Incremental CFS Clustering on Large Data

Liang Zhao*†, Zhikui Chen*, Yi Yang†‡§
*School of Software Technology, Dalian University of Technology, Dalian 116600, China.
†Department of Electrical and Computer Engineering, University of British Columbia, Vancouver V6T1Z4, Canada.
‡School of Reliability and Systems Engineering, Beihang University, Beijing 100191, China.
§Corresponding author:yang_cissy@163.com

***Abstract*—As a popular data mining tool, clustering focuses on revealing underlying patterns embedded in data. However, most existing clustering methods mainly deal with static data, which may not be suitable for analyzing large data in dynamic environments. To tackle this problem, this paper proposes an incremental clustering method based on the CFS, clustering by fast search and find of density peaks, to process large dynamic data. In the proposed method, multiple representatives are identified for each cluster to integrate new objects into previous clustering patterns at first. Then the convex hull theory is employed to modify the representatives accordingly. To further improve the generality and effectiveness, one-time cluster adjustment strategy is explored. Extensive experiments on several real-world image datasets demonstrate that the proposed method outperforms state-of-the-art methods for clustering large data.**

***Index Terms*—CFS, incremental clustering, objects assignment, clusters adjustment**

I. Introduction

As a popular data mining tool, clustering, or cluster analysis, has been recognized as an favorable technique for discovering the underlying information embedded in data. In the past decades, numerous clustering algorithms have been proposed. While they provide effective clustering results for data analysis, most of them are only designed to deal with static data [1,2]. For large data clustering, one challenge is that the data is too large to be loaded in memory. Furthermore, data are increasingly appearing in dynamic manners, such as blogs, web pages, transaction records and time series. When new objects are collected, clustering methods that utilize the stationary nature of data to find a globally optimal solution have to process the whole dataset to update changes in clusters, involving significant redundant computations. Therefore, to address the above two challenges, incremental clustering methods are preferred to process data chunk by chunk.

The characteristics of large data, including high volume and dynamically evolving [3], require incremental clustering methods to rapidly partition continuously arriving objects and effectively update the cluster structures. In the literature, many incremental clustering methods have been proposed to fulfill such requirements [12]. One group of methods is K-centroids based incremental clustering [4,5]. They employ traditional K-centroids methods to assign a new arriving object to the nearest cluster represented by the center, and the corresponding adjustment is made. Simplicity is their major characteristic. However, they have to predefine the number of clusters and are unsuitable for nonspherical data. Hierarchical-based incremental clustering [6,7] can deal with nonspherical data, as well as less susceptible to initialization, but restructuring cluster hierarchies makes them slow to converge. Another advanced group of incremental clustering is density-based methods [8,9]. They can detect clusters with arbitrary shapes and handle updates in an incremental manner. However, tuning input parameters of the methods are challenging and the time complexity of updating multiple regions is high. Recently, some incremental affinity propagation clustering algorithms, such as IAPKM and IAPNA, were proposed in [10] to handle dynamic data. By updating the messages, they adjust the current clustering results according to new arriving objects. However, there are a few concerns of these methods, e.g., IAPKM cannot adjust the number of clusters dynamically, and IAPNA requires high memory usage. For processing large data that may not be well separated, many methods employ soft or fuzzy clustering [11], such as matrix factorization, fuzzy c-means and fuzzy c-medoids, to handle data objects. Though they can capture the natural structure of a dataset more closely, they require predefining the number of clusters.

To tackle the above problems, we extend a recently proposed clustering algorithm for static data, clustering by fast search and find of density peaks (CFS)[13], to process large dynamic data in this paper. Firstly, in the proposed method, multiple representatives are identified for each existing cluster by introducing a maximum min-distance mechanism, which can capture the physical shape and geometry of clusters. After that, new arriving objects are integrated into current patterns according to the representatives, which are modified based on the convex hull theory after integrations, to detect clusters with arbitrary shapes. Further, to dynamically adjust the cluster structures (number of clusters), one-time cluster adjustment strategy is employed to split clusters with multiple dominant patterns and merge clusters with the same dominant pattern. The performance of the proposed method is evaluated on three real-world image datasets. And the experimental results demonstrate that the proposed method outperforms the compared methods in terms of the efficiency and effectiveness, supporting that they are suitable for clustering large data.

