# ICFS Clustering With Multiple Representatives for Large Data

Liang Zhao, Zhikui Chen, *Member, IEEE*, Yi Yang, Liang Zou, and Z. Jane Wang, *Fellow, IEEE*

***Abstract*— With the prevailing development of Cyber-physical-social systems and Internet of Things, large-scale data have been collected consistently. Mining large data effectively and efficiently becomes increasingly important to promote the development and improve the service quality of these applications. Clustering, a popular data mining technique, aims to identify underlying patterns hidden in the data. Most clustering methods assume the static data, thus they are unfavorable for analyzing large, unbalanced dynamic data. In this paper, to address this concern, we focus on incremental clustering by extending the novel [clustering by fast search (CFS) and find of density peaks] method to incrementally handle large-scale dynamic data. Specifically, we first discuss two challenges, i.e., assignment of new arriving objects and dynamic adjustment of clusters, in incremental CFS (ICFS) clustering. We then propose two ICFS clustering algorithms, ICFS with multiple representatives (ICFSMR) and the enhanced ICFSMR (E_ICFSMR) to tackle the two challenges. In ICFSMR, we explore the convex hull theory to modify the representatives identified for each cluster. E_ICFSMR improves the generality and effectiveness of ICFSMR by exploring one-time cluster adjustment strategy after integration of each data chunk. We evaluate the proposed methods with extensive experiments on four benchmark data sets, as well as the air quality and traffic monitoring time series, with comparisons to CFS and other three state-of-the-art incremental clustering methods. Experimental results demonstrate that the proposed methods outperform the compared methods in terms of both effectiveness and efficiency.**

***Index Terms*— Clustering by fast search (CFS), clusters adjustment, incremental clustering, large data, multiple representatives, objects assignment.**

## I. Introduction

With the widespread applications of the Cyber-physical-social systems and Internet of Things [1], [2], large data have been collected easily from various sensor nodes, social network platforms, and e-commercial websites [3], [4]. For example, the Forbes estimated in 2015 that there would be more than 25 billion devices, sensors, and chips handling upward of 50 trillion gigabytes of data within the next five years [5]. Another example is Facebook, one of the most famous social media platforms, on which 30 billion pieces of content are shared every month [6]. In addition, according to Walmart's report, it handles more than 1 million customer transactions every hour [7]. However, only having large data is inadequate, more attention should be paid to mine the valuable information embedded in the data, such as patterns, rules, and insights [8].

Clustering, which aims to find groups of patterns in a data set, called clusters, has been widely used for discovering hidden information in data [9]. Numerous clustering algorithms have been proposed in the past decades and demonstrated to be effective for data analysis. Unfortunately, most of them are specially designed to handle the static data [10]. While when dealing with large and unbalanced data in practice, there could be two additional challenges for clustering techniques: 1) one challenge is the required memory since the data could be too large to be loaded in memory and 2) the other is the increasingly dynamic nature of real-world data such as blogs, online news, web pages, and time series data (e.g., transaction records). For clustering such data, existing clustering methods assuming the static data will have to search for global optimal solutions using the whole data sets again when new arriving objects are available, and thus introduce a large computational overhead. To address the above-mentioned challenges, for large dynamic data, novel incremental clustering methods are highly desired to rapidly partition continuously arriving objects and effectively update cluster patterns.

In the literature, researchers proposed many incremental clustering methods to satisfy such requirements, including K-centroids-based incremental clustering [11]–[13], hierarchical-based incremental clustering [14]–[16], density-based incremental clustering [17]–[19], incremental affinity propagation (IAP) clustering [22], incremental fast search and find of density peaks [clustering by fast search (CFS)]-based clustering [30], [32], and soft and fuzzy-based incremental clustering [23]–[28] methods. Though promising results have been reported, these approaches require to predefine the number of clusters or adjust the parameters in dynamic processes, which is challenging and time-consuming. In addition, some methods are unfavorable when dealing with nonspherical and unbalanced data.

Manuscript received March 6, 2018; revised April 25, 2018 and June 13, 2018; accepted June 22, 2018. Date of publication July 26, 2018; date of current version February 19, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61672123, in part by the Key Science and Technology Planning Project of Guangdong, China, under Grant 2015B010110006, and in part by the Fundamental Research Funds for the Central Universities under Grant DUT18RC(3)025. *(Corresponding author: Yi Yang.)*

L. Zhao and Z. Chen are with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software Technology, Dalian University of Technology, Dalian 116023, China.

Y. Yang is with the School of Reliability and Systems Engineering, Beihang University, Beijing 100083, China (e-mail: yiyang@buaa.edu.cn).

L. Zou is with the School of Electronics and Information Engineering, Anhui University, Hefei 230039, China.

Z. J. Wang is with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC V6T 1Z4, Canada.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2018.2851979

In this paper, we also extend the recently proposed CFS method to process the large dynamic data. CFS, proposed in [29], has its basis only in the distances between data points, and it is able to detect nonspherical and unbalanced clusters and automatically find the correct number of clusters. Several experiments have shown its consistent superiority over previous algorithms for the static data clustering. Inspired by these observations, we propose designing an incremental variant of CFS with multiple representatives (ICFSMR) for clustering large dynamic data. In ICFSMR, a maximum min-distance mechanism is employed to identify multiple representatives for each existing cluster; therefore, the geometry of clusters can be described to further help integrate new arrivals into current cluster patterns by calculating their distances to the representatives. After that, the representatives are modified based on the convex hull theory to detect patterns with arbitrary shapes. The proposed ICFSMR can address the problem of new objects partitioning for the dynamic data; however, the cluster structures cannot be adjusted adaptively in ICFSMR. To further improve the generality and effectiveness of the ICFSMR, we propose the enhanced ICFSMR, E_ICFSMR. E_ICFSMR explores the one-time cluster adjustment strategy to split clusters with multiple dominant patterns and merge clusters with the same dominant pattern after the integration of each data chunk. To summarize, we have made the following contributions in this paper.

1) Two challenges in extending CFS clustering to incremental variants are discussed, and two ICFS clustering algorithms, ICFSMR and E_ICFSMR, are proposed correspondingly.
2) To integrate new arrivals into existing clustering patterns efficiently, multiple representatives are identified and updated in the proposed ICFSMR to capture geometrical properties of clusters.
3) To adjust clustering patterns dynamically, one-time cluster adjustment strategy is employed in E_ICFSMR to improve the generality and effectiveness of ICFS.

The remainder of this paper is organized as follows. Section II reviews related work on incremental clustering, and then gives a brief description of CFS clustering. Section III presents the details of the proposed ICFS algorithms, ICFSMR, and E_ICFSMR. The experiments on several data sets are conducted and the results are discussed in Section IV. Section V gives concluding remarks.

## II. Related Work

In this section, we provide the preliminaries for the rest of this paper, including incremental clustering and CFS clustering.

### A. Incremental Clustering

Many efforts have been devoted to addressing the incremental clustering problem for the dynamic data. In this section, we provide a brief review and point out the research gaps that our study aims to address.

One representative approach of incremental clustering is K-centroids-based algorithms [11]–[13]. In general, they first employ traditional K-centroids methods to identify the centers and clusters of existing data, and then process the data chunk by chunk to obtain the final results. Simplicity is their main advantage; however, they generally need to predefine the number of clustering patterns, and thus are not suitable for unbalanced, nonspherical data. Hierarchical incremental clustering methods proposed in [14]–[16] are able to deal with nonspherical data and they are not sensitive to the initialization, however, they usually maintain a structure of hierarchy. When a new object is placed in the hierarchy, a sequence of related hierarchies is restructured in regions affected by the new object, which can be computationally expensive.

Another representative approach of incremental clustering, referred as density-based incremental clustering, can overcome the difficulties of detecting arbitrary shaped clusters and handling noise [17]–[19]. Though they can partition new arriving data objects in an incremental manner, the updating process needs to search for the whole data set, resulting in huge time consumption. In addition, the methods have the shortcoming that finding appropriate input parameters is difficult. AP clustering [20], realized by passing messages on bipartite, has also been extended to the dynamic environment [21], [22], such as IAP clustering based on K-medoids (IAPKM) and IAP based on Nearest Neighbor Assignment (IAPNA). However, there are limitations of these methods, e.g., IAPKM cannot adjust the number of clusters according to the new arriving objects and IAPNA requires high memory usage.

