Dynamic Rough-Fuzzy Support Vector Clustering

Ramiro Saltos, Richard Weber, *Senior Member, IEEE*, and Sebastián Maldonado, *Member, IEEE*

***Abstract*—Clustering is one of the main data mining tasks with many proven techniques and successful real-world applications. However, in changing environments, the existing systems need to be regularly updated in order to describe in the best possible way an observed phenomenon at each point in time. Since changes lead to uncertainty, the respective systems also require an adequate modeling of the involved kinds of uncertainty. This paper presents a novel method for dynamic clustering called dynamic rough-fuzzy support vector clustering (D-RFSVC). Its main idea is to take advantage of the knowledge acquired in previous cycles to speed up model updating while tracking the structural changes that clusters can experience over time. The core method of the proposed approach is the well-known support vector clustering algorithm, which can be used for large datasets employing powerful optimization techniques. The computational experiments, together with a conceptual and numerical comparative study, highlight the potential D-RFSVC has in dynamic environments.**

***Index Terms*—Dynamic data mining, fuzzy systems, support vector clustering (SVC), visualization.**

I. Introduction

CLUSTERING is one of the main approaches for reducing the complexity related to huge volumes of data. A plethora of methods have been proposed for this purpose, which are used, e.g., to segment customers, images, products, or any other relevant kind of observations. Most of these applications, however, are static in the sense that they analyze data snapshots. On the other hand, almost all real-world phenomena are dynamic and characterized by data structures that change over time. Thus, clustering dynamically changing data plays an important role in practice and will be one of the main original contributions of this paper.

Most of the organizations are challenged, e.g., by varying environments, which underlines the necessity of adapting systems for decision support. Dynamic clustering for customer segmentation is an example where updated information on customer behavior could lead to better decisions. Obviously, market players who detect changes of customer preferences first might obtain a strategic advantage over their competitors. But business-related applications are not the only ones that require evolving systems. Other domains also have to face dynamism, e.g., in healthcare where patients are monitored over time; crime analytics, where the dynamics of the so-called hotspots are studied [1]; and analysis of weather data (see Section V-C), to mention just a few. This has motivated an increasing number of researchers to enrich and extend traditional static clustering by dynamic versions [2]–[6].

In general, changes lead to uncertainty, thus making uncertainty modeling especially important for dynamic clustering [7], [8]. Many related clustering approaches have been developed, from areas such as probability theory [9], fuzzy set theory [10], possibility theory, and rough set theory, among others, addressing different kinds of uncertainty. In this paper, we focus on a combination of fuzzy logic with rough sets, which offers particular advantages for dynamic clustering, as will be shown below.

Although many dynamic clustering algorithms have proven their advantages in real-world applications, such as traffic management [2], customer segmentation [5], and weather analysis [11], among others, they maintain the main drawbacks of the core methods used for the clustering process. These are spherical cluster shapes, cluster centers as prototypes, and an inadequate treatment of outliers. However, real-world datasets are not necessarily distributed spherically, so center-based prototypes are not always the best representatives. Furthermore, it might be important to recognize outliers accordingly [12], especially in a dynamic setting, since these outliers could provide important information on upcoming changes in the underlying data structure.

These considerations motivated the development of a dynamic clustering algorithm that is able to deal with flexible cluster structures and prototypes, providing degrees of membership of outliers to the clusters found in order to trace their evolution over time. Support vector clustering (SVC) [13] and its soft computing derivative, rough-fuzzy support vector clustering (RFSVC) [14], are two algorithms that fulfill the previously mentioned properties; however, they work only on static datasets.

The objective of this paper is to introduce dynamic rough-fuzzy support vector clustering (D-RFSVC) comprehensively. Each step of the respective method consists of a sound and novel mathematical development to treat changing data structures. D-RFSVC is applied to artificial, benchmark, and real-world datasets.

In Section II, some basic concepts of dynamic clustering and related algorithms are discussed briefly. In the subsequent

Manuscript received January 15, 2017; revised June 8, 2017; accepted August 4, 2017. Date of publication August 17, 2017; date of current version November 29, 2017. This work was supported in part by CONICYT (CONICYT-PCHA/Doctorado Nacional/2014), by SENESCYT (www.senescyt.gob.ec), by the Ph.D. program on Engineering Systems (www.dsi.uchile.cl), by the Institute on Complex Engineering Systems (www.isci.cl; ICM: P-05-004-F, CONICYT: FB0816), and by Fondecyt (1140831 and 1160738). *(Corresponding author: Ramiro Saltos.)*

R. Saltos and R. Weber are with the Department of Industrial Engineering, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, Santiago, Chile (e-mail: ramiro.saltos@ing.uchile.cl; rweber@dii.uchile.cl).

S. Maldonado is with the Facultad de Ingeniería y Ciencias Aplicadas, Universidad de Los Andes, Santiago, Chile (e-mail: smaldonado@uandes.cl).

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/TFUZZ.2017.2741442

section, we present the static version of RFSVC. Section IV introduces the details of the proposed method. The results of our experiments are shown in Section V. Finally, Section VI concludes this paper and provides ideas for possible future developments.

II. Literature Overview

A. Foundations of Dynamic Clustering

The objective of traditional (static) clustering is to group similar objects in one cluster and dissimilar objects in different clusters. The respective clustering algorithms require static datasets, in the sense that both objects to be clustered and clusters have to be static. An object is considered to be static if its feature vector contains just current values, i.e., feature values are "snapshots" and do not reflect development over time. Similarly, clusters are static if their respective structure does not change over time, i.e., the number of clusters, their shape, and position remain stable [4]. Allowing objects and/or clusters to become dynamic leads to the following categories of dynamic cluster analysis [2], [15].

1) *Static objects and static clusters:* The traditional case where static data are observed and there is no need to update the respective model.
2) *Dynamic objects and static clusters:* The case where feature values change over time, leading to feature trajectories rather than feature values.
3) *Static objects and dynamic clusters:* In this case, the existing objects maintain their previous feature values; however, new objects arrive at the dataset causing possibly a changing cluster structure.
4) *Dynamic objects and dynamic clusters:* The most general case, where objects, as well as clusters, can change over time.

To cope with the challenges posed by the dynamics of a given environment, soft computing approaches are of particular interest since changes lead to uncertainty that has to be treated adequately [4]. As a consequence, techniques for uncertainty modeling have been integrated successfully into dynamic clustering algorithms [2], [5], [7], [11], [16]–[18].

A common characteristic of most of these algorithms is the use of the so-called *dynamic clustering cycle* [4], which consists of the following generic steps.

1) The classifier with its respective parameters is applied to an initial dataset.
2) The dataset is updated in each new cycle by adding new data and, optionally, by removing data from obsolete clusters.
3) The static cluster algorithm is applied to classify the new dataset.
4) The classification results are analyzed for structural changes.
5) If relevant changes are detected, cluster parameters are updated.
6) Go back to Step 2.

In the case of a particular algorithm for dynamic clustering with soft computing approaches, these generic steps have to be specified, and criteria for model updating have to be developed. Various structural changes have been proposed in the literature; the most common ones are the following:

1) Create new clusters.
2) Move current clusters.
3) Merge existing clusters.
4) Eliminate obsolete clusters.

For each of these structural changes, specific conditions have to be accomplished in order to be recognized by the respective dynamic clustering scheme, as will be shown in Section II-B.

B. Related Dynamic Clustering Algorithms

Dynamic clustering algorithms can deal with static/dynamic objects and static/dynamic clusters; see Section II-A. The case of dynamic objects is usually known as *clustering of trajectories*, which goes beyond the scope of this paper; see, e.g., [4], [16], and [19] for related work. The case of static objects and dynamic clusters is relevant for the present paper. Hence, we briefly review the respective algorithms.

1) Dynamic Fuzzy c-Means (D-FCM): Crespo and Weber [2] presented D-FCM, one of the first dynamic clustering algorithms that integrate soft computing paradigms to deal with static objects and dynamic clusters. It is based on the dynamic clustering cycle using fuzzy c-means [20] as core clustering algorithm. Although D-FCM has been applied successfully to solve real-world problems, it needs several user-specified parameters (e.g., number of clusters), it is sensitive to outliers and noise, the membership degrees of each object to all clusters must sum 1, and the cluster shape is assumed to be spherical.

2) Dynamic Rough c-Means (D-RCM): Consolidating previous work [17], [18], Peters *et al.* [5] introduced D-RCM. The core clustering method is rough c-means [21] refined by techniques proposed by Peters [22]. Similar to D-FCM, the algorithm adapts the generic dynamic clustering cycle to address particular changes in the classifier structure. The novel contribution in D-RCM is the way changes in the level of uncertainty are managed. Peters *et al.* [5] proposed maintaining the roughness of the initial clusters, which is defined as the fraction of elements in the lower approximation of any cluster and the number of all available objects.

3) Evolving Dynamic Data Assigning Assessment (E-DDAA): Georgieva and Klawonn [11] presented E-DDAA as a dynamic generalization of their static algorithm, dynamic data assignment assessment (DDAA) [23]. The aim of DDAA is to find interesting clusters in a given dataset considering that a large part of the data can be classified as noise. It discovers interesting patterns in terms of single clusters that might cover a small proportion of the data instead of partitioning the whole dataset as in other classical clustering methods. E-DDAA has been applied successfully to weather analysis showing its potential for identifying interesting clusters without knowing the number of clusters in advance. However, cluster elimination has not been considered explicitly in this approach.