II. Incremental CFS clustering

Like the traditional CFS clustering algorithm [13], an incremental CFS should also have the characteristics of detecting clusters with arbitrary shapes and adjusting the number of clusters dynamically. Moreover, it should process new objects based on the existing clustering patterns and not re-implement CFS clustering on the whole dataset. Given a sequentially collected dataset $S = \{X_t\}_{t=1}^{T}$, in which $X_t = \{x_i^t\}_{i=1}^{n_t}$ consists of n_t objects and each object x_i^t contains m features, the main requirements of incremental CFS clustering can be described as follows: i) the new set of objects X_t at time stamp t should be assigned to the existing clustering result R_{t-1} according to the set of all available objects $S_{t-1} = X_1 \cup X_2 \cup X_3 \cup ... \cup X_{t-1}$ at time stamp $t-1$; and ii) the cluster structures should be updated dynamically to form final patterns with regard to all available objects $S_t = S_{t-1} \cup X_t$ at time stamp t. Both requirements pose great challenges on incremental CFS clustering.

In this paper, we propose two strategies to address the above challenges. (1) A new multiple representatives based partitioning that is not sensitive to the previous local density ρ and the minimum distance δ of objects is designed to assign new objects to current clustering results efficiently. (2) An enhanced cluster adjustment strategy, which employs one-time cluster splitting-merging, to partition clusters with multi-patterns and integrate clusters with the same pattern dynamically. They are presented in the following sub-sections.

A. Multiple Representatives based Partitioning

Suppose that the clustering result R_{t-1} for dataset S_{t-1} at time stamp $t-1$ has k clusters $R_{t-1} = \{r_1^{t-1}, r_2^{t-1}, ..., r_k^{t-1}\}$. To integrate the dataset $X_t = \{x_i^t\}_{i=1}^{n_t}$ at time stamp t into the previous result R_{t-1}, we have to calculate local density ρ and minimum distance δ for all new objects. Since the new objects are at initial level, the local density ρ_i for each x_i^t is equal to 0. So the minimum distance δ_i of it can be defined as:

$$\delta_i = \min(dist(x_i^t, r_j^{t-1})), j \in \{1, 2, ..., k\}. \quad (1)$$

Herein, $dist(x_i^t, r_j^{t-1})$ indicates the distance between the object x_i^t and the cluster r_j^{t-1}. And the new object will be assigned to the cluster that is nearest to it. As discussed in [11], one centroid cannot capture the underlying structure of a cluster sufficiently. In order to detect nonspherical clusters in incremental CFS clustering, the multiple representatives of a cluster are selected based on maximum min-distance to assign new data points. The selection of representatives in each cluster $r_j^{t-1} = \{y_l^{t-1}\}_{l=1}^{n_j}$ is presented in following steps:

Assuming the center of cluster r_j^{t-1} is c_j^{t-1}, the first representative object is selected as

$$RSet_{j1}^{t-1} = \underset{y_l^{t-1}}{\arg\max}(dist(y_l^{t-1}, c_j^{t-1})), l \in \{1, 2, ..., n_j\}. \quad (2)$$

After that, the remaining representative objects $\{RSet_{jo}^{t-1}\}_{o=2}^{N_r}$ are identified one by one based on Eq. (6),

$$RSet_{jo}^{t-1} = \underset{y_l^{t-1} \notin RSet_j^{t-1}}{\arg\max} (\min dist(y_l^{t-1}, q)), q \in RSet_j^{t-1}, \quad (3)$$

in which $RSet_j^{t-1}$ is the set of current representatives. And it is updated by adding the new selected representative. In this way, the representatives can be distributed evenly in the data space to avoid converging to local optimum.