Zhang *et al.* [30] extended the clustering method CFS proposed in Science in 2014 to process the large dynamic data and proposed the ICFS clustering based on k-mediods (ICFSKM) method. Though promising results have been reported, it becomes unfavorable when dealing with nonspherical and unbalanced data. For processing the dynamic data, many methods employ soft or fuzzy clustering [23]–[28] to handle data objects that may not be well separated, including the online nonnegative matrix factorization [23], the single-pass FCM [24], the online FCM [25], the kernel FCM [26], and the incremental FCM with multiple medoids [incremental multiple medoids-based fuzzy clustering (IMMFC)] [28]. However, all such methods require to predefine the number of clusters, making them unfavorable for large dynamic data.

### B. CFS Clustering

Given the data set $S = \{x_i\}_{i=1}^{N}$ and the corresponding index set $I_S = \{1, 2, \ldots, N\}$, CFS clustering defines two quantities for each object $x_i$, namely, its local density $\rho_i$ and the minimum distance $\delta_i$ from other objects with higher density, in the following equation:

$$\rho_i = \sum_{j \in I_S \setminus \{i\}} e^{-(\mathrm{dist}(i,j)/d_c)^2}, \quad 1 \le i \le N \tag{1}$$

$$\delta_i = \min_{j:\rho_j > \rho_i} \mathrm{dist}(i, j), \quad 1 \le i, \ j \le N. \tag{2}$$

Herein, $\mathrm{dist}(i, j)$ denotes the distance, Euclidean distance is employed in this paper, between $x_i$ and $x_j$, and $d_c$ is a cutoff distance, which is ranked from 1% to 2% of the distances between objects in ascending order. For the object

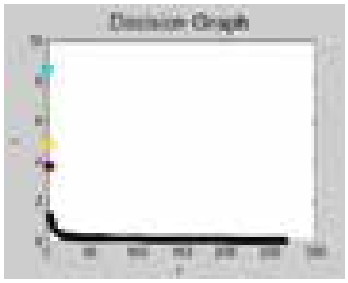


Fig. 1. Cluster center selection by decision graph. The first three points with big $\gamma$ value are the selected cluster centers.

$x_h$ with the highest density, its distance $\delta_h$ is defined as $\delta_h = \max_{j \in I_S \setminus \{h\}}(\delta_j)$.

Note that the objects with high local density $\rho_i$ and much larger distance $\delta_i$ than the neighbors will be select as cluster centers. CFS clustering defines clustering centers as the objects with a big value of $\gamma$ that is calculated in the following equation:

$$\gamma_i = \rho_i \times \delta_i. \tag{3}$$

After finding the cluster centers, as illustrated in Fig. 1, the rest of the objects are assigned to the corresponding clusters that contain their nearest neighbors with higher density.

## III. Proposed Methods

Similar to traditional CFS clustering, it is desirable that an ICFS should be able to detect nonspherical and unbalanced clusters, and can automatically adjust the number of clusters correctly. In addition, an ICFS should assign new arriving objects to existing clusters and adjust the clustering patterns dynamically. In this paper, we first propose a new multiple representatives-based ICFS clustering approach to rapidly partition new arrivals into current clusters. Furthermore, an enhanced method, employing one-time cluster adjustment strategy, is designed to split clusters (with multiple dominant clustering patterns) and merge clusters (with the same pattern) adaptively. We now elaborate the two proposed ICFS algorithms in Sections III-A–III-C.

### A. Incremental CFS Clustering With Multiple Representatives

In this section, we present an ICFS clustering algorithm, ICFSMR, in which multiple representatives are identified in each cluster to guide the assignment of new arrivals. Due to the simplicity of K-centroids-based clustering, which just calculates the distances between data points and the cluster centroids (representatives), many dynamic clustering variants of it have been proposed [11]–[13], [22], [30]. In this paper, we also design an incremental clustering algorithm based on K-centroids clustering.

The rationality of integrating CFS with K-centroids methods for incremental clustering is that the CFS clustering is good at finding initial cluster centroids, and the K-centroids methods are good at updating the clustering results according to the selected centroids dynamically. However, the K-centroids methods are unsuitable for nonspherical and unbalanced data. Thus, we identify multiple representatives in each cluster to detect arbitrary shapes of clusters for ICFS clustering.

ICFSMR consists of three major steps: 1) traditional CFS clustering is approached on the initial set of objects: 2) multiple representatives are located in each cluster to assign the new arriving objects: and 3) the representatives are modified for current clustering results. The details are introduced as follows. Given the data set $S_{t-1}$ at time stamp $(t-1)$, the clustering results are denoted as $R_{t-1} = \{r_1^{t-1}, r_2^{t-1}, \ldots, r_k^{t-1}\}$ which contains $k$ clusters. When a new data set $X_t = \{x_i^t\}_{i=1}^{n_t}$ is arrived at time stamp $t$, the local density $\rho$ and the minimum distance $\delta$ for all new arrivals should be calculated. Usually, the local density $\rho_i$ for each $x_i^t$ is initially set to be 0, and the corresponding minimum distance $\delta_i$ can be defined as

$$\delta_i = \min\left(\text{dist}\left(x_i^t, r_j^{t-1}\right)\right), \quad j \in \{1, 2, \ldots, k\}. \tag{4}$$

Herein, $\text{dist}(x_i^t, r_j^{t-1})$ indicates the distance between the object $x_i^t$ and the cluster $r_j^{t-1}$. Each object will be partitioned into the cluster with the minimum distance to it. Since one centroid usually is not able to sufficiently describe the structure of a cluster [28], to detect nonspherical clusters in ICFS, we propose identifying multiple representatives for each cluster to assign new data objects, as described in the following steps.

For each cluster $r_j^{t-1}$, given the center $c_j^{t-1}$ of the cluster $r_j^{t-1} = \{y_l^{t-1}\}_{l=1}^{n_j}$, the first representative is calculated as

$$\text{RSet}_{j1}^{t-1} = \arg\max_{y_l^{t-1}} \left(\text{dist}\left(y_l^{t-1}, c_j^{t-1}\right)\right), \quad l \in \{1, 2, \ldots, n_j\}. \tag{5}$$

After that, other representatives $\{\text{RSet}_{jo}^{t-1}\}_{o=2}^{N_r}$ are selected one by one according to the following equation:

$$\text{RSet}_{jo}^{t-1} = \arg\max_{y_l^{t-1} \notin \text{RSet}_j^{t-1}} \left(\min \text{dist}\left(y_l^{t-1}, q\right)\right), \quad q \in \text{RSet}_j^{t-1}$$
$$\text{subject to } \max\left(\min \text{dist}\left(y_l^{t-1}, q\right)\right) > e^{-s\_\text{dis}\left(r_j^{t-1}\right)}. \tag{6}$$

in which $\text{RSet}_j^{t-1}$ means the set of identified representatives, which will be updated when a new representative is added. Regarding the constraint of $\max(\min \text{dist}(y_l^{t-1}, q)) > e^{-s\_\text{dis}(r_j^{t-1})}$, it is the stop condition for the selection of representatives. Here, $s\_\text{dis}(r_j^{t-1})$ is the structure distance of clusters, which is defined in (7), where $\mu_f^j$ and $\sigma_f^j$ are the mean value and standard deviation of attribute $f$ in cluster $j$, respectively. A small structure distance represents that the cluster is compact, and less representatives will be selected to describe the cluster. Otherwise, more representatives will be selected. One advantage of the such selected representatives is that they are distributed evenly in the data space, and thus can address the potential local optimum issue

$$s\_\text{dis}\left(r_j^t\right) = \sqrt{\sum_{f=1}^{m} \left(\sigma_f^j / \mu_f^j\right)^2} * \sum_{f=1}^{m} \left(\mu_f^j\right) \Big/ m. \tag{7}$$

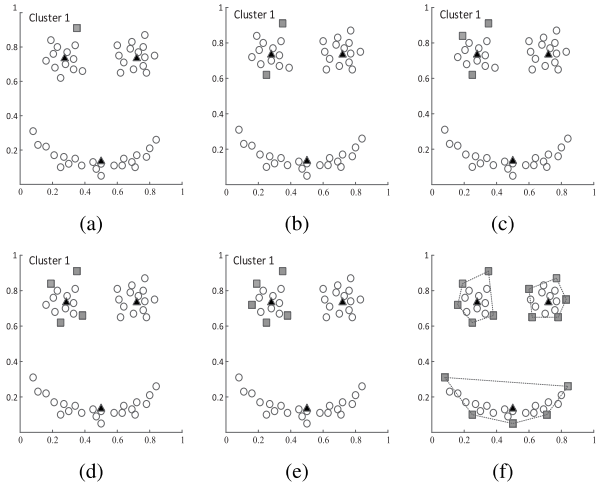



Fig. 2. (a)–(f) Illustrative example: selection of representatives for each cluster after the initial clustering by CFS.

