set of c clusters centers in Rd , m>1 is the fuzzification constant, and ||.||A is any inner product A-induced norm, i.e., $||x||A=\sqrt{x^T Ax}$. We use here Euclidean norm (A=I) in our example. Algorithm 1 outlines the steps for the literal FCM algorithm. We initialize the algorithm by choosing c objects randomly from the dataset as initial cluster centers. The most commonly used sampling technique to address big data is simple random sampling [4].

In this method, each record in the full set has equal probability of being selected into the sample set, i.e. 1/N, where N is the total number of records. The FCM algorithm then can be applied onto this sample data to get membership matrix and cluster centers. We can then extend the V to complete dataset. This can be called as rseFCM. rseFCM is shown in algorithm 2.

The step 3 is an important step. In this we extend the result of the FCM algorithm on sample dataset to the complete dataset. It is a non-iterative step as it is run only once in the end of extended FCM algorithm.

Algorithm 1: LFCM

Input: X, c, m
Output: U, V
Initialize V
while $max_{1 \le k \le c} \left\{ \left\| v_{k,new} - v_{k,old} \right\|^2 \right\} > \epsilon$ **do**

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \quad , \forall i, j \tag{3}$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m \cdot x_j}{\sum_{j=1}^{N} u_{ij}^m} \quad , \forall i \tag{4}$$

Algorithm 2: rseFCM to approximately minimize $J_m(U, V)$

Input: X, c, m
Output: U, V
1 Sample n_s objects from X without replacement, denoted X_s
2 U_s, V = LFCM(X_s, c, m)
3 Extend the partition (U_s, V) to X, $\forall x_i \notin X_s$, using Eq. (3), giving (U, V)

III. MULTISTAGE SAMPLING AND SYSTEMATIC RANDOM SAMPLING

The are other sampling methods[8] other than commonly used simple random sampling method. Here, we introduce multistage sampling and systematic random sampling techniques. ultistage sampling is a complex form of cluster sampling. It involves dividing the population into groups or strata based on the best feature[3]. With the multistage sample, there is equal chance of selecting each unit from within a particular group. In some cases, several levels of group selection may be applied before the final sample elements are reached. It should be noted that multistage and stratified sampling are different from each other as in multistage, selected clusters are studied. The multistage sampling is given in algorithm 3.

Algorithm 3: Multistage sampling based extended FCM algorithm

Input: X, c, m
Output: X_s, U, V
1 Organize the sampling process into stages where the unit of analysis is systematically grouped.
2 Select a sampling technique for each stage.
3 Systematically apply the sampling technique to each stage until the unit of analysis has been selected.
4 Execute rseFCM steps 2 and 3.

Systematic random sampling is a type of probability sampling technique. With the systematic random sample, there is an equal chance of selecting each unit from within the population when creating the sample. Here, rather than selecting referring to random number tables to select the cases (as in simple random sampling) in your sample, you select the units directly from the sample frame. Systematic sampling algorithm is given in Algorithm 4.

Algorithm 4: Systematic Random Sampling based extended FCM algorithm

Input: X, c, m
Output: X_s, U, V
1 Calculate the sampling interval, k

$$k = \frac{N}{n}$$

where, N is the population size and n is the sample size
2 Select a random start between 1 and sampling interval
3 Repeatedly add sampling interval to select subsequent record.
4 Execute rseFCM steps 2 and 3.

IV. EXPERIMENTS

We performed one set of experiment. In the experiment we compare the performance of the big data FCM algorithm on the real data or data with ground truth (Collected data) as well as on synthetic dataset. For FCM, we initialize V by randomly choosing c objects as the initial cluster centers. We fix the value of $\epsilon = 10^{-3}$ and fuzzifier m= 1.7. All code were written in JAVA eclipse environment.

A. Evaluation Criter a

We judge the performance of the VL FCM algorithm using run-time criteria. It is computed for 20 independent runs. We present statistical comparisons of the algorithms' performance over the 20 experiments.

Speed-up or Run-Time: This criterion represents an actual run-time comparison. When the LFCM is available, speedup is defined as t_{full}/t_{sample}, where these values are in milliseconds to compute the cluster centers V for data and the membership matrix, U.

B. Performance on loadable data

We compared the performance of VL FCM algorithm on the following datasets[7]. *BIRCH*-Sets (n=100000, c= 12, d=2): These data are composed of 100000 2-d vectors, with a visual preferred grouping into 15 clusters. See table I for result.

Forest-Data (n=581012, c=7, d=54): These data are composed of cartographic variables that are obtained from *United States Geological Survey* and *United State Forest Service*. There are 10 quantifiable variables, and 40 binary soil types, and one cover type. See table II for result.

We take 10% as sample size and run-time results (seconds) of various sampling technique based FCM algorithm is shown in Fig.1. It is clear from the results that systematic sampling shows far better scalability than simple random sampling. Multistage sampling can be applied only on datasets with more than two variables. It has a run-time almost near to simple random sampling.

C. Performance on unloadable data

For, our last experiment, we demonstrate the VL vector data FCM algorithm on an unloadable dataset. The performance of the VL algorithms can be measured by how well they find the clusters and measuring the run-time in milliseconds. The SUSY (Super Symmetry) data are composed of 5 million objects. In order to show how the VL algorithms could be used to process this data in PC we tested at sample size 0.01 % and 0.001% of data. Table III shows the results of this experiment. Unexpectedly, multistage sampling was faster than systematic sampling. Simple random sampling was a very expensive method among the three sampling techniques tested on the data inputs.

