Clustering of non-stationary data streams: A survey of fuzzy partitional methods

Amr Abdullatif*, Francesco Masulli*†, Stefano Rovetta*

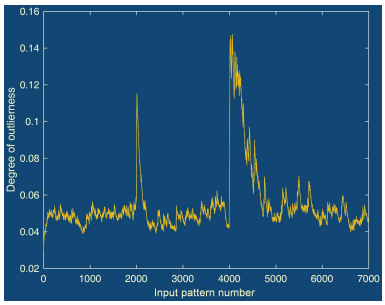
ARTICLE TYPE:

Advanced Review

ABSTRACT

Data streams have arisen as a relevant topic during the past decade. They are real-time, incremental in nature, temporally ordered, massive, contain outliers, and the objects in a data stream may evolve over time (concept drift). Clustering is often one of the earliest and most important steps in the streaming data analysis workflow. A wide literature is available about stream data clustering; however, less attention is devoted to the fuzzy clustering approach, even though the non-stationary nature of many data streams makes it especially appealing. This survey discusses relevant data stream clustering algorithms focusing mainly on fuzzy methods, including their treatment of outliers and concept drift and shift.

GRAPHICAL TABLE OF CONTENTS



Caption: Stream data change over time. Here an index of non-stationarity is shown. Fuzzy clustering methods are an effective tool to analyze data streams.

*DIBRIS - Dept of Informatics, Bioingengering, Robotics and Systems Engineering, University of Genoa, Via Dodecaneso 35, 16146 Genoa, Italy

†Sbarro Institute for Cancer Research and Molecular Medicine, College of Science and Technology, Temple University, Philadelphia, PA, USA

1 INTRODUCTION

Our society is investing massively in the collection and processing of data of all kinds, on scales unimaginable until recently[18]. The speed by which data are currently generated is growing more quickly than the memory available to practically usable computers (*information overload* challenge), reversing a trend that lasted for decades, until not many years ago. Most of these data, however, are not of archival importance. They are only interesting when they are referred to the present, and it is not useful to store all of them forever. In addition to this, pervasive computing, the Internet of Things, and ubiquitous ambient and wearable sensors provide uninterrupted flows of data that need to be taken care of in real time. In other words, many, if not most, interesting data come in the form of *streams*.

Mining data streams is a big challenge as they always depend on time, although to different degrees. They may represent actual time series, with a strong dependency on time, or quasi-stationary phenomena whose variability can be appreciated only in the long term, e.g., as evolutions in statistical distribution or a cyclical behavior. In these non-stationary conditions, any model is expected to be appropriate only in a neighborhood of the point in time where it has been learned. Over time, its validity may decrease (*concept change*). Change may be gradual (*concept drift*), or occur suddenly, for instance when switching from one operating condition to a new one (*concept shift*).

Unsupervised analysis in the form of data clustering provides a useful tool to mine data streams. Clustering methods[22,36,71] have been successfully applied in many fields such as data mining, image segmentation, fraud detection, and bioinformatics. The aim of data clustering is to group objects on the basis of a dissimilarity (or similarity) criterion obtaining clusters that are sets containing similar objects. The dissimilarity between objects is usually measured by a distance function defined on pairs of objects. The most popular distance function is the Euclidean distance[19], which is often used to measure dissimilarity between two objects and is known to work well when all clusters are well separated. Other choices may be appropriate depending on the circumstances.

In the following, we will refer to the data stream as generated by a data *source* with given characteristics, possibly non-stationary. A *clustering model* will consist in the representation of clusters as inferred from the data, along with their describing parameters. The problem of data stream clustering is then summarized as that of identifying an optimal clustering model in the presence of source variations. In the methods considered here, model identification is mostly limited to fitting (i.e., finding the optimal cluster positions/size/geometry and assigning cluster memberships to the data), although model selection (i.e., finding the optimal number of clusters and possibly other model hyperparameters) is also a relevant issue.

The representation of the model, as well as the objective that measures model quality, depend on the specific clustering method and algorithm.

In the next section we briefly introduce the problem of data streams and approaches to represent partitional clustering models.

2 DATA STREAMS AND THE PROBLEM OF CLUSTERING

The challenges of data streams clustering can be summarized as follows:

- **Single-pass operation**: Stream data naturally impose a single-pass constraint on the algorithms.
- **Robustness to outliers**: Outliers are isolated observations which are clearly not explained by the clustering model, while the majority of observations are well fitted by the current clustering model. The clustering model should be insensitive to these episodic anomalies.

- **Tracking model changes**: To remain relevant over time, the clustering model should be able to quickly adapt to concept drift and shift.
- **Balance between robustness and tracking ability**: Insensitivity to outliers and reactivity to changes are two competing goals. The balance between them is a problem-dependent design choice.

In this survey, data streams are assumed to generate independent, random instances of objects from an underlying distribution, the data source. The distribution is non-stationary, i.e., evolving, but its rate of change is mostly slow with respect to the sampling rate (this is concept drift according to our terminology) so that clustering is feasible. In the presence of concept drift only, we assume that there exist an interval around the present instant in time where the distribution can be considered stationary. We refer to this assumption as the "local i.i.d." hypothesis. Occasionally, we assume that this hypothesis is broken. These changes can either be pointwise in time (outliers) or stable over a period of time, so that the source cannot be described by the same model as before (concept shift). [1]

Many applicative problems can be modelled in this way. As an example, to monitor the health and well-being of elderly people at home alone, human activity recognition may be inferred from ambient (motion, presence, location) and wearable (3D acceleration, 3D orientation, odometry, bio-signal) sensors. In this scenario, observed patterns will be clustered in groups corresponding, not necessarily one-to-one, to individual everyday activities. Outliers are activities that are occasionally performed in an unusual way with respect to the norm, due to temporary reasons (e.g., lunch time delayed because of a long phone call from a relative). Concept shift is represented by changes in ambient conditions that induce changes in behaviour and routines (e.g., starting the habit of an outdoor walk in the evening when the weather becomes sufficiently mild), while drift can indicate either improvement (e.g., the acquisition of better skills in physical exercise) or decline (e.g., gradual loss of interest in cooking) in daily habits.

Another application can be in short-term road traffic forecasting.[1] The forecast of vehicle flow and density on a given road in the next few minutes can be based on a model of traffic conditions that represents different situations (different times of the day, different days of the week) as individual clusters. Usually these are stationary, but the importance of some routes may decrease or increase in time as a result, for instance, of the closing of a sports centre or the construction of a new shopping centre (concept drift). Unusual patterns may be observed in case of exceptional events or road accidents (outliers). Sometimes, these changed conditions may persist as a new set of traffic patterns, as in the case of extended road works (concept shift).

Let $O = \{o_1, ...o_n\}$ denote a stream of n objects, e.g., traffic flow patterns on a street network or web activity profiles on an e-commerce website. In the streaming data model, n is very large (in principle, unbounded) and objects are observed sequentially.

Individual observations in the stream are represented by a fixed set of features. Specifically, the methods considered here assume that each observed object is associated to a numeric feature vector: $o_i \rightarrow x_i \in \mathbb{R}^d$. Selecting the best representative features of the objects plays an important role in ensuring a good clustering performance.

In general, *incremental clustering* processes data in chunks[16] of size $s \ll n$. It can be *on-line* if chunks are trivially reduced to one observation ($s = 1$), *batch* if they consist of a window of larger size $s > 1$, either sliding (overlapping) or disjoint. Schemes using disjoint windows introduce a clear distinction between current observations, which are retained, and old ones, which are summarized and discarded. Temporal weights (*damped* models[64]) can also be used to account for gradual ageing, either over the whole history or

[1] Of course, depending on the length of the time interval considered and on the magnitude of the model change, the definitions of concept shift and drift can blur into each other. This depends on the method, on the source, and on the user's decision strategy. For simplicity of presentation here we assume that drift and shift are sufficiently distinguishable.