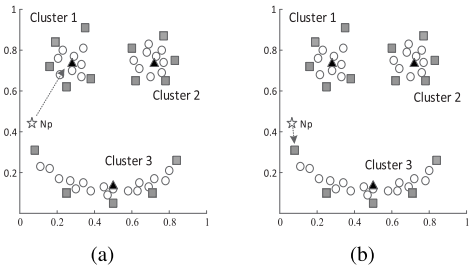



Fig. 3. Illustrative example of assigning a new arrival based on (a) one representative and (b) multiple representatives.

Moreover, to mitigate the effects of abnormal representatives, they are shrunk toward the cluster center by the factor $\alpha \in [0, 1]$ as in [31]

$$\left\{\mathrm{RSet}_j^{t-1}\right\}_1^{N_r} = \left\{\mathrm{RSet}_j^{t-1}\right\}_1^{N_r} + \alpha * \left(c_j^{t-1} - \left\{\mathrm{RSet}_j^{t-1}\right\}_1^{N_r}\right). \tag{8}$$

Fig. 2 shows an illustrative example of the selection of representatives for each cluster, where the data comes from the Smile data set introduced in Section IV. As observed in Fig. 2(a), the point (denoted as the square node) that is farthest from the cluster center, which is represented by the triangle node, is identified as the first representative in $cluster1$. After that, the remaining representatives are selected based on (6), as illustrated in Fig. 2(b)–(e), respectively. Because the maximum min-distance criterion is used for representative selection, all clusters can be described as convex hulls, which can describe cluster structures and can represent arbitrary shapes of clusters [see Fig. 2(f)].

Suppose all clustering patterns are obtained, for a new arrival, it will be assigned to the cluster that has the nearest representative to it. For example, the point $Np$ is partitioned into $cluster3$ with multiple representatives [see Fig. 3(b)], while it would be wrongly assigned to $cluster1$ if we only use the cluster center as the representative [see Fig. 3(a)].

When a new arrival $x_i^t$ is integrated into the cluster $r_j^{t-1}$, its structure is changed and the representatives should be updated correspondingly. Assuming that the nearest representative to $x_i^t$ is $rs \in \mathrm{RSet}_j^{t-1}$, we propose two strategies to update the representatives.

1) When the distance between $x_i^t$ and $rs$, $\mathrm{dist}(x_i^t, rs)$, is greater than $e^{-s\_\mathrm{dis}(r_j^{t-1})}$, $x_i^t$ is selected as a new representative.
2) Otherwise, the maximum value of the minimum distances between $p \in \{x_i^t, rs\}$ and $q \in \mathrm{RSet}_j^{t-1} \backslash \{rs\}$ is calculated based on (6). If $x_i^t$ yields the maximum value, $x_i^t$ will replace $rs$ as the new representative.

Algorithm 1 presents ICFSMR in detail. Traditional CFS clustering is applied on the first batch of objects $S_{t-1}$, and the clustering result $R_{t-1}$ is obtained. After that, the set of representative objects $\mathrm{RSet}_j^{t-1}$ for each cluster is generated. When a new set of objects $X_t = \{x_i^t\}_{i=1}^{n_t}$ are arriving, each $x_i^t$ is assigned to the cluster that contains the nearest representative object to it and the corresponding set of representatives is updated. The available objects $S_{t-1}$, clustering result $R_{t-1}$, and representatives set $\mathrm{RSet}^{t-1}$ are renewed to $S_t = S_{t-1} \cup X_t$, $R_t$, and $\mathrm{RSet}^t$, respectively.

**Algorithm 1** ICFSMR

**Input:** $S_{t-1}$, $R_{t-1}$, $RSet^{t-1}$, $X_t$.
**Output:** $S_t$, $R_t$, $RSet^t$.
1: **for** each new object $x_i^t$ in $X_t$, $i \in \{1, 2, \ldots, n_t\}$ **do**
2: Assign $x_i^t$ to the cluster $r_j^{t-1}$ that contains the nearest shrunk representative to it based on (4);
3: Update the cluster $r_j^{t-1} = r_j^{t-1} \cup x_i^t$ and the representatives set $\mathrm{RSet}_j^{t-1}$ according to the proposed strategies;
4: **end for**
5: $S_t = S_{t-1} \cup X_t$, $R_t = R_{t-1}$, $RSet^t = RSet^{t-1}$;

ICFSMR can modify the clustering result quickly and find arbitrary shapes of clusters, which makes it efficient and flexible enough to be applied in a dynamic environment. However, there is also a disadvantage of ICFSMR that the number of clusters cannot be adjusted according to the new arriving objects. In Section III-B, we will present an enhanced ICFSMR algorithm to adjust the cluster patterns automatically.

### *B. Enhanced ICFSMR to Adjust Cluster Patterns Dynamically*

Usually, the initial batch of data contains certain parts of the clusters, which could significantly affect the results for ICFS clustering. As an example shown in Fig. 4, the data set Smile consists of three clusters [see Fig. 4(a)], but only two clusters $A$ and $B$ are generated in the initialization as shown in Fig. 4(b). $B$ contains objects from $cluster3$, while $A$ contains objects from both $cluster3$ and $cluster1$. When a new object arrives, ICFSMR partitions it to the cluster that has the nearest representative to it, as illustrated in Fig. 4(c). Since no cluster describes the structure of $cluster2$ in the initialization, its objects $o$, $m$, and $n$ are wrongly assigned to $cluster B$. In the final results shown in Fig. 4(d), the objects in $cluster3$ are divided into two parts, integrated by $cluster1$

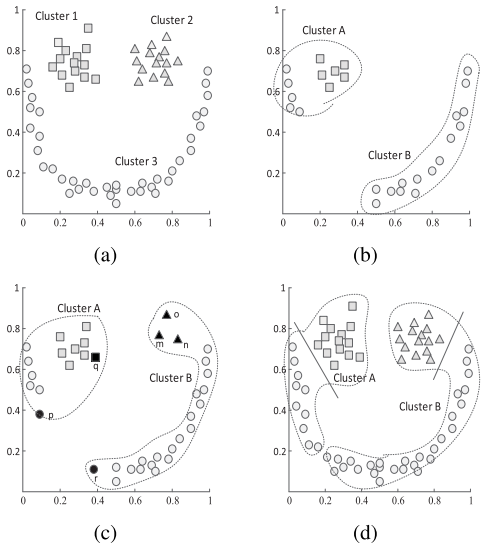



Fig. 4. Illustrative example of the evolution of clustering patterns by employing the proposed ICFSMR. (a) Original data set. (b) Initialization. (c) Assignment of new objects. (d) Final clustering results.

and $cluster2$, respectively. Therefore, adjusting the structures of clustering patterns adaptively is important for incremental clustering.

In E_ICFSMR, an enhanced cluster adjustment strategy is developed. It employs one-time cluster splitting and merging to update the clusters that consist of more than one dominant patterns of classes after integration of each data chunk. Given the clustering results $R_t = \{r_1^t, r_2^t, \ldots, r_k^t\}$ at time stamp $t$, the splitting procedures for a cluster $r_j^t = \{y_l^t\}_{l=1}^{n_j}$ are presented as follows. First, the cutoff distance $d_c$ for $r_j^t$ is calculated, thus the local density $\rho_l$ and the corresponding $\delta_l$ and $\gamma_l$ for all objects $\{y_l^t\}_{l=1}^{n_j}$ can be obtained. By sorting $\{\gamma_l\}_{l=1}^{n_j}$ in ascending order $\{\gamma_{a_l}\}_{l=1}^{n_j}$, in which $a_l$ stands for the subscript of $\gamma_l$, the centers of new clusters can be identified according to the following equations:

$$u_{a_l} = \frac{\sum_{p=1}^{l} \gamma_{a_p} + (n_j - l)\gamma_{a_l}}{l}. \tag{9}$$

$$\overline{u_{a_l}} = \frac{u_{a_{l+1}}}{u_{a_l}} = \frac{l * \left(\sum_{q=1}^{l+1} \gamma_{a_q} + (n_j - (l+1))\gamma_{a_{l+1}}\right)}{(l+1) * \left(\sum_{p=1}^{l} \gamma_{a_p} + (n_j - l)\gamma_{a_l}\right)}. \tag{10}$$

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

*Property 1:* When a hybrid cluster with multiple dominant class patterns is split, the purity of the clustering result is improved.

