IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 1, 2020, accepted August 19, 2020, date of publication September 3, 2020, date of current version September 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3021675

An Automatic Merge Technique to Improve the Clustering Quality Performed by LAMDA

LUIS MORALES[1] AND JOSE AGUILAR[2,3], (Member, IEEE)
[1]Departamento de Automatización y Control Industrial, Escuela Politécnica Nacional, Quito 170525, Ecuador
[2]CEMISID, Escuela de Ingeniería de Sistemas, Universidad de Los Andes, Mérida 5101, Venezuela
[3]GIDITIC, Universidad EAFIT, Medellín 050021, Colombia

Corresponding author: Jose Aguilar (aguilar@ula.ve)

ABSTRACT Clustering is a research challenge focused on discovering knowledge from data samples whose goal is to build good quality partitions. In this paper is proposed an approach based on LAMDA (Learning Algorithm for Multivariable Data Analysis), whose most important features are: a) it is a non-iterative fuzzy algorithm that can work with online data streams, b) it does not require the number of clusters, c) it can generate new partitions with objects that do not have enough similarity with the preexisting clusters (incremental-learning). However, in some applications, the number of created partitions does not correspond with the number of desired clusters, which can be excessive or impractical for the expert. Therefore, our contribution is the formalization of an automatic merge technique to update the cluster partition performed by LAMDA to improve the quality of the clusters, and a new methodology to compute the Marginal Adequacy Degree that enhances the individual-cluster assignment. The proposal, called LAMDA-RD, is applied to several benchmarks, comparing the results against the original LAMDA and other clustering algorithms, to evaluate the performance based on different metrics. Finally, LAMDA-RD is validated in a real case study related to the identification of production states in a gas-lift well, with data stream. The results have shown that LAMDA-RD achieves a competitive performance with respect to the other well-known algorithms, especially in unbalanced benchmarks and benchmarks with an overlapping of around 9%. In these cases, our algorithm is the best, reaching a Rand Index (RI) $>$ 98%. Besides, it is consistently among the best for all metrics considered (Silhouette coefficient, modification of the Silhouette coefficient, WB-index, Performance Coefficient, among others) in all case studies analyzed in this paper. Finally, in the real case study, it is better in all the metrics.

INDEX TERMS Automatic merging, clustering, LAMDA, unsupervised learning.

NOMENCLATURE

ADD	Autonomous Data-Driven
AGL	Artificial Gas Lift
AHT	Agglomerative Hierarchical Tree
AUC	Area Under the Curve
CMAD	Cauchy Marginal Adequacy Degree
CHP	Pressure of the Casing
DBSCAN	Density-Based Spatial Clustering of Applications With Noise
DNB	Distance Between Neighbors
DT	Density Threshold
FCM	Fuzzy C-Means
GAD	Global Adequacy Degree
GK	Gustafson & Kessel
GLDP	Differential Pressure of the Injected Gas
GLP	Pressure of the Injected Gas
GTD	Global Typicality Degree
HP	High Production
IGAD	Intuitionistic Global Adequacy Degree
KM	K-Means
KMD	K-Medoids
KNN	K-Nearest Neighbors
LAMDA	Learning Algorithm for Multivariable Data Analysis
LP	Low Production
MAD	Marginal Adequacy Degree
NIC	Non-Informative Class
NP	Normal Production
PC	Performance Coefficient
RD	Robust Distance
RI	Rand Index

The associate editor coordinating the review of this manuscript and approving it for publication was Zijian Zhang.

RMAD	Robust Marginal Adequacy Degree
SALSA	Situation Assessment using LAMDA Classification Algorithm
SC	Silhouette Coefficient
SPC	Spectral Clustering
STD	Standard Deviation
SSB	Sum-of-Squares Between Clusters
SSW	Sum-of-Squares Within Clusters
THP	Pressure of the Tubing of Production
TP	Triple PI
TIGAD	Typicality and Intuitionistic Global Adequacy Degree
VLP	Very Low Production

I. INTRODUCTION

Clustering, unlike supervised learning, is useful in problems where unlabeled data is available [1]. The aim of clustering is to separate data into partitions with elements that have similar characteristics between them. Each cluster must be separable and compact, with respect to another cluster [2]. In the literature are reported different clustering approaches, some of them are distance-based [3], [4], partitioning clustering, hierarchical clustering [4], density-based [5], [7], fuzzy logic-based [6]–[9], or Gaussian methods [10], [11], among others. All these techniques depend on a previous stage of descriptor extraction, which are later used for the individual-cluster assignment performed by the algorithm. Historical data and streaming data [12] are application scenarios of the clustering techniques, but not all the methods can work in both contexts because the data is obtained differently; in the first case, the complete database is available, while in the second case, new data arrives continuously. The importance of working in the context of data streaming is that the evolution patterns provide useful information, which can allow users to make immediate and correct decisions [13].

In classical clustering, data is assigned to exactly one cluster; on the other hand, fuzzy clustering methods are based on the fuzzy membership degree; therefore, an individual can be a member of several clusters. Fuzzy clustering is widely used in the field of machine learning [2], [14]–[17]. One of these methods is LAMDA (Learning Algorithm for Multivariable Data Analysis) [18], which requires the computation of two parameters to assign an individual to a cluster. The first one is the Marginal Adequacy Degree (MAD), which computes the contribution of all descriptors (attributes) of an individual using fuzzy probability functions [19]. Applying fuzzy aggregation operators to the MADs is computed the second parameter, called Global Adequacy Degree (GAD), which corresponds to the membership degree of the individual to a cluster. The GADs define the cluster where the individual is assigned.

LAMDA can work in supervised (classification) and unsupervised learning (clustering), which is an important advantage over other fuzzy techniques [20], [21], [61]. In unsupervised learning, the algorithm can create new clusters automatically. The creation of new clusters is based on a threshold known as Non-Informative Class (NIC). For each sample, the GADs are computed in each cluster and in the NIC. The cluster with maximum GAD is where the individual is assigned, but if the maximum GAD is the one corresponding to the NIC, then a new cluster is created.

Conventional iterative methods (based on prototypes), e.g. Fuzzy C-Means (FCM) [22], Gustafson & Kessel means (GK-means) [23], K-Nearest Neighbors (KNN), require the determination of the number of clusters K. The definition of K is not easy, and it is an open field of research, as well as the determination of the centers of the clusters [24]. In LAMDA, it is not necessary to know as an input parameter the number of partitions K which is a great advantage over the aforementioned methods. However, in the cases in which data have high levels of intra-cluster uncertainty, a large amount of undesired partitions could be created due to the comparison made with the NIC. This limitation of LAMDA was detected by observing the existence of several poor quality clusters [19], being important to propose solutions to improve this problem, especially in the computation of MADs, and incorporating an automatic merge of the clusters when it is required.

The motivation of this work is to propose an automatic merge technique to update the cluster partition performed by LAMDA to enhance the clustering quality. The proposed extension avoids the excessive and undesired cluster creation. Also, we propose a new method for the MAD computation based on a penalty factor, in order to improve individual-cluster assignment. The two improvements described above result in the LAMDA-RD algorithm, which solves the main problems that the original algorithm has when it is working in unsupervised learning.

This paper has been organized as follows: Section II briefly reviews the relevant literature related with LAMDA and its research advances in clustering, Section III shows the fundamentals of LAMDA, Section IV formalizes our approach for LAMDA in order to improve the calculation of the MADs and the implementation of the automatic merge algorithm. In Section V are presented the experiments and the statistical analysis, applied in different benchmarks and scenarios, comparing our method with other clustering algorithms. Also, the proposed extensions of LAMDA are tested in a real case study: a streaming data scenario corresponding to the identification of the rate of production in gas lift wells. Section VI presents a general analysis of the results, and finally, conclusions and further works are presented in Section VII.

II. RELATED WORKS

In the context of unsupervised learning, LAMDA has been used to identify the functional states that describe the behavior of systems, for instance, the coagulation process in water plants [25]–[27]. In these applications, the algorithm has identified eight functional states of normal and abnormal functioning of the plant, which allows a constant monitoring of the process, in order to take corrective actions when abnormal states are detected. In [28], LAMDA method is

applied for monitoring complex industrial processes, combined with Markov's theory, which allows identifying the connections between functional states (clusters) through a transition degrees matrix. Other application fields are the electrical distribution networks, to solve problems of fault detection [29], discovering information from the data. Also, this method has been used in computer vision applications [25], [30], [31], and its performance has been computed based on tests with different aggregation operators and fuzzy probability distributions. Finally, one of the most important contributions of LAMDA is its implementation in a software for the supervision of complex systems. This software is called SALSA (Situation Assessment using LAMDA classification Algorithm) [33]. SALSA has been used for functional state detection in several applications, such as those presented in [34], [35].

Different researches have proposed modifications to the original LAMDA in the field of classification and clustering. Specifically, in clustering tasks, the most important recent contributions are:

- "LAMDA Triple Pi (π) operator (LAMDA-TP)" [19], [36]. This operator is used in LAMDA as an aggregation function for the computation of the GADs, avoiding the creation of new clusters with a few individuals.
- "LAMDA clustering method based on typicality degree and intuitionistic fuzzy sets" [2]. The authors propose the calculation of three functions: the Global Typicality Degree (GTD), the Intuitionistic Global Adequacy Degree (IGAD), and the Typicality and Intuitionistic Global Adequacy Degree (TIGAD). This proposal is applied in some study cases, presenting the formation of good clusters.

The algorithms described above have some drawbacks in the cluster formation. In the first case, LAMDA-TP does not depend on the exigency parameter α (formalized in the original LAMDA), which allows calibrating the permissiveness of the algorithm. In other words, it is a control parameter linked to the quality and number of created clusters. LAMDA-TP performs the clustering process based only on the similarity computed by the triple π operator, and the user cannot calibrate the algorithm partitions. LAMDA based on intuitionistic fuzzy sets improves the clustering stage; however, in [2], a comparison of the algorithm with respect to other similar methods is not presented, and based on the results, it is observed that a merge stage is required to group clusters of similar characteristics, in order to obtain best models. In addition, the formed partitions are not analyzed in terms of performance metrics, which allow evaluating their intra and inter-cluster qualities.

Finally, the original LAMDA has the exigency parameter α used for the calibration, however, it does not guarantee the creation of good quality clusters. If this value is close to 1, then an excessive number of poor quality clusters are formed. Therefore, it is necessary to reinforce the algorithm, in order to improve the quality of the resulting partitions.

Merge techniques allow generating clusters with better intra and inter-cluster characteristics, by merging similar partitions based on certain conditions. Recent works [37]–[39] have addressed these algorithms obtaining significant improvement in cluster construction.

Therefore, the main objective of our proposal is to improve the cluster partitions based on the implementation of a merge algorithm, and a new proposal for the MAD computation, which we have identified as the main problems of LAMDA.

III. LEARNING ALGORITHM FOR MULTIVARIABLE DATA ANALYSIS "LAMDA"

LAMDA is a fuzzy method based on the concept of the adequacy degree. The versatility of this algorithm lies in the fact that it does not require the number of clusters as an input parameter [19], and it is a non-iterative method that can work online, being these the main advantages in the unsupervised learning context. The classic algorithm performs a similarity evaluation of the descriptors of a sample X and the clusters $C = \{C_1; C_2 \ldots; C_k; \ldots; C_m\}$, where m is the number of pre-existing clusters not defined by the user [18], to define where the sample should be assigned.

In order to explain the fundamentals of LAMDA, several definitions and statements are formalized in this paper.

Statement 1: Let X be an unlabeled sample (individual), represented by a vector of n descriptors [12]:

$$X = \left[x_1; x_2; \ldots; x_j; \ldots; x_n\right] \tag{1}$$

where: x_j is the descriptor j of the object X

Statement 2: Let $\bar{x}_j$ be the normalized descriptor x_j. The normalization is computed with the maximum x_{jmax} and minimum x_{jmin} limits of that descriptor as:

$$\bar{x}_j = \frac{x_j - x_{jmin}}{x_{jmax} - x_{jmin}} \tag{2}$$

The normalized sample $\bar{X}$ is used to compute the adequacy degrees in each cluster.

Definition 1 (Marginal Adequacy Degree (MAD)): This parameter computes the similarity between the descriptor of a sample and the same descriptor in each cluster. For the MAD computation, probability density functions are used, specifically, fuzzy binomial function [40]:

$$MAD_{k,j}\left(\bar{x}_j \mid \rho_{k,j}\right) = \rho_{k,j}^{\bar{x}_j}\left(1 - \rho_{k,j}\right)^{(1-\bar{x}_j)} \tag{3}$$

where $\rho_{k,j}(t)$ is calculated using Eq. (4), and it is the mean value of the descriptor j in the previously created cluster k. It is updated progressively each time that a new element is added. $n_k(t-1)$ is the number of objects previously assigned to the cluster k.

$$\rho_{k,j}(t) = \rho_{k,j}(t-1) + \frac{\bar{x}_j(t) - \rho_{k,j}(t-1)}{n_k(t-1)+1} \tag{4}$$

The value of $MAD_{k,j}$ represents the adequacy degree of $\bar{x}_j$ in the same descriptor of the cluster k. In Eq. (3), if $\rho_{k,j}(t) = 0.5$, then $MAD_{k,j}\left(\bar{x}_j \mid \rho_{k,j}\right) = 0.5$ for any value of $\bar{x}_j$. This MAD value is considered for the Non–Informative Class (NIC), $MAD_{NIC,j}\left(\bar{x}_j \mid \rho_{NIC,j}\right) = 0.5$.

Definition 2 (Global Adequacy Degree (GAD)): The adequacy of a sample to each cluster is obtained with the Global Adequacy Degree (GAD), which is computed by mixing the MADs with aggregation functions. These are linear interpolations between the t-norm and t-conorm, like the Dombi operator (Eq. (5) and (6)) [41]. This operator presents a good performance for class generation, and it has been used in the evaluation of the maximum variation of clustering (see more details in [2]).

$$T(a,b) = \frac{1}{1+\sqrt[p]{\left(\frac{1-a}{a}\right)^p+\left(\frac{1-b}{b}\right)^p}} \tag{5}$$

$$S(a,b) = 1 - \frac{1}{1+\sqrt[p]{\left(\frac{a}{1-a}\right)^p+\left(\frac{b}{1-b}\right)^p}} \tag{6}$$

Generally, in the literature $p \geq 1$. We choose $p = 1$, in order to obtain a close approximation to a linear behavior of the t-norm and t-conorm [41].

The exigency parameter $0 < \alpha < 1$ is used to calibrate the fuzzy partition data [42]. In Eq. (7), if $\alpha = 1$, then the fuzzy partition data is computed by the t-norm. It means that the clustering is stricter, therefore, more objects will be unrecognized (sent to the NIC), creating more new clusters. If $\alpha = 0$, then the fuzzy partition data is computed by the t-conorm. It means that the clustering is more permissible, therefore, samples are assigned to a cluster, despite not having enough similarity with the samples belonging to it. α produces a linear interpolation between t-norm and t-conorm for the GAD [43].

$$\begin{aligned} &GAD_{\bar{X},k}\left(MAD_{k,1},\ldots MAD_{k,n}\right) \\ &= \alpha T\left(MAD_{k,1},\ldots,MAD_{k,n}\right) \\ &+ (1-\alpha)\, S\left(MAD_{k,1},\ldots,MAD_{k,n}\right) \end{aligned} \tag{7}$$

The GAD of the NIC is computed considering $MAD_{NIC,j} = 0.5$, regardless of the value of $\bar{x}_j$, that is:

$$GAD_{\bar{X},NIC} = \alpha T(0.5,\ldots,0.5) + (1-\alpha)\, S(0.5,\ldots,0.5) \tag{8}$$

Statement 3: Let *index* (*in*) the number of the identifier of the cluster where the sample X has the highest membership degree, that is, the highest GAD, established by:

$$in = arg\max_{GAD}\left(GAD_{\bar{X},1}, GAD_{\bar{X},k},\ldots,GAD_{\bar{X},m}, GAD_{\bar{X},NIC}\right) \tag{9}$$

If the object is assigned to the NIC, then it becomes the first element of a new cluster C_{new}, The above is represented as:

$$\rho_{new,j}(t) = \bar{x}_j \tag{10}$$

The LAMDA implementation is described in the pseudo-code presented below. The process starts normalizing each descriptor of the sample X. Next, the MADs are computed for each descriptor in each cluster using the fuzzy binomial function. With the MADs, the calculation of the GAD in each cluster is performed with fuzzy connectors considering the value α for the exigency. Finally, the cluster with the higher value of the GAD is where the sample $\bar{X}$ is assigned. If the higher GAD is the GAD_{NIC}, then it is considered that the sample does not belong to any cluster, and it is sent to the NIC to create a new partition.

Algorithm 1 LAMDA for Unsupervised Learning

Input: Sample $\bar{X}$
Procedure:
1. Normalize the sample $\bar{X}$ using Eq. (2).
2. $MAD_{k,j} \leftarrow \left(\bar{x}_j \mid \rho_{k,j}\right)$
3. $GAD_{\bar{X},k} \leftarrow \left(MAD_{k,1,\ldots MAD_{k,n}}\right)$
4. Identify the index *in* of the cluster with: $in = \arg\max\left(GAD_{\bar{X},1}, GAD_{\bar{X},k},\ldots,GAD_{\bar{X},m}, GAD_{\bar{X},NIC}\right)$
5. **if** the highest GAD is the corresponding to the NIC,
6. **then** the sample is the first element of the new cluster C_{m+1}, as is shown in Eq. (10).
7. **else** the object is assigned to the cluster with the highest GAD, and update $\rho_{k,j}(t)$ using Eq. (4).
8. End.
Output: cluster updated or created by the algorithm

IV. PROPOSED APPROACH

In [2], [19] has been shown that LAMDA creates clusters that do not correspond with the number of desired groups. Clusters with a high degree of similarity should be merged in a single cluster, according to a similarity measure. Thus, the algorithm should automatically decide when a merge process between clusters is required. For that, it is proposed to hybridize the original algorithm with distance measurements, in order to improve the quality of the clusters. The split task is considered as an intrinsic LAMDA feature, because it can create new groups from the global adequacy of an individual to each existing cluster and the NIC.

Section 4.A establishes several definitions to improve the computation of the MADs based on a Robust Distance criterion. Section 4.B presents the procedure to be followed to perform the merge process, while section 4.C shows the general procedure implemented to enhance the performance of the algorithm in unsupervised learning.

A. ROBUST DISTANCE

Definition 3 (Cauchy Marginal Adequacy Degree (CMAD)): This parameter corresponds to the *MAD* computed using the fuzzy Cauchy function [44] (see Eq. 11), which corresponds to find a membership function $\mu_c(x)$ that models the similarity of an individual to a cluster. $dist(x, x_0)$ is the distance of the individual x to a prototype member x_0. This function has been chosen because it allows computing the membership degree according to a normal distribution, with the addition of a penalty factor based on distances (Definition 4), to improve

the calculation of the *MADs*.

$$\mu_c(x) = \frac{1}{1 + dist(x, x_0)} \tag{11}$$

The application of Eq. (11) in LAMDA, for $x = \bar{x}_j$ and $x_0 = \rho_{k,j}$ (descriptor j of the centroid of the cluster k), now redefines the *MAD* as *CMAD* (see [21] for more details):

$$CMAD_{k,j}\left(\bar{x}_j \mid \rho_{k,j}\right) = \frac{1}{1 + dist(\bar{x}_j, \rho_{k,j})} \tag{12}$$

To keep the MAD criterion of Definition 1, it is set as $CMAD_{NIC,j}\left(\bar{x}_j \mid \rho_{k,j}\right) = 0.5$.

