International Journal of Engineering Science Invention (IJESI)
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org ||Volume 7 Issue 6 Ver II || June 2018 || PP 01-05

Moderm Techniques used for Big Data Clustering:A Review

Sharon Susan Jacob[1], Prof. R.Vijayakumar[2]
[1](School of Computer Sciences, Mahatma Gandhi University, India)
[2](School of Computer Sciences, Mahatma Gandhi University, India)
Corresponding Auther: Sharon Susan Jacob

Abstract: *Huge quantity of data is generated in each second in the modern digital era. It became almost impossible for the conventional data analytics frameworks to handle this ever-exploding big data. Storage spaces for this huge amount of data is also running out. Big data clustering algorithms help up to huge extent where dimensionality of enormous big data is reduced to a great extent without any significant loss of relevant information. Big data clustering algorithms need not only be robust in working but also need to be compliant for distributed and parallel processing. This paper is reviewing the most modern and robust big data clustering algorithms. Four algorithms are chosen, one from each category of Partitioning based, Density based, Grid based, Model-based big data clustering algorithms. Algorithms considered for in depth review are partition based big data clustering in a distributed Apache spark framework, Density Based Spatial Clustering of Applications with Noise, grid based OptiGrid clustering algorithm and model based Self-organizing Map clustering algorithm. Detailed algorithm working and shortcomings of each of these algorithms are discussed in detail.*

Keywords *-Big data clustering, Data partition, DBSCAN, OptiGrid, SOM*

Date of Submission: 25-05-2018 Date of acceptance: 11-06-2018

I. INTRODUCTION

In this digital era, massive quantities of data are being generated each second, which shows the importance of big data analytics in the modern world. Data originating from smartphones, desktops, sensors, digital documents etc. come under the purview of big data. Big data is characterized by its Volume, Variety, Velocity, Veracity, Validity and Volatility. Many times, it becomes imperative to find a structure in a collection of unlabeled data and this process is called Clustering, which can be formally defined with respect to big data as the process of organizing the data into clusters whose members are similar in some way while data in different clusters are dissimilar. Big data in raw unprocessed form takes up huge storage spaces and may contain irrelevant information. This deficiency of big data can be solved up to a great extent through big data clustering where big data is clustered into a more compact form but still holding good informative content. There are many big data clustering methods available now. This paper is aimed at conducting a very subjective review of the most modern and proven big data clustering methods and shed some light to its shortcomings. Most of the modern day big data clustering algorithms are enhancements of conventional, original yet proven methods. Therefore, it is a good practice to first brush up such proven techniques for big data clustering.

Different category of Big data clustering methods are :

1. Partitioning based: Divides the whole big data to clusters using partitions subjected to two rules. First, each cluster must have at least one candidate and Second, Each candidate must occupy only one cluster. Example: K–Modes, K-Means, K-Medoids, PAM, CLARA, CLARANS, FCM
2. Density based: Density and connectivity of data candidates are used in dividing them to clusters. Examples: DBSCAN, OPTICS, DBCLASD, DENCLUE etc.
3. Grid based: Data space is first divided into a grid space. Grid collects statistical information of the big data as it passes through the grid in each iteration. Clustering is then performed on the grid rather than the data itself. Examples: Wave-Cluster, STING, CLIQUE, OptiGrid etc.
4. Model-based: Optimizes the match between the big data and mathematical model. This results in automatic generation of number of clusters based on the statistical aspects with noise in scope. Examples: EM, COBWEB, CLASSIT, SOMs etc.

Whole idea of this paper is to touch open at least one most prominent paper coming under each category discussed above. Section II of this paper discusses about the most widely used, discussed, relevant modern and proven big data clustering algorithms, its working, application and demerits. Partition based big data clustering in a distributed Apache spark framework, Density Based Spatial Clustering of Applications with Noise (DBSCAN), grid based OptiGrid clustering algorithm and model based Self-organizing Map clustering

algorithm (SOM) are discussed in detail. Even though the papers referred here are application specific, we are subjecting them to application independent algorithm analysis.