*Proof:* As described in Section IV, when a data set with $c$ classes is clustered into $k$ clusters, the purity of the clustering result is $\text{purity}(k) = \sum_{i=1}^{k} \max_j(n_{ij})/n,\ j \in \{1, 2, \ldots, c\}$. Suppose the hybrid cluster $r$ is split into $t$ parts, the purity is $\text{purity}(k+t-1) = (\sum_{i=1}^{r-1} \max_j(n_{ij}) + \sum_{r_p=r_1}^{r_t} \max_j(n_{r_p j}) + \sum_{i=r+1}^{k} \max_j(n_{ij}))/n$. Generally, the cluster that is split contains various patterns of classes, thus

$$\begin{aligned} n_{rj} &= \sum_{r_p=r_1}^{r_t} n_{r_p j} \\ &\Rightarrow \sum_{r_p=r_1}^{r_t} \max_j(n_{r_p j}) > \max_j(n_{rj}) \\ &\Rightarrow \left(\sum_{r_p=r_1}^{r_t} \max_j(n_{r_p j}) - \max_j(n_{rj})\right) \Big/ n > 0 \\ &\Rightarrow \text{purity}(k+t-1) - \text{purity}(k) > 0. \end{aligned} \tag{11}$$

If there are new clusters generated, we need to implement the cluster merging process. The connected graph is constructed to find connected components among clusters, where the clusters belonging to the same component should be merged together. For the construction of the connected graph, we first calculate the structure distance for clusters using (7). Then, the minimum distance between two clusters $r_u^t$ and $r_v^t$ is defined as

$$m\_\text{dis}(r_u^t, r_v^t) = \min \text{dist}(p, q), \quad p \in \text{RSet}_u^t,\ q \in \text{RSet}_v^t. \tag{12}$$

If $m\_\text{dis}(r_u^t, r_v^t)$ is smaller than $s\_\text{dis}(r_u^t)$ and $s\_\text{dis}(r_v^t)$ simultaneously, an edge between them is added. After all such edges between clusters are added, we can get the connected graph with multiple components. The final clustering result $R_t = \{r_1^t, r_2^t, \ldots, r_k^t\}$ is thus obtained by merging the clusters with the same component.

*Property 2:* When two or more clusters with the same dominant class pattern are merged, the accuracy of the clustering result is improved.

*Proof:* In the phrase of clustering splitting, a cluster with multiple patterns of classes is divided into clusters, in which one cluster only has one dominant pattern. Suppose the current clustering result consists of $k$ clusters corresponding to $c$ classes, then its accuracy is $\text{accuracy}(k) = \sum_{p=1}^{c} \text{map}(p, \delta_j(a_i))/n$ (see Section IV). When merging one of the remaining clusters with the $p$th mapped cluster for accuracy calculation, we have the new value as

$$\begin{aligned} &\text{map}(p, \delta_j(a_i)) \\ &\quad = \text{map}(p, \max(n_{ij} + \Delta n_{ij})) > \text{map}(p, \max(n_{ij})) \\ &\quad \Rightarrow \text{accuracy}(k-1) > \text{accuracy}(k) \end{aligned} \tag{13}$$

where $\Delta n_{ij}$ denotes the number of points that are merged to the $p$th mapped cluster with class $j$.

According to Properties 1 and 2, if clusters that contain multiple patterns are split and clusters that contain the same dominant pattern of clustering are merged, the performance of clustering can be improved.

Fig. 5 illustrates that, when employing E_ICFSMR on the clustering results from ICFSMR, the two clusters $A$ and $B$ shown in Fig. 4(d) are split into two parts, respectively, and

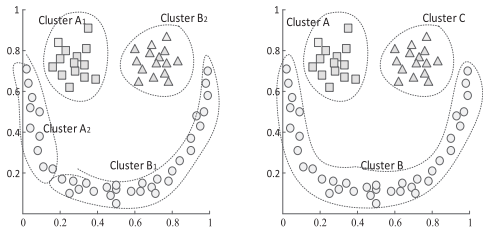



Fig. 5. Illustration of E_ICFSMR for adjusting the patterns of clustering results achieved by ICFSMR. (a) and (b) One-time cluster splitting and merging.

the newly generated clusters are merged favorably to form the final clustering patterns.

**Algorithm 2** E_ICFSMR

**Input:** $S_{t-1}$, $R_{t-1}$, $RSet^{t-1}$, $X_t$, $n_{new}$.
**Output:** $S_t$, $R_t$, $RSet^t$, $n_{new}$.

1: Algorithm 1 is employed to partition $x_i^t$ in $X_t$ to $R_{t-1}$ and the clustering result $R_{t-1}$ and $RSet^{t-1}$ are adjusted to $R_t$ and $RSet^t$, respectively;
2: $S_t = S_{t-1} \cup X_t$, $n_{new} = n_{new} + n_t$;
3: **if** any cluster contains more than one dominant patterns of classes **then**
4: The clusters containing multiple patterns of clustering in $R_t$ are split to generate $R_{t'}$ based on (9) and (10), and the corresponding RSet$^{t'}$ is created;
5: The clusters with same dominant pattern of clustering in $R'_t$ are merged to generate $R_{t''}$ based on (7) and (12), and RSet$^{t'}$ is updated;
6: $n_{new} = 0$;
7: **end if**
8: $R_t = R''_t$, $RSet^t = RSet^{t'}$;

The details of the proposed E_ICFSMR are presented in Algorithm 2. The ICFSMR algorithm is exploited to assign the new coming data set $X_t$ into the previous clustering result $R_{t-1}$, and the corresponding set of representatives for each cluster is updated. If any cluster contains more than one dominant patterns, the one-time cluster splitting and merging is conducted on the current clustering result, and new representatives are reselected accordingly.

### C. Complexity Analysis

In ICFSMR, for a new object $x_i^t$ in $X_t$, we need to calculate its distances to all the representatives in $R_{t-1}$, thus the time complexity for new objects assignment is $O(kn_tN_r)$. Here $k$ denotes the number of clusters, $n_t$ denotes the number of data objects in $X_t$, and $N_r$ is the number of representatives in each cluster. Regarding representatives updating, the maximum min-distance for each $x_i^t$ and its nearest representative to other representatives in the cluster is calculated, which yields a time complexity of $O(N_r)$. Therefore, the time complexity of ICFSMR is $O(kn_tN_r)$. When cluster splitting is implemented in E_ICFSMR, the distance matrix dist($r$) and the structure distance for each cluster are used to reassign the objects and regenerate the representatives. Therefore, the time complexity for cluster splitting is close to $O(kn_r^2)+O(k'n_{r'})$. Herein, $n_r$ is the number of data objects in each cluster, $k'$ is the number of clusters after splitting and $r' \in \{1, 2, \ldots, k'\}$. After the cluster splitting, the minimum distances between two clusters are calculated to merge clusters with the same dominate pattern of clustering, which has a complexity of $O((k'N_{r'})^2)$. $N_{r'}$ and $k'$ are usually much less than $n_r$. Therefore, the time complexity of E_ICFSMR is $O(kn_tN_r + kn_r^2 + k'n_{r'} + (k'N_{r'})^2) \approx O(kn_tN_r + kn_r^2)$. Since the traditional CFS method is implemented on the whole data set, its complexity is $O(N_t^2)$, which is much higher than that of our proposed methods. Table I provides the complexity comparisons.

TABLE I
COMPLEXITY COMPARISONS FOR CFS [29], ICFSMR, AND E_ICFSMR

| | CFS | ICFSMR | E_ICFSMR |
|---|---|---|---|
| Time Complexity | $O(N_t^2)$ | $O(kn_tN_r)$ | $O(kn_tN_r+kn_r^2)$ |

## IV. EXPERIMENTS AND ANALYSIS

This section evaluates the performances of the proposed ICFSMR and E_ICFSMR on four popular data sets, as well as the air quality and traffic monitoring time series. The goal of this paper is to propose the ICFS clustering, which can achieve comparable clustering performances as CFS yet can efficiently assign new arriving objects and adjust the cluster structures dynamically. Therefore, the traditional CFS is also implemented to provide the benchmark clustering performance. Furthermore, we compare the proposed methods with three representative incremental clustering methods in the literature, IAPNA, IMMFC, and ICFSKM, as these three methods were reported to be more effective and efficient than other methods [22], [28], [30].

Our experiments are conducted on a PC with 2.20-GHz Intel Core i5-5200U CPU and 4.00-GB memory. The evaluation criteria and experimental results are discussed in subsections IV-A–IV-D.