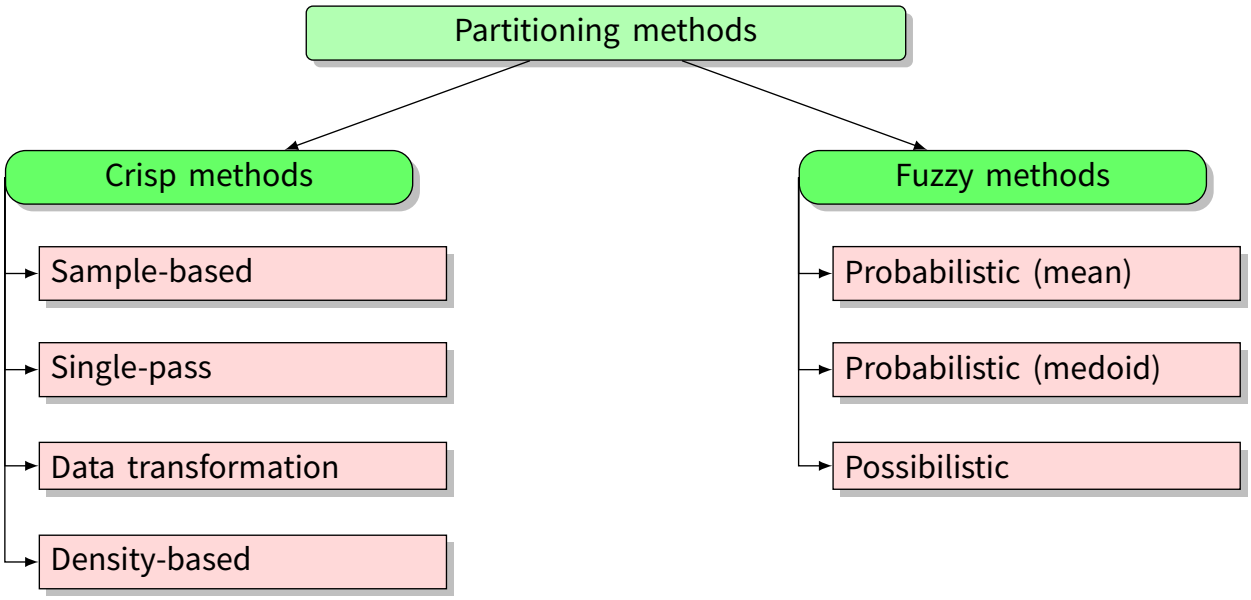


Figure 1: Partitioning methods

on a fixed-size window. With fixed window size, an important parameter that controls the tracking ability is the memory length of the method, which is related to window size and to the damping factor when fading weights are used.

Clustering methods can be roughly divided into the two categories of partitioning and hierarchical methods.[36] We focus on the partitional methods summarized in Figure 1 because in the literature this is the area where most of the research effort is spent. Partitions of a set of data (clusters) can be of two main types: crisp, or fuzzy[10,43].

The *codebook* V is defined as the set $V = \{v_1, ..., v_c\}$, with $c \leq n$ and typically $c \ll n$. Each element $v_i \in \mathbb{R}^d$ is called a *centroid* and usually is in a one-to-one relationship with a cluster. The inclusion of observation number i in clusters number j is measured by the membership μ_{ij}. Centroids are locations in the data space that minimize their average squared distance to points in their cluster. An alternative representation of a cluster is by means of the point in the cluster that minimizes the average absolute distance from all others. This is termed a *medoid*.

Crisp (or hard) clusters are characterized by binary, integer-valued memberships, $\mu_{ij} \in \{0, 1\} \subset \mathbb{N}$. Also, if $\mu_{ij} = 1$, then $\mu_{ik} = 0 \quad \forall k = 1 \dots c, \ k \neq j$. In other words, each object can be a member of only one cluster, for which it has membership degree of one. Mutual exclusion can also be expressed with the following *probabilistic constraint*:

$$\sum_{j=1}^{c} \mu_{ij} = 1 \quad \forall i : 1 \dots n \ . \tag{1}$$

In contrast to crisp partitions, *fuzzy* (or soft) partitions admit real-valued memberships, $\mu_{ij} \in [0, 1] \subset \mathbb{R}$. Each object can be a member of all clusters to which it has in general different membership degrees, each less than one. The probabilistic constraint of (1) can either be enforced or not. In the latter case, the model is termed *possibilistic*.[43?]

3 AN OVERVIEW OF CRISP CLUSTERING OF DATA STREAMS

Before focusing on fuzzy methods, this section concisely reviews the main approaches to crisp clustering of data streams.

The following types of algorithms are available in the literature:

- *Sample-based* clustering algorithms: One of the common strategies is sub-sampling the data, then clustering is executed on the samples. Popular sample-based methods include CLARA [40], CLARANS [54], CURE [28], and the coresets algorithms [30]. These algorithms may be employed for stream clustering by using as subsamples the observations included in each chunk.

- *Single-pass* algorithms: These algorithms adopt the single pass model to deal with the data stream requirement that data can be examined only once. According to this model, as data are scanned, summaries of past data are stored to leave enough memory for processing new incoming data. These algorithms are categorized into incremental approaches [12,13,48] and divide-and-conquer approaches [3,27]. Incremental approaches may either deal with one object at a time, or collect data into chunks. The goal of incremental clustering is to find K representatives (clusters) to represent the whole dataset and determine the final clustering results.

- *Data transformation* algorithms: These algorithms alter the structure of the data themselves so that they can be more effectively accessed. Popular data transformation algorithms include BIRCH, [74] CLUTO, [55] and GARDEN [39].

- *Density-based* clustering algorithms: These approaches aim to find clusters of arbitrary shape by modelling clusters as dense region separated by sparse regions [4]. Popular density-based algorithms include DBSCAN [21], OPTICS [6], DENCLUE [31], Den-Stream [14],rDen-Stream [47], and DD-STREAM [38]. Their common strategy revolves around finding core samples in high-density areas and expanding clusters from them, which is especially effective when the data contain clusters of similar density.

Many of the methods mentioned so far only focus on the progressive nature of the task, with less attention to tracking evolutions in the data stream, i.e., they do not include mechanisms to take into account possible changes in the source, but only to build the model in an incremental fashion. The ability to track changes, however, is a basic requirement for methods to be suitable for non-stationary stream learning.

4 FUZZY CLUSTERING OF DATA STREAMS

Most fuzzy clustering methods are modeled over Fuzzy c means [11], a probabilistic method where the goal is to minimize the following objective function:

$$J_{FCM} = \sum_{j=1}^{c} \sum_{i=1}^{n} \mu_{ij}^{m} \left\| x_i - v_j \right\|^2 , \tag{2}$$

where $m \geq 1$ is a fuzzification parameter. This results in a "soft partitioning" method, which allows observation to belong to all clusters to varying degrees, provided that (1) is satisfied (i.e., the total membership of a point to all clusters must be 1).

As already noted there are mainly two types of fuzzy clustering methods. If Eq. (1) is enforced we term them *probabilistic*, otherwise they are *possibilistic*. The idea of possibilistic clustering is to view membership degrees as typicalities. This allows points to have low memberships to all clusters, which is an indication of a point being an outlier.

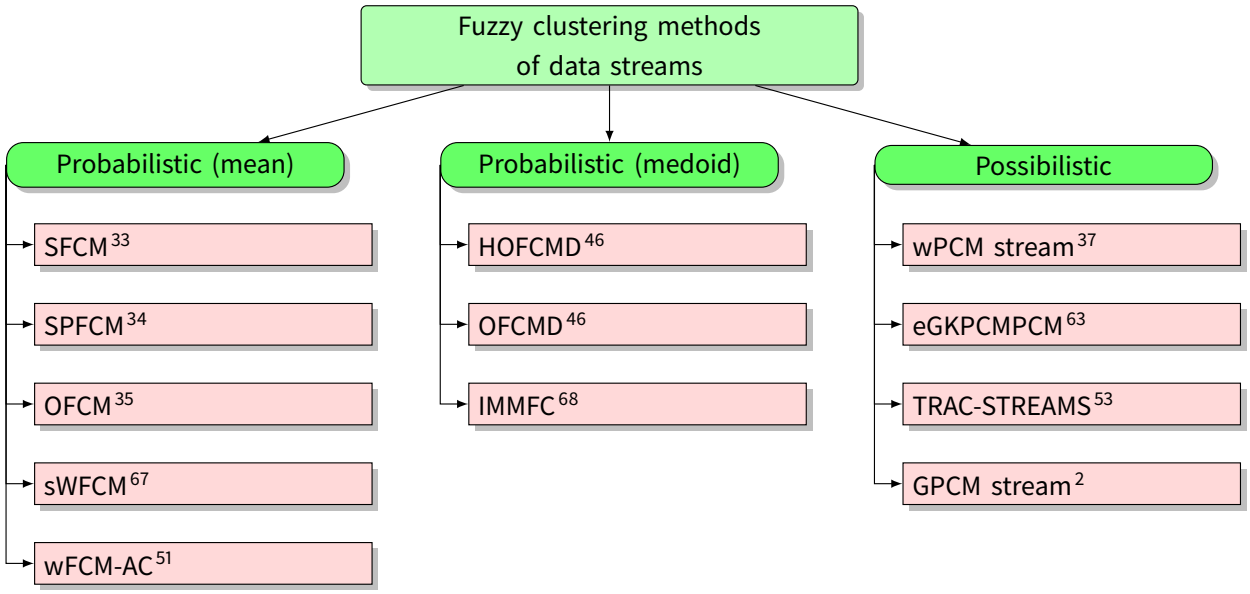


Figure 2: Fuzzy clustering methods of data streams

Fuzzy clustering, despite being less used in stream data analysis, offers potential advantages over crisp partitioning.

- In the case of non-stationary streams, i.e., in the presence of concept drift, the progressive loss of validity of a clustering model can be better represented by the real-valued membership which changes smoothly, without having to make abrupt changes in the model.
- With crisp membership, for which only two values are admissible, detection of a concept change requires a data sample sufficiently large to allow reliable statistical estimate, while with fuzzy models a pointwise evaluation is possible, thus allowing much faster detection.
- In the particular case of possibilistic models, the additional robustness property allows pointwise detection of outliers in addition to source change detection.