II. LITERATURE SURVEY

[1] is such a hybrid approach wherein, a partition based clustering algorithm, which makes use of fuzzy based scalable method is implemented over an Apache spark for real time responsive big data handling. Originally developed at UC Berkeley in2009, Apache spark is a powerful open source-processing engine built for big data analysis. It is proven to be one of the most commercially used big data analysis framework posing a serious competition to Hadoop MapReduce. The clustering algorithm used is shown in Fig. 1.

Input: n-sized data $X = \{x_1, x_2, \ldots x_n\}$; set of cluster centers $V = \{v_1, v_2, \ldots v_c\}$; number of clusters c;

Output: Updated set of cluster centers V', Membership Information I'

1: **Initialize** V by randomly choosing some data points from X.
2: **Compute** membership information.
I' = X.map(V).reduceByKey()
3: **Compute** Cluster Centers
for $< j, < sum\_d_jx, sum\_d_j >>$ in I' **do**

$V_j' = (sum\_d_jx)/(sum\_d_j)$

end for
4: Calculate change in Cluster Center values.
$e' = ||V' - V||$
5: **If** $e' < e$: Stop; **Else**: V = V'. Go to 2;
6: **Return** V' , I'

Figure 1. Scalable version of literal fuzzy c-means (slfcm) used for big data clustering.

As shown in step 2 of the above algorithm, membership function is computed by combining the map function and reduce by key function applied to the same data set. The Map function finds many key-value pairs having the same key value. Spark takes care of performing the same operations on all the key-value pairs. To add all these values we use reduceByKey function. The distributed and parallel processing of the above-mentioned algorithm in Apache spark is shown in Fig. 2.

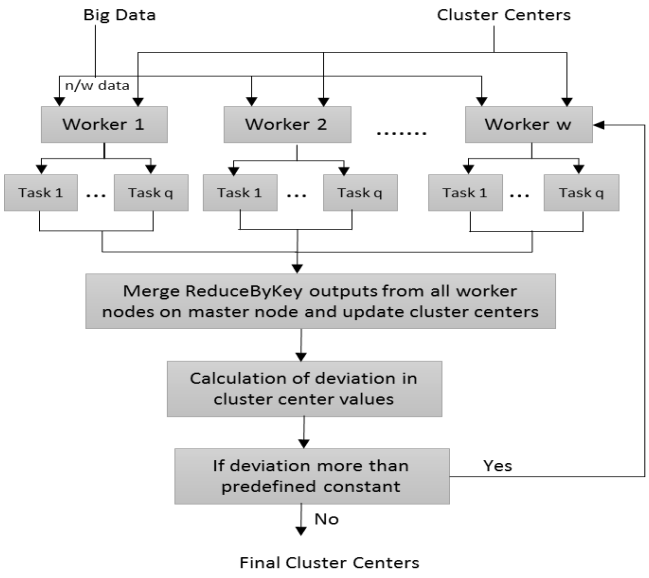

Figure 2. Implementing clustering in a distributed manner, using apache spark framework

To conclude, fuzzy based clustering algorithm attempts to partition the data points into set of c fuzzy clusters such that an objective function of a dissimilarity measure is minimized. This is then implemented on distributed apache spark framework to minimize the execution run time. This is a very valid and commercially proven technique, this can be understood by the fact that this technology is now used by technology giants like IT giants such as Yahoo, Baidu and Tencent. Even though this method is commercially proven, it still have some common demerits like:

1. Sensitivity to initial configuration
2. Lack of robustness
3. Sometimes it leads to empty clusters

[2] is a very novel attempt to detect anomalies like rule violations, accidents, unusual driving and other suspicious actions in a traffic surveillance video using DBSCAN, a density based big data clustering algorithm. Conventional attempts in anomaly detection make use of probabilistic methods, which resulted in reduced accuracy. The whole process followed is summarized in Fig. 3.

