Volume 2, Issue 1, January 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Efficient Clustering Algorithm for Large Data Set

N.Sudhakar Reddy
Professor of CSE,
S.V.College of Engineering,
Tirupati, India

KVN Sunitha
Professor in CSE &HOD,
GNITS
Hyderabad, India

***Abstract*— The concept-drift phenomenon is used for outlier detection or data labeling, which plays a vital role in detection of outlier. But in that there is a disadvantage which is of reclustering when drift occurred. In this connection two scanning operations are required, one for the drifting and another for the reclustering of sliding window. It is necessary to investigate the principal of clustering to design efficient algorithms to minimize the disk I/O and minimizing the number of scanning operations. In this paper, to overcome the problems of scanning operations and also it is extended to the categorical data where as in literature the leader algorithm for the numerical domain and sequence of data set. The main objective of the idea of clustering the outliers based on the leader algorithm instead of reclustering the entire sliding window /data set by calculating the threshold with the average method and maximal resemblance of leaders. This is more efficient than using reclustering the sliding window.**

***Keywords*— reclustering, incremental clustering, sliding window, resemblance, threshold**

I. INTRODUCTION

Clustering is one of the most important and challenging task in data mining [1]-[2]. It works based on "minimizing intra distance and maximizing inter distance" principle. But in general clusters are not static. Because data objects behaviour changes[3]-[8] over time hence there is a change in constructed clusters after certain period of time. The complete data set is divided into sliding windows [9] based on time slots or predefined number of slides. For certain period of time a part of the data set is called sliding window. This sliding window method is also used in sampling based clustering for large data sets to minimize the disk I/O operations. Here we use incremental clustering method [10]. In this method, the unlabeled data point is assigned to existing cluster or to form as new cluster depending on the maximal resemblance and threshold values [11]-[13]. These are mainly specifying the similarities among the new data points and existing clusters. The process is repeated until the outlier's size becomes zero.

In the current process, the entire data set D containing *n* data points with *d* features is stored in disk/secondary storage. The slide size is based on the availability of main memory if it is p then the number of slides $(\frac{n}{p})$. Each of the slides is transferred to the main memory and does the clustering by first choosing one represented data point per cluster, and then use incremental clustering method to form clusters.

In leader algorithm[14], representations of the clusters are called leaders. Space complexity is O (*Ld*), to store *L* leaders of dimension *d.*

Here we propose a method to extend the leader algorithm as it is simple incremental clustering algorithm with a time complexity of O (*n*d). Most of the other have discussed

leaders algorithm for numerical data sets and also sequence data sets. In the proposed method, the leader algorithm is extended to categorical data[15]-[16] with Our-NIR[17] and Pour-NIR[18] importance values.

The rest of the paper is organized as follows. In section II discussed incremental clustering, section III node importance representatives, threshold discussed, in section IV discussed labeling and outlier detection with results and finally concluded in section V.

II. INCREMENTAL CLUSTERING: LEADER ALGORITHM

Clustering of very large databases [19]-[21] is useful for both searching and browsing. An algorithm for incremental clustering is introduced in this paper. The complexity and cost analysis of the algorithm together with an investigation of its expected behaviour are presented. Though theoretical implementations shown that the algorithm achieves cost effectiveness and generates statistically valid clusters that are compatible to reclustering. The theoretical implementations evidence shows that the algorithm creates an effective and efficient retrieval environment.

Leader is an incremental algorithm in which *L* leaders each representing a cluster is generated using a Maximal resemblance and threshold values based on Our-NIR or Pour-NIR values. In this method, after finding *L* leaders using the leader algorithm, Our-NIR or Pour-NIR values

updated with in each sliding window. Thus the leader algorithm is shown in fig.1.

Algorithm 1

Leader_Adjust ($C^{[tc,t-1]}$, Outliers)
{
Initialize the leaders to clusters, add those to leader list.
***For** all the Outliers S*
{
Node importance($C^{[tc,t-1]}$) and find average threshold $a_i = \lambda_i \backslash q$
***For** all clusters $C_i^{[tc,t-1]}$ in $C^{[tc,t-1]}$*
{
Calculate Resemblance R(S, $C_i^{[tc,t-1]}$)
}
Find the maximal resemblance $C_m^{[tc,t-1]}$
***If** (R(S, $C_m^{[tc,t-1]}$)> a_m then*
S is assigned C'^t_m
***Else** S is set as a new leader*
}
}

Fig.1

III. NODE IMPORTANCE REPRESENTATIVES

In this section node importance representatives (NIR) [12]-[13] are considered which are suitable for categorical data sets. The implementation of Leaders algorithm based on Our-NIR and POur-NIR and this is series of work for the clustering of categorical data.

