A Novel Approach for Clustering Categorical Data Streams

J. Chandrika and K. R. Ananda Kumar

***Abstract*—Data stream mining research has gained importance in the recent years due to the generation of vast amount of data streams by many applications. Transactional data streams are characterized by high dimensionality and high cardinality. The transactional data streams arrive at a very high speed in an unbounded form. Clustering is an important core data mining activity that provides valuable insights into the data being processed. Clustering Transactional stream is a highly challenging activity as it is bound to single pass constraint as well as memory and CPU constraints. In this context an efficient algorithm is proposed to cluster the transactional data streams. The proposed algorithm accounts for the resource constraints. Extensive experimental analysis of the proposed algorithm on the real and synthetic data demonstrates the scalability and efficiency of the algorithm.**

***Index Terms*—Cluster histogram, data streams, data stream clustering, resource adaptation, sliding window model.**

I. Introduction

With the advances in data collection techniques and communication technology, many organizations receive vast amount of data at very high rates. Some of the examples that can be given in this context are – sensor networks, network traffic control and web usage monitoring. In all these examples there is high rate of data accumulation coupled with constant changes in the characteristics of data. Such type of data is called data streams. More formally, a data stream is an ordered sequence of data records that can be read only once or a small number of times. Retail chain transactions, web logs, credit card transaction flows and real time stock exchange data are some of the examples of data streams in real life scenario. The characteristics of data stream includes [1], [2]: huge volume of continuous data, possibly infinite, fast changing and requiring real time response, sequential single pass constraint, multi-level and multidimensional processing. Due to the unique characteristics of the data stream traditional algorithms have to be modified to handle the data streams.

A challenging problem in the area of data stream mining is to cluster the data stream [3]. Clustering can be considered the most important *unsupervised learning* problem; so, as every other problem of this kind, it deals with finding a *structure* in a collection of unlabeled data. Clustering refers to the process of grouping a collection of objects into classes or "clusters" such that objects within the same class are similar in a certain sense, and objects from different classes are dissimilar. Transactional data is a kind of special categorical data, and typical examples are market basket data, web usage data, customer profiles, patient symptoms records, and image features. Transactional data are generated by many applications from areas, such as retail industry, ecommerce, healthcare, Customer Relationship management and so forth. The volume of transactional data is usually large. Therefore, there are great demands for fast and yet high-quality algorithms for clustering transactional datasets. Fast and accurate clustering of transactional data has many potential applications in retail industry, e-commerce intelligence, etc. In the data stream settings the set of transactional data to be analyzed is application dependent. It can be the whole data stream or a part of it depending on the purpose of clustering. To gain a better understanding about this transactional data streams are clustered. Recently, more attention has been put on clustering categorical data [4], where records are made up of non-numerical attributes. When clustering transactional data streams it is very much important to account for the available resources like memory and CPU time.

The recent work in the area of clustering data streams focus only on clustering numerical data values. Even though a few algorithms are developed for clustering categorical data stream none of them make an effort to cluster transactional data stream in a resource efficient manner. As such an effort has been made to develop a novel algorithm that clusters the transactional data stream in a resource adaptive way.

Given a transactional data stream , the problem is to partition it into a number of clusters so that same type of transactions are in the same cluster, in other words intra cluster similarity is maximized and inter cluster similarity is minimized. This task is to be done in a resource adaptive manner. That is during the course of execution of the algorithm, if at any instance the algorithm runs out of memory space then the algorithm tries to continue its execution with a little compromise on the quality of output cluster. The mathematical formulation of the same can be given as

$$\text{Maximize Similarity} = \sum_{i=1}^{k}\sum_{j=1}^{n} S(Tij, rep(ck))$$

where *S* is the similarity measure, k denotes the number of clusters, n denotes the number of records in a given cluster, Tij denotes jth transaction in cluster *i*, rep (*ck*) represents the representative of cluster under consideration.

The rest of the paper is organized as follows Section II reviews the related work carried out in the context of clustering categorical streams. Section III explains the novel algorithm proposed. Section IV gives the details of experimental evaluation. Finally Section V concludes the

Manuscript received June 4, 2013; revised August 18, 2013.