Inputs:
Trajectory Vector T = [x_1 y_1, x_2 y_2,.....x_n y_n], Feature Vector F ={x_i, y_i, θ_i} which represents the position, angle of moving vehicle in i^{th} frame, θ_i= arctan(y_i/x_i),
Distance of Neighborhood (Eps)
Minimum number of points (MinPts)
Output:
Detect anomalies
Algorithm:
1: Begin with random selection of p, p->{x,y} ∈ T
2: Calculate density reachable points from p
Eps(p) = {q ∈ T|distance(p,q) <= Eps}
3: Form cluster, if
p ∈ Eps(q)
|Eps(q)| >= Minpts
4: Move to next point in T, if
p ∈ Eps(q)
|Eps(q)| < Minpts
5:Goto Step 2, until all the points in T is visited

Figure 3. DBSCAN algorithm used for finding anomalies in traffic surveillance video in an unsupervised manner

DBSCAN method does clustering based on the density of trajectory data points, to separate high and low density areas of data space. DBSCAN can handle clusters of different shapes and size in advance. The algorithm also works well in presence of noise and scalability, as it is evident from its name. It is scalable based on two essential parameters Eps (neighbors of point p) and MinPts(minimum no. of points). Here the most important step is step number 2 and 3 where a point p is checked to be density reachable from q based on the equations in these steps. The parameters Eps(p) and MinPts are the decision makers in categorizing a point into any one out of the three groups:

1. A Core point is a point if sum of points is more than specified MinPts.
2. A border point is lesser in number than MinPts, however they are neighbors of core point.
3. A point do not belong to any of the cluster is named as noise point or outlier point

Some of the commons demerits of this method are:

1. Not completely deterministic
2. Quality depends on the distance measure. For high dimensional data, this metric is useless due to the 'curse of dimensionality'
3. If the difference in data density is huge, DBSCAN possible fails
4. Without a good idea about data and scale, choosing correct distance threshold is difficult

The problem of DBSCAN in dealing with high dimensional data is solved in [3] where grid based OptiGrid clustering is used for anomaly based intrusion detection. Overview of the algorithm is shown in Fig. 4.

1) C = {c1}, first cluster center (Initially, k = 1 and cluster c1 includes all training data), q, min cut score be parameters user gives
2) BEST_CUT (A set of pair of real numbers)← Ø
3) For each d_l of c_i($1 \le l \le d$, $1 \le i \le k$), call best_local_cuts(d_l) in real d-dimensional space. As described above, the function calculates a histogram, and returns a pair of a cutting point and its score if such a point exists. The existence of the pair means that dimension d_l can be a candidate for making partition of cluster c_i.
4) For d_l($1 \le l \le d$), add pairs whose score exceeds min_cut_score to BEST_CUT, that is, If best_local_cuts(d_l) > min_cut_score then BEST_CUT = BEST_CUT U best_local_cuts(d_l)
5) Delete scores from BEST_CUT except highest q scores.
6) Generate multidimensional grids by partitioning the hyper space by axis-parallel hyperplanes from the q cutting points in BEST_CUT
7) Search a multidimensional grid which includes at least one instance. Regard the grid as a cluster whose member consists of instances within the grid. Let the clusters be c_m, c_{m+1}, …, c_{m+l} and then $C = c_m \cup c_{m+1} \ldots \cup c_{m+l}$
8) For each $c_i \in C$, execute the processes from 2 to 6 recursively.
9) If the algorithm can not find any cutting plane in every cluster, the algorithm stops

Figure 4. OptiGrid based big data clustering algorithm for intrusion detection

It was proven that the curse of dimensionality is solved using OptiGrid. The common demerits of grid based OptiGrid technology is listed below:

1. Clusters may be found in any subspace means in a single or overlapped subspace
2. The clusters may overlap each other

A very good candidate among model based big data clustering algorithms is Self-organizing Map clustering algorithm (SOM). [4] Explains SOM based clustering algorithm that can be used in MapReduce using Spark platform. SOM is a well-known unsupervised learning algorithm widely used for cluster analysis of Bigdata. SOM is a type of Artificial Neural Network (ANN) which creates a low dimensional discretized form of input data in training space.