A. Threshold value

Initially the threshold values can be chosen depending on the maximum and the minimum Euclidean distance values between the data points of a class in case of supervised learning. Threshold value should be chosen properly depending on the NIR values of clusters in the unsupervised learning. If the threshold value is too small then a large number of clusters are generated and if the threshold value is too large then very few clusters are generated. This problem can be overcome by the following proposed method.

B. Proposed Method

Leader is an incremental algorithm in which L leaders each representing clusters are generated using an average threshold value. Now select a data point from an outlier and calculate resemblance, threshold for L leaders, compare resemblance of data point with threshold value of leaders, here two ways first way is if resemblance is less than threshold value then that data point is added to that leader otherwise make that outlier data point as a new leader and increment leader count. Update the leader NIR values after adding. This process is repeated until the outlier becomes an empty based on time stamp. Maximal resemblance and average threshold value is also discussed in the following two subsections.

Algorithm 2

Leaders(*D(data set)with sliding windows $S^1, S^2 \ldots S^n$*)
t=1
Initialize $C^{[t-1]}=\Phi$ i.e no clusters.
***For** every sliding window S^t do*
***If** ($C^{[tc,t-1]}=\Phi$) then Call initial clustering and Call Node Importance($C^{[tc,]}$)*
Else
OUTH=0
***For** every data point P_j of S^t do*
***For** all clusters $C_i^{[tc,t-1]}$ in $C^{[tc,t-1]}$ do*
Calculate Resemblance R(P_j, $C_i^{[tc,t-1]}$)
End for
Find the maximal resemblance $C_m^{[tc,t-1]}$
***If** (R(P_j, $C_m^{[tc,t-1]}$)> λ_m then*
P_j is assigned C'^t_m
***Else** P_j is an Outlier*
End if
End for
/*concept drift checking*/

Numdiffclusters=0
***For** all clusters $C_i^{[tc,t-1]}$ in $C^{[tc,t-1]}$ do*

If $\left| \frac{m_i^{[te,t-1]}}{k^{[te,t-1]}} - \frac{m_i^t}{k^{[te,t-1]}} \right| > \varepsilon$

then
Numdiffclusters = numdiffclusters+1
End if
End for
If $\frac{\#ofoutiers}{N} > \theta$ *or* $\frac{numdiffclusters}{k^{[te,t-1]}} > \eta$,
then {concept drift}
Call Leader_Adjust on Outlier of S^t
Else
{concept not drifts}
Add C^{nt} in to $C_i^{[tc,t-1]}$,
Update Our-NIR and Pour-NIR as $C_i^{[tc,t]}$,
End if
End if
t=t+1
End for

Fig.2

Algorithm 3

NodeImportance($C^{[tc]}$)
{
***For** all clusters $C_i^{[tc,]}$ in $C^{[tc]}$*
{
***For** all data points Pj in $C_i^{[tc,]}$*
{
Divide P_j into nodes $I_1, I_2 \ldots I_q$ and
*Calculate Node importance W (ci, N [i, r])= pi * d(N [i, r])*
Where pi=|N[i, r]|/m_i

and d(N [r])=| Σ P(N[y, r])^2 |/2
where p(N[y, r])= |N[y, r]|/ |ΣN[z,r]|
}
Calculate threshold

$$\lambda i = \sum_{j=1}^{q} max_r \left(W(C_i, A_j = I_{ir})\right)$$

}}

1) *Maximal Resemblance*:

All the weights associated with a single data point corresponding to the unique cluster forms the resemblance. This can be given with the equation.

$$R(P_j,C_i)= \sum_{r=1}^{q} W(Ci,N_{[i,\,r]}) \qquad \text{---------- (1)}$$

Here the Our-NIR or POur-NIR values of all the nodes with all the clusters are calculated and are placed in the table. A data point P_j of the new data slide resemblance $R(P_j,C_i)$ can be obtained by summing up the Our-NIR or POur-NIR of the cluster C_i. This just gives the measurement of the resemblance of the node with cluster. And now these measurements are used to find the maximal resemblance. i.e, if data point P_j has maximum resemblance R (P_j,C_x), towards a cluster C_x, then the data point is labelled to that cluster.