Fuzzy clustering also has some disadvantages that should be taken into account, stemming from the fact that memberships are real-valued quantities.

- The search space is larger, convergence is asymptotic, and the arithmetic employed in software implementations is floating-point. These aspects make fuzzy methods intrinsically slower. However, for data streams these issues are mitigated by the need to work within a limited temporal horizon to reduce the effects of non-stationarity, so this disadvantage may not be relevant in many applications.
- The standard Euclidean distance may suffer from dimensionality effects [49] to a larger extent than in the crisp case. Countermeasures are available in the form of different distance formulations, tuning of the membership function [69], or, in extreme cases even totally different approaches, like working in kernel space [23], may be necessary. This issue is however outside the scope of this review.

The fuzzy stream clustering methods that are reviewed in the following are summarized in Figure 2.

4.1 FUZZY CLUSTERING BASED ON CENTROIDS

In these methods, centroids are computed by using an existing "batch" fuzzy clustering method on each chunk of data. Changes from one chunk to the next are smoothed by including in the next chunk a suitably summarized version of the previous chunk (a "synopsis").

A minimal synopsis is usually represented by the location of past centroids. Additional information is typically included, mostly in the form of weights associated to points. The introduction of weights into the FCM objective has been studied in the context of data reduction[20] and results in a method termed *Weighted Fuzzy c Means* (wFCM)[62]. Singleton points from the current chunk receive unit weights, while centroids from the previous iteration(s) are weighted taking into account their membership mass and previous weight (which at iteration $t = 0$ are initialised as all ones):

$$w_h(t) = \sum_{j=1}^{c} u_{hj} w_j(t-1), \quad 1 \leq h \leq c\,, t \geq 1\,. \tag{3}$$

The weights so obtained, or computed according to other formulas for other methods, are used to scale the contribution of the points to which they refer when computing averages. In this way, the importance of points can be differentiated when updating the model.

The methods described in this subsection use wFCM and differ mostly in the details of how weighting is computed and in the historical information that is included in each "batch" along with the current chunk of data.

As a convenient standard for describing the different methods we will draw inspiration from the DCC (Dynamic Clustering Cube) framework.[58] DCC views clustering from the granular computing standpoint whereby clusters represent a coarser level of granulation with respect to raw data objects.

The DCC is based on three crucial dimensions to describe any clustering algorithm:

- *Characteristics of change*: The types of concept change that are taken into account;
- *Types of granulation*: The representation of clusters;
- *Clustering processes*: The details of how the method builds and maintains the clustering model.

4.1.1 SFCM

The *Streaming Fuzzy c Means* (SFCM) method[33] is a streaming variant of the Fuzzy c Means, with the goal of balancing the trade-off between tracking an evolving distribution and summarizing the data seen so far. This trade-off is tuned by varying past history usage.

The method starts by clustering s data points that are present at time instant t and clusters are summarized by c cluster centroids. Each centroid is weighted by the sum of all the points memberships in that cluster. For s observations at time t, the weight of centroid j is computed as a sum of memberships:

$$w_j = \sum_{i=1}^{s} \mu_{ij},\ 1 \leq j \leq c \tag{4}$$

In the next chunk, the centroids from the last chunks will be used as initial centroids for clustering the new points. The length of the history taken into account, i.e., how many past centroids are stored, depends on the size of the available memory,

Dimension 1: Characteristics of change. SFCM can track any type of variation in the data, as long as they don't imply a change in the number of clusters which is fixed in advance.

Dimension 2: Types of granulation. The data are assumed to arrive in chunks where each chunk is processed as a whole. Clusters are computed on each chunk. The number of clusters stored before updating depends on the size of the available memory and can be defined by the user.

The method is based on fuzzy c means clustering and cluster representation is by means of centroids and a membership function.

Dimension 3: Clustering processes. The first chunks cluster centroids are initialized randomly while the other chunks are initialized as the last chunks cluster centroids. Using the past result as initialisation encourages, but does not guarantee, similar cluster positions from one chunk to the following one. Consequently, the dynamics of change in general is stepwise rather than continuous.

SFCM works well for large datasets. However, it uses a fixed number of clusters, which may be sub-optimal for non-stationary data streams. Additionally, the centroids are updated by doing re-clustering after a certain number of chunks defined by the user and not through an incremental learning mechanism.

4.1.2 SPFCM, SWFCM AND OFCM

Both *Single-Pass Fuzzy c Means* (SPFCM)[34] and *Online Fuzzy c Means* (OFCM)[35] are incremental fuzzy methods. Similar to SFCM, they process incoming data chunk by chunk. SPFCM and OFCM differ in the way they handle the centroids of each chunk.

In SPFCM the data are assumed to be loaded in memory, and chunks are sampled randomly. In the hypotheses stated in the previous section, data from the stream are equivalent to a random sampling, so this method can be applied to streams. The centroids obtained from the last chunks are combined into the upcoming chunk, with weights that are computed from memberships, and at the end, a final set of centroids for the entire dataset is generated. The weight for centroid j in each chunk is computed as follows:

$$w_j = \sum_{i=1}^{n+c} \mu_{ij} w_i, \quad 1 \leq j \leq c\,. \tag{5}$$

The weights of the n singleton objects are all set to one. The *Stream Weighted Fuzzy c Means* (sWFCM)[67] method is essentially similar to SPFCM. In OFCM, the weights are computed as a combination of past weights and memberships, as follows:

$$w_j = \sum_{i=1}^{n} \mu_{ij} w_i,\ 1 \leq j \leq c, \quad w_i = 1,\ \forall 1 \leq i \leq n\,. \tag{6}$$

Centroids are computed independently on each chunk, and a consensus post-processing is required.

Dimension 1: Characteristics of change. Both methods can handle large data and are able to track drift, but again they are not suitable for detecting concept change in data streams because they lack a mechanism to distinguish between outliers and steady source changes.

Dimension 2: Types of granulation.

SPFCM reads data according to the partial data accesses (PDA) model. Data are randomly accessed. The PDA size is defined by the user.

sWFCM and OFCM assumes data arriving as a stream of chunks where the chunk size is defined by the user.

The clustering model is centroid-based, with a membership function.

Dimension 3: Clustering processes. SPFCM starts by clustering the first PDA into c cluster centroids using fuzzy c means. Then the data in memory is condensed into c weighted points. Those weighted points will be clustered with new points in the next PDA.

Algorithm 1 wFCM-AC method[51]

1: Apply the wFCM on the data with current cluster number c
2: Apply the wFCM on the data with cluster number $c-1$ and choose the best structure.
3: Apply the wFCM on the data with cluster number $c+1$ and choose the best structure.
4: Choose the structure which improves the quality measure.

OFCM also clusters data in each chunk by fuzzy c means. The maximum number of clusters is assumed to be known. After data in one chunk is clustered, memory is freed by condensing the clustering solution in the memory into c weighted examples. After a certain amount of chunks defined by the user, the weighted clustering solutions are merged into c final clusters. The merging operation is done by clustering all the weighted centroids in the ensemble using their weights using Weighted FCM (wFCM).

In principle, SPFCM requires randomly sampling the data. If the source obeys the "local i.i.d." assumption, this is approximately true and the method is applicable.

These methods require a priori knowledge about the number of clusters.

4.1.3 WFCM-AC

The problems of detecting outliers and tracking concept change are not considered in the previously described methods, where concept change is handled by re-clustering. To cope with evolving nature of data streams, an extension to sWFCM called *WFCM with Adaptive Cluster number* (wFCM-AC)[51] was presented. The adaptive process is a local search by varying the number of clusters, as summarized in algorithm 1. The quality of the searched configurations is measured by the Xie-Beni index.[57,70] The advantage of this algorithm is the dynamic determination of the optimal number of clusters, although at any given step the search is limited to the options $\{c-1, c, c+1\}$. The methods work well in tracking concept drift in data streams, but it doesn't have an outlier/change detection mechanism. It is also slower than the other algorithms because it looks for the optimal number of clusters in each chunk of data.

Dimension 1: Characteristics of change. The method adapts to concept drift/shift including changes in the number of clusters.

Dimension 2: Types of granulation. The data are assumed to arrive in chunks where each chunk is processed as a whole. FCM is applied to normalized chunks from the stream, where each object is standardised by subtracting the mean and dividing the result by the standard deviation. Each cluster is weighted by summing the membership values that belong to it.

Dimension 3: Clustering processes. The method starts by clustering the first normalized chunk using fuzzy c means. The obtained centroids are used as an initialization for clustering the next chunk. For each chunk the algorithm tries to find best number of clusters by increasing and decreasing the number of clusters by one. Quality of clustering structure is measured for each iteration of the algorithm and the best structure is then chosen.

wFCM-AC requires multiple evaluations for each chunk to find the best clustering structure, which makes it slow with respect to other algorithms.

4.2 FUZZY CLUSTERING BASED ON MEDOIDS

These algorithms are medoid-based. Medoids are intrinsically more robust to random variations and outliers than centroids, since their location is not computed, but selected from the observations actually present in a cluster. For the methods considered here, a medoid is an object in a cluster that has a minimum average

dissimilarity to all the other objects in that cluster.