Definition 4 (Robust Marginal Adequacy Degree (RMAD)): This parameter corresponds to the product of the *CMAD* and a penalty factor $K_{k,\bar{X}}$ computed for each cluster k. To obtain $K_{k,\bar{X}}$, two parameters are required: the first one is the distance of the individual $\bar{X}$ to the center of each cluster k ($d_{k,\bar{X}}$), which is calculated as [21]:

$$d_{k,\bar{X}} = dist\left(\bar{x}_j, \rho_{k,j}\right) = \frac{1}{n}\sum_{j=1}^{n}\left|\bar{x}_j - \rho_{k,j}\right| \tag{13}$$

And the second parameter is the threshold $d_{nb} \in [0, 1]$, called ''average distance between neighbors'', which must be set by the user (in section 5.C is described a method to calibrate this parameter).

Statement 4: The penalty factor $K_{k,\bar{X}}$ is computed with Eq. (14). If the average distance $d_{k,\bar{X}}$ is greater than d_{nb} ($d_{k,\bar{X}} > d_{nb}$), then $K_{k,\bar{X}}$ is computed as [21]:

$$K_{k,\bar{X}} = \frac{d_{nb}}{d_{nb} + dist\left(d_{k,\bar{X}}, d_{nb}\right)} \tag{14}$$

As is shown in Eq. (14), if $dist\left(d_{k,\bar{X}}, d_{nb}\right)$ increases, then $K_{k,\bar{X}}$ decreases.

Statement 5: If the average distance $d_{k,\bar{X}}$ is less than d_{nb}, ($d_{k,\bar{X}} \leq d_{nb}$), then $K_{k,\bar{X}}$ is set to 1, because it is not required to penalize the *CMAD* of individuals that are within the threshold. Now, $RMAD_{k,j}$ is computed as [21]:

$$RMAD_{k,j}\left(\bar{x}_j \mid \rho_{k,j}\right) = K_{k,\bar{X}} \times CMAD_{k,j} \tag{15}$$

As is shown in Eq. (15), $RMAD_{k,j}$ is equal to $CMAD_{k,j}$ if the condition of statement 5 is met, this is, the distance between the individual $\bar{X}$ and the cluster k is within the threshold d_{nb}. According to statement 4, when the distance between the individual $\bar{X}$ and the cluster k is greater than the threshold, then *CMAD* is penalized; therefore, a decrease in the adequacy degree is established. The $K_{k,\bar{X}}$ parameter reinforces the measure of the degree of similarity based on distances. The two established conditions of $d_{k,\bar{X}}$ affect the computation of the *RMAD*. The following two properties *P*1 and *P*2 demonstrate it:

$$\begin{aligned} &P1: If\left(d_{k,\bar{X}} > d_{nb}\right)|d_{nb} \in [0,1] \\ &\Longrightarrow K_{k,\bar{X}} = \frac{d_{nb}}{d_{nb} + dist\left(d_{k,\bar{X}}, d_{nb}\right)} = \frac{d_{nb}}{d_{nb} + \delta}|\delta \in [0,1] \\ &\Longrightarrow K_{k,\bar{X}} < 1 \therefore RMAD_{k,j}\left(\bar{x}_j \mid \rho_{k,j}\right) < CMAD_{k,j} \end{aligned} \tag{16}$$

$$\begin{aligned} &P2: If\left(d_{k,\bar{X}} \leq d_{nb}\right)|d_{nb} \in [0,1] \\ &\Longrightarrow K_{k,\bar{X}} = 1 \therefore RMAD_{k,j}\left(\bar{x}_j \mid \rho_{k,j}\right) = CMAD_{k,j} \end{aligned} \tag{17}$$

The penalty factor for the *NIC* is set $K_{NIC,\bar{X}} = 1$, because it is not required to penalize the Non-Informative Class. As observed in Eqs. (16) and (17), the distance $d_{k,\bar{X}}$ allows penalizing the dissimilarity between the samples and the clusters. This parameter is called Robust Distance, hence, this proposal takes the name of LAMDA-RD. Once calculated *RMAD*, the computation of the *GAD* is like the original LAMDA, using Definition 2 and Statement 3, but now with *RMAD* instead of *MAD*.

B. AUTOMATIC MERGE ALGORITHM

To describe the automatic merge algorithm for LAMDA, the following definitions are formalized:

Definition 5 (A Cluster C_k): It is described by the tuple:

$$C_k = \left(\rho_{k,j}, \bar{X}_k, index, n_k\right) \tag{18}$$

where $\rho_{k,j}$ is the centroid of the descriptor j in the cluster k, which must be updated every time that a new individual is assigned to C_k (see Eq.(4)), $\bar{X}_k$ is the set of individuals in C_k, *index* is the identifier of C_k, and n_k is the number of samples in the cluster.

Definition 6 (The Neighbor Cluster C_{nb}): It is described as:

$$C_{nb} = \left(\rho_{nb,j}, \bar{X}_{nb}, index, n_{nb}\right) \tag{19}$$

where $\rho_{nb,j}$ is the centroid of the descriptor j in the cluster nb, $\bar{X}_{nb}$ is the set of individuals in C_{nb}, *index* is the identifier of C_{nb}, and n_{nb} is the number of individuals in the cluster.

LAMDA is non-iterative, therefore, in the clustering process, one individual is analyzed at a time (a useful feature in streaming data). So, according to the LAMDA fundamentals, the GADs are the membership degrees that an individual has in each cluster [45]. The maximum GAD is where the individual is assigned, so, we can conclude that the second GAD of greater value is the nearest neighbor cluster.

The main problem to solve in this paper is the drawback of the original LAMDA: the excessive creation of clusters. So, it is essential to perform an automatic merge. Our proposal is characterized by similarity measures based on distances and densities. In the merge stage, we can have two cases:

a) If the individual was assigned to the NIC, and therefore, a new cluster was created. It is the case: individual – cluster (see Figure 1-a).
b) If the individual was assigned to an existing cluster C_k. It is the case: cluster – cluster (see Figure 1-b).

The procedure of the merge process is described in the pseudo-code presented below.

Definition 7 (Measure of the Compactness of the Neighbor Cluster $(t_{nb,j})$): It is the mean value of all the distances (in each descriptor) among the individuals belonging to the neighbor cluster C_{nb}, and it is computed as:

$$t_{nb,j} = \frac{\sum_{i=1}^{n_{nb}-1}\sum_{m=i+1}^{n_{nb}}\left|\bar{x}_{nb,j}^{i} - \bar{x}_{nb,j}^{m}\right|}{n_{nb} \times (n_{nb}-1) \times \ldots \times 1};\ \forall j = 1, \ldots, n \tag{20}$$

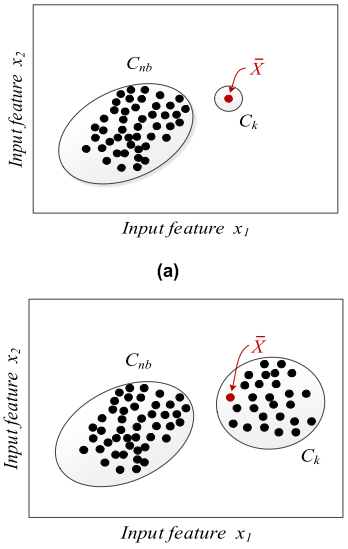

FIGURE 1. New sample assigned to (a) new cluster, (b) pre-existing cluster.

Algorithm 2 Merging Process

Input: Clusters C_k and C_{nb}.
Procedure:

1. Calculate the compactness of the neighboring cluster $t_{nb,j}$ using Definition 7.
2. Calculate the distance between the individuals of the cluster C_k and C_{nb}. Individuals whose distances are less than $t_{nb,j}$ in each descriptor j belong to the overlap zone (see Definition 8).
3. Determine the ratio of individuals in the overlap area with respect to the total of individuals between the two neighboring clusters (D_{k-nb}), as is shown in Definition 9.
4. Set a density threshold in the overlapping area D_t, and verify if the condition $D_{k-nb} \geq D_t$ is met to proceed with the merge process (Statement 6).
5. End.

Output: cluster updated or created by the algorithm

where $\bar{x}^i_{nb,j}$ is the descriptor j of the individual i in the cluster C_{nb}.

Definition 8 (Number of Individuals in the Overlapping Area (N_I)): This parameter is computed by counting the individuals in the overlapping area of the clusters C_k and C_{nb}, whose distance between its individuals is less than $t_{nb,j}$. For this, we first identify the individuals of each cluster C_k and C_{nb} that meet that condition, and then, the cardinality of the resulting subsets is calculated as:

$$N_k = \left\{\forall \bar{x}_{k,j} \in C_k \mid d\left(\bar{x}_{k,j}, \bar{x}_{nb,j}\right) < t_{nb,j}; \forall j = 1, \ldots, n\right\} \Rightarrow N_k = n\left(N_{kI}\right) \quad (21)$$

$$N_{nb} = \left\{\forall \bar{x}_{nb,j} \in C_{nb} \mid d\left(\bar{x}_{k,j}, \bar{x}_{nb,j}\right) < t_{nb,j}; \forall j = 1, \ldots, n\right\} \Rightarrow N_{nb} = n\left(N_{nbI}\right) \quad (22)$$

where N_k and N_{nb} are the number of individuals in the overlapping area for the cluster C_k and C_{nb}, respectively. The total number of individuals in the overlapping area N_I is:

$$N_I = N_k + N_{nb} \quad (23)$$

Definition 9 (D_{k-nb}): It is the density in the overlapping area between two clusters C_k and C_{nb}, and it is computed as:

$$D_{k-nb} = \frac{N_I}{n_{nb} + n_k} \quad (24)$$

Statement 6: Two clusters C_k and C_{nb} are merged, if $D_{k-nb} \geq D_t \cdot D_t \in [0, 1]$ is a density threshold set by the user. A high D_t value implies a greater density of individuals in the overlapping area.

Figure 2 shows the cases in which the condition of statement 6 is not satisfied. It is observed that the new individual $\bar{X}$ increases the density of the overlapped area between the clusters C_k and C_{nb}. However, if $D_{k-nb} < D_t$, then the algorithm does not proceed to do the merge process, considering that there is not enough similarity between the two analyzed partitions.

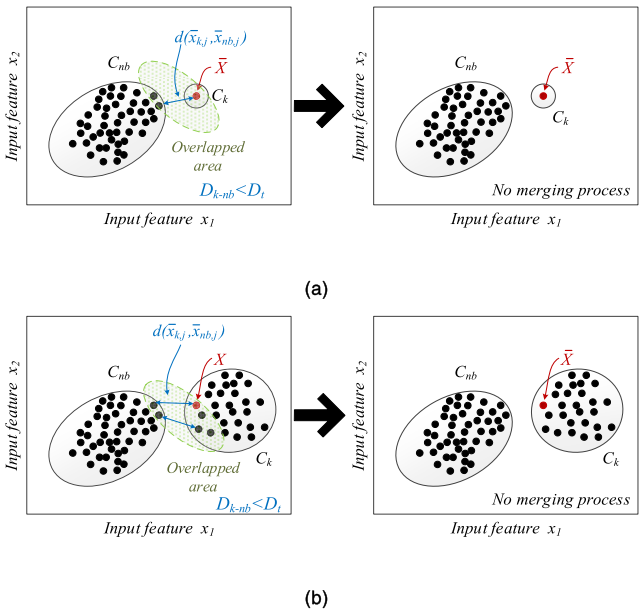

FIGURE 2. Graphical example to assign a new sample to a cluster, when statement 6 is not met, (a) the sample creates a new cluster, or (b) the sample is assigned to a pre-existing cluster.

Figure 3 shows the cases in which the statement 6 is satisfied. It is observed that the new individual $\bar{X}$ increases the density of the overlapped area between the clusters C_k and C_{nb}. If $D_{k-nb} \geq D_t$, then the algorithm proceeds to do the merge process, considering that there is enough similarity between the two analyzed groups.

Definition 10 (Resulting New Cluster (C_{new})): The resulting cluster after the merge process is given by the tuple:

$$C_{new} = \left\{\rho_{new,j}, \bar{X}_k \cup \bar{X}_{nb}, index, n_k + n_{nb}\right\} \quad (25)$$

$$\rho_{new,j} = \frac{1}{n_k + n_{nb}} \sum_{t=1}^{n_k+n_{nb}} \bar{x}^t_{new,j} \quad (26)$$

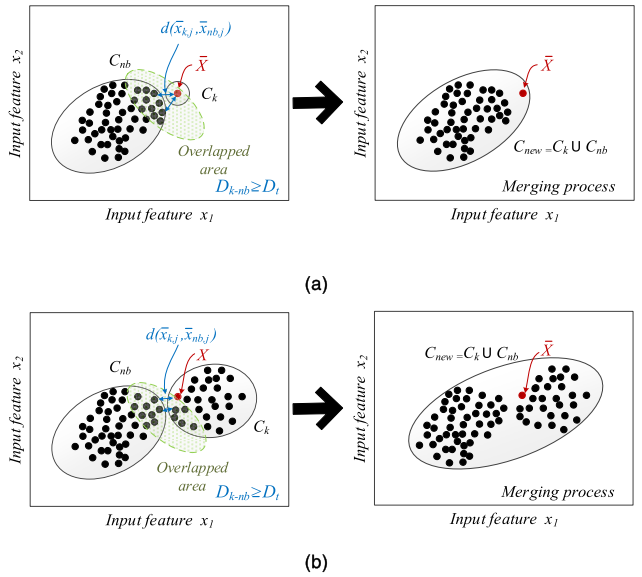

FIGURE 3. Graphical example to assign a new sample to a cluster, if statement 6 is met, (a) the algorithm merges the individual to the neighbor cluster, (b) the algorithm merges the cluster where the individual was assigned with the neighbor cluster.

where $\bar{x}^t_{new,j}$ is the descriptor j of the individual t in the clusters C_k and C_{nb} that form the new cluster C_{new}.

As shown in Figure 3, each time that an individual is assigned to a cluster, it is evaluated if the density of the overlapping area has increased. The density is considered as a requirement to determine if the merge process should be executed according to the threshold D_t.

C. GENERAL PROCEDURE OF LAMDA-RD WITH AUTOMATIC MERGE ALGORITHM

This procedure is repeated for each individual. The scheme of Figure 4 details the fuzzy clustering stage with the extension of an automatic merge stage based on distances and densities. The first step is the normalization of the descriptors of the individual to be assigned to a cluster. Next, the *RMAD* calculations are made for each descriptor in each cluster, using the Cauchy function, which considers the $K_{k,\bar{X}}$ parameter that penalizes the dissimilarity between the individual and the clusters based on distances, as is shown in the Eqs. (14) and (15). With *RMAD*, the *GAD* in each cluster is computed, setting a high value for the exigency level ($\alpha = 1$), with the aim to get a strict behavior (non-permissive algorithm).

The highest GAD defines the cluster in which the individual must be assigned (and its parameter $\rho_{k,j}$ is updated). However, if the maximum GAD corresponds to the NIC, then a new cluster is created, being this individual the first sample of the new group. In the merge stage is evaluated if this process is required between the cluster in which the individual was assigned and the neighboring cluster (defined by the second-largest GAD), this because the individual can

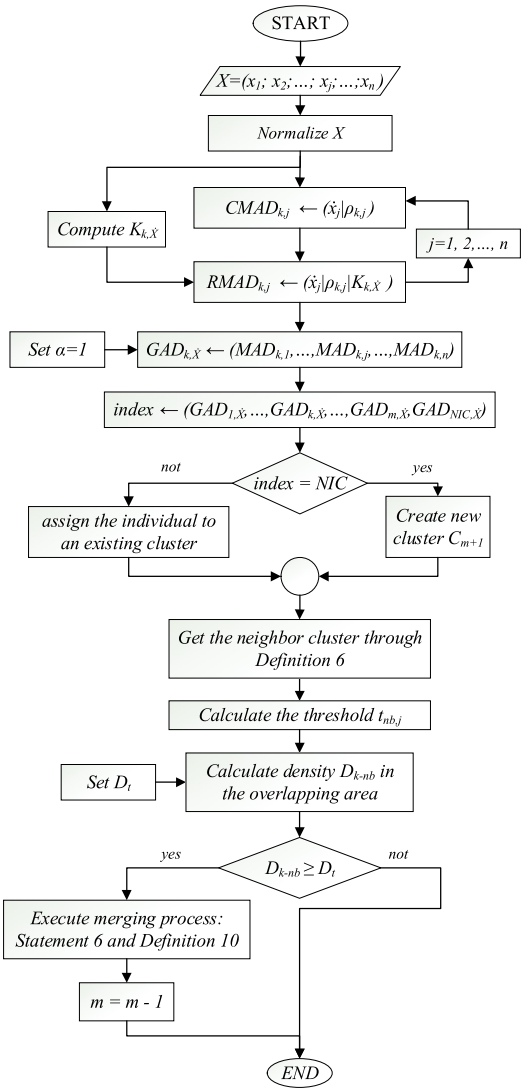

FIGURE 4. LAMDA-RD algorithm.

be located in the overlapping zone between both clusters, fulfilling the merge requirement of statement 6.

In general, the algorithm starts with $m = 0$. When the first sample to be evaluated arrives, then the first cluster is created ($m = 1$). Next, when the second sample arrives, then it is evaluated, and if the conditions established by the algorithm are met, then this sample is assigned to cluster 1, otherwise, a new cluster is created ($m = 2$). This process is followed successively for all the samples, until evaluating the last sample N, assigning it to one of the current clusters or a new one. Thus, the algorithm does not require the definition of the number of clusters (m). The number of clusters depends on the data characteristics.