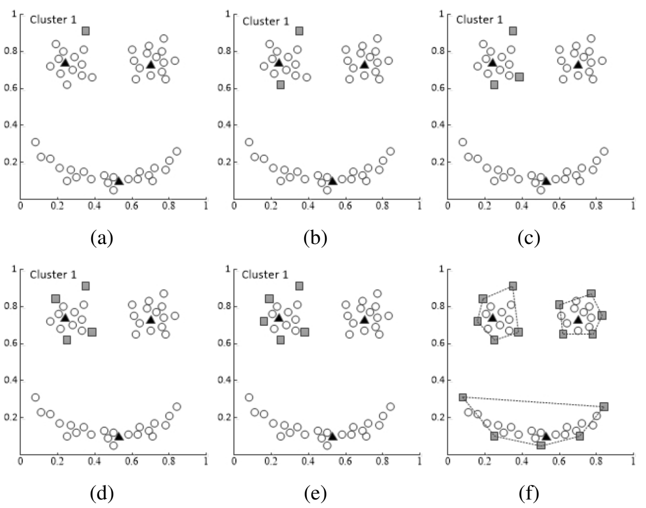

Fig. 1. Illustration of the selection of representative points for each cluster. After the initial clustering by CFS, the representatives are identified in (a)-(f).

Fig. 1 gives an example to illustrate the selection of representative points. As shown in Fig. 1(a), the first representative presented in the square node is selected in $cluster1$, because it is the farthest point from the cluster center, represented by the triangle node, compared to others. After that, the rest representative points that have the maximum min-distances to the previous representatives are identified according to Eq. (3) (see Figs. 1(b), 1(c), 1(d) and 1(e)). As a result, all the clusters can be represented as convex hulls, which can describe the structures of clustering results, as observed in Fig. 1(f).

When the cluster structures are obtained, a new object will be assigned to the cluster that contains the nearest representative to it. Fig. 2 presents the new point assignment according to one centroid and multiple representatives. For a new arriving point Np belonging to $cluster3$, it is wrongly assigned to $cluster1$ whose center is nearest to it (see Fig. 2(a)), while the representative point A of $cluster3$ can draw Np back to its own cluster (see Fig. 2(b)).

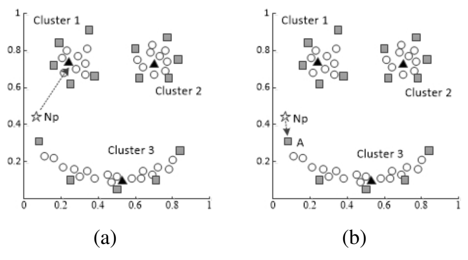

Fig. 2. Illustration of the new points assignment according to (a) one centroid and (b) multiple representatives for incremental clustering.

After the new data object x_i^t is integrated, the structure of the corresponding cluster r_j^{t-1} is changed and its representatives should be updated. Assuming the number of objects in cluster r_j^{t-1} is n_j, the number of representatives is N_r, the nearest representative to x_i^t is $rs \in RSet_j^{t-1}$. Two strategies, that can extend the physical shape and geometry of clusters, are developed to update the set of representatives.

- When N_r is not smaller than θn_j, $\theta \in (0,1)$ is a tradeoff parameter used to balance the percentage of representatives, the Eq. (3) is used to calculate the maximum value of minimum distances between $p \in \{x_i^t, rs\}$ and $q \in RSet_j^{t-1} \backslash \{rs\}$. If x_i^t achieves the maximum value, the representative object rs is replaced by x_i^t, meanwhile it is added to the candidate representatives set CaS_j^{t-1}.
- When N_r is smaller than θn_j, the object $p \in CaS_j^{t-1} \cup x_i^t$ that has the maximum value $\sum dist(p,q), q \in RSet_j^{t-1}$ is selected as a new representative in cluster r_j^{t-1}.