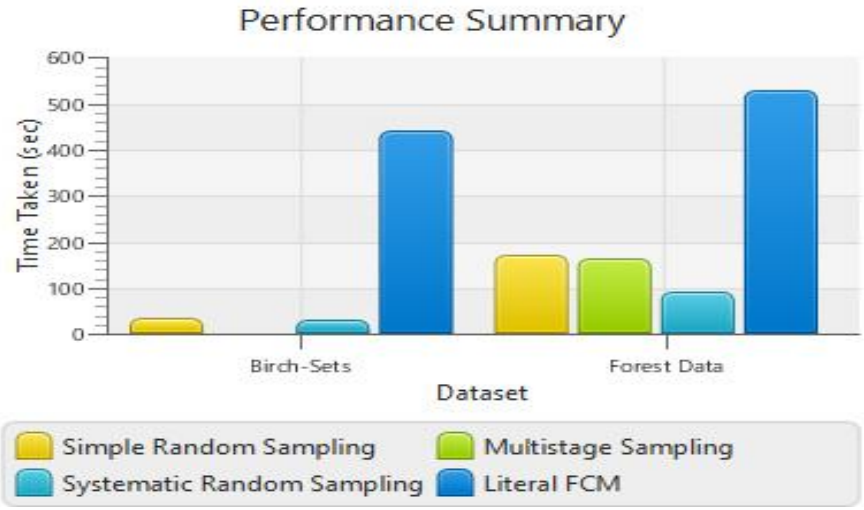

Fig. 1 Performance bar chart of the various sampling techniques on Birch-Sets and Forest Data

TABLE I
Results of VL FCM Algorithm on Birch Data (Avg. Of 20 Runs)
10% sample size

Simple Random Sampling		Systematic Random Sampling		Literal FCM	
Run-Time (seconds)	No. of Iterations	Run-Time (seconds)	No. of Iterations	Run-Time (seconds)	No. of Iterations
33.287	63	29.475	57	442.135	83

TABLE II
Results of VL FCM Algorithm on Forest Data (Avg. Of 20 Runs)
10% sample size

Simple Random Sampling		Systematic Random Sampling		Multistage Sampling		Literal FCM	
Run-Time (seconds)	No. of Iterations	Run-Time (seconds)	No. of Iterations	Run-Time (seconds)	No. of Iterations	Run-Time (seconds)	No. of Iterations
170.269	76	90.822	49	163.169	68	528.644	39

TABLE III
Results of VL Data Algorithm on SUSY Unloadable Data

	Sample Size			
	0.001%		0.01%	
Sampling Technique	Run-time (Sec)	No. Of Iterations	Run-time (Sec)	No. of Iterations
Multistage Sampling	37.050	22	43.526	21
Simple Random Sampling	460.043	43	2086.614	99
Systematic Random Sampling	64.015	36	372.497	76

V. DISCUSSION AND CONCLUSIONS

As this paper shows, there are many ways to sample the big data for VL FCM algorithm. Systematic random sampling had the best performance on large but loadable dataset. Multistage was better in case of very large or unloadable dataset. Systematic and multistage show better scalability compared to traditionally used simple random sampling for extending fuzzy c-means algorithm. In the future, we will continue to develop and investigate scalable solutions for VL fuzzy clustering.

REFERENCES

[1] Timothy C. Havens, James C. Bezdek, Christopher Leckie, Lawrence O. Hall, and Marimuthu Palaniswami, "*Fuzzy c-Means Algorithms for Very Large Data*," in IEEE Transactions On Fuzzy Systems, Vol. 20, No. 6, December 2012.
[2] Richard J. Hathway and James C. Bezdek, "Extending fuzzy and probabilistic clustering to very large data sets", Computational Statistics and Data Analysis, vol. 51, 2006
[3] V. Schwammle and O. Jensen, "*A simple and fast method to determine the parameters for fuzzy c-means cluster analysis*," Bioinformatics, vol. 26, no. 22, pp. 2841–2848, 2010.
[4] John F. Kolen and Tim Hutcheson, "Reducing the Time Complexity of the Fuzzy c-means Algorithm", IEEE Transactions On Fuzzy Systems, Vol. 10, No. 2, April 2002.
[5] James C. Bezdek, Robert L. Cannon, Jitendra V. Dave, "Efficient Implementation of the fuzzy c-Means Clustering Algorithms", Vol. PAMI-8, No. 2, March,1986.
[6] D. Dembele and P. Kastner, "Fuzzy c-means method for clustering microarray data," *Bioinformatics*, vol. 19, pp. 973–980, 2003.
[7] http://archive.ics.uci.edu/ml/datasets/
[8] http://dissertation.laerd.com/sampling-strategy.php
[9] N. Pal and J. Bezdek, "Complexity reduction for "large image" processing," *IEEE Trans. Syst., Man, Cybern.*, vol. 32, no. 5, pp. 598–611, Oct.2002.
[10] J. Bezdek and R. Hathaway, "Convergence of alternating optmization," *Nueral, Parallel, Sci. Comput.*, vol. 11, no. 4, pp. 351–368, Dec. 2003.