V. EXPERIMENTS AND RESULTS

In this section, the experimental tests in different clustering tasks are presented. The goal of the experiments is to validate the proposed method, analyzing the cluster quality and the

Algorithm 3 LAMDA-RD for Unsupervised Learning

Input: Sample X

Procedure:

1. Normalize the sample with statement 2, to obtain $\bar{X}$.
2. Set the values of d_{nb} and D_t.
3. $CMAD_{k,j} \leftarrow \left(\bar{x}_j \mid \rho_{k,j}\right)$
4. $\mathrm{R}MAD_{k,j} \leftarrow \left(\bar{x}_j \mid \rho_{k,j} | K_{k,\bar{X}}\right)$
5. $GAD_{\bar{X},k} \leftarrow \left(RMAD_{k,1}, \ldots RMAD_{k,n}\right)$
6. Identify the index in of the cluster with: $in = \text{argmax}\left(GAD_{\bar{X},1}, GAD_{\bar{X},k}, \ldots, GAD_{\bar{X},m}, GAD_{\bar{X},NIC}\right)$
7. **If** $\text{argmax}(GAD_{k,\bar{X}}) > GAD_{NIC,\bar{X}}$, **then**

- assign $\bar{X}$ to a cluster
- update $\rho_{k,j}$
- conserve current m

else

- create a new cluster
- $m = m + 1$

end

8. Identify the neighbor cluster and the two possible cases for merging stage:

- individual – cluster
- cluster – cluster

9. For the clusters k and nb :

9.1. Compute the compactness of the cluster $t_{nb,j}$ using Definition 7.

9.2. Compute the number of individuals in the overlapping area N_I using Definition 8.

9.3. Compute the density in the overlapping area D_{k-nb} using Definition 9.

9.4. **If** $D_{k-nb} \geq D_t$, **then**

- execute merging through Definition 10.
- update $\rho_{k,j}$
- $m = m - 1$

10. End.

Output: cluster updated or created by the algorithm

performance of LAMDA-RD. The following tests are carried out in this section:

i. A general validation among LAMDA-RD, original LAMDA, and other well-known clustering methods, tested in different benchmarks, making a comparative analysis of the quality of the results. In a first test, we compute statistics for clustering validation that handles the criterion that the labels of the clusters are unknown. Then, the evaluation is performed considering the intrinsic and extrinsic characteristics of the obtained model. These metrics are: Silhouette Coefficient (SC), Modification of the Silhouette coefficient ($SILA$), Sum-of-squares within clusters (SSW), Sum-of-squares between clusters (SSB), WB-index (WB), and Performance Coefficient (P_c) based on SC and $SILA$. In a second test, we consider the datasets with labeled clusters (like classes in supervised learning), to compare the formed partitions performed by the algorithms against the real classes; this is a standard evaluation procedure for clustering used to compute the Rand Index (RI).

ii. A comparative analysis in a streaming data scenario among LAMDA family (LAMDA-RD, LAMDA-TP and original LAMDA) and the algorithm called "Autonomous Data-driven Clustering for Live DataStream (ADDclustering) 46]. This algorithm has been selected since it allows online clustering, a characteristic to be considered in order to make a fair comparison with LAMDA. In this test, the individuals are acquired from streaming data, since the algorithms are based on an online operation.

iii. The applicability LAMDA-RD to a real case study (gas lift well production), to evaluate the behavior of our proposal in a real scenario.

The tests described in i) and ii) are validated using datasets from [47]–[49]. We have selected the datasets due to different characteristics such as: number of individuals and features, level of intra-cluster overlap to observe how the allocation of individuals is made in those cases, balanced and unbalanced classes, and finally, the number of clusters (see detail in Table 1). Datasets with a large number of data are: Dim 1024, Unbalance and Postures (high-dimensional) and their analysis is required to observe the cluster quality and to measure the machine time to perform the partitions. 2-dimensional datasets are used for visualization purposes, to easily observe the behavior of the different algorithms. In all benchmarks, the original dimensionality has been maintained to make a fair comparison between the algorithms (tested under the same characteristics). A correlation analysis to reduce the dimensionality of the datasets (i.e. PCA) is not the objective of this paper during the validation of our proposal, but it is an analysis that should be considered in future works.

The following metrics have been chosen, in order to evaluate the intrinsic and extrinsic characteristics of the obtained model, and the analysis of the intra and extra-cluster qualities. A more detailed description of them can be found in [50].

Silhouette coefficient (SC) : it is a metric between $[-1,1]$, -1 for incorrect clustering and 1 for highly dense clustering (dense and well separated), values around zero indicate overlapping clusters. This is composed of two values, $a_{SC}(x)$ is the mean distance between an individual and all other individuals in the same cluster, and $b_{SC}(x)$ is the mean distance between an individual and all other individuals in the nearest cluster. If the value is bigger, then the clustering is better. Considering N, the number of elements of the dataset, SC is computed as:

$$SC = \frac{1}{N} \sum_{x \in X} \left(\frac{b_{SC}(x) - a_{SC}(x)}{max\left(a_{SC}(x), b_{SC}(x)\right)} \right) \tag{27}$$

Modification of the Silhouette coefficient (SILA): It improves the analysis of the Silhouette coefficient because SC can show an incorrect partitioning scheme when there are large differences in distances between groups. The SILA

TABLE 1. Datasets used to test the clustering algorithms.

Dataset	# Individuals	# Features	Overlapping	Characteristics/number of clusters
Dim 1024	1024	1024	0%	High-dimensional datasets with 16 clusters, it is used to evaluate the behavior of the algorithms with high-dimensional features.
Segment	2310	19	unknown	7 clusters, it is a dataset of segmented images to perform classification/clustering. It is used to evaluate the behavior of the algorithms with several individuals and features.
Hepta	212	3	0%	Data created for benchmarking purposes for clustering algorithms, with 7 clearly defined clusters and different densities.
R15	600	2	0%	15 clusters corresponding to types to 2D Gaussian groups that are positioned in rings. There are 40 vectors per cluster.
Aggregation	788	2	0%	Dataset consists of 7 perceptually distinct clusters
Unbalance	6500	2	0%	Synthetic 2-d data with 8 Gaussian clusters. it is used to evaluate the behavior of the algorithms with unbalanced clusters.
s1	5000	2	9%	Synthetic data with 15 Gaussian clusters with 9% of cluster overlapping. Dataset used to evaluate the groups formed by the algorithms with 9% overlapping.
s2	5000	2	20%	Synthetic data with 15 Gaussian clusters with 20% of cluster overlapping. Dataset used to evaluate the groups formed by the algorithms with 20% overlapping.
s3	5000	2	41%	Synthetic data with 15 Gaussian clusters with 41% of cluster overlapping. Dataset used to evaluate the groups formed by the algorithms with 41% overlapping.
a1	3000	2	22%	Synthetic data with 20 clusters. There are 150 elements per cluster. Dataset used to evaluate the groups formed by the algorithms with 22% overlapping.
Postures	74975	15	Unknown	5 types of hand postures from 12 users have been recorded using unlabeled markers on the fingers of a glove in a motion capture environment. It is used to evaluate the behavior of the algorithms with high-dimensional individuals. Only used for streaming data scenario.

index contains an additional component to overcome this drawback. SILA measures the cluster compactness, which increases when a cluster size increases considerably, and it reduces the high values of the index caused by large differences between the groups [51]. This index varies in the same range as SC.

$$SILA = \frac{1}{N}\sum_{x\in X}\left(\frac{b_{SC}(x) - a_{SC}(x)}{max\left(a_{SC}(x), b_{SC}(x)\right)} \times \frac{1}{1 + a_{SC}(x)}\right) \tag{28}$$

Sum-of-squares within clusters (SSW): it is an internal measure used to evaluate the cohesion of the clusters that the algorithm has generated. The smaller the value is, the better the clustering. It is defined by Eq. (28).

$$SSW(C, m) = \frac{1}{N}\sum_{k=1}^{m}\sum_{i\epsilon C_k}\left\|\bar{X}_k^i - \rho_k\right\| \tag{29}$$

$\bar{X}_k^i$ is the $i-th$ individual in the cluster C_k, ρ_k is its centroid, and m is the number of clusters.

Sum-of-squares between clusters (SSB): it is a prototype-based separation measure used to evaluate the inter-cluster distance. If the value is bigger, then the clustering is better. It is defined by Eq. (29).

$$SSB(C, m) = \frac{1}{n_k}\sum_{k=1}^{m} n_k\left(\rho_k - \rho_g\right) \tag{30}$$

where n_k is the number of elements in the cluster k, ρ_g is the mean value of the whole data set (global center).

WB-index (WB_{index}) [50]: it is based on SSW and SSB. It emphasizes the effect of SSW multiplying it by the generated number of clusters m. This metric is an alternative to methods based on knee point detection because most indices show monotonicity with an increasing number of clusters. Therefore, indices with a clear minimum or maximum value

are preferred, being WB_{index} one of them. Being a relationship between SSW and SSB, it can be noted that the lower its value, the better the quality of the formed clusters. In cases in which it is necessary to know the optimal number of groups, the WB-index are plotted for different number of partitions, and the model with the minimum value is chosen as the optimum. This index is defined in Eq. (30).

$$WB_{index} = \frac{m \times SSW}{SSB} \tag{31}$$

Performance Coefficient (P_c)*:* it is a metric that we propose, which is a relationship between SC or $SILA$ and WB-index, in order to establish which of the tested algorithms presents the best performance. The value of P_C must be minimal and greater than zero, because WB must be small and SC or SILA must be positives and close to 1, to establish good formed clusters.

$$P_C = \frac{WB_{index}}{SC} or P_C SILA = \frac{WB_{index}}{SILA} \tag{32}$$

Rand Index (RI)*:* It is one of the most known indices for measuring the similarity between partitions, being necessary the construction of a consensus matrix [52]. Assume that there are two partitions $P^{(r)}$, $r = 1, 2$, in a set of N individuals $S = \{s_1, \ldots, s_N\}$. Partition $P^{(1)}$ has k_1 clusters, and partition $P^{(2)}$ has k_2 clusters. In order to compare the two partitions, the following terms are defined with two pairs of individuals coming from $P^{(1)}$ and $P^{(2)}$:

a: the number of the pairs of s_i and s_j belonging to the same cluster in $P^{(1)}$ and $P^{(2)}$.

b: the number of the pairs of s_i and s_j belonging to the same cluster in $P^{(1)}$ and to different clusters in $P^{(2)}$.

c: the number of the pairs of s_i and s_j belonging to different clusters in $P^{(1)}$ and to the same cluster in $P^{(2)}$

d: the number of the pairs of s_i and s_j belonging to different clusters in $P^{(1)}$ and $P^{(2)}$.

$$RI(P^{(1)}, P^{(2)}) = \frac{a+d}{a+b+c+d} = \frac{a+d}{\binom{N}{2}} \tag{33}$$

The term $a + d$ is the number of agreements between the partitions $P^{(1)}$ and $P^{(2)}$. On the other hand, $b+c$ is the number of disagreements between the partitions $P^{(1)}$ and $P^{(2)}$. The Rand Index computes the number of agreements over the total pairs.

A. COMPARISON OF LAMDA-RD WITH OTHER CLUSTERING ALGORITHMS

In the following experiments, the parameters of the compared algorithms are tuned with the same care and separately for each dataset, to make a fair comparison. The calibration procedure of LAMDA-RD is presented in section 5.C, analyzing the sensitivity of the results.

This test is done to compare the quality of the formed clusters with respect to the results original LAMDA and other methods, which are generally iterative, do not work online, and require the number of clusters as input parameter. These algorithms will serve to make a good comparison in terms of performance. Some conventional algorithms (K-means (KM), K-medoids (KMD), Fuzzy c-means (FCM), DBSCAN (DBS)), and some new algorithms, such as Agglomerative hierarchical tree (AHT) [4], Spectral clustering (SPC) [53], Hierarchical density-based clustering (HDBSCAN "HDB") [54] and Link-based cluster ensemble framework with consensus function (CON) [55], are tested for the comparison.

Figure 5 shows the methodology used for this experiment. It should be noted that LAMDA works with streaming data to form clusters, while the other algorithms require the complete dataset for this purpose; however, this test allows evaluating the quality of the created clusters in tasks in which historical data are available. For Eqs. (5) and (6), as we mentioned in Section III, we select for all the experiments $p = 1$.

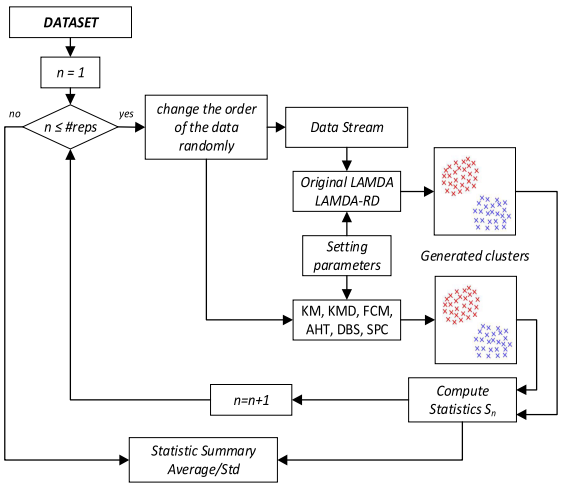

FIGURE 5. Methodology used for the comparison of LAMDA-RD with other approaches.

In order to obtain more reliable results, the experiment is repeated 20 times *(#reps=20)*, each time performance metrics are computed, and from the obtained results, the "Average" and standard deviation "Std" of the metrics are computed, for observing the repeatability and the confidence interval for the experiment in the creation of clusters

Table 2 presents the SC for each clustering algorithm, where the best average (highest value) in each benchmark has been marked in bold text. The standard deviation shows the variability of the results in the different tests.

The best algorithms have SC values the closest to 1, identifying dense and well-separated clusters. In all benchmarks, LAMDA-RD is better than original LAMDA, in most cases significantly improving the quality of the created partitions, for instance, see the results in Segment, or in the cases of Unbalance, s1, s2, s3 and a1, where SC goes from negative values (bad clustering) to positive values, in some cases better than conventional algorithms (SC close to 1). We can also observe that LAMDA-RD obtains as good performance as the best clustering algorithms in benchmarks as Dim1024,

TABLE 2. SC for different clustering algorithms.

Dataset	Statistics	LAMDA	LAMDA-RD	KM	KMD	FCM	AHT	DBS	SPC	HDB	CON
Dim1024	Average	**1.000**	**1.000**	**1.000**	**1.000**	0.581	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Std	0.000	0.000	0.000	0.000	0.031	0.000	0.000	0.000	0.000	0.000
Segment	Average	0.253	**0.569**	0.491	0.497	0.510	0.416	0.375	0.211	0.467	0.467
	Std	0.003	0.045	0.028	0.017	0.020	0.000	0.000	0.003	0.001	0.047
Hepta	Average	0.848	**0.882**	0.770	**0.882**	**0.882**	**0.882**	**0.882**	**0.882**	**0.882**	0.875
	Std	0.030	0.002	0.067	0.000	0.000	0.000	0.000	0.000	0.017	0.033
R15	Average	0.079	0.899	0.796	**0.902**	0.901	0.895	0.875	0.901	0.886	0.890
	Std	0.062	0.128	0.055	0.000	0.000	0.000	0.000	0.000	0.001	0.028
Aggregation	Average	0.339	0.565	0.638	**0.643**	0.621	0.617	0.608	0.612	0.625	0.614
	Std	0.048	0.015	0.017	0.007	0.028	0.000	0.001	0.000	0.020	0.034
Unbalance	Average	-0.164	**0.941**	0.898	0.892	0.781	0.805	0.938	0.837	0.940	0.933
	Std	0.077	0.034	0.065	0.074	0.069	0.000	0.000	0.020	0.000	0.045
s1	Average	-0.281	**0.854**	0.806	0.838	0.847	0.853	0.826	0.849	0.826	0.845
	Std	0.028	0.050	0.034	0.057	0.038	0.000	0.000	0.001	0.000	0.001
s2	Average	-0.201	0.719	0.744	0.748	0.780	0.743	0.590	**0.782**	0.658	0.717
	Std	0.027	0.063	0.040	0.046	0.033	0.000	0.000	0.004	0.000	0.012
s3	Average	-0.278	0.458	0.634	0.638	**0.653**	0.430	-0.328	0.635	0.198	0.573
	Std	0.015	0.037	0.022	0.020	0.020	0.000	0.000	0.022	0.000	0.054
a1	Average	--	0.653	0.715	0.719	0.721	0.679	0.537	**0.729**	0.563	**0.729**
	Std	--	0.050	0.019	0.026	0.026	0.000	0.001	0.047	0.000	0.021

TABLE 3. SILA for different clustering algorithms.

Dataset	Statistics	LAMDA	LAMDA-RD	KM	KMD	FCM	AHT	DBS	SPC	HDB	CON
Dim1024	Average	**0.002**	**0.002**	**0.002**	**0.002**	-0.005	**0.002**	**0.002**	**0.002**	**0.002**	**0.002**
	Std	0.000	0.000	0.000	0.000	0.031	0.000	0.000	0.000	0.000	0.000
Segment	Average	0.225	**0.489**	0.388	0.397	0.408	0.296	0.353	0.174	0.455	0.372
	Std	0.004	0.039	0.022	0.015	0.016	0.000	0.000	0.034	0.002	0.037
Hepta	Average	0.819	**0.867**	0.749	**0.867**	**0.867**	**0.867**	**0.872**	**0.867**	**0.867**	0.860
	Std	0.026	0.001	0.070	0.000	0.000	0.000	0.000	0.000	0.015	0.035
R15	Average	0.074	0.889	0.776	**0.900**	0.889	0.893	0.841	0.881	0.862	0.875
	Std	0.032	0.016	0.049	0.000	0.000	0.000	0.000	0.000	0.001	0.028
Aggregation	Average	0.310	0.551	0.619	**0.624**	0.602	0.600	0.580	0.601	0.592	0.597
	Std	0.032	0.051	0.017	0.007	0.027	0.000	0.001	0.000	0.015	0.033
Unbalance	Average	-0.202	**0.922**	0.887	0.886	0.756	0.800	0.892	0.836	0.910	0.932
	Std	0.054	0.031	0.058	0.085	0.061	0.000	0.000	0.002	0.000	0.045
s1	Average	-0.299	**0.829**	0.795	0.825	0.841	0.849	0.792	0.843	0.801	0.817
	Std	0.021	0.053	0.044	0.064	0.038	0.000	0.000	0.000	0.000	0.001
s2	Average	-0.235	0.714	0.738	0.702	0.730	0.738	0.585	**0.742**	0.639	0.713
	Std	0.019	0.071	0.046	0.033	0.035	0.000	0.000	0.020	0.000	0.012
s3	Average	-0.308	0.439	0.626	0.632	**0.643**	0.385	-0.335	0.631	0.159	0.568
	Std	0.024	0.026	0.026	0.018	0.013	0.000	0.000	0.021	0.000	0.054
a1	Average	--	0.649	0.713	0.715	0.715	0.676	0.519	**0.721**	0.550	**0.721**
	Std	--	0.049	0.021	0.032	0.030	0.000	0.001	0.054	0.000	0.021

and Hepta. Also is the best algorithm for Segment, Unbalance and s1, which are datasets of balanced and unbalanced distribution, with a maximum intra-cluster overlap of 9%. In the benchmarks R15, Aggregation and s2, our approach presents results very close to the best value (KMD). In s3 and a1, the algorithm decreases its performance due to the dispersion of the individuals (the overlap increases). Nevertheless, based on *SC*, it is observed that LAMDA-RD, in s3 and a1 datasets, presents better results with respect to DBS.

The weakness of LAMDA-RD in datasets with high overlap occurs since the number of clusters to be built is unknown. It has the same problems as density techniques as DBS and HDB, which decrease their performance since they are not based on distance optimization criteria, like KM, FCM or AHT, which is an important criterion in this context.

The Std in all cases shows good repeatability in the experiments, which makes it possible to notice that similar results are obtained in each iteration. The worst case is given in R15, where Std (0.128) reaches 14% of the average value, giving an idea of a good behavior of the algorithm.