4) Dynamic DBSCAN (D-DBSCAN): Mary and Kumar [24] proposed D-DBSCAN, which is an extension of the well-known DBSCAN algorithm [25]. The main structural changes the algorithm detects are the creation of new clusters and merging

TABLE I
RELATED APPROACHES OF DYNAMIC CLUSTERING

Author	Year	Core Method	Partition	Prototypes	Shape
Crespo *et al.*	2005	FCM	Fuzzy	Centers	Spherical
Peters *et al.*	2012	RCM	Rough	Centers	Spherical
Georgieva *et al.*	2008	DDAA	Crisp and Fuzzy	Centers	Spherical
Mary *et al.*	2012	DBSCAN	Crisp	No	Any
Dehideniya *et al.*	2013	ABC	Crisp	Centers	Spherical

TABLE II
STRUCTURAL CHANGES ADDRESSED BY DYNAMIC CLUSTERING ALGORITHMS

	D-FCM	D-RCM	E-DDAA	D-DBSCAN	ABDC
Creation	✓	✓	✓	✓	✓
Deletion	✓	✓	-	-	✓
Movement	✓	-	✓	-	-
Merging	-	-	✓	✓	✓
Change of Uncertainty	-	✓	-	-	-
Change of Shape	-	-	-	-	-
Outlier Traceability	-	-	-	-	-

of the existing ones. Given that D-DBSCAN uses the density concepts inherited from DBSCAN, Mary's algorithm has the ability to deal with clusters of any shape without knowing *a priori* their number and to detect outliers at the same time. However, elimination and movement of clusters, as well as an adequate modeling of uncertainty, are not considered.

5) Agent-Based Dynamic Clustering (ABDC): Dehideniya and Karunananda [16] introduced ABDC, a crisp dynamic partitional clustering algorithm. It is based on multiagent technology in which a group of agents, represented by their data records, negotiate decisions that can improve the configuration of the current clusters. The algorithm does not need the number of clusters as an input parameter. Since the ideas presented in [16] are mostly conceptual, without presenting mathematical details, we consider this algorithm to be still under development.

Another related research field is *data stream clustering*, which corresponds to the clustering of data points that arrive as a continuous flux [26], [27]. This is particularly interesting in online applications, like, e.g., the analysis of multimedia data or web pages, which deal with large datasets where retraining the clustering algorithms becomes intractable [26]. Several algorithms have been proposed for data stream clustering, which usually are either partitioning-based or density-based. A comprehensible literature review was developed in [28]–[30].

In contrast to the topics discussed in the present paper, data stream clustering aims at constructing a good segmentation model of the streaming data efficiently in terms of computational time and memory [26], rather than analyzing and modeling the changes that occur over time. In fact, data stream algorithms have to be able to manage the data in a single pass, i.e., they can look at a data point only once before making a decision about its membership [31]. This constraint does not exist in the context of dynamic clustering. Hence, we concentrate in our paper on dynamic clustering rather than on data stream clustering.

C. Conceptual Comparative Analysis of Algorithms

Table I presents the main characteristics of each previously explained algorithm in order to identify already achieved advances and to reveal the gaps that motivate future research. It focuses on properties such as the core method used for the clustering process, the type of partition obtained, the prototypes representing the clusters, and the assumed cluster shape.

Table II shows the main structural changes that clusters can undergo, such as creation, elimination, movement, merging, and the change in the level of uncertainty. Additionally, outlier traceability is an important issue that must be considered within dynamic clustering given that an outlier can become an element of a cluster in the course of the actions.

Tables I and II clearly identify "white spots" in the research landscape of dynamic clustering, emphasizing the necessity of developing advanced algorithms capable of revealing diverse structural changes, and providing outlier traceability at the same time.

III. ROUGH-FUZZY SUPPORT VECTOR CLUSTERING

Saltos and Weber [14] proposed RFSVC, which is the core method of the approach introduced in this paper. RFSVC constructs a fuzzy partition of the data using support vector domain description (SVDD) [32], where outliers can be clearly identified and distinguished from the clusters found.

The RFSVC algorithm consists of three steps that will be described in the following subsections in more detail. The *training step* uses SVDD to obtain a hypersphere (in a higher dimensional projected feature space) that encloses most of the data points. All observations that fall outside its boundary are considered as outliers. In the subsequent *labeling step* [13], different classes of the set of data points enclosed by the hypersphere are identified. Finally, a *fuzzification step* is performed for those objects that were considered as outliers in the training step. A formal description of the three steps follows.

A. Training Step

Let $X = \{\mathbf{x}_i \in \mathcal{R}^d / i = 1, \ldots, N\}$ be the set of N data points of dimension d. First, the observations are projected to a reproducing kernel Hilbert space, where a hypersphere with minimal radius is constructed that encloses most of the training objects. The following convex quadratic optimization problem (SVDD) is solved:

$$\text{Min}_{R,\boldsymbol{a},\boldsymbol{\xi}}\ R^2 + C\sum_{i=1}^{N}\xi_i \tag{1}$$

s.t.

$$\| \phi(\mathbf{x}_i) - \boldsymbol{a} \|^2 \leq R^2 + \xi_i \qquad \forall i = 1, \ldots, N \tag{2}$$

$$\xi_i \geq 0 \qquad \forall i = 1, \ldots, N \tag{3}$$

where R is the radius of the hypersphere and $\boldsymbol{a}$ its center, ϕ is a nonlinear mapping function, $\boldsymbol{\xi}$ is a vector of slack variables used to allow observations falling outside the boundary of the hypersphere, $\| \cdot \|$ is the Euclidean norm, and $C \in [0, 1]$ is a constant

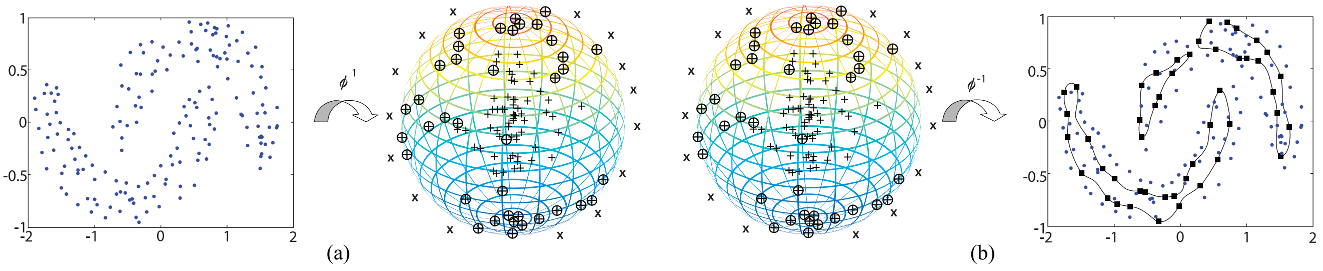


Fig. 1. General idea of RFSVC (+: IDs, ⊕: SVs, **x**: BSVs). (a) Projection from original data space to higher-dimensional space. (b) Inverse projection from enclosing sphere to cluster contours [14].

regularization parameter that controls the tradeoff between the volume of the sphere and the number of data points it includes. The dual formulation of SVDD follows:

$$\text{Max}_{\boldsymbol{\beta}} \sum_{i=1}^{N} \beta_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4}$$

s.t.

$$\sum_{i=1}^{N} \beta_i = 1 \tag{5}$$

$$0 \leq \beta_i \leq C \qquad \forall i = 1, \ldots, N \tag{6}$$

where $\boldsymbol{\beta}$ are Lagrange multipliers and $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ is the kernel function. A widely used kernel function is the radial basis function or Gaussian kernel, which has the following form:

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-q\|\mathbf{x}_i - \mathbf{x}_j\|^2} \tag{7}$$

where q is a parameter that controls the kernel's width [33]. We used the generalized sequential minimal optimization (GSMO) algorithm [34] to solve the quadratic optimization problem of the training step.

Only those observations i with $0 < \beta_i < C$ define the contours of the clusters [13]; these are called *support vectors* (SVs). Objects with $\beta_i = 0$ lie inside the hypersphere and are called *inside data points* (ID). Finally, objects with $\beta_i = C$ lie outside the hypersphere and are called *bounded support vectors* (BSV) or *outliers*.

For a given object, $\mathbf{x}$, the distance between its projection and the center of the hypersphere, $\boldsymbol{a}$, is calculated as

$$\begin{aligned} R^2(\mathbf{x}) &= \| \phi(\mathbf{x}) - \boldsymbol{a} \|^2 \\ &= K(\mathbf{x}, \mathbf{x}) - 2\sum_{i=1}^{N} \beta_i K(\mathbf{x}_i, \mathbf{x}) \\ &\quad + \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned} \tag{8}$$

The radius of the hypersphere follows:

$$R_S = \frac{1}{|\text{SSV}|} \sum_{\mathbf{x}_i \in \text{SSV}} R(\mathbf{x}_i) \tag{9}$$

where SSV is the set of SVs and $|\text{SSV}|$ its cardinality.

Fig. 1 illustrates a geometric interpretation of the SVDD algorithm, in which Fig. 1(a) describes the objects' projection to a higher dimensional space and the construction of the hypersphere, while in Fig. 1(b), the images of data points are projected back to the original space.

B. Labeling Step

The outputs of the training step are the sets of SVs, BSVs, and IDs. The main drawback from a clustering perspective is that many clusters may coexist within the hypersphere without being distinguished.

To overcome this problem, we will use the labeling strategy called SV graph proposed by Ben-Hur *et al.* [13]. It works as follows: Given two data points from different clusters, $\mathbf{x}_i$ and $\mathbf{x}_j$, any path that connects them must exit the hypersphere, i.e., $\exists\lambda \in [0, 1]$, such that $R(y_{i,j}) > R_S$, where $y_{i,j} = y_{i,j}(\lambda) = \lambda\mathbf{x}_i + (1 - \lambda)\mathbf{x}_j$. This leads to the following definition of the adjacency matrix A, whose elements $a_{i,j}$ represent whether a pair of points $\mathbf{x}_i$ and $\mathbf{x}_j$ belongs to the same cluster

$$a_{i,j} = \begin{cases} 1, & \text{if } R(y_{i,j}) \leq R_S, \forall\lambda \in [0, 1] \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Clusters are now defined as the connected components of the graph induced by A. Note that the BSVs remain unclassified since they lie outside the enclosing sphere. Studying alternative labeling approaches would go beyond the scope of this paper; see, e.g., [35], [36], and [37].

