Distance based Incremental Clustering for Mining Clusters of Arbitrary Shapes

Bidyut Kr. Patra[1,3,⋆], Ville Ollikainen[1], Raimo Launonen[1], Sukumar Nandi[2], and Korra Sathya Babu[3]

[1] VTT Technical Research Centre of Finland, Espoo, FI-02044 VTT, Finland
[2] Indian Institute of Technology Guwahati, Guwahati, Assam, PIN-781039, India
[3] National Institute of Technology Rourkela, Rourkela, Odisha, PIN-769 008, India
{ext-bidyut.patra,ville.ollikainen,raimo.launonen}@vtt.fi,
sukumar@iitg.ernet.in, ksathyababu@nitrkl.ac.in

Abstract. Clustering has been recognized as one of the important tasks in data mining. One important class of clustering is distance based method. To reduce the computational and storage burden of the classical clustering methods, many distance based hybrid clustering methods have been proposed. However, these methods are not suitable for cluster analysis in dynamic environment where underlying data distribution and subsequently clustering structures change over time. In this paper, we propose a distance based incremental clustering method, which can find arbitrary shaped clusters in fast changing dynamic scenarios. Our proposed method is based on recently proposed al-SL method, which can successfully be applied to large static datasets. In the incremental version of the al-SL (termed as *IncrementalSL*), we exploit important characteristics of al-SL method to handle frequent updates of patterns to the given dataset. The IncrementalSL method can produce exactly same clustering results as produced by the al-SL method. To show the effectiveness of the IncrementalSL in dynamically changing database, we experimented with one synthetic and one real world datasets.

Keywords: Incremental clustering, arbitrary shaped clusters, large datasets.

1 Introduction

Clustering has been recognized as one of the important tasks in data mining for data analysis. Clustering activity is to find groups of patterns, called clusters in a dataset $\mathcal{D}$, in such a way that patterns in a cluster are more similar to each other than the patterns in distinct clusters. Clustering methods are mainly divided into two categories based on the way they produce clusters *viz., partitional clustering* and *hierarchical clustering* methods. Partitional clustering methods create single clustering (also known as flat clustering) of a dataset. The partitional methods can be further categorized into two classes based on the criteria used *viz., distance based* and *density based*. Distance based methods optimize a global criteria based on the distance between patterns. Some of the

⋆ This work was carried out during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme. The research leading to these results has received funding from the European Union Seventh Framework Programme ($FP7/2007-2013$) under grant agreement 246016.

P. Maji et al. (Eds.): PReMI 2013, LNCS 8251, pp. 229–236, 2013.

popular distance based clustering methods are k-means, CLARA, CLARANS. Density based partitional clustering methods optimize local criteria based on density distribution of patterns. DBSCAN [1] is a popular example of density based partitional clustering method. Hierarchical clustering methods create a sequence of nested clusterings of a dataset. These methods can also be categorized into two classes *viz.,* distance based and density based. The single-link method [2] is a distance based hierarchical clustering method, which can find arbitrary shaped clusters in many applications.

Recent advances in storage, network and computer technology result in producing massive size of databases. To analyze clustering structures in these databases (datasets), several approaches have been proposed [3,4]. One prominent category is hybrid clustering in which a linear partitional clustering is combined with a hierarchical clustering method to reduce the computational complexity of the hierarchical method with compromising little clustering quality [5,6,7,8] or without compromising clustering quality [9,10]. However, this approach is not suitable for dynamic databases as clustering structure change over updations of the dataset. One cannot afford to apply the hybrid clustering repeatedly from scratch whenever there is an update in the database. One way to handle this problem is to adopt these clustering methods in dynamic scenarios such as purchasing patterns of a new product, financial transactions, early detection of epidemic, *etc*.

In this paper, we propose a distance based incremental clustering method which can find clusters of arbitrary shapes and sizes in fast changing databases. Our incremental clustering is based on scalable distance based clustering *al*-SL, which can find arbitrary shaped clusters in metric databases [10]. We restrict our search space for potential changes of clustering membership of patterns after inclusion of new points in the dataset using leaders clustering and metric space properties. Experimental results demonstrate the efficiency of the proposed incremental clustering (termed as IncrementalSL) in dynamic datasets.

The rest of the paper is organized as follows. Section 2 describes a summary of related works. Section 3 discusses the *al*SL clustering method. Section 4 describes the proposed IncrementalSL method. The experimental results and conclusions are discussed in Section 5 and Section 6, respectively.