The SILA index results are shown in Table 3, where it can be seen that the index decreases minimally with respect to SC (Table 2). The only benchmark where a considerable difference is seen is for Dim1024 (SC=1 and SILA =0.002). This index decreases due to the great separability that exists between the 16 clusters, however, it is shown that all the

TABLE 4. WB-index for different clustering algorithms.

Dataset	Statistics	LAMDA	LAMDA-RD	KM	KMD	FCM	AHT	DBS	SPC	HDB	CON
Dim1024	Average	**0.142**	**0.142**	**0.142**	**0.142**	2.243	**0.142**	**0.142**	**0.142**	**0.142**	**0.142**
	Std	0.000	0.000	0.000	0.000	0.068	0.000	0.000	0.000	0.000	0.000
Segment	Average	32.15	4.559	3.636	3.491	**3.452**	7.087	5.051	4.199	3.542	3.717
	Std	1.290	0.227	0.137	0.084	0.074	0.000	0.000	0.005	0.003	0.143
Hepta	Average	2.720	**1.755**	2.405	**1.755**	**1.755**	**1.755**	**1.755**	**1.755**	**1.755**	1.798
	Std	0.491	0.124	0.387	0.000	0.000	0.000	0.000	0.000	0.021	0.193
R15	Average	22.75	1.432	1.759	**1.423**	1.426	1.433	1.474	1.424	1.463	1.453
	Std	2.758	0.139	0.173	0.000	0.000	0.000	0.000	0.000	0.001	0.082
Aggregation	Average	3.857	2.288	2.147	**2.111**	2.145	2.195	2.454	2.185	2.335	2.164
	Std	0.309	0.306	0.062	0.005	0.050	0.000	0.002	0.002	0.001	0.054
Unbalance	Average	11.54	**1.033**	1.036	1.040	1.104	2.948	1.081	1.100	1.078	1.035
	Std	0.832	0.069	0.035	0.038	0.061	0.000	0.000	0.002	0.000	0.033
s1	Average	30.50	**1.772**	2.044	1.834	1.785	1.781	1.779	1.780	1.779	1.798
	Std	1.590	0.165	0.217	0.308	0.216	0.000	0.000	0.001	0.000	0.002
s2	Average	57.52	2.438	2.424	2.367	2.192	2.394	2.675	**2.127**	2.481	2.241
	Std	1.452	0.234	0.250	0.244	0.183	0.000	0.000	0.004	0.000	0.011
s3	Average	91.86	5.583	3.077	3.050	**2.922**	4.148	7.759	2.931	7.107	3.208
	Std	2.478	0.261	0.144	0.144	0.113	0.000	0.002	0.118	0.000	0.248
a1	Average	55.20	2.895	2.778	2.752	2.718	2.894	2.434	2.724	**2.311**	2.705
	Std	2.082	0.144	0.108	0.150	0.126	0.000	0.001	0.451	0.002	0.096

TABLE 5. PC metric for different clustering algorithms.

Dataset	Statistics	LAMDA	LAMDA-RD	KM	KMD	FCM	AHT	DBS	SPC	HDB	CON
Dim1024	Average	**0.142**	**0.142**	**0.142**	**0.142**	3.920	**0.142**	**0.142**	**0.142**	**0.142**	**0.142**
	Std	0.000	0.000	0.000	0.000	0.354	0.000	0.000	0.000	0.000	0.000
Segment	Average	127.0	8.012	7.433	7.039	**6.783**	17.019	13.484	8.012	7.574	8.070
	Std	15.56	0.156	0.591	0.421	0.440	0.000	0.000	0.156	0.004	1.266
Hepta	Average	3.229	**1.989**	3.181	**1.989**	**1.989**	**1.989**	**1.989**	2.104	**1.989**	2.068
	Std	0.689	0.286	0.712	0.000	0.000	0.000	0.000	0.286	0.030	0.355
R15	Average	110.6	1.592	2.237	**1.577**	1.589	1.601	1.684	1.592	1.596	1.636
	Std	32.00	0.168	0.404	0.000	0.000	0.000	0.000	0.168	0.002	0.154
Aggregation	Average	11.75	4.717	3.369	**3.281**	3.465	3.557	4.039	4.717	3.821	3.537
	Std	2.906	2.444	0.189	0.042	0.246	0.000	0.007	2.444	0.003	0.299
Unbalance	Average	-34.37	**1.095**	1.162	1.176	1.430	3.661	1.171	**1.095**	1.111	1.106
	Std	3.264	0.064	0.117	0.134	0.192	0.000	0.000	0.064	0.000	0.079
s1	Average	-109.5	**2.044**	2.551	2.223	2.121	2.078	2.150	**2.044**	2.150	2.110
	Std	10.49	0.336	0.366	0.556	0.353	0.000	0.000	0.336	0.000	0.008
s2	Average	-290.4	3.395	3.285	3.197	**2.824**	3.222	4.532	3.395	3.770	2.923
	Std	37.32	0.728	0.523	0.574	0.375	0.000	0.000	0.728	0.000	0.197
s3	Average	-329.5	12.301	4.864	4.790	**4.487**	8.641	-23.65	12.301	35.89	5.691
	Std	14.74	1.525	0.395	0.368	0.317	0.000	0.009	1.525	0.000	0.104
a1	Average	--	4.473	3.891	3.837	3.783	4.261	4.530	3.973	4.103	**3.769**
	Std	--	0.589	0.249	0.350	0.317	0.000	0.007	0.589	0.002	0.241

algorithms were able to make a good clustering since each group is well defined in the dataset. The results obtained in each benchmark are consistent between SC and SILA, showing that the best algorithms are the same for these two metrics.

The results of WB_{index} for each clustering algorithm are presented in Table 4, where the best average (lowest value) in each benchmark has been marked in bold text.

As in the previous metric, our algorithm is the best for the WB_{index} in the datasets: dim 1024, Hepta and s1, where the individuals have a percentage of overlap under (9%), and in the case where the clusters are unbalanced (Unbalance).

In the datasets Segment, R15 and Aggregation, our method is very close to the best values, as explained before, in cases where there is no overlap between groups. For s2, s3 and a1, the performance of our method decreases, due to the presence of individuals in overlapping areas. The other methods can build better models because they know the number of clusters to build; this is evidenced by the results obtained with the methods KM, KMD, FCM, AHT, and SPC whose results are quite similar in the last three benchmarks. Small values of Std, again show good repeatability in the experiments performed at each iteration.

Finally, in the previous section, we propose one way to determine the best algorithms with only one metric, called P_C and $P_C SILA$. These values are shown in Table 5 and 6, respectively. The best (lowest value) has been marked in bold text for each benchmark.

TABLE 6. PCSILA metric for different clustering algorithms.

Dataset	Statistics	LAMDA	LAMDA-RD	KM	KMD	FCM	AHT	DBS	SPC	HDB	CON
Dim1024	Average	**61.73**	**61.73**	**61.73**	**61.73**	448.6	**61.73**	**61.73**	**61.73**	**61.73**	**61.73**
	Std	0.000	0.000	0.000	0.000	2.451	0.000	0.000	0.000	0.000	0.000
Segment	Average	146.1	9.224	9.403	8.803	**8.474**	23.88	14.31	20.30	11.60	10.14
	Std	14.74	0.168	0.795	0.584	0.572	0.000	0.000	0.521	0.024	1.574
Hepta	Average	3.321	**2.021**	3.276	**2.021**	**2.021**	**2.021**	**2.012**	2.231	**2.021**	2.105
	Std	0.547	0.144	0.749	0.000	0.000	0.000	0.000	0.000	0.054	0.375
R15	Average	130.1	1.628	2.258	**1.580**	1.598	1.604	1.762	1.603	1.656	1.639
	Std	30.51	0.078	0.401	0.000	0.000	0.000	0.001	0.000	0.002	0.155
Aggregation	Average	12.65	4.819	3.472	**3.379**	3.571	3.665	4.231	4.981	4.121	3.641
	Std	2.634	2.310	0.200	0.044	0.259	0.000	0.002	1.524	0.004	0.310
Unbalance	Average	-27.12	**1.114**	1.170	1.201	1.450	3.691	1.211	1.201	1.184	1.107
	Std	2.961	0.057	0.091	0.159	0.192	0.000	0.000	0.001	0.000	0.079
s1	Average	-101.6	**2.153**	2.554	2.309	2.155	2.097	2.256	**2.111**	2.220	2.214
	Std	11.12	0.381	0.464	0.616	0.356	0.002	0.012	0.005	0.000	0.008
s2	Average	-244.7	3.414	3.311	3.297	**2.987**	3.240	4.572	3.48	3.894	2.940
	Std	35.10	0.701	0.583	0.384	0.421	0.000	0.001	0.528	0.000	0.200
s3	Average	-298.2	12.84	4.968	4.805	**4.512**	8.710	-23.16	14.65	44.69	5.733
	Std	10.94	4.128	0.298	0.3023	0.209	0.000	0.000	0.377	0.000	0.105
a1	Average	--	4.611	3.912	3.852	3.896	4.280	4.68	4.021	4.201	**3.760**
	Std	--	0.541	0.275	0.446	0.370	0.000	0.000	0.452	0.000	0.242

The results presented in Tables 5 and 6 show that LAMDA-RD is the best algorithm for the following datasets: Dim 1024, Hepta, Unbalance and s1, which implies a good quality of the clusters formed, based on P_C and $P_C SILA$. In Segment, and R15, our approach has values very close to the best algorithm (KMD). The performance for s2, s3 and a1 is reduced in LAMDA-RD, DBSCAN and HDB, which is reasonable because they are based on densities, in which, if there are scattered individuals, then the algorithms cannot make a good assignment in the clusters. Also, we can see that our proposal works quite well when the groups have not overlapping between them.

In the case of the benchmark s1 (9% of overlapping), it is the best algorithm, concluding that the performance of the algorithm is not affected by individuals slightly overlapped between clusters. Also, based on the metrics, a very good behavior of the algorithm can be observed in unbalanced datasets (unbalance). When the overlapping percentage increases, e.g. in s2 (20% overlap), our algorithm still makes a good clustering; however, in the case of a1 and s3 (22% and 40% of overlapping, respectively), based on our experiments, we can conclude that density-based methods have problems of assigning of individuals located in the overlap zone. Methods like KMD, FCM, AHT and SPC have the advantage of knowing the number of clusters a priori, which makes it easier to assign those samples to the nearest cluster, e.g. in s3 ($P_C \approx 4.48$), while LAMDA-RD decreases its performance ($P_C \approx 12.3$), a value that shows that when there is an overlap greater than 20% between clusters, our proposal builds clusters with poor quality, incorrectly assigning individuals to the most similar clusters. Particularly, our proposal is better than DBS and HDB, the methods with which a fairer comparison can be made without setting the desired number of partitions.

Based on the P_C and $P_C SILA$, the results are quite consistent with SC, $SILA$ and WB_{index}. Our proposal works quite well if the overlapping between clusters is less than 20%. If it increases, then the iterative methods are better, which is logical due to their individual assignment methodology that allows minimizing distance functions at the intra-cluster, and maximizing inter-cluster distances until reaching optimal values; however, these iterative methods increase their computation time depending on the dimensions of the data set to perform the optimization.

$P_C SILA$ shows very similar values to the PC. Because the SILA values decrease with respect to SC, the $P_C SILA$ index increases minimally. The only benchmark where a considerable difference is seen is in Dim1024 ($P_C = 0.142$ and $P_C SILA = 61.73$). The index increases considerably since SILA is close to zero, and finally, confirms the information we had regarding the separability of that benchmark. Once again, the results of PC and $P_C SILA$ are consistent in all the evaluated benchmarks.

Finally, the quality of the clusters related to the real classes of each benchmark is computed with RI. The results are computed with the best partitions obtained with each algorithm. These values are shown in Table 7. The best (highest values) has been marked in bold text. The results show that clusters constructed by LAMDA-RD have a high value of coinciding with the real classes, taking into consideration that RI is an extrinsic clustering validation measure that compares the output of the clustering method and the real results (classes). LAMDA-RD is better than LAMDA in all benchmarks, and in some datasets like Dim 1024, Hepta, R15, Unbalance and s1, the results are as good as the best algorithms, and in some cases better than them (see R15, and s1). In the rest of datasets, our method presents a good behavior, except for Segment, in which a high number of descriptors is affecting

TABLE 7. RI metric for different clustering algorithms.

Dataset	LAMDA	LAMDA-RD	KM	KMD	FCM	AHT	DBS	SPC	HDB	CON
Dim 1024	**1**	**1**	**1**	**1**	0,575	**1**	**1**	**1**		
Segment	0.534	0.425	0.621	0.741	**0.842**	0.338	0.009	0.474	0.212	0.719
Hepta	0.835	**0.990**	0,935	0,9147	0,893	**0.990**	**0.990**	0.989	0.773	0.981
R15	0.127	**0,989**	0,546	0,985	0,946	0,982	0,416	0.975	0.524	0.964
Aggregation	0.118	0.810	0.732	0.698	0,567	**0.991**	0.909	0.784	0.921	0.768
Unbalance	0.082	**0.999**	0.876	0.876	0.832	0.612	**0.999**	0.979	**0.999**	0.964
s1	0.102	**0.988**	0.834	0.801	0.966	0.981	0.855	0.985	0.855	0.778
s2	0.089	0.886	0.833	0.955	0.961	0.866	0.732	**0.978**	0.786	6.525
s3	0.077	0.524	0.802	**0.836**	0.829	0.698	0.385	0.745	0.452	0.696
a1	0.071	0.834	0.856	0.931	0.895	**0.947**	0.781	0.921	0.831	0.825

the performance of the density-based methods (see the values of LAMDA-RD, DBS and HDB), so, an evaluation of the relevant descriptors should be made, discarding those that do not adequately characterize each group. Due to the distribution and different densities of the clusters of Unbalance (see the distribution of data in [47]), LAMDA-RD, DBS and HDB algorithms are the best since they can clearly distinguish each group due to the separation that exists among them, without the existence of overlap.

B. PERFORMANCE COMPARISON OF LAMDA-RD AND OTHER ONLINE CLUSTERING ALGORITHMS

To analyze and determine how our method improves the behavior of the original algorithm and other online clustering algorithms that work with data stream, the following tests are performed. At this point, we present the time consuming (computational cost) of each proposal in a streaming data scenario. In this context, a successful algorithm must consider the following restrictions [56]:

- Individuals continually arrive;
- There is no control in the order in which the individuals are generated;
- The size of a stream is (potentially) unbounded;
- Data objects are discarded after they have been processed.

All these restrictions are considered in the test, for LAMDA family (LAMDA-RD, LAMDA-TP, original LAMDA), and another online clustering method called ADDclustering, for live data stream [46]. A maximum exigency parameter is set ($\alpha = 1$), because it is desired a strict behavior for the algorithms in the assignment process. The control parameters of LAMDA-RD (d_{nb} and D_t) have been heuristically set to obtain a number of clusters closer to the real classes in each dataset. The methodology used for this experiment is presented in Figure 6.

The experiment is repeated 20 times ($\#reps = 20$), each time performance metrics are computed. Finally, from the obtained results, are computed the "Average" and standard deviation "Std", in order to observe the repeatability in the creation of clusters of each online algorithm.

The results of these statistic metrics are shown in Table 8, and the algorithm with the best average metric is marked in bold text. According to P_C and $P_C SILA$ (the two metrics are

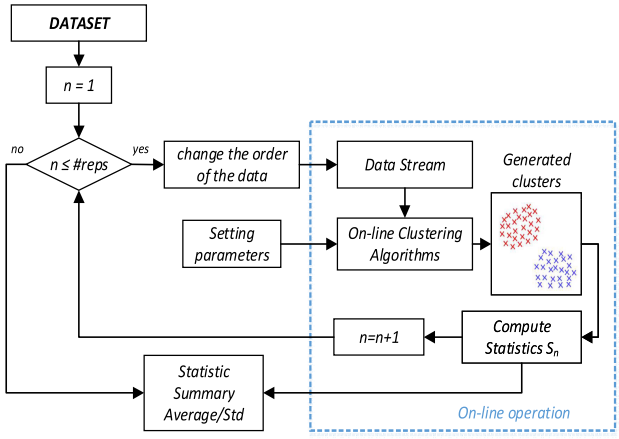

FIGURE 6. Methodology used for the comparison of the different online clustering algorithms.

similar, since *SC* and *SILA* are quite similar), our proposal is the best in all the cases, even with high-dimensional datasets (see Postures), which shows us a good scalability of our method at the cost of increasing the computational time, which is common in data stream scenarios. It can be observed that this metric increases directly proportional when the percentage of overlap between clusters increases (see the results of s1, s2, s3 and a1), which is expected because the process of clustering is more complex since different individuals can belong to two or more neighboring clusters.

LAMDA is good in Aggregation, especially in the *SSB* and *SSW*; however, it can be noticed that it creates a high number of clusters, whose quality is not good when analyzing the results of *SC*, WB_{index} or P_C, being LAMDA-RD the best.

Concerning the computational cost, it can be noticed that our approach has the highest value, this is due to the additional operations that are executed for the merging stage. This time depends especially on the number of individuals and the number of dimensions, e.g. Segment (65.23s, 2310 individuals and 19 features), and Postures (600.8s, 74975 individuals and 15 features).

Additionally, it is observed that LAMDA-RD, with respect to ADDClustering, presents better results in all the benchmarks, which makes it a good alternative in online

TABLE 8. Performance metrics among online clustering algorithms.