C. Fuzzification Step

Saltos and Weber [14] proposed a fuzzification step in order to calculate membership degrees for BSVs with respect to the clusters created in the preceding step. The strategy follows.

1) Cast the hard cluster structure established in the training step into a rough-fuzzy one based on two components: a lower approximation and a fuzzy boundary.
2) Assign the SVs and IDs to the lower approximations of their respective clusters according to the labeling step.
3) Assign the BSVs to the fuzzy boundaries of all clusters.
4) Calculate the membership degree $\mu_{i,k}$ of BSV i to cluster k using the following formula:

$$\begin{aligned} \mu_{i,k} &= \mu(\text{BSV}_i, \text{SV}_{k,i}) = K(\text{BSV}_i, \text{SV}_{k,i}) \\ &= e^{-q\|\text{BSV}_i - \text{SV}_{k,i}\|^2} \end{aligned} \tag{11}$$

TABLE III
OUTPUTS OF THE RFSVC ALGORITHM

Step	Output		
Training	Sets of SV, BSV, and ID		
Labeling	$\mu_{i,k} \in \{0,1\}$	$\forall i \in \text{SV} \cup \text{ID}$	$\forall k$
	$\mu_{i,k} = 0$	$\forall i \in \text{BSV}$	$\forall k$
Fuzzification	$\mu_{i,k} \in [0,1)$	$\forall i \in \text{BSV}$	$\forall k$

where $\text{SV}_{k,i}$ is the SV in cluster k, which is closest to the BSV i.

The use of the kernel function as the membership function in (11) is justified by Theorem 1, which has already been formulated and proven by Saltos and Weber [14].

Theorem 1: Let $\mathbf{x}_i,\ \mathbf{x}_j,\ \mathbf{x}_k\ \in \mathcal{R}^d$ be elements of a d-dimensional data space. Let ϕ be a nonlinear transformation from $\mathcal{R}^d$ to some higher dimensional space with $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = e^{-q\|\mathbf{x}_i - \mathbf{x}_j\|^2}$ the Gaussian kernel function with width parameter $q > 0$. Let $\|\cdot\|$ be the Euclidean norm and $d(\cdot,\cdot)$, $d'(\cdot,\cdot)$ be the Euclidean distance in the original data space and the higher dimensional projected space, respectively. Then,

$$d(\mathbf{x}_i, \mathbf{x}_j) \le d(\mathbf{x}_i, \mathbf{x}_k) \iff d'(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \le d'(\phi(\mathbf{x}_i), \phi(\mathbf{x}_k)).$$

For the static case, this theorem guarantees that the order relation in the data space is maintained in the higher dimensional space after applying the nonlinear transformation ϕ. Below, we will extend this Theorem to the dynamic case; see Section IV-B3.

To summarize, Table III provides a description of the outputs of each step of RFSVC, where $\mu_{i,k}$ is the membership degree of data point i to cluster k.

IV. DYNAMIC ROUGH-FUZZY SUPPORT VECTOR CLUSTERING

We first introduce structural changes that D-RFSVC will detect when new objects arrive. Then, in Section IV-B, the respective method will be presented in detail, and we will show how the classifier is adapted to those changes. Finally, in Section IV-C, we will present the algorithm's pseudocode and compare it conceptually with alternative algorithms.

A. Addressed Structural Changes

As shown in Table II, the possible structural changes that clusters can experience are creation, elimination, movement, merging, and splitting, along with changes of their level of uncertainty. Most of the existing dynamic clustering algorithms do not treat all of these changes. In order to overcome these limitations, the method proposed in this paper addresses the structural changes as is explained in the following subsections (without loss of generality and to simplify presentation, we treat these changes independently).

1) Creation of New Clusters: A cluster can be considered to be a cloud of points with high relative density. As a consequence, new clusters have to be created when new objects

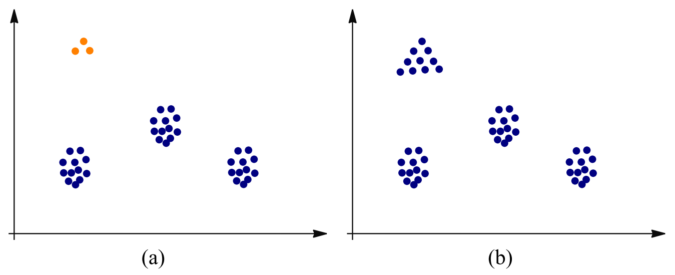


Fig. 2. Creation of clusters by accumulation. (a) Cycle $t = s$. (b) Cycle $t = s + \Delta t$.

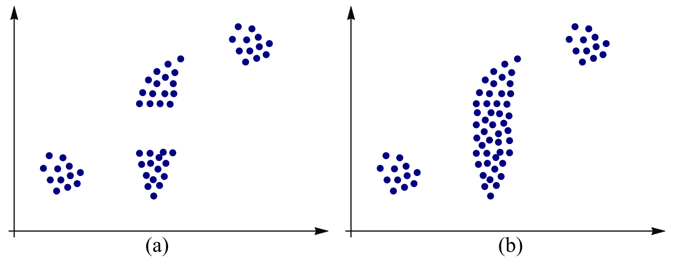


Fig. 3. Creation of clusters by merging. (a) Cycle $t = s$. (b) Cycle $t = s + \Delta t$.

enter the dataset forming clouds of high density (*Creation by Accumulation*). If one or more new clusters appear by accumulation, the number of columns of the membership matrix [see formula (11)] will be increased by the number of clusters that were created in that period. At the same time, a subset of BSVs will become SVs or IDs. Fig. 2 shows this phenomenon, where the three orange (light) points in the upper left of part (a) are not yet a cluster.

2) Merging Clusters: Another scenario is when the new objects arrive in between two or more clusters. In this case, the clusters affected could merge in future periods. Under this premise, it could be argued that merging clusters is a particular case of cluster creation (see Fig. 3), but we treat this kind of change separately due to some conceptual differences.

If $cm > 1$ clusters merge, the number of clusters and, hence, the number of columns of the membership matrix will be reduced by $cm - 1$. At the same time, a subset of BSVs will become SVs or IDs, and a subset of SVs will become IDs.

3) Elimination of Clusters: When a cluster does not receive enough new observations to maintain its relative density compared to the other ones, it should be eliminated. This case can be inferred from the membership matrix as follows.

1) If one or more clusters disappear, the number of columns of the membership matrix will be reduced by the number of dying clusters. At the same time, the set of SVs and IDs that belonged to those clusters will become BSVs.

It is important to highlight that objects of dying clusters will not be eliminated from the dataset, given that in future cycles they can receive new observations as neighbors and revive as a cluster. This is an important difference between the proposed method and current state-of-the-art dynamic clustering algorithms.

4) Splitting Clusters: The opposite case of merging clusters is splitting clusters. If a connecting area of a large cluster does not receive new observations, its density will go down, possibly leading to a split into two or more smaller clusters. In this case, the number of columns of the membership matrix will be increased by $cs - 1$, where $cs > 1$ is the number of fragments into that the larger cluster splits. Additionally, some SVs and IDs will become BSVs.

5) Other Structural Changes: Apart from creating, merging, eliminating, and splitting clusters, our algorithm is capable of detecting other kinds of structural changes, such as the following.

1) *Change of shape:* New observations that appear in the existing clusters can alter their original silhouette. Since D-RFSVC uses SVs as cluster representation instead of cluster centers, changing shapes can be accomplished easily.
2) *Contraction:* When a cluster receives fewer new observations than others, its relative density will decrease gradually. This fact can be observed when its SVs get closer to each other and the number of BSVs increases. If this phenomenon continues for several periods, all SVs and IDs will become BSVs, finally causing the elimination of the cluster. Identifying cluster contraction is especially important in dynamic clustering since it could anticipate elimination of clusters, as will be shown in our experiments; see, e.g., Fig. 5.
3) *Dilatation:* This is the opposite case of contraction. In this scenario, a cluster receives many more new objects than the others, dispersing its SVs and, as a consequence, absorbing part of the nearby BSVs. Cluster dilatation could anticipate the merging of clusters.
4) *Change of the level of uncertainty:* The level of uncertainty, also called roughness in D-RCM, is automatically maintained stable, as will be shown in Section IV-B.
5) *Outlier traceability:* Tracing outliers (BSVs) could provide important information about structural changes in the respective class structure to happen in subsequent cycles. This can be accomplished by monitoring their membership degrees over the respective cycles.

B. Development of D-RFSVC

The following subsections show how to adapt each step of static RFSVC to the dynamic case.

1) Training Update: As mentioned in Section III-A, we used the GSMO algorithm [34] to solve the quadratic optimization model of the training step. In the static case, it starts with a feasible solution of the quadratic problem, checks first-order conditions, and improves this solution iteratively until convergence is reached.

In the dynamic case, however, new observations arrive, increasing the number of components of vector $\boldsymbol{\beta}$ as solution of model (4)–(6), thus making it infeasible. As a consequence, we propose applying the idea of the two-phase method of optimization from [38], with which we determine a feasible solution in Phase I, using the information provided by the optimal solution of the previous cycle. In Phase II, we then optimize starting from the feasible solution that was determined in Phase I. Next, we introduce the necessary mathematical notation.