J. Chandrika is with Dept. of Computer Science and Engineering, MCE, Hassan, India (e-mail; jc@mcehassan.ac.in).

K. R. Ananda Kumar is with Department of Computer Science, SJBIT, Bangalore (e-mail:kr_mega@hotmail.com).

paper with directions for the possible future work.

II. Related Work

The main challenge in clustering the data streams is to handle the evolving data. Aggarwal *et al*. [1] has proposed a framework for clustering the evolving streams. It splits the process of clustering into an online micro clustering component subjected to single pass constraint and an offline macro clustering component which is not constrained. The Clustream algorithm which is based on this framework deals with d dimensional numeric data. The summary information is stored as the sum of squared data values and sum of data values for each dimension together with sum of squares of timestamps and sum of timestamps for the data points in the cluster. This approach for storing summary information is suitable for handling numeric data and not categorical data. In the recent past a few algorithms are developed for handling categorical data. The Kmode algorithm [5] is the extension of Kmeans [6] which can handle only numerical data streams. The cluster is represented by a vector that has minimal distance to all the points. The distance is measured by number of common attributes shared by two data values, considering optimal weights for different attribute values. Since this algorithm requires the user to specify the number of clusters as input it produces K clusters regardless of underlying cluster structure. Wang *et al.* have proposed Largeitem [7] .It works in two phases: Allocation phase and refinement phase. In the allocation phase every transaction read will be moved to an existing cluster or a new cluster will be created whichever minimizes the cost. In refinement phase the original allocation made will be changed based on re computation of cost function. Because of two passes it is not suitable for data streams. ROCK [8] is based on the assumption that an attribute value, in addition to its occurrence, should be examined according to the number of other attribute values it exists with. ROCK works totally differently than k-modes not only because it is hierarchical but also due to the fact that it works on samples of the data. The main disadvantage of ROCK is that it employs sampling for scaling up to larger datasets and the results of clustering highly depend on it. CACTUS [9], an approach that entails the computation of summaries from the underlying data set (a concept similar to the one used in BIRCH [10]. CACTUS's summaries have to do with the support of pairs of attribute values from different attributes and the similarity of values that belong to the same attribute. The disadvantage of this algorithm is that, as the number of dimensions grows, when computing the summaries, each attribute value of a tuple is compared with every other value while the results can be kept in main memory only if the domain size of every attribute is very small. CLOPE [4] uses histogram as the summary structure for representing the cluster. The cost function used for clustering tries to minimize the height to width ratio of the histogram because a larger height implies more overlapping items and hence more intra cluster similarity. CLOPE is a cluster seeking algorithm that produces clusters based on user expectation of intra cluster similarity controlled by a user input called repulsion factor. Choosing a proper value for the repulsion factor is a challenging task. CLOPE requires multiple scans of the data, where the number of iteration depends on the desired level of intra-cluster similarity. This violates the one-pass requirement. Furthermore, CLOPE requires multiple evaluation of the clustering criterion for each record, an expensive operation when the size of the stream is massive. INCLUS [11] is an algorithm that clusters transactional data streams using two variants of sliding window model one is equal width model and the other is elastic model. This algorithm uses the similarity measure as given by equation (4) of section 2. Most of these algorithms try to produce the clusters taking into account the available computational requirements like memory and CPU time and the required resources as such the scalability of these algorithms is questionable. However some of the recent works focus on resource adaptation and quality assurance. Two such algorithms are proposed for finding frequent itemsets in data streams [12], [13]. RVFKM [14] is a resource aware very fast Kmeans algorithm that is applicable to numeric data streams in ubiquitous environment using the algorithm output granularity approach that is the algorithm controls the amount of output generated so that it can be handled with available resources. The concept of resource awareness is incorporated for processing the data streams generated by wireless sensor network by designing a resource aware framework for sun SPOT sensor node [15]. An online adaptive clustering algorithm called ERA Cluster is developed in Java. It is an online clustering algorithm which uses a threshold value to reduce the data generated by the sensor node so that it can be processed offline later. The algorithm is capable of adapting itself to the change in the battery level, remaining memory and CPU utilization. The algorithm uses Manhattan distance rather than the normally used Euclidian distance. Input adaptation using load shedding and data synopsis creation using wavelets have been proposed in [16], The Ideas proposed by these resource adaptive algorithms lay the foundation for our work.