If any data point is not similar or has any resemblance to any of the cluster then that data point is considered to be an outlier. We even introduce the threshold to simplify the outlier detection [11]. With the threshold value the data points with small resemblance towards many clusters generated and as the outlier if the resemblance is less than the threshold. Data labeling or outlier detection can be found using the equation (2).

$$Label = \begin{cases} C_i^*, & if \max R(p_j,c_i) \geq \lambda_i, where \quad 1 \leq i \leq k, \\ outliers, & otherwise. \end{cases} \qquad \text{---------- (2)}$$

2) Average threshold

In this section, we introduce the decision function that is average threshold, which decides the quality of the cluster and the number of the clusters. Here we have to calculate the threshold (λ) for every cluster can be set identical, i.e., $\lambda_1=\lambda_2=\ldots=\lambda_n=\lambda$. Even then we have a problem to find the main λ(threshold) that can be find with comparing all the clusters. Hence an intermediate solution is chosen to identify the threshold (λ_i) the smallest resemblance value of the last clustering result is used as the new threshold for the new clustering. After data labeling we obtain clustering results which are compared to the clusters formed at the last clustering result which are base for the formation of the new clusters. This leads to the "Cluster Distribution Comparison" step. In case of concept drift with too many data points in outliers, to avoid reclustering by the average threshold *a* for every cluster and to merge all the outliers to exiting clusters or making those as leaders of new clusters. The following equation (3) is used to find average threshold value for every cluster.

$$a_i = \lambda_i \backslash q \qquad \text{----------(3)}$$

where

$$\lambda i = \sum_{j=1}^{q} max_r \left(W(C_i, A_j = I_{ir})\right)$$

IV. LABELING AND OUTLIER DETECTION USING AVERAGE THRESHOLD

The data point is identified as an outlier if it is outside the radius of all the data points in the resemblance methods. Therefore, if the data point is outside a cluster, but is very close to a point in that cluster, it will still be an outlier. However, this case might be frequent due to concept- drift or noise, As a result, detecting existing clusters as novel would be high. In order to solve this problem here we adapted the average threshold for detecting the outliers/labeling. The most important step in the detection of the drift in the concept starts at the data labeling. The concept formation from the raw data which is used for the decision making is to be perfect to produce proper clusters after the decision. Hence the formation of clustering with the incoming data points is an important step. Comparison of the incoming data point with the initial clusters generated with the previous data available gives rise to the new clusters.

If a data point P_j is the next incoming data point in the current sliding window, this data point is checked with the initial cluster C_i, for doing so the resemblance R (C_i, P_j) is measured, and the appropriate cluster is the cluster to which the data point has the maximum similarity or resemblance. Our-NIR or POur-NIR is used to measure the resemblance. Maximal Resemblance was discussed in III.B.1 section**.**

A. *Illustration of LOur-NIR (Leader Our-NIR)*

Consider the data set in fig.4 and perform initial clustering on first sliding window S^1 by using any clustering method then we get C_1^1 and C_2^1clustres as shown in fig.5. Then calculated Node Importance values using Our-NIR for these two clusters and shown in table 1. The threshold values are λ_1=1+0.5+0.33=1.83 and λ_2=0.33+0.5+1=1.83 with respect to these clusters.

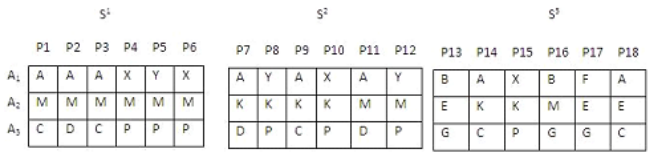

Fig. 4: Data set with 3 sliding windows of size 6.