### A. Evaluation Criteria

Three popular evaluation criteria, including purity [8], normalized mutual information (NMI) [28], and accuracy [22], are studied to evaluate the effectiveness of different algorithms, meanwhile the computational time is employed to evaluate the efficiency of different algorithms. If we refer to class as the ground truth and cluster as the results of clustering algorithms, the purity is defined as

$$\text{purity} = \sum_{i=1}^{k} \frac{\max_j(n_{ij})}{n} \tag{14}$$

where $n_{ij}$ indicates the number of common points in class $j \in \{1, 2, \ldots, c\}$ and cluster $i \in \{1, 2, \ldots, k\}$ and $n$ stands for the size of the data set.

NMI is an information theoretic measure, which evaluates the effectiveness of clustering algorithms by calculating the

TABLE II
BRIEF DESCRIPTION OF THE TESTED DATA SETS

| Dataset | Number of objects | Number of attributes | Number of Categories |
|---|---|---|---|
| Smile | 266 | 2 | 3 |
| Iris | 150 | 4 | 3 |
| Wine | 178 | 13 | 3 |
| Yeast | 1,484 | 8 | 10 |

TABLE III
EXPERIMENTAL DETAILS ON THE TESTED DATA SETS

| Dataset | Number of initial objects | Number of new arriving objects |
|---|---|---|
| Smile | 66 | 50 |
| Iris | 30 | 30 |
| Wine | 58 | 30 |
| Yeast | 324 | 290 |

mutual information between the clustering results and real cluster labels

$$\text{NMI} = \frac{\sum_{j=1}^{c}\sum_{i=1}^{k} n_{ij}\log\left(\frac{nn_{ij}}{n_j n_i}\right)}{\sqrt{\left(\sum_{j=1}^{c} n_j \log\left(\frac{n_j}{n}\right)\right)\left(\sum_{i=1}^{k} n_i \log\left(\frac{n_i}{n}\right)\right)}} \quad (15)$$

in which $n_i$ and $n_j$ are the numbers of objects in $i$th cluster and $j$th class, respectively.

Accuracy is a more direct measure to reflect the effectiveness of clustering results, which is calculated as follows:

$$\delta_j(a_i) = \max(n_{ij}), \quad (16)$$

$$\text{accuracy} = \sum_{p=1}^{c} \frac{\text{map}(p, \delta_j(a_i))}{n}. \quad (17)$$

Herein, $\delta_j(a_i)$ is the maximum number of common objects in cluster $i$ corresponding to different class $j \in \{1, 2, \ldots, c\}$, and $\text{map}(p, \delta_j(a_i)$ indicates the $p$th largest value for all $\delta_j(a_i), i \in \{1, 2, \ldots, k\}$ with no duplication of corresponding classes label $j \in \{1, 2, \ldots, c\}$.

Higher values of the above-mentioned three criteria imply that better clustering results are achieved by algorithms.

### B. Results on Standard Data sets

In this section, four standard data sets are used to evaluate ICFS clustering algorithms. Table II gives the brief description of the data sets. Smile is a synthetic data set, whose characteristics are given in Section III. Iris, Wine, and Yeast are benchmark data sets for clustering algorithms in the UC Irvine Machine Learning Repository [22].

Each data set is randomly divided into five chunks with certain specified sizes. The first chunk is utilized for initialization, and then a certain number of objects (considered as new arrivals) are added four times. More details are presented in Table III. For example, the traditional CFS is implemented on the initial batch of objects (66 2-D points) in Smile, and the left points are added 4 times (50 points each time). When new objects arrive, ICFSMR is employed to update the clustering results. When any cluster contains more than one dominant patterns of classes after integration of each data chunk, E_ICFSMR is employed to adjust the structure of clusters adaptively.

In the first experiment, the proposed methods are compared with the traditional CFS clustering, which is implemented on the entire available objects to provide the benchmark clustering performance. Every experiment is repeated 20 times by randomly selecting the objects, and mean values of measures are calculated. Regarding the parameters, $d_c$ is ranked at 2% of the distances between objects in ascending order [29], and $\alpha$ is empirically set to be 0.5. The results are given in Tables IV–VII, in terms of the purity, NMI, accuracy, and computational time, respectively.

TABLE IV
COMPARISON OF PURITY ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | CFS[29] | **0.939** | 0.767 | 0.783 | 0.801 | 0.797 |
| | ICFSMR | **0.939** | 0.922 | 0.922 | 0.917 | 0.910 |
| | E_ICFSMR | **0.939** | **0.942** | **0.958** | **0.944** | **0.951** |
| Iris | CFS[29] | **0.833** | **0.833** | 0.789 | **0.875** | 0.780 |
| | ICFSMR | **0.833** | 0.767 | 0.700 | 0.692 | 0.667 |
| | E_ICFSMR | **0.833** | 0.797 | **0.800** | 0.792 | **0.807** |
| Wine | CFS[29] | **0.948** | **0.943** | **0.966** | 0.890 | 0.785 |
| | ICFSMR | **0.948** | 0.920 | 0.928 | 0.903 | 0.890 |
| | E_ICFSMR | **0.948** | 0.937 | 0.941 | **0.939** | **0.927** |
| Yeast | CFS[29] | **0.478** | 0.472 | 0.466 | 0.482 | 0.484 |
| | ICFSMR | **0.478** | 0.469 | 0.461 | 0.455 | 0.444 |
| | E_ICFSMR | **0.478** | **0.496** | **0.538** | **0.527** | **0.518** |

TABLE V
COMPARISON OF NMI ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | CFS[29] | **0.740** | 0.576 | 0.573 | 0.594 | 0.587 |
| | ICFSMR | **0.740** | 0.735 | 0.730 | 0.728 | 0.717 |
| | E_ICFSMR | **0.740** | **0.795** | **0.830** | **0.816** | **0.817** |
| Iris | CFS[29] | **0.695** | 0.695 | 0.670 | **0.791** | 0.669 |
| | ICFSMR | **0.695** | 0.674 | 0.673 | 0.665 | 0.657 |
| | E_ICFSMR | **0.695** | **0.774** | **0.786** | 0.771 | **0.761** |
| Wine | CFS[29] | **0.798** | 0.797 | **0.805** | 0.733 | 0.629 |
| | ICFSMR | **0.798** | 0.746 | 0.756 | 0.736 | 0.707 |
| | E_ICFSMR | **0.798** | **0.806** | 0.785 | **0.765** | **0.752** |
| Yeast | CFS[29] | **0.236** | 0.197 | 0.191 | 0.204 | 0.232 |
| | ICFSMR | **0.236** | 0.216 | 0.183 | 0.165 | 0.163 |
| | E_ICFSMR | **0.236** | **0.276** | **0.279** | **0.253** | **0.237** |

From Tables IV–VI, we can see that the three methods achieve the same clustering results on the first batch of objects, because the traditional CFS is implemented on the initial data. It can be observed from Table IV that ICFSMR achieves consistent lower purity than that of CFS on data sets Iris and Yeast. Similar observations are noted for the NMI and accuracy measures in Tables V and VI. Such observations imply that the initial clustering may only identify the part of the cluster patterns. As presented in Fig. 4(c), ICFSMR cannot always correctly recognize all the cluster patterns of Iris (only 4 out of 20 times identify all 3 cluster patterns) and Yeast (only 2 out of 20 times identify all 10 cluster patterns) based on their initial set of objects. When objects

TABLE VI
COMPARISON OF ACCURACY ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | CFS[29] | **0.788** | 0.767 | 0.783 | 0.801 | 0.797 |
| | ICFSMR | **0.788** | 0.819 | 0.819 | 0.829 | 0.823 |
| | E_ICFSMR | **0.788** | **0.869** | **0.873** | **0.861** | **0.865** |
| Iris | CFS[29] | **0.700** | **0.700** | 0.578 | **0.875** | 0.573 |
| | ICFSMR | **0.700** | 0.617 | 0.578 | 0.575 | 0.547 |
| | E_ICFSMR | **0.700** | 0.697 | **0.700** | 0.692 | **0.687** |
| Wine | CFS[29] | **0.897** | **0.928** | **0.931** | 0.881 | 0.778 |
| | ICFSMR | **0.897** | 0.875 | 0.889 | 0.884 | 0.861 |
| | E_ICFSMR | **0.897** | 0.905 | 0.894 | **0.892** | **0.876** |
| Yeast | CFS[29] | **0.319** | **0.321** | 0.275 | 0.284 | **0.273** |
| | ICFSMR | **0.319** | 0.279 | 0.242 | 0.226 | 0.217 |
| | E_ICFSMR | **0.319** | 0.319 | **0.316** | **0.289** | 0.248 |