2 Related Work

Many incremental clustering methods have been developed by many researchers over the decades. The study of incremental clustering can be traced back to late 70's in the last century [11,12]. The leaders clustering is a distance based linear incremental clustering method. For a given threshold distance τ, it produces a set of leaders $\mathcal{L}$ incrementally. For each pattern x, if there is a leader $l \in \mathcal{L}$ such that $||x - l|| <= \tau$, then x is assigned to the cluster represented by l. If there is no such leader, then x becomes a new leader. Each leader can be seen as a representative of a cluster of patterns which are grouped with it. However, it cannot find arbitrary shaped clusters and it is an order depended clustering method. Other variant of leaders method can be found in [13].

It may be noted that clustering in incremental databases has significant different from the clustering applied to data stream. A clustering algorithm in data streams works

in different restricted scenarios. An incremental clustering can have access the whole dataset along with the new updates whereas data stream algorithm can view a limited number of objects at a given time.

The BIRCH clustering proposed in [14] has an incremental phase called *Clustering Feature (CF)*, which utilizes vector space (Euclidean space) properties to store summarized information of a micro cluster of k data points incrementally. Charikar et al. in [15] developed a randomized incremental clustering method, which minimizes maximum diameter of a cluster. Chen et al. [16] introduced an incremental hierarchical clustering method GRIN for numerical datasets, which is based on the gravity theory in physics. In the first phase it uses an agglomerative hierarchical clustering method to produce clustering hierarchy (dendogram) of the given dataset. Then it restructures the hierarchy by merging the lower levels. Each leaf node in the restructured dendogram is represented by the centroid, radius and number of points formed the leaf node. In the second phase, a new data points is either inserted to a leaf node or gravity theory is applied to decide the proper position of the new point. The method may apply the agglomerative hierarchical method to the new set of points if its number exceeds a predefined threshold and follow same restructuring process. However, these methods are not suitable for non Euclidean datasets.

Widyantoro el. al [17] proposed an incremental agglomerative hierarchical clustering, which also uses building and restructuring process of the dendogram of the datasets. It uses minimum spanning tree (MST) to find neighborhood of each data point and it uses vector space property to represent centre of the neighborhood. It is a computationally expensive approach, which cannot be used in large dynamic databases.

IncrementalDBSCAN [18] is a first density based incremental clustering method, which is based on the popular density based clustering DBSCAN [1]. Due to density-based characteristics of DBSCAN, the effects of insertions and deletions are restricted to neighborhood of these data points. Initially, classical DBSCAN is applied to the available dataset at a point of time. Then, IncrementalDBSCAN finds the neighborhood objects and its affected objects in successive updates of the datasets. However, both methods (DBSCAN AND IncrementalDBSCAN) needs two input parameters, which are very difficult to find. The DBSCAN is known an expensive clustering method. IncrementalDBSCAN also needs on an average more than one region quarries for an incremental update.

3 The *al*-SL Clustering Method

Our proposed incremental clustering method IncrementalSL is based on recently proposed distance based *al*-SL method. We choose to adopt this method in dynamic scenarios because the *al*-SL method needs only a single input parameter (h), which is the inter-cluster distance and the value of h can be estimated using approaches given in [10]. The *al*-SL can find arbitrary shaped clusters in large datasets and datasets can be from any metric space (vector or non vector space). In this section, a brief description of the *al*-SL method is reported.

The *al*-SL clustering method is a two-phase clustering method. In the first phase, a hybrid clustering method called *l*-SL, which is the combination of leaders clustering

Algorithm 1. *al*-SL($\mathcal{D}$, h)

Apply leaders clustering method with threshold $\tau = h/2$ to $\mathcal{D}$.
Apply SL method to the leaders set.
Identify a set of pairs of clusters (subsequently leaders) with distance less than $2h$
Apply the leaders clustering method and store followers of above mentioned leaders.
Find two nearest patterns x and y from each pair of clusters with distance less than $2h$.
if $||x - y|| \leq h$ **then**
  Merge the pair of clusters into a single cluster;
end if
Each leader in each cluster is replaced by its followers set. This gives a clustering of the dataset.

method and classical single-link (distance h stopping condition called SL method) is applied to a large dataset. However, the clustering results produced by *l*-SL method may deviate from the final clustering produced by the SL method applied directly to the dataset. A few clusters in the final results of *l*-SL method may be required to be merged in order to obtain the exact final results as that of the SL method. Therefore, the *al*-SL method merges the required clusters in the second phase. The method is reproduced in Algorithm 1. Objects in a cluster produced by *al*-SL method are related to each other by a binary relation 'reachable', which is formally defined as follows.