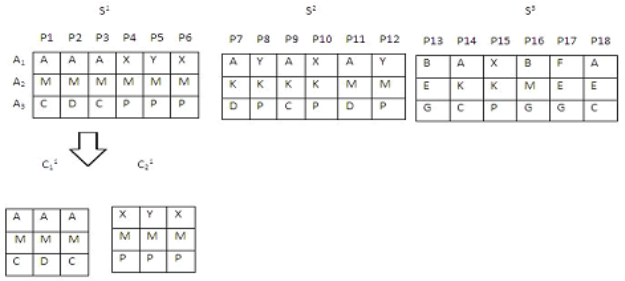

Fig.5: Clusters after the initial clustering is performed on S^1.

Now consider data points of S^2 one-by-one and find the Resemblance to C_1^2 and C_2^2 and compare with threshold values λ_1 and λ_2 respectively. The final results have shown in fig.6. In this case concept drift is occurred because of too many data points in outliers. Now this is time to do reclustering but here considered leader algorithm to merge outliers one after another by updating node importance values into clusters or make them as leaders then all these clusters also shown in fig.6. The leaders algorithm designed based on the average threshold instead of actual threshold.

C_1^1

Node	Importance
A1=A	1
A2=M	0
A3=C	0.66
A3=D	0.33

C_2^1

Node	Importance
A1=X	0.66
A1=Y	0.33
A2=M	0
A3=P	1

Table 1: Node importance values using Our-NIR

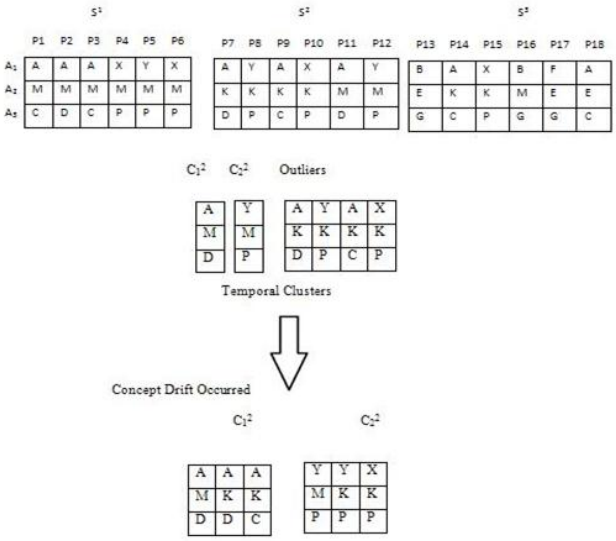

Fig.6: Clusters after Leader algorithm for sliding window S^2.

For the remaining sliding windows of data set apply the process as said above and those results shown in fig.7. In the previous sections of this paper explained leader algorithm with the Our-NIR representation. That same thing can be applied for the Pour-NIR representation.

V. CONCLUSIONS

In this paper, Our-NIR and Pour-NIR methods are considered to find node importance in every cluster and also published by us. In these methods reclustering process is needed in case of concept drift occurred, that is time consuming. This problem has been over come by the leader algorithm with average threshold value. It only applied for outliers. The future work is to use leader-sub leader algorithm which is hierarchical structured.

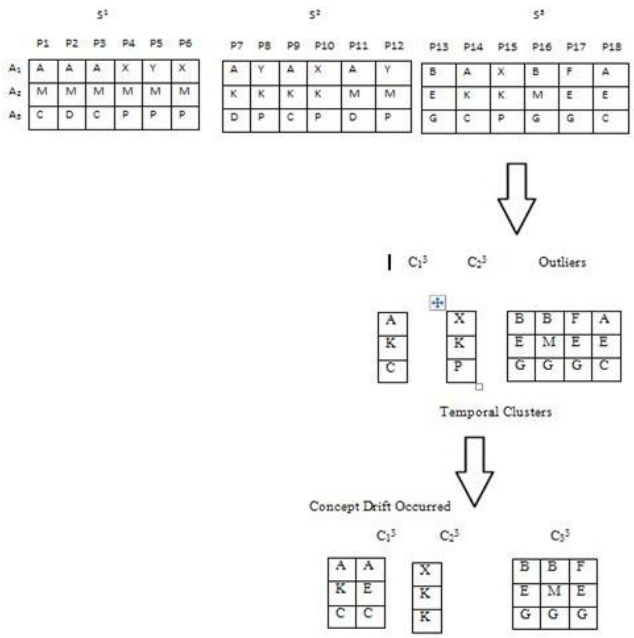

Fig.7 Clusters after leader algorithm for sliding window S^3.