Dataset	Method	Statistics	SC	SILA	SSW	SSB	#Clust.	WB_{index}	Time (s)	PC	PCSILA
Dim 1024	LAMDA-RD	Average	**1.000**	**0.002**	**0.089**	**9.994**	**16**	**0.142**	21.12	**0.142**	**61.73**
		Std	0.000	0.000	0.000	0.000	0	0.000	0.902	0.000	0.000
	LAMDA-TP	Average	**1.000**	**0.002**	**0.089**	**9.994**	**16**	**0.142**	**4.085**	**0.142**	**61.73**
		Std	0.000	0.000	0.000	0.000	0	0.000	0.110	0.000	0.000
	LAMDA	Average	**1.000**	**0.002**	**0.089**	**9.994**	**16**	**0.142**	21.12	**0.142**	**61.73**
		Std	0.000	0.000	0.000	0.000	0	0.000	0.902	0.000	0.000
	ADDClustering	Average	0.785	0.751	0.084	9.994	41	0.343	71.46	0.436	0.456
		Std	0.000	0.000	0.000	0.000	0	0.000	0.000	0.000	0.000
Segment	LAMDA-RD	Average	**0.569**	0.489	0.426	0.654	**7**	**4.559**	65.23	**8.012**	**9.224**
		Std	0.045	0.039	0.025	0.014	1	0.227	5.185	0.156	0.168
	LAMDA-TP	Average	0.190	0.043	0.797	0.019	3	125.8	2.496	662.1	2253
		Std	0.025	0.064	0.064	0.002	1	6.237	0.330	37.48	90.00
	LAMDA	Average	0.253	**0.225**	**0.183**	**0.774**	136	32.15	13.70	127.0	146.1
		Std	0.003	0.004	0.007	0.004	4	1.290	0.424	15.56	14.74
	ADDClustering	Average	0,550	0.521	0,609	0,445	3	4,571	**1,780**	8.309	8.758
		Std	0,032		0,003	0,018	0	0,683	0,332	1,549	1.625
Hepta	LAMDA-RD	Average	**0.865**	0.867	0.082	**0.330**	**7**	**1.808**	0.213	**1.989**	**2.021**
		Std	0.043	0.001	0.002	0.000	1	0.124	0.055	0.286	0.144
	LAMDA-TP	Average	0.704	0.703	0.089	0.327	9	2.469	**0.141**	3.608	3.880
		Std	0.075	0.066	0.012	0.007	1	0.368	0.039	1.087	1.214
	LAMDA	Average	0.848	**0.819**	**0.005**	0.329	192	2.720	0.454	3.229	3.321
		Std	0.030	0.026	0.001	0.001	2	0.491	0.089	0.689	0.547
	ADDClustering	Average	0.882	0.854	0.082	0.330	7	1.754	1.112	1.988	2.057
		Std	0.000	0.000	0.000	0.000	0	0.000	0.241	0.000	0.000
R15	LAMDA-RD	Average	**0.899**	**0.889**	**0.027**	**0.285**	**15**	**1.432**	0.590	**1.592**	**1.628**
		Std	0.128	0.016	0.006	0.003	1	0.439	0.073	0.168	0.078
	LAMDA-TP	Average	0.373	0.351	0.105	0.268	**15**	6.046	0.381	17.42	20.19
		Std	0.079	0.093	0.014	0.004	1	0.657	**0.052**	6.230	9.27
	LAMDA	Average	0.079	0.074	0.062	0.284	105	22.75	0.638	110.6	130.1
		Std	0.062	0.032	0.007	0.001	2	2.758	0.066	32.00	30.51
	ADDClustering	Average	0.643	0.625	0.093	0.215	15	6.258	14.07	9.931	10.25
		Std	0.057	0.055	0.013	0.003	3	1.706	4.163	3.249	3.102
Aggregation	LAMDA-RD	Average	**0.565**	0.551	0.122	0.386	**7**	**2.288**	1.367	**4.017**	**4.819**
		Std	0.015	0.051	0.010	0.005	1	0.306	0.135	2.444	2.310
	LAMDA-TP	Average	0.323	0.314	0.125	0.383	10	3.230	**0.469**	12.64	13.14
		Std	0.115	0.101	0.015	0.006	1	0.419	0.056	8.307	6.57
	LAMDA	Average	0.339	**0.310**	**0.080**	**0.397**	19	3.857	0.535	11.75	12.65
		Std	0.048	0.032	0.004	0.001	1	0.309	0.060	2.906	2.634
	ADDClustering	Average	0.562	0.544	0.140	0.335	6	2.330	3.860	4.125	4.297
		Std	0.035	0.041	0.016	0.006	1	0.221	1.606	0.480	0.504
Unbalance	LAMDA-RD	Average	**0.941**	**0.922**	**0.020**	**0.155**	**8**	**1.033**	33.25	**1.095**	**1.114**
		Std	0.034	0.031	0.007	0.003	1	0.069	1.195	0.064	0.057
	LAMDA-TP	Average	0.934	0.914	0.103	0.105	4	3.627	**5.219**	3.900	3.980
		Std	0.017	0.016	0.001	0.000	1	0.791	1.483	0.928	0.944
	LAMDA	Average	-0.164	-0.202	0.054	0.144	31	11.54	5.660	-34.37	-27.12
		Std	0.077	0.054	0.000	0.000	2	0.832	0.479	326.4	2.961
	ADDClustering	Average	0.742	0.721	0.067	0.129	10	4.244	59.89	5.702	5.987
		Std	0.197	0.188	0.036	0.023	6	2.460	10.44	2.793	2.575
s1	LAMDA-RD	Average	**0.854**	**0.829**	**0.041**	**0.341**	**15**	**1.772**	26.82	**2.044**	**2.153**
		Std	0.050	0.053	0.004	0.002	1	0.165	4.231	0.336	0.381
	LAMDA-TP	Average	0.468	0.450	0.092	0.332	**15**	4.166	**3.838**	9.413	9.878
		Std	0.084	0.085	0.010	0.004	1	0.435	0.472	3.087	3.368
	LAMDA	Average	-0.281	-0.299	0.067	0.340	155	30.50	7.026	-109.5	-101.6
		Std	0.028	0.021	0.003	0.000	5	1.590	0.827	10.49	11.12
	ADDClustering	Average	0.512	0.498	0.177	0.305	4	2.323	8.590	4.532	6.708
		Std	0.066	0.062	0.080	0.005	0	0.022	0.003	0.014	0.018
s2	LAMDA-RD	Average	**0.719**	**0.714**	**0.051**	**0.316**	**15**	**2.438**	31.84	**3.395**	**3.414**
		Std	0.063	0.071	0.005	0.001	1	0.234	6.151	0.728	0.701
	LAMDA-TP	Average	0.390	0.377	0.084	0.310	22	5.927	**3.917**	16.59	17.16
		Std	0.094	0.069	0.010	0.002	2	0.691	0.332	6.647	4.428
	LAMDA	Average	-0.201	-0.235	0.058	0.319	315	57.52	9.722	-290.4	-244.7
		Std	0.027	0.019	0.001	0.000	4	1.452	1.130	37.32	35.10
	ADDClustering	Average	--	--	0.320	0.000	1	Inf	1.218	--	--
		Std	--	--	0.000	0.000	0	--	--	--	--
s3	LAMDA-RD	Average	**0.458**	**0.439**	**0.053**	**0.284**	30	**5.583**	30.25	**12.30**	**12.84**
		Std	0.037	0.026	0.003	0.001	1	0.261	1.481	1.525	4.128
	LAMDA-TP	Average	0.346	0.311	0.075	0.281	**24**	6.224	**3.882**	18.46	20.72
		Std	0.047	0.028	0.005	0.002	2	0.483	0.085	3.993	3.98
	LAMDA	Average	-0.278	-0.308	0.063	0.285	412	91.56	11.16	-329.5	-298.2
		Std	0.015	0.024	0.001	0.000	5	2.478	0.649	14.74	10.94
	ADDClustering	Average	--	--	0.288	0.000	1	Inf	1.246	--	--
		Std	--	--	0.000	0.000	0	--	--	--	--

TABLE 8. *(Continued.)* **Performance metrics among online clustering algorithms.**

a1	LAMDA-RD	Average	**0.653**	**0.649**	**0.047**	**0.324**	**20**	**2.895**	5.981	**4.473**	**4.611**
		Std	0.050	0.049	0.002	0.000	1	0.144	0.780	0.589	0.541
	LAMDA-TP	Average	0.324	0.315	0.101	0.314	17	5.594	**1.548**	17.81	18.52
		Std	0.054	0.053	0.007	0.002	1	0.444	0.077	3.739	3.71
	LAMDA	Average	-0.120	--	0.063	**0.324**	283	55.20	3.943	--	--
		Std	0.033	--	0.002	0.000	4	2.082	0.630	--	--
	ADDClustering	Average	0.529	0.504	0.107	0.311	11	3.788	26.98	7.161	7.528
		Std	0.000	0.000	0.000	0.000	0	0.000	0.000	0.000	0.000
Postures	LAMDA-RD	Average	0,032	0.026	0,493	0,185	**6**	**16,00**	600,8	**497,3**	**628.7**
		Std	0.002	0.002	0.065	0.012	1	1.898	5.238	6.874	7.852
	LAMDA-TP	Average	0.045	0.036	0,403	0,349	25	28,87	**110,8**	644,4	657.7
		Std	0.001	0.001	0.087	0.047	0	2.687	3.650	5.238	5.144
	LAMDA	Average	-0,132	**-0.142**	**0,391**	**0,356**	106	116,5	173,0	-881,4	-815.4
		Std	0.006	0.005	0.058	0.089	2	2.218	4.844	-6.854	-8.711
	ADDClustering	Average	**0,677**	0.614	0,536	0,000	2	6547	766.8	9672	10612
		Std	0.025	0.021	0.025	0.000	0	10.25	10.98	35.24	42.87

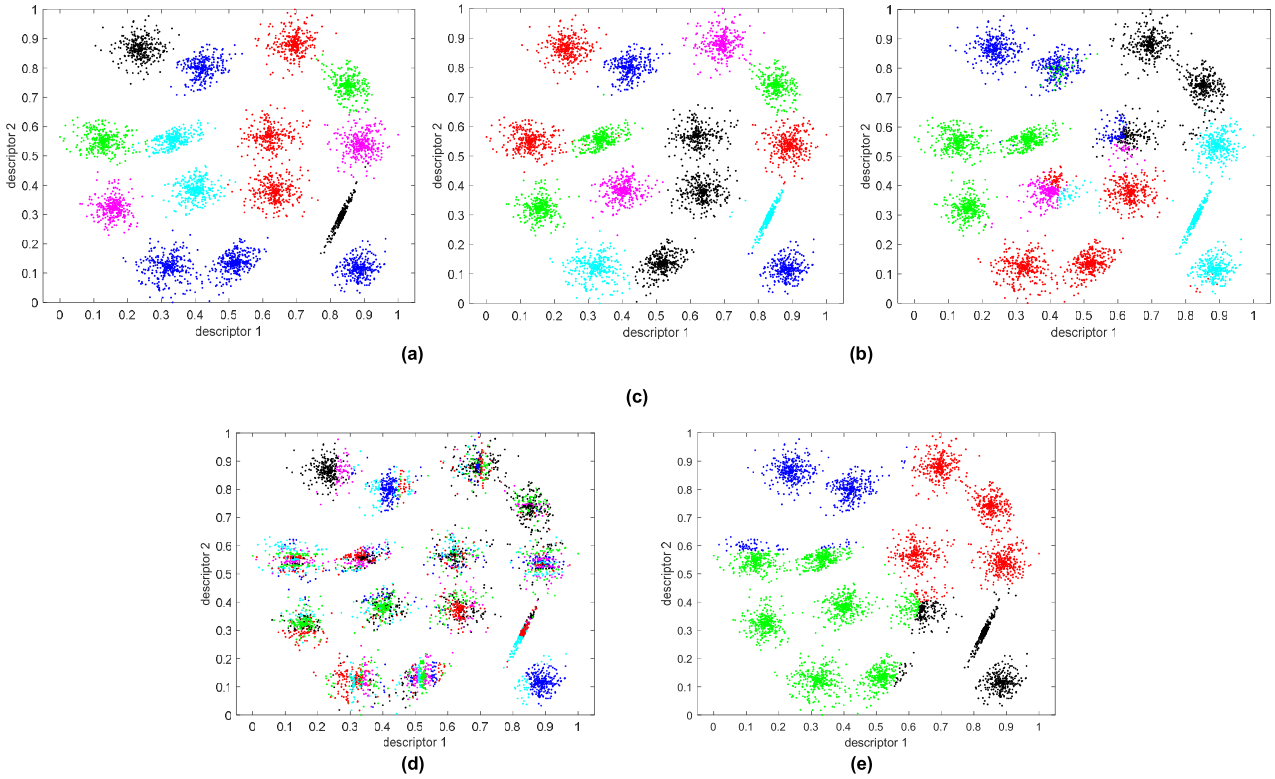

FIGURE 7. Tests performed with s1 dataset; (a) Original partition, clusters generated by (b) LAMDA-RD (15 clusters), (c) LAMDA-TP (15 clusters), (d) LAMDA (155 clusters) and (e) ADDclustering (4 clusters), see the detailed statistics in Table 6.

clustering because both algorithms work with the density criterion to form the clusters. Evaluating P_C, our method is always the best, in Segment (LAMDA-RD: 65.23s and ADDClustering: 1.780s) ADDclustering is faster, which shows that this algorithm works better with data of several descriptors, decreasing its performance when the number of individuals increases, e.g. Postures (LAMDA-RD: 600.8s and ADDClustering: 766.8s).

Comparing PC and $P_C SILA$, it can be seen that the latter presents a higher value because $SILA$ penalizes the large differences of distances between clusters in a dataset, which is clearly observed in the Dim1024 benchmark. Especially, in this benchmark can be observed how the $SILA$ index, which uses a measure of cluster compactness, has a less value with respect to SC, and so, it makes the value of $P_C SILA$ bigger with respect to PC due to the large differences between clusters

An illustration of the obtained clusters with the different algorithms in s1, is presented in Figure 7. The parameters of our approach (d_{nb} and D_t) have been calibrated to obtain the desired number of clusters. On the other hand, LAMDA-TP creates 15 clusters of poor quality because it incorrectly

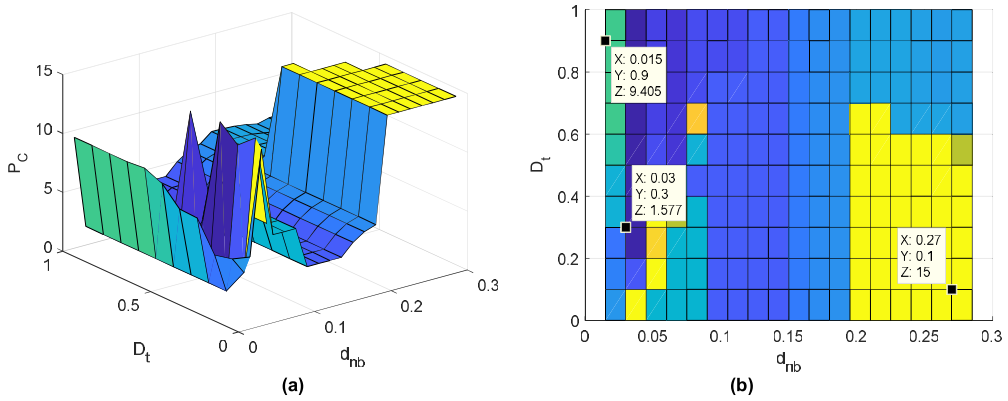

FIGURE 8. Obtained results for $R15$, in function of d_{nb} and D_t: (a) P_C. (b) top view of P_C.

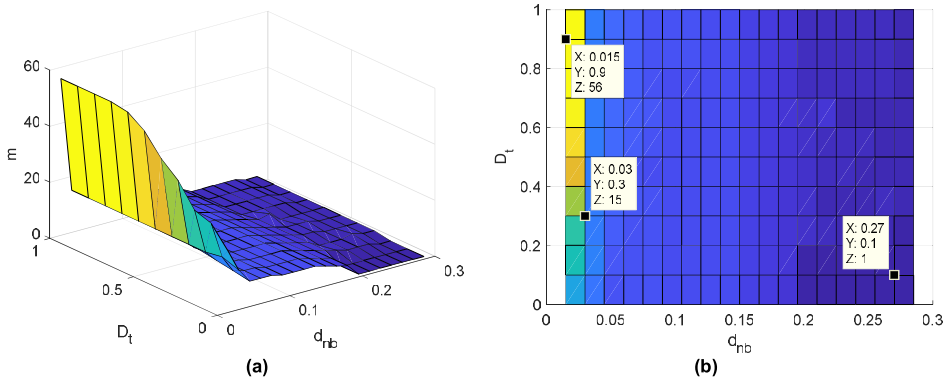

FIGURE 9. Obtained results for $R15$, in function of d_{nb} and D_t: (a) number of clusters "m". (b) top view of the number of clusters "m".

assigns individuals in different clusters (bad quality clusters); whereas, in the case of LAMDA, it creates a very large amount of clusters, in which a manual merge process should be applied. Finally, ADDClustering builds 4 clusters, and according to the results of Table 8 (for s1), it can be noted that the quality of the clusters is not as good as that obtained by LAMDA-RD, where all quality metrics are the best, e.g. $P_C = 2.044$ and $P_C SILA = 2.153$. The method that follows is ADDClustering, with $P_C = 4.532$ (almost double), this is, the groups formed have better inter-cluster (the individuals in the same group are very similar to each other) and intra-cluster characteristics (dissimilar individuals are in different groups).

C. LAMDA-RD PARAMETER CALIBRATION

LAMDA-RD has two calibration parameters that affect the quality and number of the formed clusters: d_{nb} (Definition 4) and D_t (Statement 6).

A guideline for the calibration is presented below, which shows how the variation of the parameters d_{nb} and D_t (necessary to be set by the user) affects the quality of the clusters for the case of R15. Figures 8-a shows the variations of P_C, depending on the parameters d_{nb} and D_t, and Figure 8-b shows its top view, in which the different areas are represented in colors. The yellow zone, e.g. ($D_t = 0.1$, $d_{nb} = 0.27$, $P_C = 15$) presents high P_C values, which as detailed in the experimental tests, this implies poor quality in the created clusters. Based on this, it is necessary to look for the zone with the minimum P_C, in this case, the dark blue zones (which shows the next values: $D_t = 0.3$, $d_{nb} = 0.03$, $P_C = 1.577$), that is, good quality clusters (the lowest P_C). However, the number of created clusters m must also be considered, which is represented in Figure 9-a as a function of the parameters d_{nb} and D_t, and Figure 9-b shows its top view. In this case, the yellow areas represent a high number of created clusters ($D_t = 0.9$, $d_{nb} = 0.015$, $m = 56$ clusters), while dark blue areas ($D_t = 0.1$, $d_{nb} = 0.27$, $m = 1$ cluster) are not useful because all data has been grouped into a single cluster. Finally, we observe the green zone ($D_t = 0.3$, $d_{nb} = 0.03$, $m = 15$ clusters), which coincides with the values of d_{nb} and D_t with the minimum P_C (see Fig. 8). So, the general idea of the method is to find a balance between P_C and m.

Based on the results of Figure 8 and 9, the following criteria can be established:

- Low values of D_t make the merging process between neighboring clusters with low or no density in the overlapped area (see Figure 10-a), which is not adequate since they

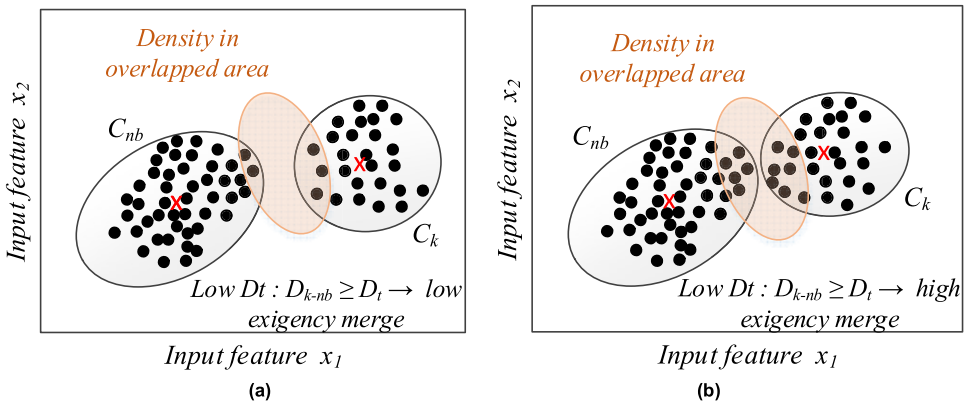

FIGURE 10. Illustrative example of: (a) low value of D_t. (b) high value of D_t.

produce a non-demanding or low exigency algorithm (as is shown in Figure 9 for dark blue zones), performing the merge process with separate or dissimilar neighboring clusters, which leads to poor quality clusters, as is shown by high P_C in Figure 8 for the equivalent zone (yellow area).