B. One-time Cluster Adjustment Strategy

As observed in Fig. 3(a) that, the dataset consists of three clusters, while the initial batch of data S_{t-1} only contains objects in $cluster1$ and $cluster3$ (see Fig. 3(b)). By implementing CFS clustering on S_{t-1}, two clusters, A and B, are generated. B is only comprised of data points in $cluster3$, but A is comprised of the data points in $cluster3$ and $cluster1$ owing to the incompleteness of S_{t-1}. When new points arriving, they are partitioned to the corresponding clusters based on the representatives, as shown in Fig. 3(c). Note that the points o, m and n of $cluster2$ in original dataset are wrongly assigned to $clusterB$, because no cluster describes the structure of $cluster2$ in S_{t-1}. Fig. 3(d) presents the finial incremental clustering results by ICFSMR. It can be seen that, the $cluster3$ is divided into two parts, which are merged with $cluster1$ and $cluster2$ respectively. Hence, how to adjust the structure of clustering result to make it as reasonable as possible is important.

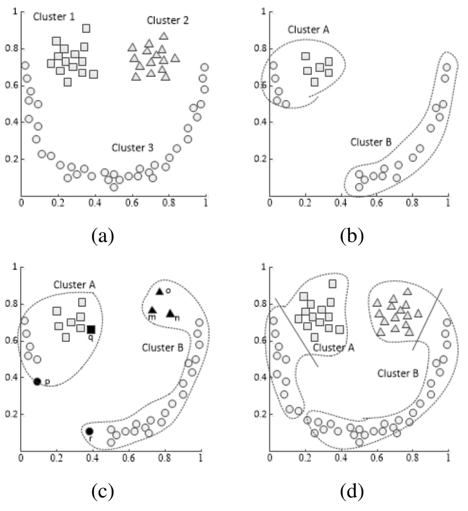

Fig. 3. The evolution of the clustering result. (a) shows the original dataset, (b) shows the first batch of data, (c) shows the result of new points assignment, and (d) shows the finial clustering result.

We proposes an enhanced cluster adjustment strategy, which employs one time splitting and merging of clusters when the number of new arriving objects exceeds the threshold $\lambda \in (0, +\infty)$ of percentage of the total. Given the clustering results $R_t = \{r_1^t, r_2^t, ..., r_k^t\}$ at time stamp t, the processes of splitting for each cluster $r_j^t = \{y_l^t\}_{l=1}^{n_j}$ is described in following steps. The cutoff distance d_c of cluster r_j^t is calculated to achieve the local density ρ_l and the corresponding δ_l and γ_l for all points $\{y_l^t\}_{l=1}^{n_j}$. By sorting $\{\gamma_l\}_{l=1}^{n_j}$ in ascending order $\{\gamma_{a_l}\}_{l=1}^{n_j}$, in which a_l indicates the subscript of γ_l, the centers of new clusters can be selected based on Eqs. (4) and (5).

$$u_{a_l} = \frac{\sum_{p=1}^{l} \gamma_{a_p} + (n_j - l)\gamma_{a_l}}{l}. \tag{4}$$

$$\overline{u_{a_l}} = \frac{u_{a_{l+1}}}{u_{a_l}} = \frac{l * \left(\sum_{q=1}^{l+1} \gamma_{a_q} + (n_j - (l+1))\gamma_{a_{l+1}}\right)}{(l+1) * \left(\sum_{p=1}^{l} \gamma_{a_p} + (n_j - l)\gamma_{a_l}\right)}. \tag{5}$$

Herein, $\overline{u_{a_l}}$ is the jumping degree between point $y_{a_l}^t$ and $y_{a_{l+1}}^t$.When reversing the series of jumping degrees $\overline{U} =< \overline{u_{a_1}}, \overline{u_{a_2}}, ..., \overline{u_{a_{n_j-1}}} >$, the first $\overline{u_{a_l}}$, $l \in \{2, 3, ..., n_j - 1\}$, that meets $\overline{u_{a_l}} > \overline{u_{a_{l-1}}}$ is defined as the borderline. And the points $\{y_{a_{l+1}}^t, ..., y_{a_{n_j}}^t\}$ are selected as the candidate cluster centers. If there are more than one candidate centers, the remaining points are assigned as CFS clustering to split current cluster r_j^t, otherwise no new cluster is generated.