A drawback shared by methods based on medoids consists in the reduced reactivity to concept changes, because robust methods are inherently based on the concept of outlier rejection rather than detection.

4.2.1 OFCMD, HOFCMD

The *Online Fuzzy c Medoids* (OFCMD)[46] method is similar to OFCM in that the final set of reference points (medoids rather than centroids) is generated by the *weighted fuzzy medoid* (wFCMD) algorithm on medoids obtained from all the chunks.

In the *History-Based Online Fuzzy c Medoids* (HOFCMD)[46] model, similar to SPFCM, a subset of medoids from the previous chunk are combined with the upcoming chunk as history information. The final set of medoids is generated after processing all the chunks.

These methods add robustness to the basic methods on which they are respectively modelled.

4.2.2 IMMFC

Different from previous two methods, the *Incremental Multiple Medoids-based Fuzzy Clustering* (IMMFC)[68] model provides a new mechanism by:

- Selecting multiple medoids instead of one to represent the clusters in each chunk,
- Iteratively updating both fuzzy membership and cluster weight,
- Automatically generating pairwise constraints from the medoids obtained from every chunk, which is used to help the final data partition.

The IMMFC proves to be relatively insensitive to the order of data and it was mainly used for handling large multi-dimensional data.

4.3 POSSIBILISTIC CLUSTERING BASED ON CENTROIDS

In general, possibilistic partitions can be obtained by the *Possibilistic c Means* (PCM)[44?] , a possibilistic variant of FCM. Since clusters are not mutually exclusive, in this method cluster memberships do not "compete" with each other, and the extension of each cluster must be specified with an explicit, additional *scale* or *width* parameter. In general this is a scalar, reflecting the assumption that in the data space there are no preferential directions. One method, however, uses full matrices to be able to specify arbitrary hyperellipsoidal clusters.

Possibilistic clustering was designed to be robust to outliers, so it lends itself very well to noisy stream learning. On the other hand, the number of parameters to train is higher.

It has been observed that PCM may produce overlapping clusters[7]. In the context of non-stationary streams, clusters that start collapsing on each other are used to identify a possible inadequacy in the number of clusters, providing an additional indicator of concept change. However, this often requires some additional processing. One of the methods described below[2] is based on a variant of possibilistic clustering[50] that limits this phenomenon by allowing the user to set the desired balance between competitive behavior (partitional or "probabilistic" clustering) and cooperative behavior (mode-seeking or possibilistic clustering). This is beneficial because it avoids the need of an accurate initialization or post-processing. The possibilistic stream clustering methods considered here are all based on the PCM method.

4.3.1 WPCM STREAM[?]

The *Weighted Possibilistic c Means for streams* (wPCM stream)[37] method is a modified version of WFCM in a possibilistic variant (WPCM, Weighted Possibilistic c Means) to cope with changes in data streams. It assigns weights to the incoming data streams based on the existence of concept drift, which must be assessed separately by the user.

If concept change is absent, all the weights are set equal to one for all data elements:

$$w_j = 1,\ j = 1, ..., n. \tag{7}$$

In case of the existence of concept change, the weights are updated as follow:

$$w_{j+1} = w_j 2^{\lambda},\ w_1 = 1,\ j = 1, ..., n\,, \tag{8}$$

where $\lambda > 0$ is a decay rate parameter which reflects the speed of forgetting the influence on the clustering results of the old data chunks.

Dimension 1: Characteristics of change. The method is suitable for non-stationary streams, but its sensitivity to change is pre-determined by the user-selected parameter λ.

Dimension 2: Types of granulation. The method is based on possibilistic clusters described by a location (centroid) and a scale.

Dimension 3: Clustering processes. The Possibilistic c means method is used on each chunk. The method is of the dixed-window-size, damped memory type.

Due to the user-selected parameter and to the fact that there is no adaptivity in weight computation, the authors report that the method performs well when it is tuned to the type of change present in the source, but is outperformed by the base PCM method when no concept change is present.

4.3.2 EGKPCM

The *evolving Gustafson-Kessel possibilistic c Means* (eGKPCM)[63] approach is a possibilistic, stream-oriented variation over the Gustafson-Kessel clustering method. In this approach, the scalar width parameters of possibilistic clustering are replaced by scale matrices, so as to allow general elliptic clusters. This is the "Gustafson-Kessel Possibilistic c Means" (GKPCM). To obtain the evolving-stream oriented variant, data are evaluated for their typicality. Typical points are discarded after storing summary information, in the form of location and (matrix) scale of centroids. Atypical points are stored in a buffer and used to find new centroids.

The algorithm starts by defining a buffer size for storing data objects, minimum membership, fuzziness value, and a termination criteria. In the initialization also the first data objects stored in a buffer are used to calculate the initial cluster centres using GKPCM algorithm. When a new data sample is obtained, the membership of the object to the current set of cluster prototypes are calculated. If the maximal membership is bigger than the minimum initial membership then the sample belongs to the cluster with the biggest membership. If it is lower, the data object is stored in the buffer. When the data buffer is full, new clusters are identified by using GKPCM algorithm on the buffer data.

Dimension 1: Characteristics of change. The method is able to track several degrees of changes from concept drift to shift, including the ability to vary the number of centroids.

Dimension 2: Types of granulation. The cluster model, although in a possibilistic variant, is based on the Gustafson-Kessel method, so it includes both a centroid and a scale matrix.

Dimension 3: Clustering processes. The method is based on a fixed window and its operation can be interpreted as a non-damped, weighted model. Weights are implicitly implemented by selecting which data objects to store and which ones to discard.

The model's ability to represent non-spherical clusters gives this method an added flexibility. On the other hand, the number of model and optimisation parameters is higher, with several pre-specified quantities, for instance on the maximum number of prototypes, the maximum aspect ratio for ellipsoids, the minimum cluster size, the typicality threshold for inclusion in the buffer, and others. The method, being specific for evolving streams, can adapt to non-stationary clusters. On the other hand, it does this by subsequent re-clusterings based on whole chunks, not by incremental updates of the model.

4.3.3 TRAC-STREAMS AND GPCM STREAM

The two methods described here exploit the robustness of the possibilistic clustering model to evaluate the degree of typicality of incoming points. This allows making decisions for each observation, so they are able to update the clustering model with continuity without the need to work by chunks, i.e., in batch mode. They both belong to the family of damped models.

The **Tracking Robust Adaptive Clusters in evolving data STREAMS** (TRAC-STREAMS)[53] method builds a continuous synopsis including centroid locations and scales, and a time parameter (e.g., the number of points so far). Weights are computed from the observation-centroid distance and from the difference between their time of occurrence and the current time. The method focuses on more recent data thanks to the time-dependent weights and distinguishes between typical data and outliers by testing against a threshold obtained by the Chebychev inequality. The Chebychev bound allows creating new clusters (within a pre-set maximum number) and merging "compatible" ones, i.e., those too similar to be kept distinct.

The **Graded Possibilistic c Means stream clustering** (GPCM stream)[2] method uses as its basic clustering model the Graded Possibilistic c Means (GPCM)[50] adapted to on-line (or by-object) operation. It learns from either individual input objects as soon as they arrive, or a sliding window of fixed size that includes the newest object while forgetting the oldest one. Measures of fuzzy outlierness and fuzzy outlier density are computed from memberships. Specifically, for each incoming observation i its outlierness is its total membership to all clusters.

$$\Omega_i = \sum_{j=1}^{c} \mu_{ij} \,, \tag{9}$$

and outlier density is a time-discounted average of outlierness so far

$$\rho_i = \eta\Omega_i + (1-\eta)\rho_{i-1} \,, \tag{10}$$

with $0 < \eta < 1$ an ageing factor depending on the desired reactivity of the method. Outlier density is then used as a feedback control signal to speed up or slow down the updates to the clustering model, modulating the amount of incremental learning. This gives the method the ability to operate in the different regimes required by non-stationary data stream clustering:

- Slow learning (Concept drift),
- No learning (Outliers),
- Fast learning (Concept shift).

Dimension 1: Characteristics of change. Both methods operate on an object-by-object basis, and include adaptive parameters to be able to track changes of different intensity from drift to shift. TRAC-STREAMS measures outlierness by means of the Chebychev inequality, while GPCM stream employs a user-selected criterion to update learning parameters on the basis of outlierness.

Dimension 2: Types of granulation. The model representation includes centroids and scale, as usual for possibilistic methods. TRAC-STREAMS additionally stores current and historical weights. In GPCM

stream, there is a user-selectable parameter that sets the *degree of possibility*, making it possible to balance the trade-off between robust, mode-seeking behaviour of possibilistic models and avoidance of degenerated solutions of probabilistic models.[50]

Dimension 3: Clustering processes. Despite being based on possibilistic clustering, neither of the two methods employs PCM directly. TRAC-STREAMS uses a density-based procedure, while GPCM stream is based on a stochastic version of the maximum entropy clustering criterion.[59]

These methods have good reactivity to change, with GPCM stream giving the user more control over the reactivity-rejection trade-off.