ACKNOWLEDGMENT

We would like to thank Director of SIT, JNTUHyderabad, INDIA, for his support and encouragement. We would also like to thank our teacher Dr. Vinay Babu for his advice of clearing the doubts. Last but not the least we thank to our friends who helped us in doing this work.

REFERENCES

[1] C.C. Aggarwal, J.L. Wolf, P.S. Yu, C. Procopiuc, and J.S. Park, "Fast Algorithms for Projected Clustering," *Proc. ACM SIGMOD" 1999,* pp. 61-72.

[2] AK Jain MN Murthy and P J Flyn "Data Clustering: A Review," *ACM Computing Survey*, 1999.

[3] C. Aggarwal, J. Han, J. Wang, and P. Yu, "A Framework for Clustering Evolving Data Streams," *Proc. 29th Int'l Conf.Very Large Data Bases (VLDB)* ,2003.

[4] MM Gaber and PS Yu "Detection and Classification of Changes in Evolving Data Streams," *International .Journal .Information Technology and Decision Making*, v5 no 4, 2006.

[5] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-Based Clustering over an Evolving Data Stream with Noise," *Proc. Sixth SIAM Int'l Conf. Data Mining (SDM),* 2006.

[6] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary Clustering,"*Proc. ACM IGKDD" 2006,* pp. 554-560.

[7] O.Narsoui and C.Rojas,"Robust Clustering for Tracking Noisy Evolving Data Streams" SIAM I*nt. Conference Data Mining* , 2006.

[8] G Hulton and Spencer, "Mining Time-Changing Data Streams" *Proc. ACM SIGKDD,* 2001.

[9] C.E. Shannon, "A Mathematical Theory of Communication," Bell System Technical J., 1948.

[10] D.H. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering," Machine Learning, 1987.

[11] Viswanadha Raju, H.Venkateswara Reddy andN.Sudhakar Reddy," A Threshold for clustering Concept – Drifting Categorical Data", IEEE Computer Society, ICMLC 2011.

[12] H.-L. Chen, K.-T. Chuang and M.-S. Chen, "Labeling Unclustered Categorical Data into Clusters Based on the Important Attribute Values," *Proc. Fifth IEEE Int'l Conf. Data Mining (ICDM),* 2005.

[13] H.-L. Chen, M.-S. Chen, and S-U Chen Lin "Frame work for clustering Concept –Drifting categorical data," *IEEE Transaction Knowledge and Data Engineering v21 no 5* , 2009.

[14] S. Asharaf ,M. Narasimha Murty "An adaptive rough fuzzy single pass algorithm for clustering large data sets" Pattern Recognition 36 (2003) 3015 – 3018 December 2002.

[15] P. Andritsos, P. Tsaparas, R.J. Miller, and K.C. Sevcik, "Limbo: Scalable Clustering of Categorical Data," *Proc. Ninth Int'l Conf. Extending Database Technology (EDBT),* 2004.

[16] D. Barbará, Y. Li, and J. Couto, "Coolcat: An Entropy-Based Algorithm for Categorical Clustering," *Proc. ACM Int'l Conf. Information and Knowledge Management (CIKM),* 2002.

[17] S.Viswanadha Raju,H.Venkateswara Reddy and N.Sudhakar Reddy " Our-NIR:Node Importance Representation of Clustering Categorical Data ", IJCST June 2011.

[18] S.Viswanadha Raju, N.Sudhakar Reddy and H.Venkateswara Reddy " POur-NIR:Node Importance Representation of Clustering Categorical Data ", IJCSIS April 2011.

[19] V. Ganti, J. Gehrke, and R. Ramakrishnan, "CACTUS-Clustering Categorical Data Using ummaries," Proc. Fifth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining KDD), 1999.

[20] S. Guha, R. Rastogi, and K. Shim, "CURE: An Efficient Clustering Algorithm for Large atabases," Proc. ACM SIGMOD, 1998.

[21] Z. Huang, "Extensions to the K-Means Algorithm for Clustering Large Data Sets with ategorical Values," Data Mining and Knowledge Discovery, 1998.