TABLE VII
COMPARISON OF COMPUTATIONAL TIME (IN SECOND) ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | CFS[29] | **0.033** | 0.072 | 0.106 | 0.163 | 0.209 |
| | ICFSMR | 0.034 | **0.033** | **0.028** | **0.033** | **0.037** |
| | E_ICFSMR | 0.035 | 0.053 | 0.089 | 0.106 | 0.093 |
| Iris | CFS[29] | **0.025** | 0.036 | 0.095 | 0.194 | 0.416 |
| | ICFSMR | 0.026 | **0.021** | **0.017** | **0.020** | **0.023** |
| | E_ICFSMR | 0.026 | 0.032 | 0.115 | 0.052 | 0.065 |
| Wine | CFS[29] | **0.028** | 0.100 | 0.195 | 0.366 | 0.686 |
| | ICFSMR | 0.030 | 0.024 | **0.019** | **0.020** | **0.022** |
| | E_ICFSMR | 0.031 | **0.022** | 0.152 | 0.122 | 0.066 |
| Yeast | CFS[29] | **0.478** | 2.156 | 5.329 | 14.21 | 47.01 |
| | ICFSMR | 0.503 | **0.438** | **0.590** | **0.752** | **0.889** |
| | E_ICFSMR | 0.514 | 1.043 | 1.298 | 2.972 | 4.283 |

representing new patterns of clustering arrive, they are wrongly assigned to the current patterns, which will affect the clustering performance. However, the proposed E_ICFSMR can achieve higher purity, NMI, and accuracy than CFS on these data sets when the one-time splitting-merging for clusters is performed after integration of each data chunk. For data sets Smile and Wine, ICFSMR achieves better or comparable clustering results when compared with CFS, which indicates that multiple representatives can effectively extend the structure of clusters if all cluster patterns are captured favorably in initialization.

The efficiency results of the methods are reported in Table VII. As noted from Table VII, the computational time of CFS increases dramatically on all data sets. This is because the CFS method is conducted on all available data objects at the corresponding time step. The proposed ICFSMR spends the least time to cluster new objects for all data chunks, and E_ICFSMR requires slightly more time than ICFSMR to adjust the cluster structures dynamically.

In summary, the proposed ICFS methods can achieve better, or at least comparable, clustering performance than that of the traditional CFS clustering. More significantly, the computational time of CFS is reduced dramatically, making it suitable for clustering large data in the dynamic environment.

In the second experiment, we compare the performance of E_ICFSMR with IAPNA, IMMFC, and ICFSKM in terms of purity, NMI, and computational time on the standard data sets. It is worth mentioning that, E_ICFSMR, IAPNA, and ICFSKM can automatically determine the number of clusters $k$, while IMMFC requires such information and IMMFC's accuracy is sensitive to the number of clusters $k$. We therefore do not report the accuracy results as the comparison is not fair. The IAPNA, IMMFC, and ICFSKM are reimplemented based on the data chunks described in Table III, and all other settings are same as previous experiments. About the parameters of the compared methods, they are similar to those in [22], [28], and [30]. Specifically, for IAPNA, same as in [22], the preference coefficient $pc$ is set to be 0.015 for all data sets, and the preference $p$ is calculated according to the guideline. For IMMFC, same as in [28], the parameter $\varepsilon$ is set to be $10^{-5}$, $10^{-5}$, $10^{-5}$, and $10^{-2}$ on Smile, Iris, Wine, and Yeast data sets, respectively. The number of medoids $t$ for each cluster is set to be 2, and the parameter $T_u$ is set to be 0.1. The results are given in Tables VIII–X.

TABLE VIII
PURITY COMPARISONS BETWEEN IAPNA [22], IMMFC [28], ICFSKM [30], AND E_ICFSMR ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | IAPNA | 0.924 | 0.922 | 0.952 | 0.940 | 0.929 |
| | IMMFC | 0.864 | 0.655 | 0.735 | 0.718 | 0.722 |
| | ICFSKM | **0.939** | 0.911 | 0.894 | 0.886 | 0.902 |
| | E_ICFSMR | **0.939** | **0.942** | **0.958** | **0.944** | **0.951** |
| Iris | IAPNA | **0.837** | **0.841** | 0.784 | 0.787 | 0.792 |
| | IMMFC | 0.784 | 0.805 | **0.853** | 0.747 | 0.782 |
| | ICFSKM | 0.833 | 0.835 | 0.794 | 0.783 | 0.805 |
| | E_ICFSMR | 0.833 | 0.797 | 0.800 | **0.792** | **0.807** |
| Wine | IAPNA | 0.931 | 0.909 | 0.932 | 0.926 | 0.921 |
| | IMMFC | 0.914 | 0.827 | **0.943** | 0.857 | 0.888 |
| | ICFSKM | **0.948** | 0.922 | 0.894 | 0.913 | 0.896 |
| | E_ICFSMR | **0.948** | **0.937** | 0.941 | **0.939** | **0.927** |
| Yeast | IAPNA | 0.463 | 0.484 | 0.471 | 0.462 | 0.474 |
| | IMMFC | **0.504** | **0.515** | 0.493 | 0.467 | 0.483 |
| | ICFSKM | 0.478 | 0.464 | 0.476 | 0.462 | 0.473 |
| | E_ICFSMR | 0.478 | 0.496 | **0.538** | **0.527** | **0.518** |

Tables VIII and IX show that the proposed E_ICFSMR performs better in terms of purity and NMI than other three methods on third, fourth, and fifth chunks of Yeast, first, second, fourth, and fifth chunks of Wine, fourth and fifth chunks of Iris, and all chunks of Smile, respectively. Regarding other data chucks, E_ICFSMR has lower while still comparable performance with the best ones. This is because the initial CFS cannot identify all the cluster patterns, E_ICFSMR has to adjust the clustering results with the new objects added continuously. Thus, it can achieve the best results for the final clustering on all data sets. Since IAPNA and ICFSKM cannot exactly find the clusters with small size in Yeast, which is also called unbalanced data, they yield much lower purity and NMI values than other two methods in most cases, while they are comparable with other two methods on Smile, Iris, and Wine. Sometimes, IMMFC produces the highest purity and NMI values; however, it is worth mentioning that its computational time is the highest on all data sets, as given in Table X. From

TABLE IX

NMI COMPARISONS BETWEEN IAPNA [22], IMMFC [28], ICFSKM [30], AND E_ICFSMR ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | IAPNA | 0.708 | 0.720 | 0.720 | 0.706 | 0.681 |
| | IMMFC | 0.670 | 0.481 | 0.530 | 0.492 | 0.495 |
| | ICFSKM | **0.740** | 0.782 | 0.776 | 0.784 | 0.781 |
| | E_ICFSMR | **0.740** | **0.795** | **0.830** | **0.816** | **0.817** |
| Iris | IAPNA | **0.749** | 0.756 | 0.754 | 0.753 | 0.755 |
| | IMMFC | 0.689 | 0.736 | **0.811** | 0.701 | 0.736 |
| | ICFSKM | 0.695 | 0.733 | 0.741 | 0.747 | 0.738 |
| | E_ICFSMR | 0.695 | **0.774** | 0.786 | **0.771** | **0.761** |
| Wine | IAPNA | 0.773 | 0.727 | 0.771 | 0.757 | 0.746 |
| | IMMFC | 0.781 | 0.696 | **0.793** | 0.698 | 0.745 |
| | ICFSKM | **0.798** | 0.780 | 0.772 | 0.744 | 0.736 |
| | E_ICFSMR | **0.798** | **0.806** | 0.785 | **0.765** | **0.752** |
| Yeast | IAPNA | 0.208 | 0.232 | 0.221 | 0.218 | 0.225 |
| | IMMFC | **0.247** | 0.258 | 0.239 | 0.221 | 0.231 |
| | ICFSKM | 0.236 | 0.242 | 0.234 | 0.227 | 0.213 |
| | E_ICFSMR | 0.236 | **0.276** | **0.279** | **0.253** | **0.237** |