Recall model (4)–(6). The variable transformation $\alpha_i = \frac{\beta_i}{C}$ $\forall i = 1, \ldots, N$ leads to the model given in (12)–(14):

$$\text{Max}_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{12}$$

s.t.

$$\sum_{i=1}^{N} \alpha_i = \frac{1}{C} \tag{13}$$

$$0 \leq \alpha_i \leq 1 \qquad \forall i = 1, \ldots, N \tag{14}$$

where $C = \frac{1}{\upsilon N}$ is the penalty constant, $\upsilon \in \left[\frac{1}{N}, 1\right]$ is the ratio of data points that we allow to lie outside the hypersphere in the higher dimensional space, and $\boldsymbol{\alpha}$ is the N-dimensional solution vector of the above model. Now, the sets of SVs, BSVs, and IDs are given by

1) ID $= \{\mathbf{x}_i \in X / \alpha_i = 0\}$.
2) BSV $= \{\mathbf{x}_i \in X / \alpha_i = 1\}$.
3) SV $= \{\mathbf{x}_i \in X / 0 < \alpha_i < 1\}$.

Let $X^{(t)} = \{\mathbf{x}_i \in \mathcal{R}^d / i = 1, 2, \ldots, n_t\}$ be the set of n_t objects that arrived at period t, with $t = 0, 1, 2, \ldots, T$. We define the set:

$$X_t = \bigcup_{s=0}^{t} X^{(s)} \tag{15}$$

as the set of all data points that arrived until period t, with $N_t = |X_t|$. Let $\boldsymbol{\alpha}^t$ be the optimal solution of model (12)–(14) applied to the dataset X_t.

Denote as $C_t = \frac{1}{\upsilon N_t}$ the value of the penalty constant when using the dataset X_t. Since in each cycle t the number of elements in the dataset X_t increases, it follows that $N_0 < N_1 < \ldots < N_t$, and therefore, $C_0 > C_1 > \ldots > C_t$. These facts indicate that the stability of the clusters' roughness is automatically maintained without any additional parameter.

Since the value of C_t decreases, the value of $\frac{1}{C_t}$ increases. This fact leaves a slack in the constraint (13), allowing the variables α_i, associated with the new observations, to become part of the feasible solution of the model (12)–(14).

To summarize, in each period t, given the dataset X_t, we have to solve the quadratic model:

$$\text{Max}_{\boldsymbol{\alpha}} \sum_{i=1}^{N_t} \alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i=1}^{N_t}\sum_{j=1}^{N_t} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{16}$$

s.t.

$$\sum_{i=1}^{N_t} \alpha_i = \frac{1}{C_t} \tag{17}$$

$$0 \leq \alpha_i \leq 1 \qquad \forall i = 1, \ldots, N_t. \tag{18}$$

Since the optimal solution vector for period $t-1$ ($\boldsymbol{\alpha}^{t-1}$) has fewer components than the optimal solution vector for period t

Algorithm 1: Phase I: Determine Feasible Solution.

```
Input: Solution vector $\boldsymbol{\alpha}^{t-1}$, number of data points
    $N_{t-1}$ and $N_t$, parameter $\upsilon \in \left[\frac{1}{N_t}, 1\right]$.
Output: Feasible solution $\boldsymbol{\alpha}^t$ of model (16) - (18).
1  Calculate $C_t = \frac{1}{\upsilon N_t}$
2  Set $s = 0$
3  Set $\boldsymbol{\alpha}^t = 0$
4  for $i = 1, \ldots, N_{t-1}$ do
5      $\alpha_i^t = \alpha_i^{t-1}$
6      $s = s + \alpha_i^t$
7  Set $i = N_{t-1} + 1$
8  while $s < \frac{1}{C_t}$ do
9      if $\frac{1}{C_t} - s \geq 1$ then
10         $\alpha_i^t = 1$
11     else
12         $\alpha_i^t = \frac{1}{C_t} - s$
13     $s = s + \alpha_i^t$
14     $i = i + 1$
```

($\boldsymbol{\alpha}^t$), we apply Phase I in order to determine a feasible solution; see Algorithm 1.

In Phase II, we run the GSMO algorithm using $\boldsymbol{\alpha}^t$ from Phase I as the initial solution in order to determine the set of SVs, BSVs, and IDs for the current cycle. Finally, we compare these new sets with their predecessors to detect the changes between the previous cycle and the current one, among which are the following:

1) data points that become SVs;
2) data points that leave the set of SVs.

These changes are the most important ones because SVs are the prototypes of the clusters, define their contours, and are the basis for the labeling step.

2) Labeling Update: After we have changed the set of SVs from the previous period to the current one, labeling can be updated as follows:

1) If no changes are revealed, the current classifier remains unchanged, and it is only necessary to classify the new objects into the current clusters.
2) If some observations left the set of SVs, we remove their respective rows and columns from the adjacency matrix A.
3) If some data points became SVs, we add their respective rows and columns to the adjacency matrix A.

After updating the adjacency matrix A, it is necessary to identify connected components of the new graph induced by A. This can be done from scratch, or by modifying the algorithm used for this purpose, in order to take advantage of the previous knowledge. Finally, IDs have to be assigned to the new clusters according to the changes made during labeling.

3) Fuzzification Update: All changes applied in the beforementioned steps call for updated membership degrees of previously existing BSVs. For new observations, these values have to be calculated. This can be done as follows.

1) If the set of SVs does not change, we only need to calculate the membership degrees for the new BSVs using (11).
2) If the set of SVs changes, we must check whether the closest SV to the respective BSV is the same or not for each cluster. If so, it is not necessary to change membership values. In the opposite case, the membership degrees have to be updated according to the new closest SV using (11).

In order to use (11) to update or calculate the membership degrees for the BSVs, it is necessary to guarantee that the order relation of the data space has not been changed after the arrival of new data points. To do so, we first postulate the following corollary that follows from Theorem 1.

Corollary 1: Let $X \subset \mathcal{R}^d$ be a set that fulfills Theorem 1. Any set $S \subset X$ with $card(S) \geq 3$ also fulfills Theorem 1.

Proof: By contradiction, if subset S does not fulfill Theorem 1, then $\exists \mathbf{x}_i,\ \mathbf{x}_j,\ \mathbf{x}_k\ \in S$ that do not fulfill $d(\mathbf{x}_i, \mathbf{x}_j) \leq d(\mathbf{x}_i, \mathbf{x}_k) \iff d'(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \leq d'(\phi(\mathbf{x}_i), \phi(\mathbf{x}_k))$. This contradicts the corollary's assumption. ■

This allows us to generalize Theorem 1 to be used in a dynamic environment.

Theorem 2: Let $t = 0, 1, \ldots, T$ be the index for the dynamic clustering cycles and X_t be the database of observations that arrived until cycle t. Let $\mathbf{x}_i^t$, $\mathbf{x}_j^t$, and $\mathbf{x}_k^{t+\Delta t} \in \mathcal{R}^d$ be observations, which entered the database X_t and $X_{t+\Delta t}$, respectively.

Let $\|\cdot\|$ be the Euclidean norm and $d(\cdot,\cdot)$, $d'(\cdot,\cdot)$ be the Euclidean distance in the original data space and in the higher dimensional projected space, respectively. Let ϕ be a nonlinear transformation from $\mathcal{R}^d$ to some higher dimensional space and $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = e^{-q\|\mathbf{x}_i - \mathbf{x}_j\|^2}$ be the Gaussian kernel function with width parameter $q > 0$. Then,

$$d\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) \leq d\left(\mathbf{x}_i^t, \mathbf{x}_k^{t+\Delta t}\right) \iff d'\left(\phi\left(\mathbf{x}_i^t\right), \phi\left(\mathbf{x}_j^t\right)\right) \leq d'\left(\phi\left(\mathbf{x}_i^t\right), \phi\left(\mathbf{x}_k^{t+\Delta t}\right)\right).$$

Proof: Without loss of generality, we establish a time index $s \in \{t + \Delta t, ..., T\}$. Given that data points $\mathbf{x}_i^t$, $\mathbf{x}_j^t$, and $\mathbf{x}_k^{t+\Delta t}$ had entered the dataset X_s in previous cycles, they are now elements of X_s. On the other hand, X_s fulfills Theorem 1 because it can be ignored that it was built by adding new elements over time, i.e., it can be seen as a static dataset. By Corollary 1, any subset of X_s also fulfills Theorem 1. This implies that observations $\mathbf{x}_i^t$, $\mathbf{x}_j^t$, and $\mathbf{x}_k^{t+\Delta t}$ also fulfill Theorem 1, whose equivalence is a particular case of Theorem 2 without time index. ■

Theorem 2 guarantees that the order relation of the data space is maintained over time when new observations enter the database.

An important difference between our Theorem 2 and Lemma 1 presented in [39] is that we explicitly consider BSVs, i.e., outliers. In [39], outliers are explicitly not allowed. We think that this assumption is too restrictive in general in real-world datasets, and here is where the special contribution of this paper takes place: we model outliers explicitly in dynamic soft clustering.

At the end of this step, all necessary updates have been performed, leading to an updated classifier. Note that each of the

Algorithm 2: Dynamic Rough-Fuzzy Support Vector Clustering.

```
Input: Data set $X_0$, parameters $q > 0$, $\upsilon \in \left[\frac{1}{N_0}, 1\right]$, final
       period $T \in \mathbb{N}$.
Output: Clusters, membership matrix, and clusters'
        changes from $t = 0$ to $t = T$.
1 Run RFSVC algorithm using data set $X_0$
2 for $t = 1 : T$ do
3     Collect new data $X^{(t)}$
4     $X_t \leftarrow X_t \cup X^{(t)}$
5     Run the Training Update Algorithm
6     Run the Labeling Update Algorithm
7     Run the Fuzzification Update Algorithm
8     Save the results and changes
```

update steps uses previous knowledge to modify the current classifier, saving important resources.

C. Pseudocode, Computational Complexity, and Conceptual Comparison

Next, we present the pseudocode of D-RFSVC and show how to obtain its crisp version (dynamic support vector clustering—D-SVC). Thereafter, we analyze its computational complexity. Finally, a conceptual comparison between D-RFSVC and related cluster approaches is presented, highlighting its potential for dynamic clustering.

1) Pseudocode of D-RFSVC: Combining the ideas from Sections IV-B1–IV-B3 leads to Algorithm 2, which is capable of dealing with dynamic databases in the manner presented in this paper without building the classifier from scratch after each update.