III. Proposed Algorithm

The algorithm is implemented in java. It uses a sliding window model for processing the elements of the data stream. In this model, the range of mining is confined to the elements contained in a window which will slide with time. The window always covers a certain number of most recent elements and the mining task focuses on these elements at any point.

The proposed algorithm is based on the Clustream framework proposed by Agarwal *et al.* It is composed of an online clustering component which is a single pass procedure that reads the transactions of the data stream as it arrives and forms the cluster snapshots. Unlike the Clustream framework that handles numeric data and stores summary as sum of squared errors the proposed algorithm uses transactional data and stores cluster snapshots are in terms of histograms. A histogram records the frequency of every distinct item in the sliding window in the non decreasing order of frequencies.

An offline component is used to output the clusters formed for a particular time period. The input parameters required by the algorithm are, the sliding window size sw, the similarity threshold λ , the frequency threshold φ, the memory threshold Lm and the maximum amount of available memory Memorymax. The similarity threshold is required to

determine the cluster into which the newly arriving transaction is to be moved. The frequency threshold is helpful in selecting a cluster representative. The memory threshold depicts the minimum amount of memory to be available at the given instance of time to ensure that possibility of algorithm running out of memory resource which may lead to the crash of the software due to non availability of memory resources is avoided.

In order to partition a data stream into clusters a quantitative measure to assess the degree of similarity between the two transactions is required. There are several similarity measures proposed for clustering transactional databases [4], [17]-[19]. The similarity measure used in the proposed algorithm is the one proposed by Li *et al*. [11]

$$S(Ti, Tj)=(ITi\backslash TjI)/\max(ITiI, ITjI) \quad (1)$$

Based on our earlier work [20] we try to identify the parameters in the algorithm that can be fine tuned in order to avoid the algorithm running out of memory. Such parameters are called as adaptation parameters. Among the input parameters, sliding window size SW, λ similarity threshold and frequency threshold φ are used as adaptation parameters. Sliding window size is a parameter that is directly proportional to the memory and the CPU resource. Because greater the value for the sliding window size more will be the number of transactions to be handled and higher will be the memory and processing time. Obviously whenever resource critical condition is detected during the course of execution the sliding window size is to be varied according to the following condition:

$$Rm=(\text{Memorymax}–\text{Um})/\text{Memorymax} \quad (2)$$

In (2), Rm denotes amount of remaining memory and Um denotes the amount of memory being utilized up to this point of time. When Rm - Lm <= 0, it depicts the memory critical condition that calls for adjusting the sliding window size according to the following equation:

$$sw=Rm\times sw \quad (3)$$

For example if initial window size is 10 and *Rm*=0.8 then the new window size will be 10×0.8 = 8. The size of the sliding window is changed only when Rm falls below Lm indicating a memory critical condition.

The algorithm works in three phases. First is the initialization phase that sets the initial window size according to the user defined input and forms the clustering for the initial sliding window. Next an adaptation phase is executed before the window slides. In this phase the adaptation parameters viz, sliding window size *sw* and the similarity threshold λ are changed based on the current resource usage. The resource accounted for is the memory utilized and memory available at a particular time instance (2). If it falls below minimum threshold value the input granularity is adjusted by reducing the window size (3).