After the cluster splitting, the merging processes for new clusters are implemented. We construct the connected graph of clusters to find the connected components, which indicate that the clusters in the same component should be merged together. The construction of connected graph is as follows. First, we define the structure distance for each cluster in Eq. (6), where σ_f^j and μ_f^j are the standard deviation and mean value of attribute f in cluster j, respectively.

$$s\_dis(r_j^t) = \sqrt{\sum_{f=1}^{m} \left(\sigma_f^j / \mu_f^j\right)^2} * \sum_{f=1}^{m} \left(\mu_f^j\right) \bigg/ m. \tag{6}$$

For two clusters r_u^t and r_v^t, we also define the minimum distance between them by

$$m\_dis(r_u^t, r_v^t) = \min dist(p,q), p \in RSet_u^t, q \in RSet_v^t. \tag{7}$$

If $m\_dis(r_u^t, r_v^t)$ is smaller than $s\_dis(r_u^t)/2$ and $s\_dis(r_v^t)/2$ simultaneously, an edge between them is added. When no edge can be added between clusters, the connected graph with multiple components is obtained. The clusters in the same component are merged to generate the final clustering result $R_t = \{r_1^t, r_2^t, ..., r_k^t\}$.

III. Experiments

A. Experimental Settings

The performance of the proposed method is evaluated on three real-world image datasets (MNIST [11], Olivetti Faces[1] and CIFAR-10[2]). And the proposed method is compared with other two representative incremental clustering methods, IAPNA and IMMFC [10, 11], which have been reported to be

[1]http://www.datatang.com/data/11904
[2]http://www.cs.utoronto.ca/ kriz/cifar.html

more effective and efficient than others. The popular evaluation criteria normalized mutual information (NMI) [11] is used to evaluate the effectiveness, while the computational time is used to evaluate the efficiency of the algorithms.

B. Results and Analysis

In the experiment, each data set is divided into five parts. The first part is used as initial objects, and equal number of left objects are added by four times. For the MNIST and CIFAR-10 datasets, we conduct experiments with chunk size being 0.5% of the entire dataset size, and the first five clustering results are selected for analysis. The results are presented in TABLES I and II.

TABLE I
NMI PERFORMANCE COMPARISONS BETWEEN IAPNA [10], IMMFC [11] AND OUR METHOD ON THE IMAGE DATA SETS.

Data set	Method	1st	2nd	3rd	4th	5th
MNIST	IAPNA	0.335	0.388	0.444	0.440	0.414
	IMMFC	0.324	0.277	0.269	0.273	0.244
	Our Method	0.491	0.413	0.441	0.442	0.439
Olivetti Faces	IAPNA	0.322	0.326	0.370	0.470	0.518
	IMMFC	0.449	0.438	0.453	0.438	0.415
	Our Method	0.590	0.525	0.547	0.530	0.525
CIFAR-10	IAPNA	0.125	0.126	0.106	0.102	0.108
	IMMFC	0.136	0.111	0.100	0.096	0.085
	Our Method	0.175	0.125	0.151	0.138	0.128

TABLE II
COMPUTATIONAL TIME (IN SECONDS) COMPARISONS BETWEEN IAPNA [10], IMMFC [11] AND OUR METHOD ON THE IMAGE DATA SETS.

Data set	Method	1st	2nd	3rd	4th	5th
MNIST	IAPNA	2.180	4.363	8.544	15.03	23.59
	IMMFC	56.86	74.02	83.68	71.87	92.42
	Our Method	0.918	0.725	7.254	1.585	1.666
Olivetti Faces	IAPNA	0.569	2.589	6.062	13.14	22.31
	IMMFC	23.22	31.19	21.85	30.25	33.10
	Our Method	0.793	0.551	7.286	1.102	1.184
CIFAR-10	IAPNA	2.104	9.182	24.85	48.82	81.45
	IMMFC	53.07	65.12	59.09	62.65	81.83
	E_ICFSMR	2.926	1.922	20.05	3.781	4.389

From TABLE I, we can see that our method almost produces the highest MNI values on all image sets while the NMI values of IAPNA and that of IMMFC are comparable. Sometimes, IAPNA achieves larger NMI values than the other two methods, e.g. 3rd and 2nd data chunks on MNIST and CIFAR-10, respectively, but overall IMMFC performs better than IAPNA on the data set of Olivetti Faces.