Definition 1 (Reachable) *A pattern $x \in \mathcal{D}$ is reachable from another pattern $y \in \mathcal{D}$, if there exists a sequence of patterns $x_1, x_2, \ldots, x_p, \quad x_1 = y, x_p = x$ such that $||x_i - x_{i+1}||_{i=1..p-1} <= h, x_i \in \mathcal{D}$.* ■

This 'reachable' relation $R \in \mathcal{D} \times \mathcal{D}$ is an equivalence relation. Therefore, a cluster C in *al*-SL method is an equivalence class of $\mathcal{D}$ by R, *i.e.* $C = [x]_R = \{y \in \mathcal{D} \mid xRy\}$.

4 IncrementalSL: Proposed Incremental Clustering Method

The *al*-SL clustering method works with static large datasets. However, financial transactional database, Recommendation engine database, birth/death registration databases in populated country like India, China are large and dynamic in nature, *i.e.,* these databases have frequent updates of objects. Therefore, clustering structures of these datasets change over time. To capture changes in clustering structures, we present a novel incremental approach called IncrementalSL in this section.

Proposed IncrementalSL updates clustering structures whenever new points are added to the datasets. The method starts with the clustering output produced by the *al*-SL method at a certain point of time. A new object x may influence more than one existing clusters and merge them into one. The set of objects which are influenced over insertion of an object x can be defined as follows.

Definition 2 (Influenced objects) *Let π_{alSL} be the clustering obtained by applying* al-*SL method over the dataset $\mathcal{D}^t$ and x be an object to be inserted to $\mathcal{D}^t$. The set of objects influenced by x is*

$INFLO(x) = \{y \in \mathcal{D}^t \mid xRy,$ *R is a reachable relation on* $\mathcal{D}^t \cup \{x\}\}$. ■

The point x may change the cluster membership of objects in $INFLO(x)$. Finding $INFLO(x)$ is computationally expensive. Therefore, we use leaders and their cluster memberships to find the set $INFLO(x)$. It may be noted that a cluster in *al*-SL consists of a number of leaders with threshold $h/2$, where h is the cluster separation. We only retrieve some key leaders of $\mathcal{D}$ to update the cluster membership information of $INFLO(x)$ instead of entire set $INFLO(x)$. We introduce the following definition to identify the key leaders, which are responsible for possible change in clustering structures.

Definition 3 (Key Leaders) *Let π_{alSL} be the clustering obtained by applying* al-*SL method with parameter h over the dataset $\mathcal{D}^t$ and x be an object to be inserted to $\mathcal{D}^t$. The set of leaders influenced by x called Key Leaders set is*

$Key\_Leaders(x) = \{l \in \mathcal{D}^t \mid ||l - x|| \leq 1.5h, l \text{ is a leader in } \mathcal{D}^t\}$. ■

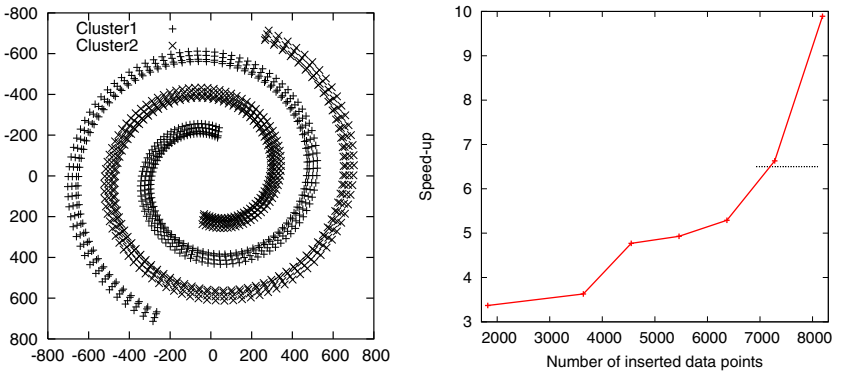
The key leaders of x is the set of leaders located not more than $1.5h$ distance away from x. We use leaders in $Key\_Leaders(x)$ to update the membership information of $INFLO(x)$. We utilize triangle inequality property to quickly locate non-influenced regions in the dynamic datasets using the following Lemma 1.