- High values of D_t produce a more demanding algorithm (as is shown clearly in Figure 9 for yellow zones) since it requires a higher percentage of individuals in the overlapping area (see in Figure 10-b), performing the merge process only when the neighboring clusters are very close, which improves the quality of them, as is shown by low P_C in Figure 8 for the equivalent zone (dark blue area).
- Low values of distance between neighbors d_{nb} allow obtaining a more demanding algorithm (see Figures 8 and 9, the best P_C and a non-excessive number of clusters m is presented with a low d_{nb}), since the calculation of $K_{k,j}$ is stricter (strongly penalizing the dissimilarity between samples). High values of d_{nb} produce a non-demanding or low exigency algorithm, by weakening the penalization for the dissimilarity between samples.

Figure 11 shows the recommended zone for the initial parameter calibration, looking for a balance zone in Figure 8 and 9 to obtain a good P_C, without creating an excessive number of clusters m. Based on this, it is possible to verify the quadrants of maximum and minimum exigency, and the balanced zone, which can be taken as a starting point to perform the search of the most appropriate D_t and d_{nb}.

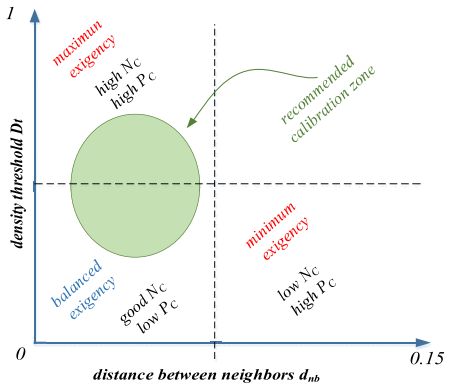

FIGURE 11. Recommended calibration of D_t and d_{nb}.

The critical cases occur in the yellow zones of Figure 9, low D_t and high d_{nb}, which generate the minimum number of clusters. In the case of R15, the best results are obtained by calibrating d_{nb} to a small value, as is shown in Figure 9, where we have a great variation of D_t. Now, contrasting the results with Figure 8, the smaller P_C must be located on the graph.

From the experimentation, a generic behavior could be observed, concluding that the parameter calibration can start with a value of $D_t \approx 0.5$, and $d_{nb} \approx 0.1 \times D_t$, e.g. for R15 the best values are: $D_t = 0.3$, $d_{nb} = 0.03$ (shaded area of Figure 11).

For example, for s1 dataset the formed clusters with different parameter values are shown below, in which the parameter D_t is initially set $D_t = 0.54$, and d_{nb} is changed until finding the minimum P_C (Figure 12 shows all the values of this metric when the parameters are changed), looking for an adequate number and quality of clusters (in this case, 15 clusters). By increasing the value of d_{nb}, the algorithm creates fewer clusters, which are better constituted by covering the more dispersed individuals. On the other hand, by setting the value of d_{nb}, fixed, and changing the values of D_t, we observe that the algorithm is less strict when it is small, which implies a decrease in the number of clusters (less strict), as is shown in Figure 13.

The behavior of P_C for the case when D_t is fixed and d_{nb} is changed, is shown in Figure 14-a, while the behavior of P_C for the case when d_{nb} is fixed and D_t is changed, is shown in Figure 14-b; in both cases, the minimum P_C is a guide to calibrate these parameters.

D. LAMDA-RD AND OUTLIERS

As detailed in Section III, LAMDA-RD initially has a descriptor normalization stage, which is essential to know how the algorithm works with the samples, allowing us

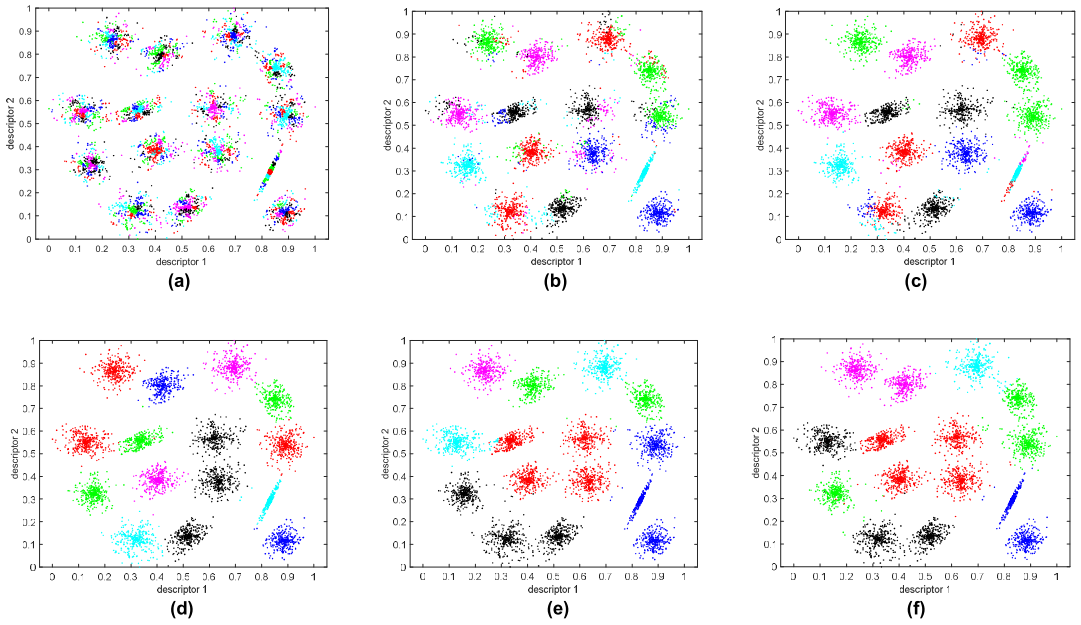

FIGURE 12. Clusters formed for $D_t = 0.54$ and different values of d_{nb} (a) $d_{nb} = 0.005$, (b) $d_{nb} = 0.02$, (c) $d_{nb} = 0.04$, (d) $d_{nb} = 0.06$, (e) $d_{nb} = 0.08$, and (f) $d_{nb} = 0.1$.

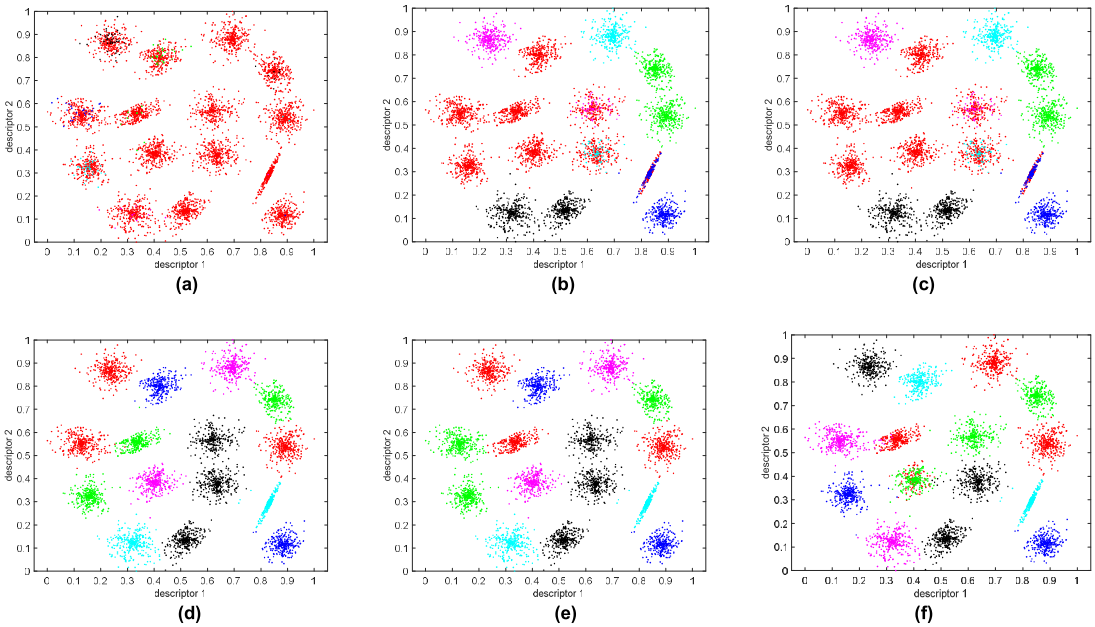

FIGURE 13. Clusters formed for $d_{nb} = 0.06$ and different values of D_t (a) $D_t = 0$, (b) $D_t = 0.25$, (c) $D_t = 0.35$, (d) $D_t = 0.54$, (e) $D_t = 0.75$, and (f) $D_t = 1$.

to analyze the behavior, performance and sensitivity of LAMDA-RD in the presence of outliers. Normalization is essential in LAMDA-RD since the algorithm does not eliminate outliers, which is not recommended because valuable information can be obtained from this data, especially in the context of a stream mining scenario [46]. For the normalization stage, the maximum and minimum values of each descriptor/feature are required, as shown in Eq. (2). The normalization process is detailed in the pseudo-algorithm presented with the label "Algorithm 4".

If there are descriptors with outliers, then the normalization process limits them to values of 0 or 1, in order to calculate the marginal and global adequacy. To observe the behavior of LAMDA-RD with outliers, we use for simplicity the R15 benchmark (600 samples and 2 descriptors). The Maximum and Minimum Limits (MML) of the original dataset are $X_{min} = [3.402; 3.178]$ and $X_{max} = [17.124; 17.012]$.

Figure 15 shows the different partitions obtained by the algorithm when the MML of the dataset are modified, so that data outside of these limits are considered outliers.

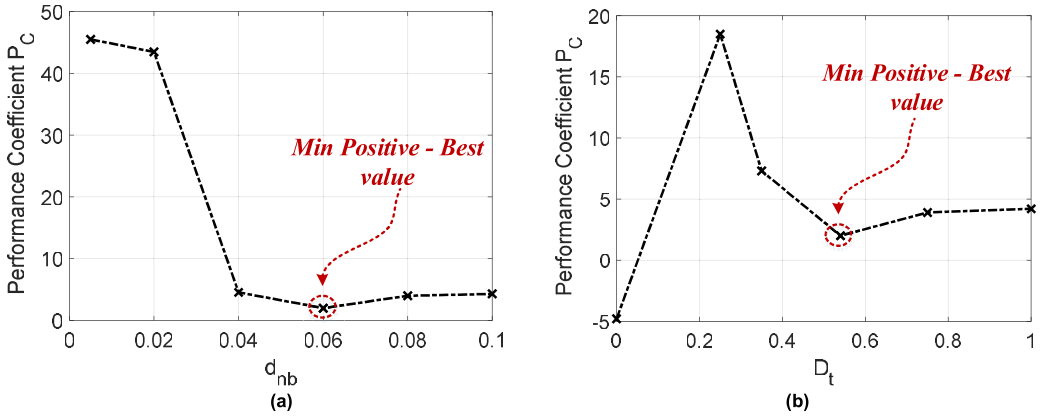

FIGURE 14. Performance Coefficient for different values of d_{nb} and D_t (marked in red the best value).

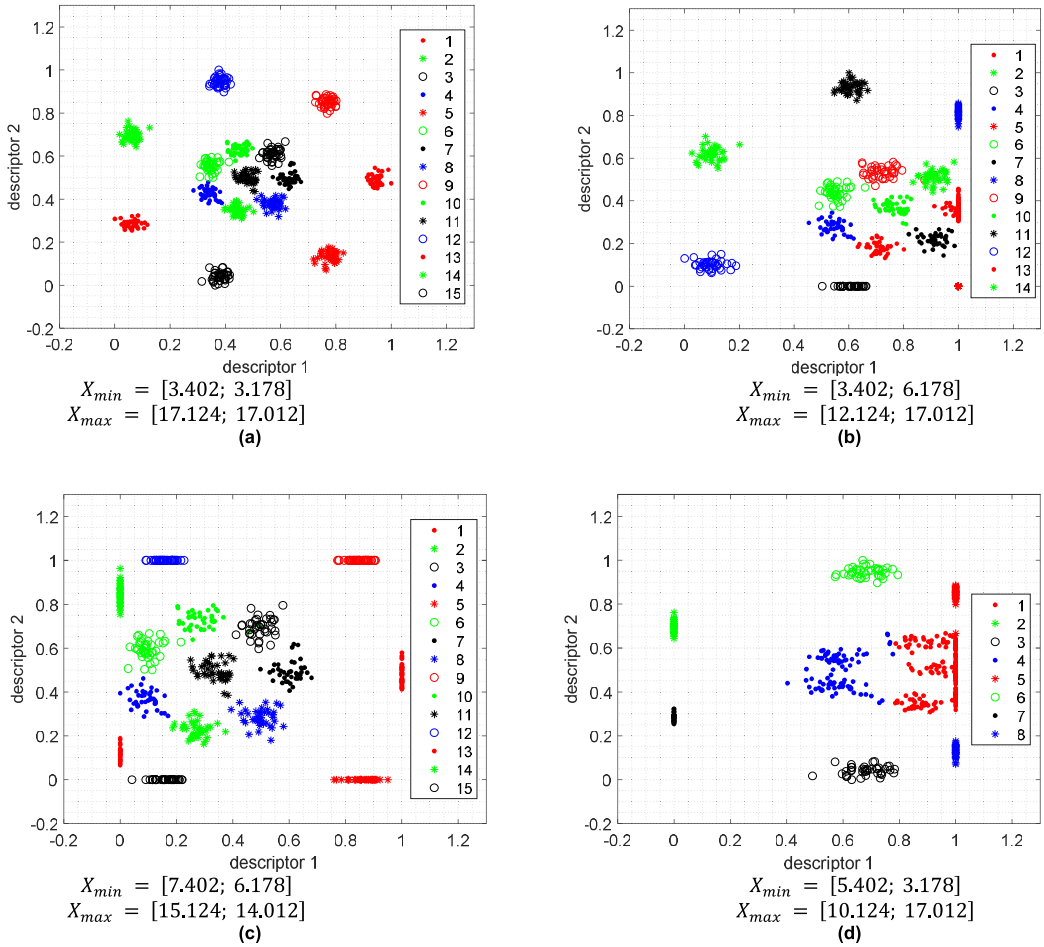

FIGURE 15. Clusters created by LAMDA-RD for benchmark R15 with different MML: (a) Case 1: 0 outliers, Case 2: 180 outliers, (c) 280 outliers and (d) 343 outliers.

Table 9 presents the evaluation metrics of the formed clusters, in order to analyze the performance and sensitivity.

Table 9 shows the performance of LAMDA-RD when the MML of the dataset are modified, in order to artificially create outliers with its own data. When we consider the original MML of the dataset (0 outliers), it is observed that the algorithm correctly assigns the samples in each group (see Figure. 15-a). In case 2, two MML have been reduced, observing that the algorithm groups the outliers on the edge of the normalization values, merging the clusters 7 and 13 of

TABLE 9. Performance metrics for different number of outliers in R15.

	# Outliers	SC	SILA	SSW	SSB	#Clust.	WB	P_C	P_C SILA
Case 1	0	0.893	0.891	0.027	0.286	15	1.441	1.612	1.615
Case 2	180	0.886	0.883	0.033	0.343	14	1.367	1.543	1.547
Case 3	281	0.875	0.872	0.041	0.402	15	1.532	1.749	1.757
Case 4	343	0.772	0.755	0.071	0.343	8	1.668	2.158	2.184

Algorithm 4 Normalization Stage of LAMDA-RD

Input:
Sample $X = [x_1; x_2; \ldots; x_j; \ldots; x_n]$
$X_{min} = [x_{1min}; x_{2min}; \ldots; x_{jmin}; \ldots; x_{nmin}]$
$X_{max} = [x_{1max}; x_{2max}; \ldots; x_{jmax}; \ldots; x_{nmax}]$
Procedure:
1. Apply (2) in each descriptor x_j to obtain the normalized descriptor $\bar{x}_j$.
2. **If** $\bar{x}_j > 1$, **then**
- $\bar{x}_j = 1$
else if $\bar{x}_j < 0$
- $\bar{x}_j = 0$
End
Output: Normalized sample $\bar{X}$

Figure 15-a, resulting in the cluster 13 of Figure 15-b. In this case, the metrics change minimally with respect to case 1, since in this experiment there are 14 well-separated clusters. In case 3, the four MML have been reduced, observing that the algorithm is able to group the outliers on the edge of the normalization values without affecting the creation of the internal clusters. Finally, in case 4, the MML of descriptor 1 have been reduced considerably, obtaining 343 outliers, which affects the distribution of the samples. The algorithm groups the outliers on the edge of the normalization values, and merges the groups (7, 8, 10, 11, 13, 14, 15) and (4,6) of Figure 15-a, forming the clusters 1 and 4 of Figure 15-d, respectively, in which 8 partitions are obtained. The performance metrics are quite good and vary from the original partition (case 1) because the samples have a new distribution due to the high percentage of outliers, which are greater than 50% of the entire dataset. Based on the results obtained, we can determine that LAMDA-RD presents a good response working with outliers, and it is not sensitive to them due to the initial normalization. Finally, in real applications, the MML values could be statistically determined by calculating the interquartile range.

E. REAL CASE STUDY: GAS-LIFT METHOD

Gas-lift is a technology to extract oil from wells that have low reservoir pressure, by reducing the hydrostatic pressure in the tubing. Gas is injected into the tubing, as deep as possible, and mixed with the fluid from the reservoir (see Figure 16). The gas reduces the density of the fluid in the tubing, which reduces the bottom pressure (PWF), and thereby, increases the production from the reservoir. The dynamics of highly

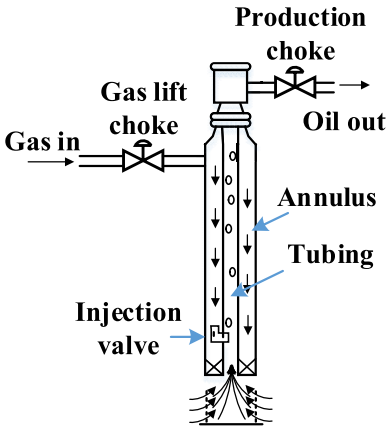

FIGURE 16. The Artificial Gas-Lift.

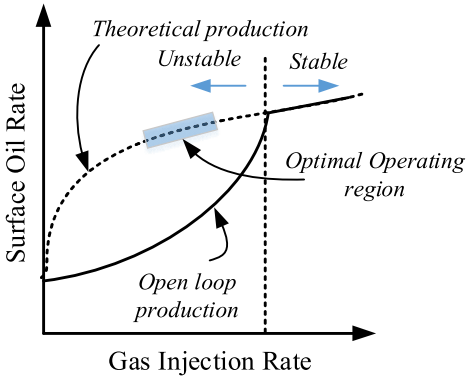

FIGURE 17. Artificial gas-lift well behavior'S model.

oscillatory flow in a gas lifted well can be described as follows: i) Gas from the casing starts to flow into the tubing. As gas inputs in the tubing, then the pressure in the tubing falls. This accelerates the inflow of gas; ii) The gas pushes the major part of the liquid out of the tubing; iii) Liquid in the tubing generates a blocking constraint in the injection orifice. Hence, the tubing gets filled with liquid and the annulus with gas; iv) When the pressure on the injection orifice overcomes the pressure on the tubing side, then a new cycle starts.