TABLE II shows that our mwthod requires significantly less time than other two methods to adjust the current clustering patterns when new objects arrive, though it needs some additional overhead for new pattern generation sometimes, as noted in the 3rd experiment on all data sets. The computational time of IAPNA increases with the data being added chunk by chunk, because the new arriving objects make the message passing on all available data. Since the required training time for IMMFC is high, it has the maximum time overhead on all image sets.

In summary, the proposed method performs better than the compared two methods not only in terms of the NMI, but also in terms of the computational time, which demonstrates that our proposed incremental CFS clustering is promising for clustering large dynamic data.

IV. CONCLUSION

In this paper, we focus on extending CFS into incremental clustering for large dynamic data. The difficulties in incremental CFS clustering are first pointed out, and then two strategies are developed correspondingly. Firstly, multiple representatives, which can capture the physical shape and geometry of the clusters, are identified to describe the patterns of clustering. However, new data arrives continuously, the structure and the number of clusters may be changed, thus the one-time cluster splitting-merging strategy is employed to adjust clusters automatically. Experiments conducted on three real-world image datasets demonstrate that the proposed method is superior to the compared methods and promising for clustering large data.

V. ACKNOWLEDGEMENT

This full work of this paper has been submitted to IEEE Transactions on Knowledge and Data Engineering.

REFERENCES

[1] A. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letter*, vol.31, no.8, pp.651-666, 2010.
[2] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol.16, no.3, pp.645-678, 2005.
[3] L. Zhao, Z. Chen, Y. Hu, G. Min and Z. Jiang, "Distributed feature selection for efficient economic big data analysis," *IEEE Transactions on Big Data*, 2016, DOI: 10.1109/TBDATA.2016.2601934.
[4] S. Chakraborty and N. K. Nagwani, "Analysis and study of incremental k-means clustering algorithm," *High Performance Architecture and Grid Computing, Springer Berlin Heidelberg*, pp.338-341, 2011.
[5] G. F. Tzortzis and A. C. Likas, "The global kernel-means algorithm for clustering in feature space," *IEEE Transactions on Neural Networks*, vol.20, no.7, pp.1181-1194, 2009.
[6] N. Sahoo, J. Callan, R. Krishnan, G. Duncan and R. Padman, "Incremental hierarchical clustering of text documents," in *Proc. the 15th International Conference on Information and Knowledge Management, ACM*, 2006, pp.357-366.
[7] P. A. Vijaya, M. N. Murty and D. K. Subramanian, "Leaders-subleaders: an afficient hierarchical clustering algorithm for large dats sets," *Pattern Recognition Letters*, vol.25, no.4, pp.505-513, 2004.
[8] S. Singh and A. Awekar, "Incremental shared nearest neighbor density-based clustering," in *Proc. the 22nd ACM international conference on Information & Knowledge Management, ACM*, 2013, pp.1533-1536.
[9] A. M. Bakr, N. M. Ghanem and M. A. Ismail, "Efficient incremental density-based algorithm for clustering large datasets," *Alexandria Engineering Journal*, vol.54, no.4, pp.1147-1154, 2015.
[10] L. Sun and C. Guo, "Incremental affinity propagation clustering based on message passing," *IEEE Transactions on Knowledge and Data Engineering*, vol.26, no.1, pp.2731-2744, 2014.
[11] Y. Wang, L. Chen and J. P. Mei, "Incremental fuzzy clustering with multiple medoids for large data," *IEEE Transactions on Fuzzy Systems*, vol.22, no.6, pp.1557-1568, 2014.
[12] L. Zhao, Z. Chen, Z. Yang, Y. Hu and M. S. Obaidat, "Local similarity imputation based on fast clustering for incomplete data in Cyber-Physical Systems," *IEEE Systems Journal*, 2016, DOI: 10.1109/JSYST.2016.2576026.
[13] A. Rodriguez and A Laio, "Clustering by fast search and find of density peaks," *Science*, vol.344, no.6191, pp.1492-1496, 2014.