Lemma 1. *Let l be the closest leader to a newly inserted data point x. If distance between x and l is more than $1.5h$, then no cluster in the dataset $\mathcal{D}$ will be affected due to this insertion.*

Proof: Let y be an arbitrary follower of the leader l. As y is a follower of l, we get $||l - y|| \leq h/2$. By assumption, we get $||l - x|| > 1.5h$. Using triangle inequality, one can write, $||x - y|| + ||l - y|| \geq ||l - x||$. So, $||x - y|| \geq ||l - x|| - ||l - y|| > h$. There is no point in $\mathcal{D}$, which is reachable from x. Therefore, $INFLO(x) = \{x\} \nsubseteq \mathcal{D}$. ■

The IncrementalSL method works as follows while inserting an object x into the dataset $\mathcal{D}$. It calculates distance from x and all leaders in the dataset $\mathcal{D}$. Let $l_{closest}$ be the closest leader with cluster membership C_i.

I. If the distance between x and $l_{closest}$ is less than or equal to $h/2$, the point becomes follower of the leader $l_{closest}$ and x is merged with the cluster in which $l_{closest}$ belongs. Identify the $Key\_Leaders$ of x and if distance between x and a follower of a leader $l \in Key\_Leaders \backslash \{l \in C_i\}$ is less than or equal to h, then C_i is merged with the cluster in which l belongs. Similarly, C_i may be merged with more than one clusters if each of them has a follower f' such that $||x - f'|| \leq h$.
II. If distance between x and $l_{closest}$ satisfies $h/2 < ||x - l_{closest}|| \leq h$, then x becomes a new leader and its membership is C_i. Identify the $Key\_Leaders$ of x and if distance between x and a follower of a leader $l \in Key\_Leaders \setminus \{l \in C_i\}$ is less than or equal to h, then C_i is merged with the cluster in which l belongs. Similarly, C_i may be merged with more than one clusters if each of them has a follower f' such that $||x - f'|| \leq h$.
III. If distance between x and $l_{closest}$ satisfies $h < ||x - l_{closest}|| \leq 1.5h$, then x becomes a new leader. Identify the $Key\_Leaders$ of x and if distance between x and a follower of a leader $l \in Key\_Leaders$ is less than or equal to h, then x is merged with the cluster in which l belongs. Similarly, more than one clusters may be merged if each of them has a follower f' such that $||x - f'|| \leq h$.



(a) Spiral Dataset. (b) Speed-up factor of IncrementalSL.

Fig. 1. Performance of IncrementalSL on Spiral Dataset

IV. If distance between x and $l_{closest}$ is more than $1.5h$, x will be a new leader and creates a new cluster.

It can be shown that the clustering results produced by the IncrementalSL is exactly same as the results produced by the *al*-SL method after inserting all datapoints.

5 Performance Evaluation

In this section, we evaluate efficiency of the IncrementalSL method with one synthetic and one real world datasets. Plot of a synthetic data (Spiral Data) is shown in Fig. 1(a). As we discussed earlier that the IncrementalSL method produces same clustering results as produced by the *al*-SL method, therefore, we do not discusses about the clustering results produced by IncrementalSL in this section. We implemented IncrementalSL and *al*-SL method in C on Intel® Core^{TM} i5 Processor with 4 GB RAM Laptop. We use *speed-up factor*, which is defined as the ratio of the time taken by *al*-SL method and the time taken by our proposed IncrementalSL method. Experiments with Spiral dataset of size 9100 shows the IncrementalSL is upto one order faster (speed-up factor 9.89) than the *al*-SL method. To simulate incremental nature with this dataset, we added points with different sizes $(1820, 3600, 4500, 5500, 6000, 7000, 8000)$ in such a way that total points in the dataset does not exceed its size. It may be noted that inserting a set of points with size m indicates database had initially $(9100 - m)$ points. In the experiments, the *al*-SL was run on entire datasets while we ran IncrementalSL on incremental nature of the Spiral dataset. Plot in Fig. 1(b) shows the speed-factors of IncrementalSL over different sizes of inserted points.