The drawback of both algorithms is a downside of the same feature, i.e., the need for setting several initial parameters, although for both some guidance is given in the original proposal.

5 COMPARATIVE EVALUATION

This section deals with the comparative experimental evaluation of the fuzzy-based clustering algorithms.

For comparison the following methods were selected: OFCM, SPFCM, eGKPCM, and GPCM stream, as representative of the various approaches.

5.1 QUALITY INDICES

For the comparison between fuzzy and possibilistic clustering methods of data stream, we have selected three validity indices, extended to encompass possibilistic models as described by Yang and Wu[72]. They normalized the possibilistic c-memberships $\{\mu_1, \cdots, \mu_c\}_P$ to be $\left\{\mu_1^{'}, \cdots, \mu_c^{'}\right\}_F$ where for an input x the $\mu^{'}$ is defined as follow:

$$\mu_j^{'}(x) = \frac{\mu_j(x)}{\sum_{k=1}^{c} \mu_k(x)},\ j = 1, \ldots, c\,. \tag{11}$$

The first and second validity indices involve only the membership values and are based on the assumption that the outputs are better if they are close to a crisp partition. The third validity index takes into account a measure of compactness involving data and clusters, and a separation measure between clusters.

- PC (partition coefficient)[10]

$$PC(c) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c}\mu_{ij}^2, \tag{12}$$

  where $1/c \leq PC(c) \leq 1$. The higher the value of PC the better the clustering quality.

- PE (partition entropy)[8,9]

$$PE(c) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c}\mu_{ij}^2 log_2\,\mu_{ij}, \tag{13}$$

  where $0 \leq PE(c) \leq log_2\mu_{ij}$. The lower the value of PE the better the clustering quality.

- Xie-Beni(XB)[57,70] A validity function proposed by Xie and Beni with $m = 2$ and then generalized by Pal and Bezdek is defined by

$$XB(c) = \frac{\sum_{j=1}^{c}\sum_{i=1}^{n}\mu_{ij}\left\|x_i - v_j\right\|^2}{n\,min_{i,j}\left\|v_j - v_i\right\|^2} = \frac{J_{FCM}(\mu, v)/n}{Sep(v)}\,. \tag{14}$$

  J_{FCM} is the compactness measure, and $Sep(a)$ is the separation measure between clusters. The lower the value of XB the better the clustering quality.

Data are organized in chunks of observations, where the size of each chunk is ≥ 1. For each chunk, we compute the validity indices and summarised the index distribution it in a box-and-whisker plot. Box-and-whisker plots are based on a five-number summary: w_1, q_1, q_2, q_3, and w_2, where:

- the box extends from q_1 to q_3, the first and third quartiles;
- the intermediate line is at q_2, the median;
- the lower whisker extends to $w_1 = q_1 - 1.5(q_3 - q_1)$;
- the upper whisker extends to $w_2 = q_3 + 1.5(q_3 - q_1)$;
- values below w_1 or above w_2 are individually marked with crosses.

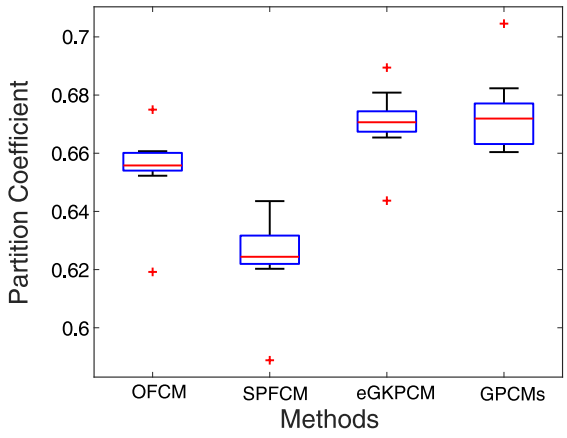
This provides information about the distribution of the variations in quality of the models along the duration of each experiment, and therefore about their ability to track changes and to remain adequate even in the presence of shift, drift, and outliers.

5.2 DATA SETS

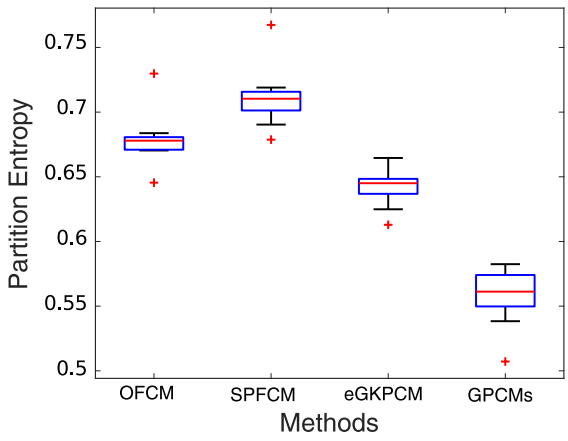
The datasets employed in our experimental analysis are the following:

- **Gaussian dataset** is a synthetic data set with four evolving two-dimensional Gaussian distributions[17]. Along time, one new data point is added and one removed so that the total number stays constant. However, the underlying data source (centroid positions) is slowly changed, leading to concept drift. Concept shift is obtained by removing a whole segment of the sequence at time 4000 where the stream changes abruptly. The data set was generated using the Matlab program ConceptDriftData.m .[2]
- **Smtp (KDDCUP99) dataset** The original KDD Cup 1999 dataset [3] contains 41 attributes (34 continuous, and 7 categorical), however, they are reduced to 4 attributes (service, duration, src_bytes, dst_bytes) as these attributes are regarded as the most basic attributes, where only "service" is categorical. Here, only "smtp" service data is used. The original data set has 3,925,651 attacks 80.1% out of 4,898,431 records. A smaller set is forged by having only 3,377 attacks 0.35% of 976,157 records, where attribute "logged in" is positive. From this forged dataset 95,156 "smtp" service data is used to construct the Smtp (KDDCUP99) dataset, The stream is stationary but contains outliers.
- **Soil moisture and temperature dataset** This dataset contains air temperature and soil temperature from the Soil Climate Analysis Network (SCAN) site 2026, "Walnut Gulch", Soil Moisture Percent: SMS.I-1:-8, Temperature Soil Temperature Observed: STO.I-1:-8. We have selected the data from the United State Department of Agriculture [4] as follows:
  - Report : Soil moisture and temperature(1999-03-19),
  - Time: Hourly,
  - Format: csv,
  - Year: 2001 (Calendar year-all days).

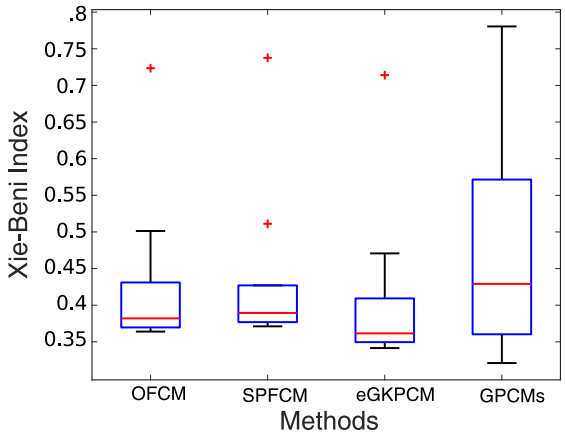
  This dataset contains concept drift and few outliers.



(a)

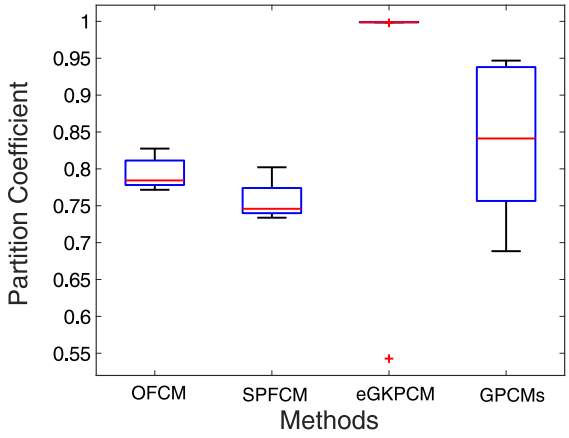


(b)

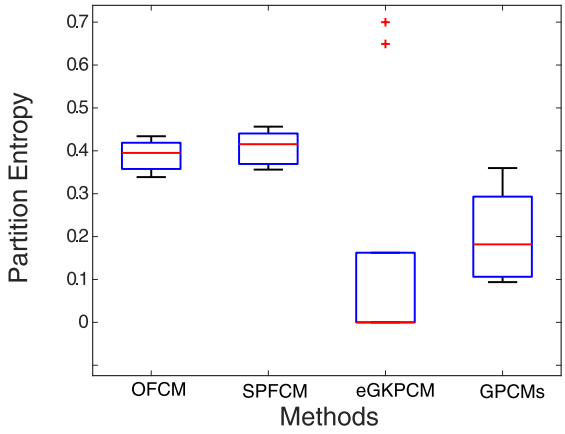


(c)

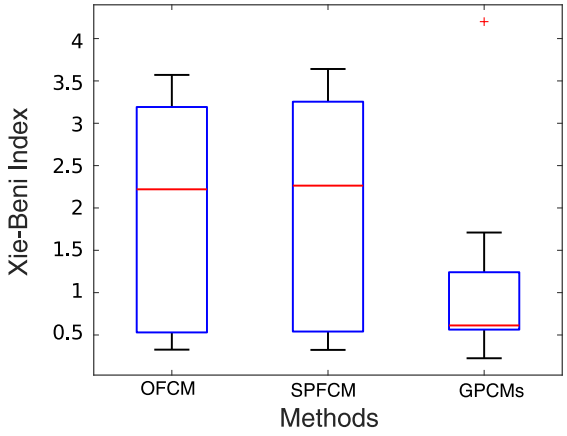
Figure 3: Quality indices of the fuzzy clustering methods on the Gaussian dataset: (a) PC, (b) PE, (c) XB.