Finally there is an offline macro clustering phase that will print the output result for the given query period. The detailed algorithm is outlined below:

Input: Sliding window size SW, λ similarity threshold and frequency threshold φ, Memorymax, Lm (Memory available cannot be lesser than this)
// online micro clustering component
While not the end of the stream do
Begin
//Initialization phase
Read the first transaction
Form a cluster
Repeat
While not the end of the sliding window do //on line micro clustering
Found=0 // indicates if the candidate cluster is found
Read the next transaction t
For each of the existing cluster Ci
Find the similarity St with respect to transaction t
If St >= λ
Found=1
insert t into *Ci*
Update the cluster representative and its histogram
break;
End If
End For
If not found form a new cluster.
End While
// Adaptation phase
Monitor the current memory usage
Rm = (Memorymax – Um) / Memorymax
If Rm < Lm then sw = Rm × sw
//end adaptation phase
Till the stream arrives

The space requirement of the algorithm is small as the algorithm represents the cluster with a histogram in the main memory, the space consumed will be O(dk) where k is the number of clusters and d is the dimension of the transaction. Since the algorithm is non iterative the processing time will be O(n) where n is the total number of transactions in the cluster.

IV. Empirical Analysis

The proposed algorithm uses the initial parameter settings as. λ =60% and φ=40%. The first perspective for analyzing the algorithm is to demonstrate how the processing is continued by resource adaptation that is the adjustment of the size of the input sliding window. The concept is demonstrated by the graph given in Fig. 1. It is evident from the graph that the algorithm terminates after executing for a while without the resource adaptation. The resource adaptation mechanism enables the continuation of execution by suitable adjustment of the input granularity. The sliding window size adjustment controls the number of input transactions considered for processing.

Another crucial aspect of analyzing an online technique is to measure the running time. The computational complexity can be better illustrated considering various threshold values. It is observed that the run time of the algorithm will take very small leaps for a larger increase in the threshold value as illustrated in the graph below (Fig. 2).

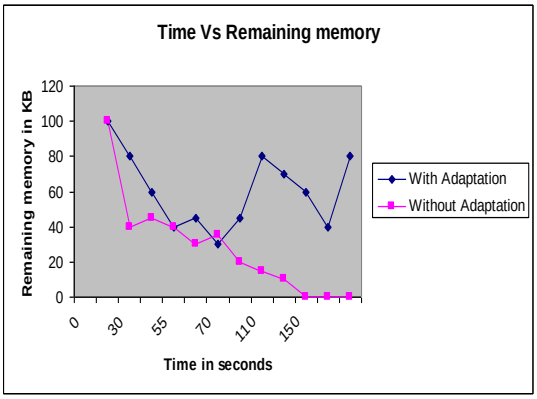

Fig. 1. The impact of adaptation on memory usage.

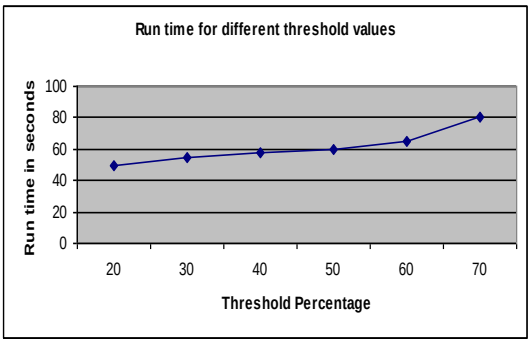

Fig. 2. The impact of threshold on execution time

Finally scalability tests are conducted by increasing the number of transactions keeping the dimensionality of data set constant. The empirical results generated are depicted by the graph below (Fig. 3).

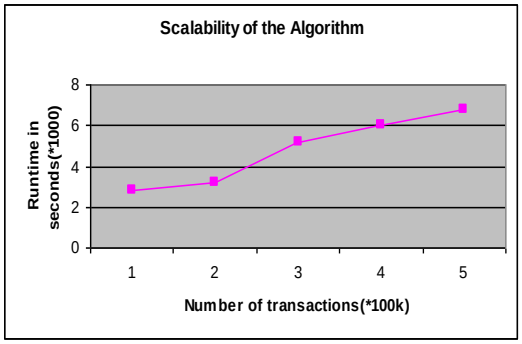

Fig. 3. Scalability of the algorithm for varying number of transactions.

The above graph illustrates the scalability of the algorithm as the number of transactions increased.

V. Conclusion and Future Work

In this study we have proposed an efficient scalable algorithm for clustering the transactional data streams in a resource constrained environment. An important aspect of the proposed algorithm is that it is structure seeking and it is not a structure imposing algorithm. Further the algorithm monitors the resource usage and always tries to keep the processing going on with an acceptable compromise on quality. The empirical evaluation demonstrates the efficiency of the algorithm. The proposed algorithm can be extended to cluster multiple data streams that originates from distributed locations.