The parameter T in Algorithm 2 represents the final of a series of periods observed over time, but all completely in the past. This approach is used if we want to understand behavior changes from the past, e.g., past customers' purchasing behavior in order to reveal patterns on which to base future decisions. Algorithm 2 could also be applied continuously over time without assuming a final period. In that case, upcoming patterns could be detected as they appear, in order to provide timely information in order to make the most appropriate decisions.

Step 7 is optional. If it is omitted, the classes found will be crisp, providing the dynamic version of SVC, i.e., D-SVC can be considered as a special case of the proposed D-RFSVC. As a consequence, the BSVs detected in each cycle will remain unclassified, and outlier traceability will be lost since it is based on the membership degrees calculated for these data points.

2) Computational Complexity: Following Ben-Hur *et al.* [13] and Li and Ping [40], the computational complexity of the proposed method can be approximated as follows: The quadratic programming problem of the training step can be solved by the GSMO algorithm [34]. Benchmarks reported in [34] show that this algorithm converges after approximately $O\left(N^2\right)$ kernel evaluations. The complexity of the labeling part of the algorithm is $O\left(mSV^2\right)$, since we do not compute the entire adjacency matrix, but only adjacencies with SVs.

TABLE IV
CONCEPTUAL COMPARISON BETWEEN D-RFSVC AND OTHER DYNAMIC CLUSTER ALGORITHMS

	D-FCM	E-DDAA	D-RCM	D-DBSCAN	ABDC	D-RFSVC
Creation	✓	✓	✓	✓	✓	✓
Deletion	✓	-	✓	-	✓	✓
Movement	✓	✓	-	-	-	✓
Merging	-	✓	-	✓	✓	✓
Splitting	-	-	-	-	-	✓
Change of Uncertainty	-	-	✓	-	-	✓
Change of Shape	-	-	-	-	-	✓
Contraction	-	-	-	-	-	✓
Dilatation	-	-	-	-	-	✓
Outlier Traceability	-	-	-	-	-	✓

Saltos and Weber [14] presented a rough-fuzzy version of SVC. This method for static environments proposes to model uncertainty via rough-fuzzy concepts adding a fuzzification step, which uses already computed kernel values. As a consequence, the static RFSVC algorithm maintains the computational complexity of Ben-Hur's SVC.

Finally, D-RFSVC also maintains this computational complexity, since it applies RFSVC in each iteration of the proposed updating scheme, but it uses previously computed information (kernel values, adjacency matrix, and membership values) to reduce the computational time required to update the classifier.

3) Conceptual Comparison: To conclude this section, Table IV presents a conceptual comparison between the algorithms discussed in Section II and D-RFSVC, indicating which structural change the respective algorithms address.

D-FCM and E-DDAA perform the movement of clusters by updating their centers. Since D-RFSVC uses SVs instead of centers as prototypes, it is not possible to perform the movement of clusters in the same sense defined by Crespo and Weber [2] and Georgieva and Klawonn [11]. However, the effects of changing shape or dilatation can be interpreted as movement of clusters, recalling that Crespo and Weber [2] update the centers in order to include the new observations close to those clusters. Change of shape and dilatation have the same purpose, i.e., to include the new observations that are located close to the changing cluster. It can be concluded that D-RFSVC has the intrinsic ability to perform cluster movements but in a way different from that proposed by other authors.

V. EXPERIMENTS AND DISCUSSION

A. Description of the Datasets and Experimental Setup

Table V summarizes the main characteristics of the datasets used. The column "Total Instances" refers to the total number of objects that arrived until the final period T.

The parameters for $t = 0$ are given in Table VI. They were set using the ideas reported in [13], similarly to those of the algorithm's static version. For comparison, Crespo's D-FCM [2] was selected since other dynamic algorithms do not provide membership degrees nor detect enough structural changes.

TABLE V
DATASET CHARACTERISTICS

Name	Type	Total Instances	Attributes	Periods (T)
Four Squares	Synthetic	9000	2	10
Three Spheres	Synthetic	17 250	3	10
XO	Benchmark	3600	2	10
S1-Gaussian	Benchmark	5000	2	10
Unbalance	Benchmark	6500	2	10
NOAA Weather	Real-World	8759	8	12
Electric	Real-World	50 000	7	10
Retail	Real-World	12 170	3	12
IndapU2	Real-World	22 380	3	48
Pokemon	Real-World	800	6	6

TABLE VI
ALGORITHM'S PARAMETERS FOR $t = 0$

	D-RFSVC		D-FCM
	q	v	c
Four Squares	12	$\frac{1}{3}$	2
Three Spheres	8	$\frac{1}{3}$	3
XO	6	$\frac{1}{6}$	2
S1-Gaussian	20	0.4	2
Unbalance	10	0.05	2
NOAA Weather	0.5	0.05	2
Electric	0.0345	0.05	2
Retail	0.12	0.05	2
IndapU2	10	0.05	2
Pokemon	0.0002	0.05	2

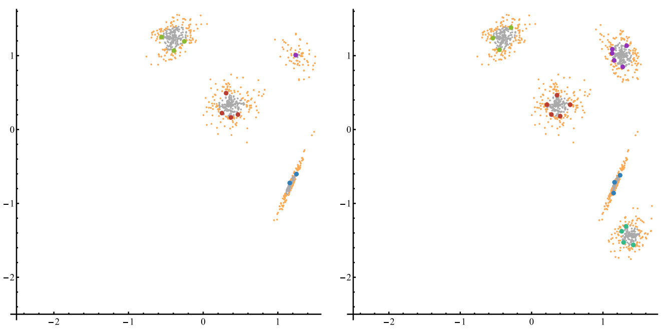


Fig. 4. Creation and expansion of clusters for S1-Gaussian dataset. (a) Cycle $t = 1$. (b) Cycle $t = 2$.

B. *Results for Benchmark and Synthetic Datasets*

Using the benchmark and synthetic datasets as described in Section V-A, we illustrate how the proposed method identifies the structural changes introduced in Section IV-A.

For Figs. 4–6, orange (light) points belong to fuzzy boundaries (BSV), color highlighted points are each cluster's prototypes (SV), while the remaining gray/cyan (dark) data points are elements of the lower approximations of their respective clusters (ID).

Fig. 4 shows a sample of the results obtained for the S1-Gaussian dataset. In cycle $t = 1$, four clusters are present, while for $t = 2$, we have five clusters. The main changes observed in the transition between these two cycles are a new cluster (C_5), and cluster C_4 expanded. In $t = 1$, this cluster has only one SV and a few IDs, while in $t = 2$, it has more objects in its lower approximation and new SVs covering a larger area.

Fig. 5 shows how clusters can contract and disappear between cycles. The initial cluster structure is shown in Fig. 5(a), where three clusters were found, two of them having most of the existing objects. Then, in cycle $t = 4$, the three clusters have received new data points, but the red one (the one in front of the other two) has a significantly higher number than the others, causing their contraction. However, until period $t = 7$, only the blue (in the upper right corner) and red (the one in front) clusters have received new objects. This fact causes the blue class to expand while the purple one (in the left upper corner) disappears [see Fig. 5(c)]. Note that all the data points of the disappeared (purple) cluster are now BSVs, becoming elements of the remaining clusters' fuzzy boundaries. The proposed method does not remove former objects of eliminated classes, given that they might belong to clusters that could appear in future cycles.

Fig. 6 shows how clusters can change their shape and merge. From period $t = 1$ to $t = 4$, the red cluster (initially a rectangle) changed its silhouette becoming an "X." At the same time, two new clusters appeared: the upper and lower part of what will later become the "O." Finally, in cycle $t = 7$, these clusters merged leading to the "O" cluster.

To conclude this section, we emphasize that all changes shown in Figs. 4–6 can be detected using the membership matrices and the sets SV, BSV, and ID. As an example, a small section of the membership matrix for the S1-Gaussian dataset is presented in Tables VII and VIII. These data points correspond to elements located in the top right corner, with values close to 1 on both dimensions (see Fig. 4).

The membership values of each of the four clusters in Table VII reveal that four of the six observations are outliers related to $\boldsymbol{C_4}$, while the remaining two belong to this cluster. Table VIII shows the membership degrees for the five clusters found in period $t = 2$. This reveals the fact that a new cluster appeared from one cycle to another, but more importantly, we infer that $\boldsymbol{C_4}$ has expanded because three of the four outliers related to this cluster in the previous cycle became IDs. This phenomenon can be observed in Fig. 4 for the cluster in the top right corner.

An interesting alternative for inferring the same results described above for 2-D or 3-D datasets, but with high-dimensional data, is using aggregated indicators characterizing the clusters found. This is performed based exclusively on information provided by the membership matrix. In this work, we calculate the following indicators.

1) The cardinality of each cluster's lower approximation (CardLA). When this value increases, we postulate that the cluster is dilating or changing its shape to cover more data points. Otherwise, the cluster is contracting.
2) The cardinality of the fuzzy boundary (CardFB), given by the number of BSVs surrounding each cluster's lower approximation. These BSVs fulfill $\mu_{i,k} \in (0.001, 1)$. An increasing number of BSVs indicate a contracting cluster.