TABLE X

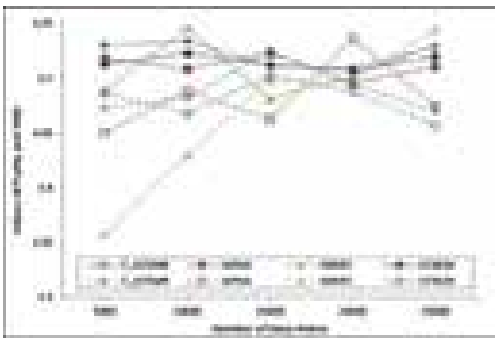
COMPUTATIONAL TIME (IN SECOND) COMPARISONS BETWEEN IAPNA [22], IMMFC [28], ICFSKM [30], AND E_ICFSMR ON THE STANDARD DATA SETS

| Dataset | Method | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|---|
| Smile | IAPNA | **0.020** | 0.054 | **0.068** | 0.124 | 0.144 |
| | IMMFC | 1.457 | 0.911 | 0.603 | 0.604 | 0.759 |
| | ICFSKM | 0.034 | 0.056 | 0.083 | 0.114 | 0.132 |
| | E_ICFSMR | 0.035 | **0.053** | 0.089 | **0.106** | **0.093** |
| Iris | IAPNA | **0.026** | 0.033 | **0.034** | 0.085 | 0.097 |
| | IMMFC | 0.259 | 0.168 | 0.336 | 0.392 | 0.529 |
| | ICFSKM | 0.028 | 0.037 | 0.052 | 0.066 | 0.092 |
| | E_ICFSMR | **0.026** | **0.032** | 0.115 | **0.052** | **0.065** |
| Wine | IAPNA | **0.027** | 0.034 | **0.054** | 0.174 | 0.118 |
| | IMMFC | 0.677 | 0.339 | 0.317 | 0.380 | 0.630 |
| | ICFSKM | 0.030 | 0.043 | 0.056 | **0.078** | 0.116 |
| | E_ICFSMR | 0.031 | **0.022** | 0.152 | 0.122 | **0.066** |
| Yeast | IAPNA | **0.489** | 4.295 | 3.639 | 6.832 | 10.79 |
| | IMMFC | 24.78 | 21.00 | 21.42 | 22.70 | 23.36 |
| | ICFSKM | 0.506 | 1.642 | 2.357 | 3.066 | 6.464 |
| | E_ICFSMR | 0.514 | **1.043** | **1.298** | **2.972** | **4.283** |

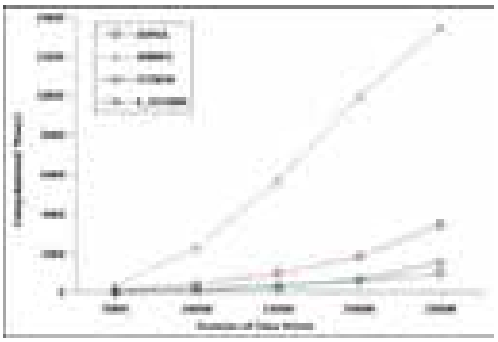
Table X, it can also be observed that the computational times of IAPNA and ICFSKM increase steadily with the data added chunk by chunk, while that of E_ICFSMR changes slightly. Most importantly, the proposed E_ICFSMR requires much less time than IAPNA, IMMFC, and ICFSKM in most cases. Such results indicate that E_ICFSMR can achieve higher efficiency than IAPNA, IMMFC, and ICFSKM in terms of computational time.

### C. Results on Monitoring Time Series

In this section, E_ICFSMR, IAPNA IMMFC, and ICFSKM are tested to discover patterns in air quality and traffic monitoring time series AQMTS and traffic monitoring time series (TMTS). AQMTS [33] is collected every hour from 437 air quality stations covering 43 cities in China over a period of one year (from May 1, 2014 to April 30, 2015). In total, there are 2 891 393 air quality records with six



(a)



(b)

Fig. 6. Experimental results on air quality monitoring time series with different number of data points. (a) Purity results (solid line) and NMI results (dotted line). (b) Computational time comparison for the methods.

indicators, including PM2.5, PM10, $NO_2$, $SO_2$, $O_3$, and CO. We test the methods E_ICFSMR, IAPNA, IMMFC, and ICFSKM on AQMTS to cluster the data objects based on PM2.5. After deleting the incomplete records in the preprocessing, the first 500 objects are used as the initial batch of data, and other remaining objects are added group by group (500 points in each group) according to their acquisition time. Since the accuracy of clustering is sensitive to the cluster number, we study the performance of the methods in terms of purity, NMI, and computational time. All other settings are similar to previous experiments and the experimental results are shown in Fig. 6.

It is noteworthy that the proposed E_ICFSMR is superior to other three methods in terms of computational time, meanwhile it yields better or comparable purity and NMI on this large data set.

TMTS consists of 254 video sequences collected from a single stationary traffic camera interstate I-5 in Seattle within two days [34]. Each video is recorded in 5 s with a resolution of $320 \times 240$ pixels. In this paper, each sequence is converted to grayscale, clipped to $48 \times 48$ pixels over the area with the most total motion and normalized to [0, 1] pixel values. Fig. 7 shows four typical sequences for low-, medium-, and high-traffic patterns. Similar to the previous experiment on AQMTS, the first 54 videos are selected as the initial objects, and the remaining videos are added one by one. For IMMFC, we set the number of clusters $k$ to be 3 to include different traffic patterns, and all other parameters are set similar to previous experiments. After the 254th video is assigned, the final clustering results are obtained in Table XI.

Fig. 7. Four typical video sequences corresponding to low-, medium-, and high-traffic patterns.

TABLE XI
EXPERIMENTAL RESULTS ON TRAFFIC MONITORING TIME SERIES

| | E_ICFSMR | IAPNA | IMMFC | ICFSKM |
|---|---|---|---|---|
| Purity | **0.748** | 0.650 | 0.685 | 0.715 |
| NMI | **0.289** | 0.129 | 0.158 | 0.245 |
| Accuracy | 0.638 | 0.650 | **0.661** | 0.630 |
| Time(s) | **2.294** | 2.430 | 6.050 | 2.656 |

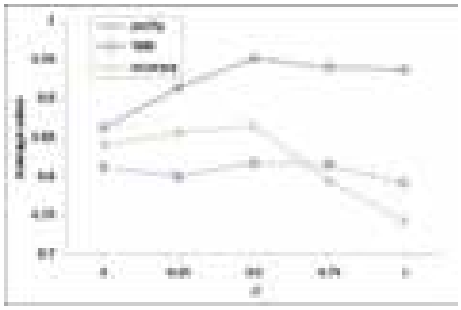


Fig. 8. Parameter influence of $\alpha$ on E_ICFSMR in terms of purity, NMI, and accuracy, respectively.

It can be observed from Table XI that the proposed E_ICFSMR yields higher purity, NMI, and comparable accuracy compared to IAPNA, IMMFC, and ICFSKM on the video data set. Moreover, it outperforms other three methods in terms of the computational time.

The above-mentioned results indicate that the proposed method is promising for clustering large time series data.

### *D. Parameter Sensitivity Tests*

In this section, we investigate the influence of the parameter $\alpha$ on the performance of the proposed method. Specifically, $\alpha$ is sampled from {0, 0.25, 0.5, 0.75, 1}. As an example, we test on the Smile data set and investigate the performances in terms of purity, NMI, accuracy, and computational time. The results are shown in Fig. 8. It can be seen that, with $\alpha$ increasing from 0 to 1, the average values of purity and accuracy increase first and then decrease, while the NMI performance is relatively stable. This is because when $\alpha \to 0$, no shrink happens for all the representatives toward the cluster center. Thus, the clusters (or objects) with different dominant patterns may be merged due to the effect of abnormal representatives. When $\alpha \to 1$, all the representatives are shrunk to the cluster center, meanwhile the distances between clusters become larger. So the clusters with the same dominant pattern cannot be merged. Both will affect the purity and accuracy of clustering results. By selecting a suitable $\alpha$, E_ICFSMR can mitigate the effects of abnormal representatives and capture the geometry of clusters robustly. Moreover, the corresponding computational times {0.288, 0.271, 0.282, 0.287, 0.267} are slightly different for different values of $\alpha$. Therefore, we set $\alpha = 0.5$ to achieve promising and stable results in the proposed methods.

## V. CONCLUSION

In this paper, we focus on incremental clustering for large dynamic data by improving the traditional CFS. We first point out the difficulties in designing new ICFS clustering methods. We then propose two novel incremental clustering algorithms, referred to as ICFSMR and E_ICFSMR. The proposed ICFSMR is inspired by combing CFS with the K-centroids-based clustering, since CFS clustering is good at finding initial cluster centroids while the K-centroids-based method is good at modifying the clustering structure according to new arriving objects. However, one concern is that K-centroids-based methods, usually with one representative for one cluster, are generally unsuitable for nonspherical and unbalanced data. Therefore, to dynamically describe the clustering patterns, multiple representatives are identified to capture the physical shape and geometry of the clusters automatically. To further tackle the problem that the number of clusters cannot be automatically adjusted according to new arriving objects in ICFSMR and E_ICFSMR is proposed. It employs the one-time cluster splitting and merging strategy to adapt clustering patterns periodically.