(a)

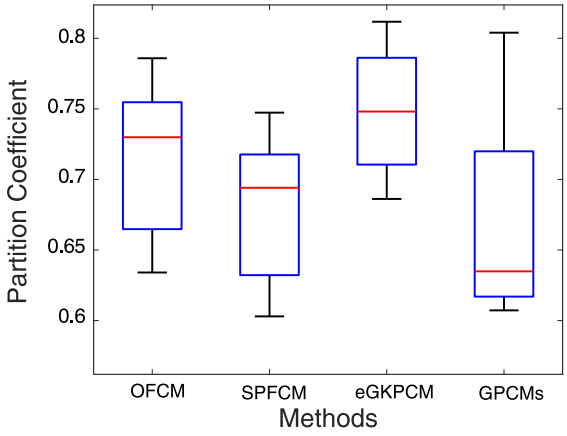


(b)

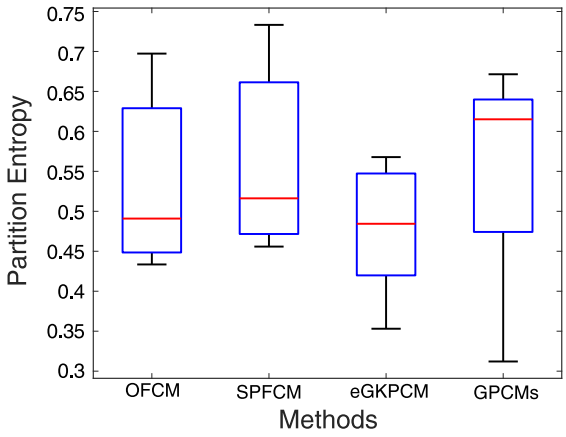


(c)

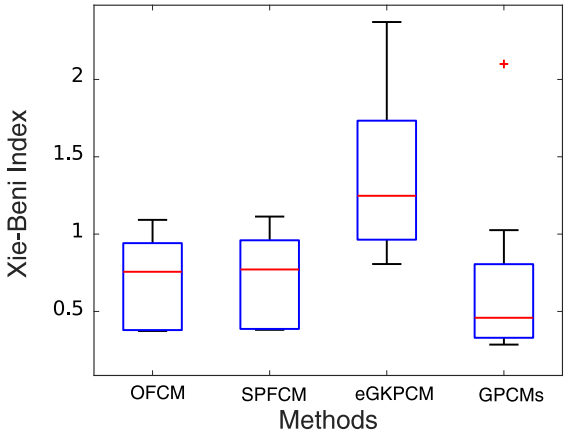
Figure 4: Quality indices of the fuzzy clustering methods on the Smtp (KDDCUP99) dataset: (a) PC, (b) PE, (c) XB.



(a)



(b)



(c)

Figure 5: Quality indices of the fuzzy clustering methods on the Soil moisture and temperature dataset: (a) PC, (b) PE, (c) XB.

5.3 RESULTS

Figures 3, 4 and 5 show the results on the Gaussian, Smtp, and Soil datasets, respectively, in the form of box plots as previously described. (In the figures GPCM stream appears shortened as GPCMs for space reasons.)

For the Gaussian dataset, the input data were organised as follows: For OFCM, SPFCM and eGKPCM they were split in chunks of size $s = 800$. For GPCM stream the same size was used for initializing the centroids, then the chunk size was reduced to $s = 1$ for by-object learning.

Figure 3 shows that according to the PC and PE indexes the fuzzy possibilistic models outperform the probabilistic ones when dealing with non-stationary data. According to XB the GPCM model experiences a higher variability of results. This is the only method that has the time resolution of one observation, so this behaviour is understandable and may indicate a higher sensitivity of the XB index with respect to the other two.

Regarding the experiments on the Smtp (KDDCUP99) dataset, here the chunk size selected is $s = 3000$ and the number of clusters was fixed at 4.

Figure 4 shows the performance drop in OFCM and SPFCM in the presence of outliers in data streams. eGKPCM shows a very low separation between clusters and its XB value was not even included in the figure. According to the three indexes GPCM stream appears to be the most suitable method in the presence of outliers.

Finally, with the Soil moisture and temperature dataset, the selected chunk size is $s = 800$ and the number of clusters was fixed at 4.

The PC and PE indexes do not reveal strong differences between the four methods, while XB seems to indicate GPCM stream as the method that achieves the best top performance, although with the largest variation; again this is understandable, given that it is the only method that deals with the stream object by object rather than chunk by chunk, so pointwise variations are not averaged out within chunks.

6 DISCUSSION

6.1 SUMMARY OF METHODS PROPERTIES

In table 1 we summarize the properties of some clustering algorithms, of both the fuzzy and non-fuzzy types, showing the computational complexities of each one. In the table the assumption is made that the number of iterations required for termination is fixed. The following parameters are used: n is the number of objects in the s-dimensional chunk; c is the number of clusters; m is the number of micro-clusters for Den-Stream; for the grid-based clustering algorithm (DUCstream) c_b is the result in bits and g the number of grid points.

The columns contain the acronym of the clustering method (Method), the basic clustering technique on which the stream method is based (Type), the input organisation, by chunks or online (Input), the expected shape of clusters, the degree of sensitivity to concept changes, and finally the time complexity.

About computational complexity, we remark that the convergence of all these optimization procedures depends on the data and particularly on source dynamics (drift and shift), so the time required to reach a stable state is difficult to estimate in general terms. In addition, computational complexity is defined as an asymptotic limit. It is only useful for comparing the rate of variation of computational time with respect

[2] https://github.com/gditzler/ConceptDriftData
[3] http://archive.ics.uci.edu/ml/datasets/kdd+cup+1999+data
[4] https://wcc.sc.egov.usda.gov/nwcc/site?sitenum=2026

Method	Type	Input	Cluster shape	Concept change	Time complexity
BIRCH	k-means	Chunks	hyper-sphere	Low	$O(ncd)$
CluStream	k-means	Chunks	hyper-sphere	Low	$O(ncd)$
Den-Stream	DBSCAN	Chunks	arbitrary	Moderate	$O(m)$
DD-STREAM	DBSCAN	Chunks	arbitrary	Moderate	$O(g^2)$
DUCstream	Density grid-based	Chunks	arbitrary	Moderate	$O(c_b)$
OFCM	Fuzzy C-Means	Chunks	hyper-sphere	Moderate	$O(c^2dn)$
sWFCM	Fuzzy C-Means	Chunks	hyper-sphere	Moderate	$O(c^2dn)$
SPFCM	Fuzzy C-Means	Chunks	hyper-sphere	Moderate	$O(c^2dn)$
SFCM	Fuzzy C-Means	Chunks	hyper-sphere	Moderate	$O(c^2dn)$
HOFCMD	Fuzzy C-Medoids	Chunks	hyper-sphere	High	$O(n^2d)$
OFCMD	Fuzzy C-Medoid	Chunks	hyper-sphere	High	$O(n^2d)$
IMMFC	Fuzzy C-Medoid	Chunks	hyper-sphere	High	$O(n^2d)$
wPCM stream	Possibilistic c-means	Chunks	hyper-sphere	Moderate	$O(c^2dn)$
TRAC-STREAMS	Possibilistic c-means	Chunks and by pattern	hyper-sphere	Moderate	$O(w^2dn)$
GPCM stream	Possibilistic c-means	Chunks and by pattern	hyper-sphere	High	$O(c^2dn)$

Table 1: Comparative analysis of data stream clustering algorithms.

to variations in problem size, not the actual computational time. This makes it only indirectly relevant since, even when re-clustering is performed, the data are always split into smaller chunks. The details of the implementation (for instance, compiled versus interpreted software; integer versus float; single precision versus double precision; data acquisition and storage overhead) and the selected chunk length may be more important than the asymptotic behaviour.