We experimented with real Shuttle dataset (`http://archive.ics.uci.edu/ml/`), which has 9 integer-valued attributes of $58,000$ patterns distributed over 7 classes (after merging training and test sets). Class labels are eliminated from the all patterns. In experiments with Shuttle dataset, we inserted fixed size points $(3,000)$ to the

Table 1. Experiment with Shuttle Dataset

Size of Dataset	Time taken by IncrementalSL (in seconds)	Time taken by *al*-SL (in seconds)
5,000	1.09	1.32
10,000	2.31	3.41
20,000	5.11	8.15
30,000	8.21	14.31
40,000	11.01	23.21
58,000	16.21	36.21

different sizes datasets to simulate the incremental nature. It is found that the our proposed IncrementalSL is faster than the *al*-SL method. The detailed results are given in Table 1.

6 Conclusions and Future Research

The *al*-SL is a distance based clustering method, which is suitable for finding arbitrary shaped clusters in a static dataset. In this paper, we presented an incremental version of the *al*-SL method to work with dynamic scenarios. The clustering results produced by the proposed incremental method is exactly same as the results produced by *al*-SL method and the IncrementalSL is faster than the *al*-SL method. The IncrementalSL method utilizes the metric space properties to find affected region over insertion of data points. In the present work, we do not consider the effect of deletions of points in a dataset. Finding the clustering structures over deletions of points can be future research direction of this work. Another potential research direction can be the study of effect in clustering structures over change of the value of input parameter (h).

References

1. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proceedings of 2nd ACM SIGKDD, pp. 226–231 (1996)
2. Sneath, A., Sokal, P.H.: Numerical Taxonomy. Freeman, London (1973)
3. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern Recognition Letters 31(8), 651–666 (2010)
4. Patra, B.K.: Mining Arbitrary Shaped Clusters in Large Dataset. PhD thesis, Indian Institute of Technology Guwahati, Guwahati, INDIA (2012)
5. Murty, M.N., Krishna, G.: A hybrid clustering procedure for concentric and chain-like clusters. Int. J. Comput. Inform. Sci. 10(6), 397–412 (1981)
6. Wong, M.A.: A hybrid clustering algorithm for identifying high density clusters. Journal of the American Statistical Association 77(380), 841–847 (1982)

7. Vijaya, P.A., Murty, M.N., Subramanian, D.K.: Efficient bottom-up hybrid hierarchical clustering techniques for protein sequence classification. Pattern Recognition 39(12), 2344–2355 (2006)
8. Lin, C.R., Chen, M.S.: Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging. IEEE Trans. on Knowl. and Data Eng. 17(2), 145–159 (2005)
9. Chaoji, V., Hasan, M.A., Salem, S., Zaki, M.J.: Sparcl: an effective and efficient algorithm for mining arbitrary shape-based clusters. Knowl. Inf. Syst. 21(2), 201–229 (2009)
10. Patra, B.K., Nandi, S., Viswanath, P.: A distance based clustering method for arbitrary shaped clusters in large datasets. Pattern Recognition 44(12), 2862–2870 (2011)
11. Hartigan, J.A.: Clustering Algorithms. John Wiley & Sons, Inc., New York (1975)
12. Spath, H.: Cluster Analysis Algorithms for Data Reduction and Classification of Objects. Ellis Horwood, UK (1980)
13. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 3rd edn. Academic Press, Inc., Orlando (2006)
14. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. In: Proceedings of ACM SIGMOD International Conference on Management of Data, SIGMOD 1996, pp. 103–114 (1996)
15. Charikar, M., Chekuri, C., Feder, T., Motwani, R.: Incremental clustering and dynamic information retrieval. In: Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, pp. 626–635 (1997)
16. Chen, C.-Y., Hwang, S.-C., Oyang, Y.-J.: An incremental hierarchical data clustering algorithm based on gravity theory. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS (LNAI), vol. 2336, pp. 237–250. Springer, Heidelberg (2002)
17. Widyantoro, D., Ioerger, T., Yen, J.: An incremental approach to building a cluster hierarchy. In: Proceedings of IEEE International Conference on Data Mining, ICDM 2003, pp. 705–708 (2002)
18. Ester, M., Kriegel, H.P., Sander, J., Wimmer, M., Xu, X.: Incremental clustering for mining in a data warehousing environment. In: Proceedings of 24th International Conference on Very Large Data Bases (VLDB 1998), pp. 323–333 (1998)