The Artificial Gas-Lift (AGL) well behavior's model (see Figure 17) shows that when the gas injection rate increases, then the production also increases until reaching its maximum value; but additional increases in the gas injection will cause a production diminution [57], [58].

TABLE 10. Comparative performance of online clustering algorithms in gas lift wells using all descriptors.

Method	Statistics	SC	SILA	SSW	SSB	#Clust.	WB	Time	P_C	P_C SILA	RI	Accuracy	Fmeasure
LAMDA-RD	Average	**0.745**	**0.716**	**0.133**	**0.444**	**4**	**1.254**	4.354	**1.692**	**1.752**	**0.919**	**0.947**	**0.908**
	Std	0.024	0.028	0.008	0.006	0	0.068	0.041	0.141	0.129	0.046	0.059	0.122
LAMDA-TP	Average	0.733	0.723	0.111	0.461	6	1.413	**1.886**	1.920	1.959	0.742	0.747	0.563
	Std	0.042	0.050	0.006	0.004	1	0.090	0.045	0.208	0.195	0.099	0.096	0.104
LAMDA	Average	0.550	0.539	0.083	0.471	14	2.405	2.188	4.398	4.482	0.485	0.303	0.119
	Std	0.033	0.0277	0.006	0.001	1	0.203	0.043	0.512	0.485	0.097	0.124	0.045
ADDClustering	Average	0.643	0.602	0.188	0.404	8	2.618	8.051	4.334	4.620	0.643	0.275	0.145
	Std	0.107	0.098	0.089	0.037	6	0.912	3.578	1.984	1.872	0.107	0.079	0.097

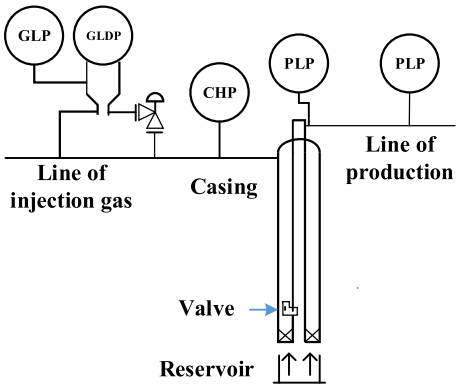

FIGURE 18. Schematic design of a Well based on the Gas-Lift Method.

For the implantation in the field of the AGL method, it is needed an instrumentation and control arrangement [57], [59]. For such task, we need the measurement and control of the following variables (see Figure 18): Pressure of the Injected Gas (GLP), Differential Pressure of the Injected Gas (GLDP), Pressure of the Casing (CHP), Pressure of the Tubing of Production (THP).

The measurement of the injected flow is carried out using the GLP and GLDP variables. The measurement of the pressure casing (CHP) allows knowing the pressure that the gas exercises in the casing (THP), the pressure exercised by the fluids in the pipeline, and the pressure of the line of production (PLP). Other important variables are the Gas flow of lift (FGL. expressed as ''mpcgd''-thousands of gas cubic feet per day), and the Rate of Production (Qprod: expressed as ''BNPD''- barrels net of production per day).

In [60], this case study is used to evaluate the performance of LAMDA in the context of a classification problem (supervised learning), to explain the behavior of these oil wells. In this paper, the goal is to identify the most consistent clusters according to the Rate of Production (Qprod: ''in a gas-lift well), using unsupervised learning, based on the following set of descriptors:

Descriptor 1: Casing Pressure

Descriptor 2: Production Tubing Pressure.

Descriptor 3: Gas Lift Flow.

Descriptor 4: Bottom Pressure.

These variables have been suggested by the experts as the most appropriate for the identification. The historical data has 1187 individuals, corresponding to 4 classes: very low production (VLP), low production (LP), normal production (NP), and high production (HP).

Table 7 reports the results of the metrics after evaluating the performance of each algorithm, in which it can be noted that LAMDA-RD again presents the best results. Based on the proper calibration of its parameters, it is possible to identify the 4 clusters, corresponding to the number of classes of Qprod. On the other hand, ADDClustering presents good results, however, the standard deviation of the number of classes found in each iteration is high, which shows that for this case study, the change in the order of the data that arrives to the algorithm considerably affects the performance of this algorithm.

The results show that LAMDA-RD is better in all the metrics, especially the $P_C = 1.692$, and in the external metrics $RI = 0.919$, $Accuracy = 0.947$ and $Fmeasure = 0.908$. P_C metric defines the quality of the clusters, and the external validation indices show the level of correspondence between the created clusters and the real classes of the dataset. Based on this, in LAMDA-RD the metrics exceed 90%, which indicates an excellent performance of the proposed algorithm, if we compare it with the other methods where the external validation indices are close to 70% in the best case for LAMDA- TP, or with values less than 40%, as in LAMDA and ADDClustering. Another detail to be observed is the machine time of LAMDA-RD, which is around twice that the best (LAMDA-TP), with a difference of 2.468s.

The real distribution of the data (after ordering them in each class) is shown in Figure 19-a, the best clusters created by LAMDA-RD (Figure 19-b) are very similar to the real ones, with a small amount of misassigned individuals grouped in the clusters. In other words, the algorithm is able to identify the correct number of clusters, performing a good assignment of the individuals. The best results of LAMDA-TP (see Figure 19-c) show that clusters 1 and 2 are correctly constructed in relation to the real labels; however,

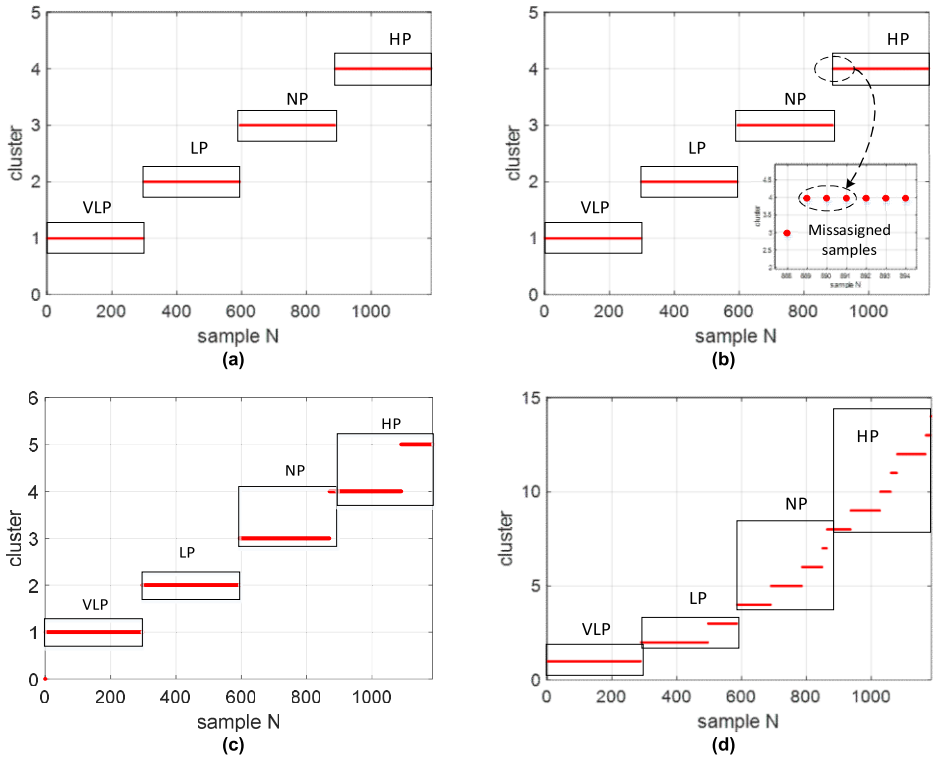

FIGURE 19. Clustering results of the LAMDA approaches: (a) real classes, (b) LAMDA-RD, 4 clusters related to 4 classes were generated, (c) LAMDA-TP, 5 clusters related to 4 classes were generated. (d) LAMDA, 13 clusters related to 4 classes were generated.

TABLE 11. Clustering results of different LAMDA approaches.

Class	Production	Acronym	Number of elements	Number of Clusters		
				LAMDA-RD	LAMDA-TP	LAMDA
1	Very Low Production	VLP	297	1	1	1
2	Low Production	LP	297	1	1	2
3	Normal Production	NP	297	1	1	5
4	High Production	HP	295	1	2	6

in cluster 3 there are partition errors and incorrect assignment of individuals, with individuals assigned to cluster 4, such that is partitioned into two groups incorrectly in relation to the labeled data.

LAMDA for class 1 (see Figure 19-d) generates 1 cluster, in class 2 it generates 2 clusters, in class 3 the algorithm creates 5 clusters, and 6 clusters have been generated in class 4, which is inadequate and impractical, evidencing the need of a merging algorithm. The results detailed above are summarized in Table 11.

Figure 20 shows the ROC (Receiver Operating Characteristic) curve for LAMDA-RD (because only our approach builds the same number of clusters as the real classes), to analyze their sensitivity and specificity in the diagnostic processes. Diagnostic methods with high specificity are required because we are interested in seeing the negative results identified by the algorithms correctly, and also, high sensitivity is required since each state of the system must give a positive result during the diagnostic test,

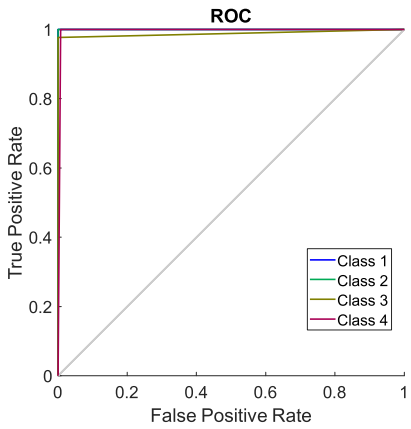

FIGURE 20. ROC of the Rate of Production diagnosed by LAMDA-RD.

according to the class that is represented by each functional state. In ROC curve, the ideal value is close to the point (0,1), which represents a very good diagnostic method.

ROC curves have been drawn for each class in the case study. Table 12 reports average metrics of sensitivity, specificity and Area Under the Curve (AUC), where it can be observed quantitatively that LAMDA-RD performs a very good clustering. Note that curves for the classes 1 and 4 are overlapped.

TABLE 12. Results of the diagnostic metrics of LAMDA-RD for the best partition.

Sensitivity	Specificity	AUC
0.9942	0.9981	0.9961

In general, although LAMDA-TP allows a good clustering results, the algorithm still makes mistakes; however, it considerably reduces the number of created clusters, but individuals not adequately characterized by the descriptors are misassigned. LAMDA creates an excessive number of clusters because, as seen in Figure 19, it is evident that the real number of clusters is much lower than the one identified by the algorithm. Also, the quality of the clusters is not good, which is supported by the metrics of Table 10. It is clear that LAMDA-RD corrects the problems of the other two algorithms, forming good quality clusters if an appropriate parameter calibration is performed.

F. COMPUTATIONAL COMPLEXITY

We proceed to analyze the computational complexity of LAMDA-RD in terms of memory usage, computation time and number of operations on the clustering tasks. Our program is implemented in Matlab R2020a, and it is run on an Intel (R) Core (TM) i7-8750H @ 2.2GHz microprocessor. The analysis is based on the spatial (memory usage and arithmetic complexity) and temporal complexity of the proposed algorithm.

1) MEMORY USAGE

In this subsection, the permanent usage of memory is counted. The number of parameters required to perform the clustering tasks is based on the number of descriptors and formed clusters, n and m, respectively. According to Eqs. (2) to (15), the number of parameters (*#parameters*) to be stored in memory is:

$$\#parameters = nm + 2n + 3 \tag{34}$$

In addition, if there are N samples, each with n descriptors, the total number of stored values is:

$$\#stored\_values = Nn \tag{35}$$

It is assumed that each value is stored in 2 bytes of memory [62]. It can be concluded that its complexity linearly increases.

2) NUMBER OF OPERATIONS

In this subsection is evaluated the number of arithmetic operations (arithmetic complexity) used to solve a problem. Addition, subtraction, multiplication, division, power and root are considered as basic operations. Following the procedure of pseudo-algorithm of LAMDA-RD, the number of operations in each step to assign one sample to a cluster is detailed in Table 13.

TABLE 13. Datasets used to test the clustering algorithms.

	Arithmetic Complexity	
	LAMDA	LAMDA-RD
For normalization (2)	3	3
For $MAD_{k,j}$, (3)	$4mn$	--
For update $\rho_{k,j}$, (4)	4	4
For $CMAD$, (12)	--	$3mn$
For $d_{k,\bar{x}}$, (13)	--	$m(n-1)+2$
For $K_{k,\bar{x}}$, (14)	--	$3m$
For $RMAD_{k,j}$, (15)	--	m
For $GAD_{k,j}$, (7)	$21(m+1)(n-1)$	$21(m+1)(n-1)$
Cluster identif. (9)	1	1
For $t_{nb,j}$, (18)-(20)	--	$n(n_{nb}^2 - n_{nb} + 1)$
Count individuals in overlapping (21)-(23):	--	$n_k + n_{nb} + 1$
For D_{k-nb}, (24) and Statement 6	--	3
In the case of merge, to update $\rho_{new,j}$, (26)	--	$n_k + n_{nb} + 1$

The arithmetic complexity of LAMDA (C_L), compared with LAMDA-RD (C_{RD}), is computed in the case of Table 13:

$$C_L = 25nm - 21m + 21n - 13 \tag{36}$$

$$C_{RD} = 25nm - 18m + n\left(n_{nb}^2 - n_{nb} + 22\right) + 2\left(n_{nb} + n_k\right) - 7 \tag{37}$$

If LAMDA-RD clustering process is considered without the merge stage, then it presents similar arithmetic complexity with respect to the original LAMDA. Both algorithms have linear characteristics in the terms n and m, especially depending on the number of descriptors, and increasing as new clusters are created. On the other hand, when we consider the addition of the merge algorithm that is the case of LAMDA-RD, it is observed a quadratic exponent in the term of the number of elements n_{nb} in the neighbor cluster C_{nb}, which is multiplied by the number of descriptors n. It can be concluded that its complexity increases quadratically as more samples are added to the clusters.

3) TEMPORAL COMPLEXITY

The temporal complexity is used to verify the increase of the operations performed in the evaluation of each benchmark. Based on the number of samples, we can compute the average time required to evaluate each sample, as is shown in Table 14.

The computation time of LAMDA-RD increases in the most of cases respect to LAMDA due to the merge algorithm. Generally, the computational time increases as the number of descriptors increases, i.e., in Segment with 19 descriptors and Postures with 15 descriptors, the time required by LAMDA-RD is 4 times greater than the required by

TABLE 14. Computational time (seconds) of LAMDA and LAMDA-RD.

Benchmark	Descriptors	Clusters	LAMDA	LAMDA-RD
Dim 1024	1024	16	20.6e-3 s	20.6e-3 s
Segment	19	7	5.93e-3 s	28.2e-3 s
Hepta	3	7	2.14e-3 s	1.00e-3 s
R15	2	15	1.06e-3 s	0.98e-3 s
Aggregation	2	7	0.67e-3 s	1.73e-3 s
Unbalance	2	8	0.87e-3 s	5.11e-3 s
S1	2	15	1.40e-3 s	5.36e-3 s
S2	2	15	1.94e-3 s	6.36e-3 s
S3	2	15	2.23e-3 s	6.05e-3 s
A1	2	20	1.31e-3 s	1.99e-3 s
Postures	15	5	2.31e-3 s	8.01e-3 s
Gas lift	4	4	1.84e-3 s	3.66e-3 s

LAMDA. In general, for LAMDA-RD, it is between two and four times larger, but in some cases, it is the same (Dim 1024, R15), or even smaller (Hepta). However, as has been shown in the experimental results, the benefits of our proposal are observed by analyzing the quality of the formed clusters. Thus, the temporal complexity increases, but at the same time, a considerable improvement in the results of LAMDA-RD is evident.

VI. GENERAL ANALYSIS OF THE RESULTS

With the different tests carried out, we summarize the following results:

- The main objective of this paper is to improve LAMDA in clustering tasks, for which we have proposed LAMDA-RD, an algorithm that in all cases considerably improves the performance of the original algorithm (see the results from Table 2 to Table 10).
- Based on the results of P_C, metric that considers SC and WB-index, we can notice that in 4 of the 10 datasets tested (Dim1024, Hepta, Unbalance and s1), LAMDA-RD obtains the best results, in terms of performance, while in Segment (high dimensionality), R15 and Aggregation, is close to the best algorithms. In Postures (benchmark with a large number of samples and high dimensionality) is the best algorithm when compared to other clustering methods focused on data streams, achieving the objective of this paper of obtaining a competitive algorithm that in all cases improves the performance of LAMDA.
- The tests have been performed on balanced and unbalanced datasets with different overlapping. In cases where there is no overlap, the algorithm works as well as KM, KMD and AHT. With an overlap of 9%, as in the case of s1, LAMDA-RD presents the best results, while with an overlap of less than 20% (s2 and a1), the algorithm has an intermediate performance. Also, it is noted that the performance decays in s3, which has a 40% overlap (strong non-Gaussian distribution of feature values), where it is complicated to make an online assignment of elements working in a streaming data scenario, based on distances and densities.
- In the context of the data stream scenario, LAMDA-RD, based on the performance metrics of Table 6, is widely superior to LAMDA, LAMDA-TP and ADDClustering, since our proposal has a merging process that allows avoiding the creation of an excessive number of clusters. Particularly, the expert must calibrate the parameters to get an adequate model.
- Our proposal is able to work on clustering problems in an appropriate manner. Our method reaches the best results comparing with other well-known clustering algorithms in benchmarks with overlap <20% and unbalanced datasets, such as: Dim1024, Segment, Unbalance and s1. While increasing the individuals in the overlap area for instance s2, s3, and a1, the algorithm decreases its performance since it is based on density measurements. The advantage of our algorithm is that it can work in online mode with streaming data. However, it is not adequate when the dataset has a high number of individuals, because the algorithm's execution time will considerably increase.
- The advantage of LAMDA-RD is that it can discover new groups with a low computational cost. The addition of a merging algorithm avoids the creation of an excessive number of poor quality clusters, which is demonstrated by the performance metrics.
- LAMDA has the characteristic of making intrinsically a split process, generating new classes when it does not identify the similarity between the individual and the clusters. However, the quality of the clusters generated is not good as we show in the experiments, especially in cases of high overlapping, which LAMDA-RD corrects presenting better results in all the benchmarks.
- The parameters d_{nb} and D_t regulates the requirements in the clusters to be merged. Specifically, when setting D_t at a low value, then a merging process is made between nearby, but dissimilar clusters, whereas when setting it to a high value, then the merge is made between nearby and similar clusters.
- In gas lift wells, LAMDA creates an excessive number of clusters of bad quality (see Table 11), obtaining a model that does not characterize the system based on real classes. LAMDA-TP considerably reduces the number of clusters created; however, it has the issue with the low production cluster, since it divides it into two groups, which is not adequate since its characteristics are similar in all the individuals of that state. LAMDA-RD identifies the majority of individuals in their respective clusters compared to real classes, and perfectly improves the assignment made by LAMDA, performing the merging process in order to reduce the number of classes to the correct value.
- The robust distance (RD) related to d_{nb} allows improving the quality of the resulting clusters, since this term penalizes the dissimilarity between the individuals and the clusters. Figures 8 and 9 show how this term affects the quality of clusters related to the minimum P_C and the final number of clusters created m. These figures have shown

that proper calibration of the d_{nb} parameter (related to RD) plays an important role in the final result.