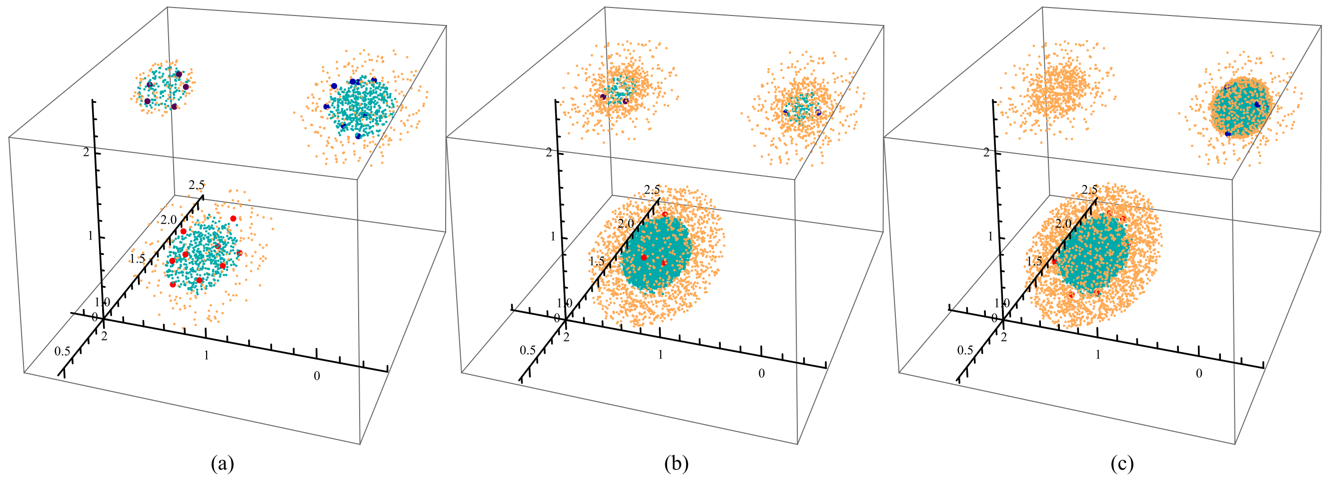


Fig. 5. Contraction and elimination of clusters for three spheres dataset. (a) Cycle $t = 0$. (b) Cycle $t = 4$. (c) Cycle $t = 7$.

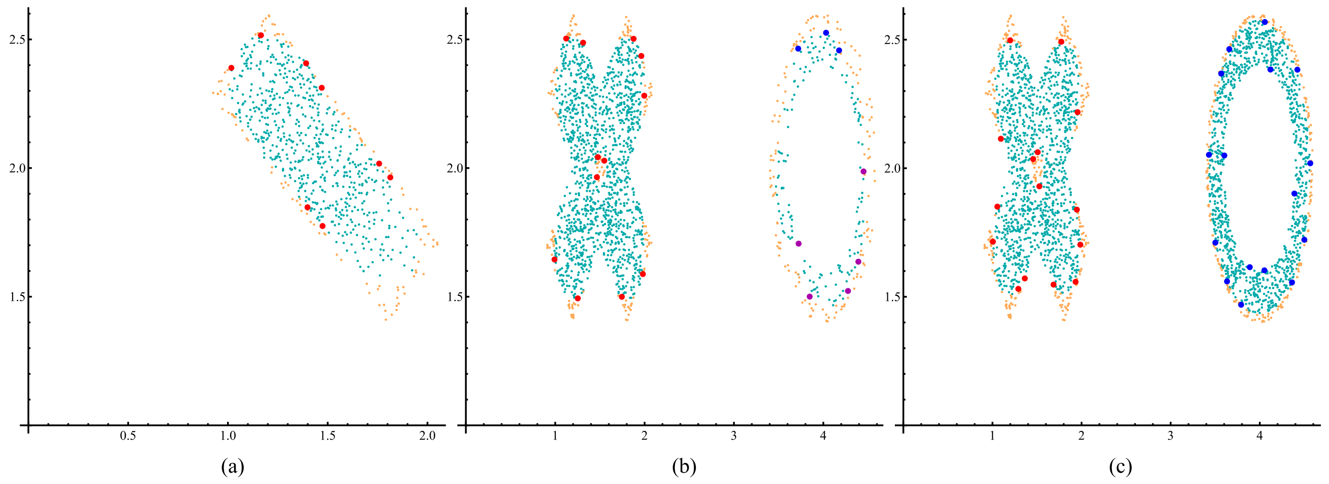


Fig. 6. Changing shape and merging clusters for XO dataset. (a) Cycle $t = 1$. (b) Cycle $t = 4$. (c) Cycle $t = 7$.

TABLE VII
S1-GAUSSIAN: MEMBERSHIP MATRIX FOR $t = 1$

Data ID	X	Y	C_1	C_2	C_3	C_4
995	1.249	0.932	4.11E-10	3.50E-21	6.67E-21	0.892
996	1.239	1.007	0	0	0	1
997	1.349	0.798	1.35E-10	7.55E-18	2.29E-24	0.328
998	1.356	0.953	3.99E-12	8.05E-22	1.02E-23	0.716
999	1.140	0.974	9.01E-09	2.03E-22	5.40E-18	0.804
1000	1.256	1.010	0	0	0	1

TABLE VIII
S1-GAUSSIAN: MEMBERSHIP MATRIX FOR $t = 2$

Data ID	X	Y	C_1	C_2	C_3	C_4	C_5
995	1.249	0.932	0	0	0	1	0
996	1.239	1.007	0	0	0	1	0
997	1.349	0.798	2.91E-08	2.78-18	5.83E-27	0.858	2.38E-39
998	1.356	0.953	0	0	0	1	0
999	1.140	0.974	0	0	0	1	0
1000	1.256	1.010	0	0	0	1	0

3) The mean of the membership degrees of the elements in each cluster's fuzzy boundary (MeanFB). This indicator can be interpreted in two ways: The first one is based on a static snapshot. If the value is high, we can infer that the cluster's fuzzy boundary is very compact, i.e., the dispersion of BSVs is low. Otherwise, their dispersion is high. The second interpretation is the change from one period to another. Whether this value increases or decreases indicates whether the cluster is receiving BSVs with high membership or low membership, respectively. This phenomenon could "anticipate" changing data structures.
4) The standard deviation of the membership degrees of the elements in each cluster's fuzzy boundary (StdFB) complementing the information provided by MeanFB.
5) The mean of the membership degrees of each cluster's elements (Mean). This is similar to MeanFB but includes the elements in the lower approximation.
6) The standard deviation of the membership degrees of each cluster's elements (Std) complementing the indicator Mean.

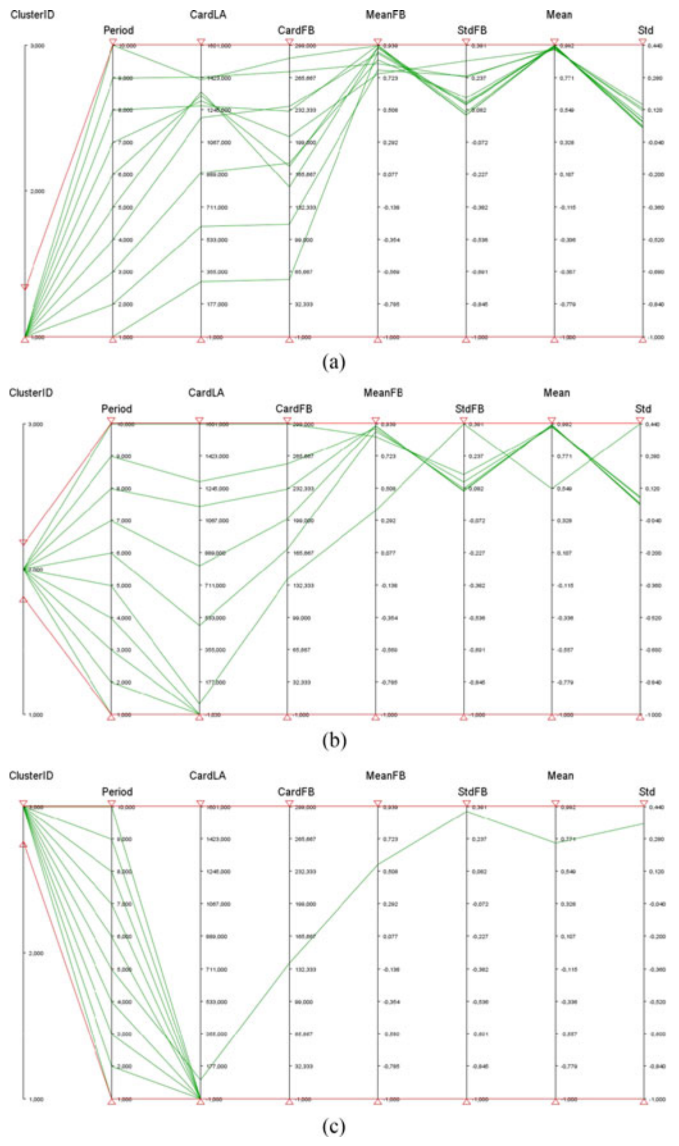


Fig. 7. Parallel coordinates plot for the XO dataset. (a) Cluster 1. (b) Cluster 2. (c) Cluster 3.

If a cluster appears in one cycle but disappears in others, we set the previously mentioned indicators at -1 for those clusters that do not exist in certain cycles.

Using a parallel coordinates plot [41], e.g., for XO data, we obtain Figs. 7 and 8. This is a powerful visualization tool that allows users to evaluate several dimensions graphically in a single plot and, in our case, to assess the evolution over time for various indicators. Fig. 7 shows the parallel coordinate plot for each cluster and for different periods.

We use Fig. 8 to explain how to identify changes using parallel coordinates. This figure shows the trajectories for cluster 3 at $t = 5$ and $t = 6$ (blue and green lines, respectively). The blue line shows the values that the previously introduced indicators take in each period, while the green line goes to the value -1. This indicates that cluster 3 disappears from period 5 to period 6. Appearing clusters follow a similar reasoning. The curve of the previous cycle will start with value -1, and in a posterior cycle, the indicator will take values different from -1. Similar results can be inferred for the remaining datasets.