To help in the selection of the most suitable method, the following guidelines may be adopted:

- When irregular cluster shapes are expected, density-based and grid-based method may be preferred.
- If non-stationarity is expected, methods with a higher ability to track concept change are required. For strongly non-stationary streams, methods able to work by-pattern may ensure faster tracking.
- Soft clustering (fuzzy or possibilistic) is to be preferred when concept drift is prevalent over concept shift and the user wants to avoid discontinuities in cluster memberships. This may happen with crisp clustering when observations suddenly switch from a cluster to a neighbouring one.
- Asymmetric clusters can be dealt with by medoid-based methods.
- To counterbalance the effect of working in high dimensionality d[41,61], one strategy consists in using order statistics rather than means[60]. This translates again into a preference for medoid-based methods.
- Computational complexity should be used only as a general estimation; for time-critical problems, e.g., in control applications with real-time requirements, software profiling on the actual data should be used.

While a good practice in data science is to try different methods with the same dataset, we remark that the literature also offers **ensemble** or **consensus clustering** frameworks[24,45] that may be adapted to the stream setting to achieve more consistent clustering results.

6.2 SOURCES OF UNCERTAINTY IN DATA STREAM CLUSTERING

Several crisp methods and all fuzzy methods considered in this survey are based on the implicit assumption that the number of clusters c is known, or at least that it can be reasonably estimated. This is a common

trait of all centroid-based methods, and many cluster validation methods have been developed to help in this estimation.

In general, the stream clustering task is confronted with two main sources of uncertainty:

1. The nature of the source is not known (tackled by the task of source modeling, or learning).

2. The nature of the source is not constant in time (tackled by the task of source tracking).

When a good model can be assumed for the source, for instance when it is known that clusters are convex, the source modeling problem can be split into **model selection** and **model fitting**. Any residual variation after these two tasks can be ascribed to random variability of the data, and analyzed for stationarity. If changes are detected, these may indicate concept change and the model can be updated to track the source. This may require finding a good balance between memory of the model (e.g., window size) and statistical variability of the data [65].

However, when the model selection and model identification problems are not easily split due to uncertainty on the model itself, then distinguishing between model effects and random effects is conceptually impossible.

In these cases, methods that do not rely on model identification can be adopted. These methods assume an unspecified probabilistic model, without distinguishing between model effects and noise, and check for stationarity by using distribution-independent inequalities.

As an example of this category we can mention the martingale framework. [32] This approach uses Doob's martingale inequality, a concentration inequality which holds pretty generally due to its mild hypotheses, to monitor a property termed exchangeability [66] that holds only when the "local i.i.d." assumption is satisfied.

Of the methods covered in this survey, those based on the possibilistic approach, namely, wPCM stream, TRAC-STREAMS, and GPCM stream, feature inherent robustness to outliers that favours model identification even when model selection is not optimal. TRAC-STREAMS and GPCM stream include explicit mechanisms to monitor stationarity.

6.3 OTHER APPROACHES

In this survey we have focused mainly on different types of fuzzy stream clustering. The literature about source change detection and tracking is vast and spans several different types of approaches. However, a large part of this literature is not directly applicable to clustering, since it focuses on supervised tasks and relies on a well-defined loss function that makes it easier to compute indicators of change.

The methods that we have described are all heuristic procedures. In principle, a number of methods with more solid theoretical grounds can be used instead of the fuzzy/possibilistic formalism. In the following discussions, we highlight some reasons why in the case of data streams these more principled solutions may, in practice, be less convenient.

Among the competing approaches available for the unsupervised case, we can distinguish between probabilistic and statistical methods. **Probabilistic** methods for source tracking, change detection, and outlier detection/rejection are based on explicit models of data density, for instance mixture models (Gaussian or other). Their advantage is clear provided that (1) the model is appropriate and (2) its parameters can be fitted reliably. Unfortunately, the validity of condition (1) is difficult to assess, unless prior knowledge is available. Failing that, the value of a mixture-based method is similar to that of a heuristic procedure, so it becomes essentially equivalent to a fuzzy probabilistic model. Coming to requisite (2), the main fitting algorithm, Expectation-Maximization (EM), may not converge very quickly. [52] In this case, too, heuristic algorithms may become equivalent, but simpler. Stochastic-approximation-type versions do exist. [15] However,

having to fit more parameters unavoidably requires more data than optimization of simpler cluster models. This limits its applicability to cases where the "local i.i.d." assumption holds on extended intervals, i.e., cases with slow concept change.

One particular approach within the probabilistic family is Bayesian filtering. The Kalman filter has been employed in this task,[42] resulting in a speed-up of one order of magnitude with respect to EM iterations. However this is exactly applicable only when source change follows a linear dynamics plus Gaussian evolution. Discontinuity, i.e., concept shift, is difficult to deal with, unless, again, prior knowledge about the possible evolutions is available. In this case one option (with drawbacks similar to the general Bayesian filtering approach) is particle filtering[73] on an explicit model of shift, e.g., a Markov model.

Regarding **statistical** methods, these have the advantage that they may be non-parametric, that is, they may not require the identification of a probabilistic model. The statistics used are indicators of change computed on (suitably-sized) samples. A basic but common example of this class of methods is the CUSUM criterion,[56] as well as other forms of sequential analysis.[26] Concentration inequalities can also be used (this is also done in TRAC-STREAMS with a Chebychev bound), but these have the disadvantage of becoming trivial, and therefore of little use, when the quantity of data considered is not sufficient.

Other, more sophisticated statistics are based on permutation tests,[29] multi-dimensional variants of statistical tests,[25] distances between density estimates.[5] These are often computationally expensive, requiring the construction of combinatorial graphs or the computation of complete density estimates.

Some of these indicators are sensitive only to some types of changes. For instance, CUSUM in its basic form only detects unidirectional changes in a scalar estimator. Finally, the focus of most of these tests is in detecting concept shift, not a trend that indicates concept drift.

It should be noted that all of the cited methods compare stationary distributions, and usually the theory behind them does not cover damping factors or forgetting weights to model explicitly the gradual loss of validity in time. This makes the "local i.i.d." assumption even more critical.

As a conclusion, despite being only heuristically motivated, fuzzy approaches actually constitute an interesting solution to the problem, especially under resource and time constraints and in the absence of detailed knowledge about the source and its evolution.

7 CONCLUSIONS

This survey has covered the available methods for clustering data streams with a particular focus on fuzzy models. While not all fuzzy methods have the important properties of non-stationarity tracking and outlier rejection, fuzzy models offer the potentiality to implement these properties in a *continuous* fashion. Some of the methods presented exploit this potentiality to cluster data streams in an uninterrupted fashion.

In particular, the possibilistic model, if appropriate constraints are used in training to make it converge properly, possesses intrinsic robustness properties which make it a very effective basis for tracking model changes while being insensitive to outliers.

8 SOFTWARE AVAILABILITY

For OFCM and SPFCM the C implementation that was used for this study is available at:
http://www.csee.usf.edu/~lohall/scalable/
For GPCM stream the following MATLAB implementation is available:
https://it.mathworks.com/matlabcentral/fileexchange/64318-onlinegradedpossibilisticclustering

9 ACKNOWLEDGEMENTS

The authors would like to acknowledge the contribution of COST Action IC1406 High-Performance Modelling and Simulation for Big Data Applications (cHiPSet).

Part of the work was done under the project *Piattaforma per la mobiLità Urbana con Gestione delle INformazioni da sorgenti eterogenee* (PLUG-IN) funded by the Italian Ministry of Education, University and Research.

Amr Abdullatif holds a doctoral fellowship from the University of Genova, Italy.

References

1. Amr Abdullatif, Francesco Masulli, and Stefano Rovetta. Tracking time evolving data streams for short-term traffic forecasting. *Data Science and Engineering*, Oct 2017. ISSN 2364-1541.

2. A. Abdullatif, F. Masulli, S. Rovetta, and A. Cabri. *Graded Possibilistic Clustering of Non-stationary Data Streams*, pages 139–150. Springer International Publishing, Cham, 2017. ISBN 978-3-319-52962-2. doi: 10.1007/978-3-319-52962-2_12.

3. Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '03, pages 81–92. VLDB Endowment, 2003. ISBN 0-12-722442-4.

4. Amineh Amini, TehYing Wah, and Hadi Saboohi. On density-based data streams clustering algorithms: A survey. *Journal of Computer Science and Technology*, 29(1):116–141, 2014. ISSN 1000-9000. doi: 10.1007/s11390-014-1416-y.

5. Niall H Anderson, Peter Hall, and D Michael Titterington. Two-sample test statistics for measuring discrepancies between two multivariate probability density functions using kernel-based density estimates. *Journal of Multivariate Analysis*, 50(1):41–54, 1994.

6. Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, SIGMOD '99, pages 49–60, New York, NY, USA, 1999. ACM. ISBN 1-58113-084-8.

7. M. Barni, V. Cappellini, and A. Mecocci. Comments on "a possibilistic approach to clustering". *IEEE Transactions on Fuzzy Systems*, 4(3):393–396, Aug 1996. ISSN 1063-6706. doi: 10.1109/91.531780.

8. James C. Bezdek. Cluster validity with fuzzy sets. *Journal of Cybernetics*, 3(3):58–73, 1973. doi: 10.1080/01969727308546047.