Experiment results on several standard data sets and real-word time series demonstrate that the proposed methods outperform the non-ICFS in terms of effectiveness and efficiency. Moreover, the proposed E_ICFSMR achieves not only higher or comparable clustering results, but also requires less computational time when compared with several state-of-the-art incremental algorithms, IAPNA, IMMFC, and ICFSKM. Such results indicate that the proposed methods are promising for clustering large data.

Our future research will focus on further improvements of the ICFS clustering, such as extending the methods to heterogeneous data clustering and incomplete data clustering.

## ACKNOWLEDGMENT

The authors would like to thank L. Sun for providing the source code of IAPNA algorithm.

## REFERENCES

[1] L. Zhao, Z. Chen, and Y. Yang, "Parameter-free incremental co-clustering for multi-modal data in cyber-physical-social systems," *IEEE Access*, vol. 5, pp. 21852–21861, 2017.

[2] Z. Ma and A. Leijon, "Bayesian estimation of beta mixture models with variational inference," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2160–2173, Nov. 2011.

[3] L. Zhao, Z. Chen, Y. Hu, G. Min, and Z. Jiang, "Distributed feature selection for efficient economic big data analysis," *IEEE Trans. Big Data*, vol. 4, no. 2, pp. 164–176, Jun. 2018.

[4] Z. Ma, A. E. Teschendorff, A. Leijon, Y. Qiao, H. Zhang, and J. Guo, "Variational Bayesian matrix factorization for bounded support data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 876–889, Apr. 2015.
[5] Forbes. *In Search of the True Value in the Internet Of Things*. Accessed: May 10, 2015. [Online]. Available: http://www.forbes.com/sites/joemckendrick/2015/05/10/in-search-of-the-true-value-in-the-internet-of-things/#12bf3e345f00
[6] J. Manyika *et al.*, *Big Data: The Next Frontier for Innovation, Competition, and Productivity*. New York, NY, USA: McKinsey Global Institute, 2011, pp. 1–137.
[7] D. Ryżko, P. Gawrysiak, M. Kryszkiewicz, and H. Rybiński, *Machine Intelligence and Big Data in Industry*. Berlin, Germany: Springer, 2016.
[8] Z. Ma, J.-H. Xue, A. Leijon, Z.-H. Tan, Z. Yang, and J. Guo, "Decorrelation of neutral vector variables: Theory and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 129–143, Jan. 2018.
[9] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.
[10] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "High-order possibilistic c-means algorithms based on tensor decompositions for big data in IoT," *Inf. Fusion*, vol. 39, pp. 72–80, Jan. 2018.
[11] S. Chakraborty and N. K. Nagwani, "Analysis and study of incremental K-means clustering algorithm," in *High Performance Architecture and Grid Computing*. Berlin, Germany: Springer, 2011, pp. 338–341.
[12] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 3, pp. 515–528, May/Jun. 2003.
[13] G. F. Tzortzis and A. C. Likas, "The global kernel *k*-means algorithm for clustering in feature space," *IEEE Trans. Neural Netw.*, vol. 20, no. 7, pp. 1181–1194, Jul. 2009.
[14] D. H. Widyantoro, T. R. Ioerger, and J. Yen, "An incremental approach to building a cluster hierarchy," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2002, pp. 705–708.
[15] N. Sahoo, J. Callan, R. Krishnan, G. Duncan, and R. Padman, "Incremental hierarchical clustering of text documents," in *Proc. 15th Int. Conf. Inf. Knowl. Manage.*, 2006, pp. 357–366.
[16] P. A. Vijaya, M. N. Murty, and D. K. Subramanian, "Leaders–Subleaders: An efficient hierarchical clustering algorithm for large data sets," *Pattern Recognit. Lett.*, vol. 25, no. 4, pp. 505–513, 2004.
[17] M. Ester, H.-P. Kriegel, J. Sander, M. Wimmer, and X. Xu, "Incremental clustering for mining in a data warehousing environment," in *Proc. 24th Int. Conf. Very Large Data Bases*, 1998, pp. 323–333.
[18] S. Singh and A. Awekar, "Incremental shared nearest neighbor density-based clustering," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 1533–1536.
[19] A. M. Bakr, N. M. Ghanem, and M. A. Ismail, "Efficient incremental density-based algorithm for clustering large datasets," *Alexandria Eng. J.*, vol. 54, no. 4, pp. 1147–1154, 2015.
[20] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.
[21] X. Zhang, C. Furtlehner, and M. Sebag, "Frugal and online affinity propagation," in *Proc. Conf. Francophone sur l'Apprentissage (CAP)*, 2008. [Online]. Available: https://hal.inria.fr/inria-00287381
[22] L. Sun and C. Guo, "Incremental affinity propagation clustering based on message passing," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 11, pp. 2731–2744, Nov. 2014.
[23] F. Wang, C. Tan, A. C. König, and P. Li, "Efficient document clustering via online nonnegative matrix factorizations," in *Proc. SIAM Int. Conf. Data Mining*, 2011, pp. 908–919.
[24] P. Hore, L. O. Hall, D. B. Goldgof, and W. Cheng, "Online fuzzy c means," in *Proc. IEEE Annu. Meeting North Amer., Fuzzy Inf. Process. Soc.*, 2008, pp. 1–5.
[25] P. Hore, L. O. Hall, D. B. Goldgof, Y. Gu, A. A. Maudsley, and A. Darkazanli, "A scalable framework for segmenting magnetic resonance images," *J. Signal Process. Syst.*, vol. 54, nos. 1–3, pp. 183–203, 2009.
[26] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, "Fuzzy c-means algorithms for very large data," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, Dec. 2012.
[27] J.-P. Mei and L. Chen, "Fuzzy clustering with weighted medoids for relational data," *Pattern Recognit.*, vol. 43, no. 5, pp. 1964–1974, 2010.
[28] Y. Wang, L. Chen, and J.-P. Mei, "Incremental fuzzy clustering with multiple medoids for large data," *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 6, pp. 1557–1568, Dec. 2014.
[29] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.
[30] Q. Zhang, C. Zhu, L. T. Yang, Z. Chen, L. Zhao, and P. Li, "An incremental CFS algorithm for clustering large data in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 1193–1201, Jun. 2017.
[31] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 1998, pp. 73–84.
[32] L. Zhao, Z. Chen, and Y. Yang, "Incremental CFS clustering on large data," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Nov. 2017, pp. 687–690.
[33] Y. Zheng *et al.*, "Forecasting fine-grained air quality based on big data," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 2267–2276.
[34] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 909–926, May 2008.

**Liang Zhao** received the Ph.D. and M.S. degrees from the Dalian University of Technology, Dalian, China, in 2018 and 2014, respectively.

He is currently a Teacher with the School of Software Technology, Dalian University of Technology. His current research interests include big data and artificial intelligence.

**Zhikui Chen** (M'11) received the Ph.D. and M.S. degrees from Chongqing University, Chongqing, China, in 1998 and 1993, respectively.

He is currently a Professor with the Dalian University of Technology, Dalian, China. He is a Lead of the Institute of Ubiquitous Network and Computing, Dalian University of Technology. His current research interests include big data processing, mobile cloud computing, ubiquitous network and its computing.

**Yi Yang** received the Ph.D. degree from the Nanjing University of Science and Technology, Nanjing, China, in 2008.

Since 2013, she has been an Associate Professor with the School of Reliability and Systems Engineering, Beihang University, Beijing, China. Her current research interests include reliability analysis and design, repairable system, and control science and engineering.

**Liang Zou** received the Ph.D. degree from the University of British Columbia, Vancouver, BC, Canada, in 2017, and the M.Sc. degree from the University of Science and Technology of China, Hefei, China, in 2013.

His current research interest includes biomedical signal processing and bioinformatics.

**Z. Jane Wang** (F'17) received the B.Sc. degree from Tsinghua University, Beijing, China, in 1996, and the Ph.D. degree from the University of Connecticut, Storrs, CT, USA, in 2002.

Since 2004, she has been with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada, where she is currently a Professor. Her current research interests include the broad areas of statistical signal processing theory and applications.