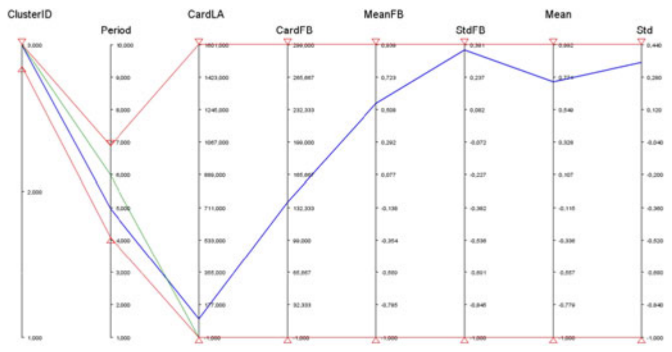


Fig. 8. Parallel coordinates plot for the XO Dataset: Tracking Cluster 3.

TABLE IX
WEATHER DATASET MEMBERSHIP MATRIX

Data Id.	$t = 1$ C_1	$t = 2$ C_1	$t = 3$ C_1
10	1	1	1
12	1	1	1
14	1	1	1
15	0.9706	0.7110	0.9277
16	1	1	1
22	1	0.6898	0.6333

C. Application to the NOAA Weather Dataset

The NOAA Weather dataset [42] is a database containing information related to the normal climate variables from 1981 to 2010 for the 12 months of each of those years in San Francisco, CA, USA. A normal climate variable is defined as the 30-year average of that particular variable. More details can be found in [42]. In order to emulate dynamism, the method receives monthly chunks of observations.

As mentioned in Table V, the dataset contains 8759 instances for the observed period. The parameters for the first cycle were set at $q = 0.5$ and $\upsilon = 0.05$. In the first month, the algorithm found three clusters, but in the next one, these clusters merge into only one for the whole remaining observation period. The conclusion we can obtain is that only one cluster exists in the dataset. This makes sense since the climate conditions commonly do not change abruptly, i.e., the weather usually has gradual changes from one day to another. As a consequence, only the change of shape and expansion of the found cluster occurs over time.

Table IX shows a sample of the membership matrix for three cycles. The main fact we can highlight from this sample is that data point 22 lost its status as an inside vector, becoming a BSV with a 0.68 membership degree. This may have happened because SVs moved away in order to cover new objects that arrived in $t = 2$.

TABLE X
VALIDATION MEASURES WHEN D-FCM DOES NOT ELIMINATE CLUSTERS

		D-FCM			D-RFSVC		
		PC	PE	α	PC	PE	α
S1 Gaussian	Mean	**0.8419**	0.4029	0.6887	0.8057	**0.0875**	**0.9372**
	Std. Dev.	0.0824	0.2250	0.1419	**0.0124**	**0.0066**	**0.0138**
Unbalance	Mean	0.8668	0.2441	0.9339	**0.9896**	**0.0057**	**0.9982**
	Std. Dev.	0.0827	0.1378	0.0612	**0.0040**	**0.0021**	**0.0013**
XO	Mean	0.8568	0.2493	0.7772	**0.9655**	**0.0176**	**0.9735**
	Std. Dev.	0.1049	0.1439	0.2437	**0.0093**	**0.0057**	**0.0112**
Three Spheres	Mean	**0.8958**	0.2359	0.8129	0.7898	**0.0826**	**0.9181**
	Std. Dev.	**0.0117**	0.0207	**0.0212**	0.0282	**0.0064**	0.0683
NOAA Weather	Mean	0.7101	0.4544	0.6403	**0.9883**	**0.0055**	**0.9975**
	Std. Dev.	0.0163	0.0210	0.0457	**0.0056**	**0.0019**	**0.0015**
Electric	Mean	0.6862	0.4795	0.5124	**0.9761**	**0.0096**	**0.7896**
	Std. Dev.	0.0116	0.0152	0.0901	**0.0015**	**0.0007**	**0.0892**
Retail	Mean	0.9567	0.0847	0.9866	**0.9868**	**0.0053**	**0.9975**
	Std. Dev.	**0.0018**	0.0034	0.0053	0.0033	**0.0013**	**0.0001**
IndapU2	Mean	0.7242	0.4359	0.5527	**0.9912**	**0.0043**	**0.9987**
	Std. Dev.	0.0109	0.0139	0.1125	**0.0028**	**0.0014**	**0.0003**
Pokemon	Mean	0.4979	0.8346	0.1217	**0.9873**	**0.0053**	**0.8687**
	Std. Dev.	0.0647	0.1341	0.1025	**0.0066**	**0.0029**	**0.0879**
Four Squares	Mean	**0.8883**	0.2110	0.7790	0.8445	**0.0687**	**0.9476**
	Std. Dev.	0.1135	0.1568	0.3010	**0.0264**	**0.0089**	**0.0163**

TABLE XI
VALIDATION MEASURES WHEN D-FCM ELIMINATES CLUSTERS

		D-FCM			D-RFSVC		
		PC	PE	α	PC	PE	α
S1 Gaussian	Mean	**0.8675**	0.3125	0.7010	0.8057	**0.0875**	**0.9372**
	Std. Dev.	0.0555	0.1340	0.1352	**0.0124**	**0.0066**	**0.0138**
Unbalance	Mean	0.9216	0.1423	0.9594	**0.9896**	**0.0057**	**0.9982**
	Std. Dev.	0.0854	0.1302	0.0586	**0.0040**	**0.0021**	**0.0013**
XO	Mean	0.5775	0.2052	0.5919	**0.9655**	**0.0176**	**0.9735**
	Std. Dev.	0.4087	0.1887	0.3888	**0.0093**	**0.0057**	**0.0112**
Three Spheres	Mean	0.5544	0.1037	0.5290	**0.7898**	**0.0826**	**0.9181**
	Std. Dev.	0.4775	0.0995	0.4574	**0.0282**	**0.0064**	**0.0683**
NOAA Weather	Mean	0.7236	0.4368	0.6403	**0.9883**	**0.0055**	**0.9975**
	Std. Dev.	0.0119	0.0158	0.0457	**0.0056**	**0.0019**	**0.0015**
Electric	Mean	0.6862	0.4795	0.5124	**0.9761**	**0.0096**	**0.7896**
	Std. Dev.	0.0116	0.0152	0.0901	**0.0015**	**0.0007**	**0.0892**
Retail	Mean	0.9567	0.0847	0.9866	**0.9868**	**0.0053**	**0.9975**
	Std. Dev.	**0.0018**	0.0034	0.0053	0.0033	**0.0013**	**0.0001**
IndapU2	Mean	0.7243	0.4359	0.5527	**0.9912**	**0.0043**	**0.9987**
	Std. Dev.	0.0109	0.0139	0.1125	**0.0028**	**0.0014**	**0.0003**
Pokemon	Mean	0.4330	1.3050	0.1515	**0.9873**	**0.0053**	**0.8687**
	Std. Dev.	0.1666	0.4152	0.1443	**0.0066**	**0.0029**	**0.0879**
Four Squares	Mean	**0.9110**	0.1557	0.8209	0.8445	**0.0687**	**0.9476**
	Std. Dev.	0.1168	0.1666	0.3179	**0.0264**	**0.0089**	**0.0163**

D. Numerical Comparison

It is important to highlight that the existing validation measures only work for static cluster solutions, making an objective quantitative comparison between state-of-the-art dynamic clustering algorithms and D-RFSVC difficult. In particular, classical cluster validity measures such as, e.g., DB-index or Maji's Indices [43], are not applicable here, since we are not only interested in validating static cluster structures found each time period. We are interested in validating how well the algorithms detect changes between two consecutive periods. To the best of our knowledge, there are still no such dynamic cluster validity measures. As a consequence, we used the partition coefficient (PC), partition entropy (PE), and Maji's α index to compare the results of D-RFSVC and D-FCM. Since these validation measures are computed for each cycle, we report their mean and standard deviation.

Since cluster elimination has to be guided by a parameter in D-FCM, whereas in D-RFSVC, it is simply based on the clusters' density, we present two comparisons. First, we analyze the case where D-FCM is not allowed to eliminate clusters followed by the case where such an elimination is based on a user-defined parameter.

TABLE XII
HOLM'S TEST WITHOUT CLUSTER ELIMINATION

Index	PC		PE		α	
Method	D-FCM	D-RFSVC	D-FCM	D-RFSVC	D-FCM	D-RFSVC
Mean Rank	1.7	1.3	2	1	2	1
Mean Index	0.7925	0.9326	0.363	0.029	0.681	0.943
p-value	0.2059		0.0016***		0.0016***	

*** = 0.01, ** = 0.05, * = 0.1 significance level.

TABLE XIII
HOLM'S TEST WITH CLUSTER ELIMINATION

Index	PC		PE		α	
Method	D-FCM	D-RFSVC	D-FCM	D-RFSVC	D-FCM	D-RFSVC
Mean Rank	1.8	1.2	2	1	2	1
Mean Index	0.7357	0.9326	0.366	0.029	0.644	0.943
p-value	0.0578*		0.0016***		0.0016***	

*** = 0.01, **]= 0.05, * = 0.1 significance level.

Table X shows the mean and standard deviation of validity indices when D-FCM does not eliminate clusters compared with D-RFSVC. In each row, the bold number indicates the best solution for each indicator. Clearly, D-RFSVC outperforms D-FCM in most of the tested datasets and provides more stable results since the respective standard deviations are lower than those for D-FCM. Similar results are shown in Table XI when eliminating clusters is allowed for D-FCM. The number of periods required to eliminate clusters was set to 3 following the ideas reported in [2].

We perform Holm's test [44] to compare the performance of both algorithms and present the respective results in Tables XII and XIII. The proposed D-RFSVC has the best overall performance for all experiments, achieving differences that are statistically significant at the respective levels mentioned in both tables.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented the novel D-RFSVC algorithm. It allows updating clusters built from previous cycles to the current one, taking advantage of the knowledge generated in each period. Since it is a generalization of the static version of RFSVC, it inherits and adapts the following fundamental properties.