A. ADVANTAGES AND DRAWBACKS OF THE PROPOSED EXTENSION

The main advantages of the algorithm are:

- Competitive results in the context of a stream mining scenario.
- A Cluster quality improvement over the original LAMDA algorithm, results supported by several performance metrics.
- It does not require knowing a priori the number of clusters in the clustering process.
- It works correctly with individuals of 20 descriptors, as is seen in the results, i.e. in Segment benchmark.
- It is a non-iterative method to obtain the model, reducing the number of operations compared to other clustering methods.
- It is a white box with simple operations that are easily modified to obtain better results.

The main drawbacks of the algorithm are:

- Two parameters for calibration that must be fine-tuned for best results.
- Increased computational complexity compared to the original LAMDA algorithm.
- The temporal complexity increases as the number of descriptors and the number of elements in the clusters to be merged increases

VII. CONCLUSION

In this paper has been proposed extensions for the LAMDA algorithm in the clustering context. These extensions are based on two strategies, the first one by calculating the MAD with the Cauchy function, adding a factor to penalize the individual-cluster dissimilarity, to make a better assignment to a cluster. Additionally, an automatic algorithm has been added to LAMDA to perform the merge process, which analyses the similarity between neighboring clusters to decide if this process is carried out or not. The merging algorithm has an additional execution time, because it evaluates the overlap based on distance and density measures of similarity of the clusters to perform the merge; however, the computational cost is compensated with the algorithm's ability to avoid creating an excessive number of clusters. In addition, parameters can be calibrated to increase or decrease the number of clusters, a feature that is not possible in LAMDA-TP and in LAMDA. In general, in the comparative study with LAMDA and LAMDA-TP, it was possible to demonstrate that LAMDA-RD significantly improves the performance and the clusters formed, especially with the metrics *SC*, WB_{index} and *RI*.

LAMDA-RD has been tested in several benchmarks with different overlapping percentage, and its results have been compared with other clustering algorithms. In these comparisons, it was determined that in cases when the overlapping is 0-20%, our method presents results as good as iterative methods (KM, KMD and FCM), and as the overlapping increases its performance decreases because it is more complex to make an assignment when the elements have characteristics of several clusters at the same time, which are difficult to differentiate, cases where iterative methods are the best. Specifically, our proposal is the best in the cases: Dim1024, Segment, Unbalance and s1, that is, when the amount of elements in overlapping areas is not excessive.

The results of the gas lift well are satisfactory since the algorithm has been able to identify the expected production states, partitioning the individuals (wells) into good quality clusters with the greatest similarity. The density threshold D_t, and the distance between neighbors d_{nb} allow to the expert to calibrate the number of the desired clusters.

As future work, we propose to improve the performance of the algorithm when it is tested in strong non-Gaussian distribution of feature values, and address in detail the curse of dimensionality in datasets with a very high number of features. Also, we want to combine the clustering algorithm with supervised learning features to implement a hybrid algorithm based on LAMDA, which can be applied in systems with labeled and unlabeled data. Additionally, we want to improve the algorithm performance computing the optimum threshold for each cluster for the merge process, and formalize in a more precise way the parameter calibration of the algorithm.

REFERENCES

[1] L. F. Zhu, J. S. Wang, and H. Y. Wang, "A novel clustering validity function of FCM clustering algorithm," *IEEE Access*, vol. 7, pp. 152289–152315, 2019.
[2] J. F. B. Valderrama and D. J. L. B. Valderrama, "On LAMDA clustering method based on typicality degree and intuitionistic fuzzy sets," *Expert Syst. Appl.*, vol. 107, pp. 196–221, Oct. 2018.
[3] D. G. Ferrari and L. N. de Castro, "Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods," *Inf. Sci.*, vol. 301, pp. 181–194, Apr. 2015.
[4] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping Multidimensional Data*. Berlin, Germany: Springer-Verlag, vol. 2012, pp. 25–71.
[5] 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, Jan. 2014.
[6] A. Ben Ayed, M. Ben Halima, and A. M. Alimi, "Survey on clustering methods: Towards fuzzy clustering for big data," in *Proc. 6th Int. Conf. Soft Comput. Pattern Recognit. (SoCPaR)*, Aug. 2014, pp. 331–336.
[7] H. Parvin and B. Minaei-Bidgoli, "A clustering ensemble framework based on selection of fuzzy weighted clusters in a locally adaptive clustering algorithm," *Pattern Anal. Appl.*, vol. 18, no. 1, pp. 87–112, Feb. 2015.
[8] A. Bagherinia, B. Minaei-Bidgoli, M. Hossinzadeh, and H. Parvin, "Elite fuzzy clustering ensemble based on clustering diversity and quality measures," *Int. J. Speech Technol.*, vol. 49, no. 5, pp. 1724–1747, May 2019.
[9] H. Alizadeh, B. Minaei-Bidgoli, and H. Parvin, "Optimizing fuzzy cluster ensemble in string representation," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 27, no. 2, Mar. 2013, Art. no. 1350005.
[10] M.-S. Yang, C.-Y. Lai, and C.-Y. Lin, "A robust EM clustering algorithm for Gaussian mixture models," *Pattern Recognit.*, vol. 45, no. 11, pp. 3950–3961, Nov. 2012.
[11] M.-S. Yang, S.-J. Chang-Chien, and Y. Nataliani, "Unsupervised fuzzy model-based Gaussian clustering," *Inf. Sci.*, vol. 481, pp. 1–23, May 2019.
[12] C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2013.
[13] J. Sui, Z. Liu, A. Jung, L. Liu, and X. Li, "Dynamic clustering scheme for evolving data streams based on improved STRAP," *IEEE Access*, vol. 6, pp. 46157–46166, 2018.

[14] J. Nayak, B. Naik, and H. S. Behera, "Fuzzy C-means (FCM) clustering algorithm: A decade review from 2000 to 2014," in *Computational Intelligence in Data Mining—Volume 2*, vol. 411, H. S. Behera and D. P. Mohapatra, Eds. Springer, 2015, pp. 133–149.
[15] M. Gong, L. Su, M. Jia, and W. Chen, "Fuzzy clustering with a modified MRF energy function for change detection in synthetic aperture radar images," *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 1, pp. 98–109, Feb. 2014.
[16] S. A. Sert, H. Bagci, and A. Yazici, "MOFCA: Multi-objective fuzzy clustering algorithm for wireless sensor networks," *Appl. Soft Comput.*, vol. 30, pp. 151–165, May 2015.
[17] H. Izakian, W. Pedrycz, and I. Jamal, "Fuzzy clustering of time series data using dynamic time warping distance," *Eng. Appl. Artif. Intell.*, vol. 39, pp. 235–244, Mar. 2015.
[18] J. Aguilar-Martín and R. L. De Mantaras, "The process of classification and learning the meaning of linguistic descriptors of concepts," in *Approximate Reasoning in Decision Analysis*. Amsterdam, The Netherlands: North Holland, 1982, pp. 165–175.
[19] C. Bedoya, J. Waissman Villanova, and C. V. Isaza Narvaez, "Yager-Rybalov triple? Operator as a means of reducing the number of generated clusters in unsupervised Anuran vocalization recognition," in *Proc. Mex. Int. Conf. Artif. Intell.*, 2014, pp. 382–391.
[20] T. Kempowsky, A. Subias, and J. Aguilar-Martin, "Process situation assessment: From a fuzzy partition to a finite state machine," *Eng. Appl. Artif. Intell.*, vol. 19, no. 5, pp. 461–477, Aug. 2006.
[21] L. Morales, C. A. Ouedraogo, J. Aguilar, C. Chassot, S. Medjiah, and K. Drira, "Experimental comparison of the diagnostic capabilities of classification and clustering algorithms for the QoS management in an autonomic IoT platform," *Service Oriented Comput. Appl.*, vol. 13, no. 3, pp. 199–219, Sep. 2019.
[22] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy C-means clustering algorithm," *Comput. Geosci.*, vol. 10, nos. 2–3, pp. 191–203, Jan. 1984.
[23] D. Gustafson and W. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *Proc. IEEE Conf. Decis. Control Including 17th Symp. Adapt. Processes*, Sep. 1978, pp. 761–766.
[24] Y. Li, J. Cai, H. Yang, J. Zhang, and X. Zhao, "A novel algorithm for initial cluster center selection," *IEEE Access*, vol. 7, pp. 74683–74693, 2019.
[25] B. Lamrini, M.-V. Le Lann, A. Benhammou, and E. K. Lakhal, "Detection of functional states by the 'LAMDA' classification technique: Application to a coagulation process in drinking water treatment," *Comp. Rendus Phys.*, vol. 6, no. 10, pp. 1161–1168, Dec. 2005.
[26] H. R. Hernandez, J. L. Camas, A. Medina, M. Perez, and M. V. Le Lann, "Fault diagnosis by LAMDA methodology applied to drinking water plant," *IEEE Latin Amer. Trans.*, vol. 12, no. 6, pp. 985–990, Sep. 2014.
[27] M. Ruiz, J. Colomer, M. Rubio, and J. Meléndez, "Combination of multivariate statistical process control and classification tool for situation assessment applied to a sequencing batch reacto wastewater treatment," in *Proc. 8th Int. Workshop Intell. Stat. Qual. Control*. Zaklad Poligraficzny, Warszawa: Printing House, 2004, pp. 1–9.
[28] J. F. Botía, C. Isaza, T. Kempowsky, M. V. Le Lann, and J. Aguilar-Martín, "Automaton based on fuzzy clustering methods for monitoring industrial processes," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1211–1220, Apr. 2013.
[29] J. Mora-Florez, V. Barrera-Nunez, and G. Carrillo-Caicedo, "Fault location in power distribution systems using a learning algorithm for multivariable data analysis," *IEEE Trans. Power Del.*, vol. 22, no. 3, pp. 1715–1721, Jul. 2007.
[30] J. G. Zambrano, E. Guzmán-Ramirez, and O. Pogrebnyak, *Search Algorithms for Engineering Optimization*. Rijeka, Croatia: InTech, 2013.
[31] J. G. Z. Nila, E. Guzman, and O. Pogrebnyak, "Search algorithm for image recognition based on learning algorithm for multivariate data analysis," in *Search Algorithms for Engineering Optimization*. Rijeka, Croatia: InTech, 2013, pp. 3–22.
[32] E. Guzman, J. G. Zambrano, A. Orantes, and O. Pogrebnyak, "A theoretical exposition to apply the lamda methodology to vector quantization," in *Proc. 52nd IEEE Int. Midwest Symp. Circuits Syst.*, Aug. 2009, pp. 743–746.
[33] T. Kempowsky, "Surveillance de procédés base de méthodes de classification?: Conception d'un outil d'aide pour la détection et le diagnostic des défaillances," Ph.D. dissertation, INSA-Toulouse, Toulouse, France, 2004.
[34] J. Waissman, R. Sarrate, T. Escobet, J. Aguilar, and B. Dahhou, "Wastewater treatment process supervision by means of a fuzzy automaton model," in *Proc. IEEE Int. Symp. Intell. Control. Held Jointly 8th IEEE Medit. Conf. Control Autom.*, Jun. 2000, pp. 163–168.
[35] V. Krivanek, "Application LAMDA algorithm for fault detection and isolation," in *Proc. 14th Int. Conf. Mechatronika*, Jun. 2011, pp. 46–51.
[36] A. Doncescu, S. Regis, and N. Kabbaj, "Reinforced operators in fuzzy clustering systems," in *Complex Intelligent Systems and Their Applications*, vol. 41, F. Xhafa, L. Barolli, and P. J. Papajorgji, Eds. New York, NY, USA: Springer, 2010, pp. 247–266.
[37] M. Cerrada, J. Aguilar, J. Altamiranda, and R.-V. Sánchez, "A hybrid heuristic algorithm for evolving models in simultaneous scenarios of classification and clustering," *Knowl. Inf. Syst.*, vol. 61, no. 2, pp. 755–798, Nov. 2019.
[38] M. Cheng, T. Ma, and Y. Liu, "A projection-based split-and-merge clustering algorithm," *Expert Syst. Appl.*, vol. 116, pp. 121–130, Feb. 2019.
[39] T. P. Q. Nguyen and R. J. Kuo, "Partition-and-merge based fuzzy genetic clustering algorithm for categorical data," *Appl. Soft Comput.*, vol. 75, pp. 254–264, Feb. 2019.
[40] A. Doncescu, J. Aguilar-Martin, and J.-C. Atine, "Image color segmentation using the fuzzy tree algorithm T-LAMDA," *Fuzzy Sets Syst.*, vol. 158, no. 3, pp. 230–238, Feb. 2007.
[41] M. Mizumoto, "Pictorial representations of fuzzy connectives, part I: Cases of t-norms, t-conorms and averaging operators," *Fuzzy Sets Syst.*, vol. 31, no. 2, pp. 217–242, Jun. 1989.
[42] F. A. Ruiz, C. V. Isaza, A. F. Agudelo, and J. R. Agudelo, "A new criterion to validate and improve the classification process of LAMDA algorithm applied to diesel engines," *Eng. Appl. Artif. Intell.*, vol. 60, pp. 117–127, Oct. 2017.
[43] C. Bedoya, C. Uribe, and C. Isaza, "Unsupervised feature selection based on fuzzy clustering for fault detection of the tennessee eastman process," in *Advances in Artificial Intelligence—IBERAMIA* (Lecture Notes in Computer Science), vol. 7637. Berlin, Germany: Springer, 2016, pp. 350–360.
[44] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 2, pp. 98–110, May 1993.
[45] J. Aguilar-Martín and N. López De Mantaras, *The Process of Classification and Learning the Meaning of Linguistic Descriptors of Concepts*. Amsterdam, The Netherlands: North Holland, 1982.
[46] X. Gu and P. P. Angelov, "Autonomous data-driven clustering for live data stream," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 1128–1135.
[47] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," *Int. J. Speech Technol.*, vol. 48, no. 12, pp. 4743–4759, Dec. 2018.
[48] A. Ultsch, "Clustering wih SOM: U*C," in *Proc. 5th Workshop Self-Organizing Maps*, vol. 2, 2005, pp. 75–82.
[49] C. J. Veenman, M. J. T. Reinders, and E. Backer, "A maximum variance cluster algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 9, pp. 1273–1280, Sep. 2002.
[50] Q. Zhao, M. Xu, and P. Fränti, "Sum-of-squares based cluster validity index and significance analysis," in *Proc. Int. Conf. Adapt. Natural Comput. Algorithms*, 2009, pp. 313–322.
[51] A. Starczewski and A. Krzyżak, "A modification of the silhouette index for the improvement of cluster validity assessment," *Lect. Notes Comput. Sci.*, vol. 9693, pp. 114–124, 2016.
[52] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Amer. Stat. Assoc.*, vol. 66, no. 336, pp. 846–850, Dec. 1971.
[53] U. von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, Dec. 2007.
[54] J. Sorokin. Jorsorokin/HDBSCAN. GitHub. Accessed: Jul. 15, 2020. [Online]. Available: https://github.com/Jorsorokin/HDBSCAN
[55] N. Iam-on and S. Garrett, "LinkCluE: A MATLAB package for link-based cluster ensembles," *J. Stat. Softw.*, vol. 36, no. 9, pp. 1–36, 2010.
[56] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. Gama, "Data stream clustering," *ACM Comput. Surv.*, vol. 46, no. 1, pp. 1–31, Oct. 2013.
[57] E. Camargo, J. Aguilar, A. Ríos, F. Rivas, and J. Aguilar-Martin, "Nodal analysis-based design for improving gas lift wells production," *WSEAS Trans. Inf. Sci. Appl.*, vol. 5, no. 5, pp. 706–715, 2008.
[58] E. Camargo and J. Aguilar, "Advanced supervision of oil wells based on soft computing techniques," *J. Artif. Intell. Soft Comput. Res.*, vol. 4, no. 3, pp. 215–225, Jul. 2014.
[59] E. Camargo, J. Aguilar, A. Ríos, F. Rivas, and J. Aguilar-Martin, "A neo-fuzzy approach for bottom parameters estimation in oil wells," *WSEAS Trans. Syst. Control*, vol. 4, no. 9, pp. 445–454, 2009.
[60] L. Morales, H. Lozada, J. Aguilar, and E. Camargo, "Applicability of LAMDA as classification model in the oil production," *Artif. Intell. Rev.*, vol. 53, no. 3, pp. 2207–2236, Mar. 2020.

[61] L. Morales, J. Aguilar, D. Chávez, and C. Isaza, "LAMDA-HAD, an extension to the LAMDA classifier in the context of supervised learning," *Int. J. Inf. Technol. Decis. Making*, vol. 19, no. 1, pp. 283–316, Jan. 2020.
[62] Y. H. Kim, S. C. Ahn, and W. H. Kwon, "Computational complexity of general fuzzy logic control and its simplification for a loop controller," *Fuzzy Sets Syst.*, vol. 111, no. 2, pp. 215–224, Apr. 2000.

LUIS MORALES received the degree in electronics and control engineering from the Escuela Politécnica Nacional, Quito, Ecuador, in 2010, and the M.Sc. degree in automatic and robotics from the Universitat Politécnica of Catalunya, Spain, in 2012. He is currently pursuing the Ph.D. degree in control systems with the Escuela Politécnica Nacional. He is also an Assistant Professor with the Escuela Politécnica Nacional, where he taught electronics and automation engineering. His research interests include automatic systems and artificial intelligence applied to control systems.

JOSE AGUILAR (Member, IEEE) received the M.Sc. degree in computer science from Université Paul Sabatier, France, in 1991, and the Ph.D. degree in computer science from the Universite Rene Descartes, France, in 1995.

He held the postdoctoral studies with the Department of Computer Science, University of Houston, from 1999 to 2000, and the Laboratoire D'analyse et D'architecture des Systèmes (LAAS)-CNRS, Toulouse, France, from 2010 to 2011. He is currently a Full Professor with CEMISID, Escuela de Ingeniería de Sistemas, Universidad de Los Andes, Mérida, Venezuela. He has published more than 500 articles and ten books in parallel and distributed computing, computer intelligence, and science and technology management. His research interests include artificial intelligence, semantic mining, big data, emerging computing, and intelligent environments. He is a member of the Mérida Science Academy and the IEEE CIS Technical Committee on Neural Networks. He received the title of Systems Engineer from the Universidad de Los Andes in 1987.