References

[1] A. Charu, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proc. VLDB*, Berlin, Germany, September 2003.

[2] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. "Models and issues in data stream systems," in *Proc. ACM Symp*, PODS, June 2002.

[3] D. Barbara, "Requirements for clustering data streams," *SIGKDD Explorations*, 2002.

[4] Y. Yang, X. Guan, and J. You, "CLOPE: a fast and effective clustering algorithm for transactional data," in *Proc. SIGKDD*, Edmonton, Canada, July 2002.

[5] Z. Huang, "A fast clustering algorithm to cluster very large categorical data sets in data mining," in *Proc. SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 1997.

[6] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5-th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, vol. 1, pp. 281-297, 1967.

[7] K. Wang, C. Xu, and B. Liu, "Clustering transactions using large items," in *Proc. CIKM'99*, Kansas, Missouri, 1999.

[8] S. Guha, R. Rastogi, and K. Shim, "ROCK: a robust clustering algorithm for categorical attributes," in *Proc. ICDE'99*, Sydney, Australia, 1999.

[9] G. Venkatesh, G. Johannes, and R. Ramakrishnan, "CACTUS: clustering categorical data using summaries," in *Proc. 5th International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 73–83, 1999.

[10] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," in *Proc. SIGMOD*, Canada, June 1996.

[11] Y. Li, and R. P. Gopalan, "Clustering high dimensional sparse transactional data with constraints," in *Proc. IEEE International Conference on Granular Computing*, pp. 692 – 695, 2009.

[12] J. Chang and W. Lee, "Finding recent frequent itemsets adaptively over online data streams," in *Proc. ACM SIGKDD*, pp. 487–492, 2003.

[13] C. Giannella , J. W. Han, J. Pei , X. F. Yan, and P. S. Yu. Mining frequent patterns in data streams at multiple time granularities. [Online]. Available: www.cs.uiuc.edu/~hanj/pdf/fpstm03.pdf, 2003.

[14] C. Giannella, J. W. Han, J. Pei , X. F. Yan, P. S. Yu. Mining frequent patterns in data streams at multiple time granularities. [Online]. Available: www.cs.uiuc.edu/~hanj/pdf/fpstm03.pdf, 2003.

[15] G. M. Medhat and P. S. Yu, "A holistic approach for resource-aware adaptive data stream mining," *New Generation Computing*, Ohmsha, Ltd. and Springer-Verlag., 2006.

[16] W. Teng, M. Chen, and P. S. Yu, "Resource-aware mining with variable granularities in data streams," in *Proc. SIAM SDM*, 2004.

[17] L. O. Callaghan, A. Meyerson, R. Motwani, N. Mishra, and S. Guha. "Streaming data algorithms for high quality clustering," in *Proc. ICDE*, San Jose, CA, February 2002.

[18] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: theory and practice," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, June 2003.

[19] J. Yang, "Dynamic clustering of evolving streams with a single pass," in *Proc. 19th International Conference on Data Engineering, ICDE'03*.

[20] J. Chandrika and K. R. Ananda Kumar, "An adaptive framework for clustering data streams," in *Proc. ACC 2011*, Kochi, India, July 22-24, 2011.

J. Chandrika is a M.Tech graduate in computer science and engineering. Currently works as associate professor in the department of computer science and engineering at Malnad college of Engineering, Hassan, Karnataka. Currently she is doing her research work in the area data stream mining under visvesvaraya technological university, Belgaum. She has four international conference publications and four international journal publications and one national conference publication to her credit.

K. R. Ananda Kumar holds a doctoral degree in computer science and engineering. Currently works as Professor and Head of the department in the department of computer science and engineering, SJBIT Bangalore. He has a vast teaching experience of about 25 years. His research interest includes medical data mining; data stream mining, Artificial intelligence, intelligent agents and web mining. He is currently guiding five research scholars.He has fourteen international journal publications and two national journal to his credit.