1) D-RFSVC identifies soft clusters of any silhouette with crisp lower approximations and fuzzy boundaries. It is able to detect complex structural changes such as change of shape, merging clusters, contraction, and dilatation.
2) The membership degrees of the BSVs are calculated using the Gram matrix, and they are not constrained to sum 1. As a consequence, in each cycle, we can identify new outliers and trace the behavior of previously detected ones.
3) Similar to its static version, D-RFSVC does not require the number of clusters as an input parameter. This decision is just based on the clusters' density.

The applications presented in this paper highlight the potential dynamic clustering has in general and particularly underline the original contribution of D-RFSVC, which is an adequate modeling of outliers as BSVs indicating changing data structures.

The following ideas represent fruitful avenues for future research. Further applications could discover D-RFSVC's benefits and limitations. It would also be interesting to see how other density-based clustering algorithms could adapt ideas we have implemented in the proposed D-RFSVC in order to detect changing data structures.

REFERENCES

[1] G. Espejo, G. L'Huillier, and R. Weber, "A game-theoretical approach for policing decision support," *Eur. J. Appl. Math.*, vol. 27, no. 3, pp. 338–356, 2016.

[2] F. Crespo and R. Weber, "A methodology for dynamic data mining based on fuzzy clustering," *Fuzzy Sets Syst.*, vol. 150, no. 2, pp. 267–284, 2005.

[3] E. Lughofer, "A dynamic split-and-merge approach for evolving cluster models," *Evolving Syst.*, vol. 3, no. 3, pp. 135–151, 2012.

[4] G. Peters and R. Weber, "Dynamic clustering with soft computing," *Wiley Interdisciplinary Rev.: Data Mining Knowl. Discovery*, vol. 2, no. 3, pp. 226–236, 2012.

[5] G. Peters, R. Weber, and R. Nowatzke, "Dynamic rough clustering and its applications," *Appl. Soft Comput.*, vol. 12, no. 10, pp. 3193–3207, 2012.

[6] T. M. Nguyen, Q. M. J. Wu, and S. Member, "Dynamic fuzzy clustering and its application in motion segmentation," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 6, pp. 1019–1031, Dec. 2013.

[7] G. Peters, R. Weber, and F. Crespo, "Uncertainty modeling in dynamic clustering: A soft-computing perspective," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2010, pp. 1–6.

[8] G. Peters and R. Weber, "DCC: A framework for dynamic granular clustering," *Granular Comput.*, vol. 1, no. 1, pp. 1–11, 2016.

[9] T. C. Glenn, A. Zare, and P. D. Gader, "Bayesian fuzzy clustering," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 5, pp. 1545–1561, Oct. 2015.

[10] L. Silva, R. Moura, A. M. P. Canuto, R. H. N. Santiago, and B. Bedregal, "An interval-based framework for fuzzy clustering applications," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 6, pp. 2174–2187, Dec. 2015.

[11] O. Georgieva and F. Klawonn, "Dynamic data assigning assessment clustering of streaming data," *Appl. Soft Comput.*, vol. 8, no. 4, pp. 1305–1313, 2008.

[12] R. Saltos and R. Weber, "Rough-fuzzy support vector domain description for outlier detection," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Istanbul, Turkey, 2015, pp. 1–6.

[13] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *J. Mach. Learn. Res.*, vol. 2, no. 12, pp. 125–137, 2001.

[14] R. Saltos and R. Weber, "A rough-fuzzy approach for support vector clustering," *Inf. Sci.*, vol. 339, pp. 353–368, 2016.

[15] R. Weber, "From operations research to dynamic data mining and beyond," in *Zukunftsperspektiven des Operations Research*, M. Lübbecke, A. Weiler, and B. Werners, Eds. Wiesbaden, Germany: Springer, 2014, pp. 343–356.

[16] D. Dehideniya and A. S. Karunananda, "Dynamic partitional clustering using multi-agent technology," in *Proc. IEEE Int. Conf. Adv. ICT Emerg. Regions*, 2013, pp. 228–233.

[17] G. Peters and R. Weber, "A dynamic approach to rough clustering," in *Rough Sets and Current Trends in Computing*. New York, NY, USA: Springer, 2008, pp. 379–388.

[18] G. Peters and R. Weber, "A class of dynamic rough partitive algorithms," *Int. J. Intell. Syst.*, vol. 26, no. 6, pp. 540–554, 2011.

[19] A. Joentgen, L. Mikenina, R. Weber, and H.-J. Zimmermann, "Dynamic fuzzy data analysis based on similarity between functions," *Fuzzy Sets Syst.*, vol. 105, no. 1, pp. 81–90, 1999.

[20] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York, NY, USA: Plenum, 1981.

[21] P. Lingras and C. West, "Interval set clustering of web users with rough k-means," *J. Intell. Inf. Syst.*, vol. 23, no. 1, pp. 5–16, 2004.

[22] G. Peters, "Some refinements of rough k-means clustering," *Pattern Recognit.*, vol. 39, no. 8, pp. 1481–1491, 2006.

[23] O. Georgieva and F. Klawonn, "Cluster analysis via the dynamic data assigning assessment algorithm," *Inf. Technol. Control*, vol. 2, pp. 14–21, 2006.

[24] A. S. L. Mary and K. R. S. Kumar, "A density based dynamic data clustering algorithm based on incremental dataset," *J. Comput. Sci.*, vol. 8, no. 5, pp. 656–664, 2012.

[25] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.

[26] 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.

[27] C. C. Aggarwal, Ed., *Data Streams: Models and Algorithms*. New York, NY, USA: Springer, 2007, vol. 31.

[28] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2013.

[29] Y. Wang and S. Chen, "Soft large margin clustering," *Inf. Sci.*, vol. 232, pp. 116–129, 2013.

[30] A. Amini, T. Y. Wah, and H. Saboohi, "On density-based data streams clustering algorithms: A survey," *J. Comput. Sci. Technol.*, vol. 29, no. 1, pp. 116–141, 2014.

[31] P. Lingras and M. Triff, "Advances in rough and soft clustering: Meta-clustering, dynamic clustering, data-stream clustering," in *Proc. Int. Joint Conf.*, Santiago, Chile, Oct. 7–11, 2016, pp. 3–22.

[32] D. Tax and R. Duin, "Support vector domain description," *Pattern Recognit. Lett.*, vol. 20, no. 11, pp. 1191–1199, 1999.

[33] B. Schölkopf and A. Smola, *Learning with Kernels*. Cambridge, U.K.: MIT Press, 2002.

[34] S. S. Keerthi and E. G. Gilbert, "Convergence of a generalized SMO algorithm for SVM classifier design," *Mach. Learn.*, vol. 46, pp. 351–360, 2002.

[35] J. Yang, V. E. Castro, and S. K. Chalup, "Support vector clustering through proximity graph modelling," *Proc. 9th Int. Conf. Neural Inf. Process.*, 2002, vol. 2, pp. 898–903.

[36] S. H. Lee and K. M. Daniels, "Gaussian kernel width exploration and cone cluster labeling for support vector clustering," *Pattern Anal. Appl.*, vol. 15, no. 3, pp. 327–344, 2012.

[37] H. Li, "A fast and stable cluster labeling method for support vector clustering," *J. Comput.*, vol. 8, no. 12, pp. 3251–3256, 2013.

[38] M. Bazaraa, J. Jarvis, and H. Sherali, *Linear Programming and Network Flows*, 4th ed. New York, NY, USA: Wiley, 2010.

[39] C. D. Wang, J. H. Lai, and D. Huang, "Incremental support vector clustering," in *Proc. IEEE Int. Conf. Data Mining*, 2011, pp. 839–846.

[40] H. Li and Y. Ping, "Recent advances in support vector clustering: theory and applications," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 29, no. 1, 2015, Art. no. 1550002.

[41] A. Inselberg, *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*, 1st ed. New York, NY, USA: Springer, 2009.

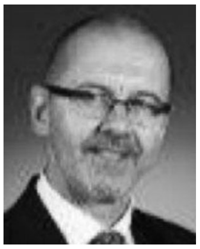
[42] A. Arguez *et al.*, "NOAA's U.S. Climate Normals (1981–2010)," 2010. [Online]. Available: https://goo.gl/ahuRI3

[43] P. Maji and S. Pal, "Rough set based generalized fuzzy c-means algorithm and quantitative indices," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 37, no. 6, pp. 1529–1540, Dec. 2007.

[44] S. Holm, "A simple sequentially rejective multiple test procedure," *Scand. J. Statist.*, vol. 6, no. 2, pp. 65–70, 1979.

Ramiro Saltos received the B.S. degree in logistics and transportation from Escuela Superior Politécnica del Litoral, Guayaquil, Ecuador, in 2011, the M.S. degree in operations research from the Universidad Nacional Autónoma de México, Mexico City, Mexico, in 2014, and the Ph.D. degree in engineering systems - operations management from the University of Chile, Santiago, Chile, in 2016.

His current research interests include fuzzy sets, rough sets, and data mining, with a special focus on data clustering and its applications.



Richard Weber (SM'10) received the B.S. degree in mathematics, the M.S. degree in operations research, and the Ph.D. degree in business and economics all from RWTH Aachen University, Aachen, Germany, in 1986, 1988, and 1992, respectively.

He is an Associate Professor with the Department of Industrial Engineering and Director of the Graduate School at the Faculty of Physical and Mathematical Sciences, University of Chile, Santiago, Chile. His research interests include data mining, machine learning, and soft computing.

Dr. Weber was the President of the Chilean chapter on computational intelligence within the IEEE.

Sebastián Maldonado (M'10) received the B.S. in industrial engineering and M.S. degrees in operations management in 2007, and the Ph.D. degree in engineering systems in 2011, all from the University of Chile, Santiago, Chile.

He is currently an Associate Professor with the Facultad de Ingeniería y Ciencias Aplicadas (Faculty of Engineering and Applied Sciences), Universidad de Los Andes, Santiago. His research interests include statistical learning, data mining, and business analytics.