Variables

1. s is the current iteration
2. λ is the iteration limit
3. t is the index of the target input data vector in the input data set D
4. D(t) is a target input data vector
5. v is the index of the node in the map
6. W_v is the current weight vector of node v
7. u is the index of the best matching unit (BMU) in the map
8. θ(u, v, s) is a restraint due to distance from BMU, usually called the neighborhood function, and
9. α(s) is a learning restraint due to iteration progress.

Algorithm

1. Randomize the map's nodes' weight vectors
2. Grab an input vector D(t)
3. Traverse each node in the map
   - Use the Euclidean distance formula to find the similarity between the input vector and the map's node's weight vector
   - Track the node that produces the smallest distance (this node is the best matching unit, BMU)
4. Update the nodes in the neighborhood of the BMU (including the BMU itself) by pulling them closer to the input vector

   $W_v(s+1)=W_v(s)+ \theta(u,v,s). \alpha(s).(D(t)-W_v(s))$
5. Increase s and repeat from step 2 while $s < \lambda$

A variant algorithm:

1. Randomize the map's nodes' weight vectors
2. Traverse each input vector in the input data set
   - Traverse each node in the map
     - ✓ Use the Euclidean distance formula to find the similarity between the input vector and the map's node's weight vector
     - ✓ Track the node that produces the smallest distance (this node is the best matching unit, BMU)
   - Update the nodes in the neighborhood of the BMU (including the BMU itself) by pulling them closer to the input vector

   $W_v(s+1)=W_v(s)+ \theta(u,v,s). \alpha(s).(D(t)-W_v(s))$
3. Increase s and repeat from step 2 while $s < \lambda$

Figure 5. Algorithm used in Model based SOM algorithm in big data clustering

Just like any ANN, SOMs also operate in two modes, training and mapping. A self-organizing map comprises of components called neurons. Fig.5 shows the overall working of the SOM. SOM can be implemented in Map reduce or Spark apache framework for time critical big data applications.
Even though model based big data clustering algorithms like SOM utilizes the most modern ANN like architecture, it is prone to following demerits:

1. Higher execution time
2. One model working well for a specific data set may prove to be erroneous for a different data set

III. CONCLUSION

This paper made an attempt to review the most widely used big data clustering algorithms in a very subjective manner. Four algorithms are chosen, one from each category of Partitioning based, Density based, Grid based, Model-based big data clustering algorithms. These algorithms are the best among the currently available clustering algorithms. Algorithm working of each of them is explained in detail and their shortcomings are also discussed. Future work shall be the designing of a new big data clustering algorithm which can alleviate the demerits of the existing methods discussed in this paper.

REFERENCES

[1]. Neha Bharill, Aruna Tiwari, Aayushi Malviya, "Fuzzy Based Scalable Clustering Algorithms for Handling Big data using Apache Spark", IEEE Transactions on Big Data (Volume: 2, Issue: 4, Dec. 1 2016), DOI: 10.1109/TBDATA.2016.2622288
[2]. R. Ranjith, J. Joshan Athanesious, V. Vaidehi, "Anomaly Detection using DBSCAN Clustering Technique for Traffic Video Surveillance", Seventh International Conference onAdvanced Computing (ICoAC), 2015, DOI: 10.1109/ICoAC.2015.7562795
[3]. Moriteru Ishida, Hiroki Takakura, Yasuo Okabe,"High-Performance Intrusion Detection Using OptiGrid Clustering and Grid-basedLabelling", 11th International Symposium on Applications and the Internet (SAINT), 2011 IEEE/IPSJ, DOI: 10.1109/SAINT.2011.12
[4]. Tugdual Sarazin, Hanane Azzag, Mustapha Lebbah, " SOM Clustering Using Spark-MapReduce", IEEE InternationalParallel & Distributed Processing Symposium Workshops (IPDPSW), 2014, DOI: 10.1109/IPDPSW.2014.192

Sharon Susan Jacob "Moderm Techniques used for Big Data Clustering:A Review "International Journal of Engineering Science Invention (IJESI), vol. 07, no. 06, 2018, pp 01-05