9. J. C. Bezdek. Numerical taxonomy with fuzzy sets. *Journal of Mathematical Biology*, 1(1):57–71, May 1974. ISSN 1432-1416.

10. James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981. ISBN 0306406713.

11. James C. Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers and Geosciences*, 10(2):191 – 203, 1984. ISSN 0098-3004. doi: http://dx.doi.org/10.1016/0098-3004(84)90020-7.

12. Fazli Can. Incremental clustering for dynamic information processing. *ACM Trans. Inf. Syst.*, 11(2):143–164, April 1993. ISSN 1046-8188. doi: 10.1145/130226.134466.

13. F. Can and II Drochak, N.D. Incremental clustering for dynamic document databases. In *Applied Computing, 1990., Proceedings of the 1990 Symposium on*, pages 61–67, Apr 1990. doi: 10.1109/SOAC.1990.82141.

14. Feng Cao, Martin Ester, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *In 2006 SIAM Conference on Data Mining*, pages 328–339, 2006.

15. Gilles Celeux, Didier Chauveau, and Jean Diebolt. Stochastic versions of the em algorithm: an experimental study in the mixture case. *Journal of Statistical Computation and Simulation*, 55(4):287–314, 1996. doi: 10.1080/00949659608811772.

16. Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 626–635, New York, NY, USA, 1997. ACM. ISBN 0-89791-888-6. doi: 10.1145/258533.258657.

17. G. Ditzler and R. Polikar. Incremental learning of concept drift from streaming imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 25(10):2283–2301, Oct 2013. ISSN 1041-4347. doi: 10.1109/TKDE.2012.136.

18. D. L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality, plenary lecture. In *Mathematical Challenges of the 21st Century*. AMS, 2000.

19. R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973. ISBN 9780471223610.

20. Steven Eschrich, Jingwei Ke, Lawrence O Hall, and Dmitry B Goldgof. Fast accurate fuzzy clustering through data reduction. *IEEE transactions on fuzzy systems*, 11(2):262–270, 2003.

21. Martin Ester, Hans peter Kriegel, Jorg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231. AAAI Press, 1996.

22. Maurizio Filippone, Francesco Camastra, Francesco Masulli, and Stefano Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recognition*, 41(1):176 – 190, 2008. ISSN 0031-3203. doi: http://dx.doi.org/10. 1016/j.patcog.2007.05.018.

23. Maurizio Filippone, Francesco Masulli, and Stefano Rovetta. Applying the possibilistic c-means algorithm in kernel-induced spaces. *IEEE Transactions on Fuzzy Systems*, 18:572–584, June 2010. ISSN 1063-6706.

24. Ana L. N. Fred and Anil K. Jain. Data clustering using evidence accumulation. *Pattern Recognition, International Conference on*, 4, 2002. ISSN 1051-4651. doi: http://dx.doi.org/10.1109/ICPR.2002.1047450.

25. Jerome H. Friedman and Lawrence C. Rafsky. Multivariate generalizations of the wald-wolfowitz and smirnov two-sample tests. *The Annals of Statistics*, 7(4):697–717, 1979. ISSN 00905364.

26. BK Ghosh. *Sequential tests of statistical hypotheses*. Addison-Wesley, 1970.

27. Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams: Theory and practice. *IEEE Trans. on Knowl. and Data Eng.*, 15(3):515–528, March 2003. ISSN 1041-4347. doi: 10.1109/TKDE.2003.1198387.

28. Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: An efficient clustering algorithm for large databases. *SIGMOD Rec.*, 27(2):73–84, June 1998. ISSN 0163-5808.

29. Peter Hall and Nader Tajvidi. Permutation tests for equality of distributions in high-dimensional settings. *Biometrika*, 89(2):359–374, 2002.

30. Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 291–300, New York, NY, USA, 2004. ACM. ISBN 1-58113-852-0.

31. Alexander Hinneburg, Er Hinneburg, and Daniel A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *KDD*, pages 58–65. AAAI Press, 1998.

32. S. S. Ho and H. Wechsler. A martingale framework for detecting changes in data streams by testing exchangeability. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2113–2127, 2010. ISSN 0162-8828.

33. P. Hore, L. O. Hall, and D. B. Goldgof. A fuzzy c means variant for clustering evolving data streams. In *2007 IEEE International Conference on Systems, Man and Cybernetics*, pages 360–365, Oct 2007. doi: 10.1109/ICSMC.2007.4413710.

34. P. Hore, L. O. Hall, and D. B. Goldgof. Single pass fuzzy c means. In *2007 IEEE International Fuzzy Systems Conference*, pages 1–7, July 2007. doi: 10.1109/FUZZY.2007.4295372.

35. P. Hore, L. O. Hall, D. B. Goldgof, and W. Cheng. Online fuzzy c means. In *NAFIPS 2008 - 2008 Annual Meeting of the North American Fuzzy Information Processing Society*, pages 1–5, May 2008. doi: 10.1109/NAFIPS.2008. 4531233.

36. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, September 1999. ISSN 0360-0300. doi: 10.1145/331499.331504.

37. Maciej Jaworski, Piotr Duda, and Lena Pietruczuk. *On Fuzzy Clustering of Data Streams with Concept Drift*, pages 82–91. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-29350-4. doi: 10.1007/ 978-3-642-29350-4 10.

38. Chen Jia, Chengyu Tan, and Ai Yong. A grid and density-based clustering algorithm for processing data stream. In *Genetic and Evolutionary Computing, 2008. WGEC '08. Second International Conference on*, pages 517–521, Sept 2008. doi: 10.1109/WGEC.2008.32.

39. George Karypis. *CLUTO: A Clustering Toolkit*, 2002.

40. Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 1 edition, 2005. ISBN 0471735787.

41. Frank Klawonn, Frank Höppner, and Balasubramaniam Jayaram. What are clusters in high dimensions and are they difficult to find? In *International Workshop on Clustering High-Dimensional Data*, pages 14–33. Springer, 2012.

42. Georg Krempl, Zaigham Faraz Siddiqui, and Myra Spiliopoulou. *Online Clustering of High-Dimensional Trajectories under Concept Drift*, pages 261–276. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-23783-6.

43. R. Krishnapuram and J. M. Keller. A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98–110, May 1993. ISSN 1063-6706. doi: 10.1109/91.227387.

44. Raghu Krishnapuram and James M. Keller. The possibilistic *C*-Means algorithm: insights and recommendations. *IEEE Transactions on Fuzzy Systems*, 4(3):385–393, August 1996.

45. Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

46. Nicolas Labroche. Online fuzzy medoid based clustering algorithms. *Neurocomput.*, 126:141–150, February 2014. ISSN 0925-2312. doi: 10.1016/j.neucom.2012.07.057.

47. Liu Li-xiong, Huang Hai, Guo Yun-fei, and Chen Fu-Cai. rdenstream, a clustering algorithm over an evolving data stream. In *Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on*, pages 1–4, Dec 2009. doi: 10.1109/ICIECS.2009.5363379.

48. Sebastian Lühr and Mihai Lazarescu. Incremental clustering of dynamic data streams using connectivity based representative points. *Data Knowl. Eng.*, 68(1):1–27, January 2009. ISSN 0169-023X. doi: 10.1016/j.datak.2008.08.006.

49. Francesco Masulli and Stefano Rovetta. Clustering high-dimensional data. In Francesco Masulli, Alfredo Petrosino, and Stefano Rovetta, editors, *Clustering High–Dimensional Data: First International Workshop, CHDD 2012, Naples, Italy, May 15, 2012, Revised Selected Papers*, pages 1–13. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

50. F. Masulli and S. Rovetta. Soft transition from probabilistic to possibilistic fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, 14(4):516–527, Aug 2006. ISSN 1063-6706. doi: 10.1109/TFUZZ.2006.876740.

51. S. Mostafavi and A. Amiri. Extending fuzzy c-means to clustering data streams. In *20th Iranian Conference on Electrical Engineering (ICEE2012)*, pages 726–729, May 2012. doi: 10.1109/IranianCEE.2012.6292449.

52. Iftekhar Naim and Daniel Gildea. Convergence of the em algorithm for gaussian mixtures with unbalanced mixing coefficients. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1655–1662, 2012.

53. Olfa Nasraoui and Carlos Rojas. Robust clustering for tracking noisy evolving data streams. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 619–623. SIAM, 2006. doi: 10.1137/1. 9781611972764.72.

54. Raymond T. Ng and Jiawei Han. Clarans: A method for clustering objects for spatial data mining. *IEEE Trans. on Knowl. and Data Eng.*, 14(5):1003–1016, September 2002. ISSN 1041-4347.

55. Ratko Orlandic, Ying Lai, and Wai Gen Yee. Clustering high-dimensional data using an efficient and effective data space reduction. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 201–208, New York, NY, USA, 2005. ACM. ISBN 1-59593-140-6.

56. E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954. ISSN 00063444.

57. N. R. Pal and J. C. Bezdek. On cluster validity for the fuzzy c-means model. *IEEE Transactions on Fuzzy Systems*, 3(3):370–379, Aug 1995. ISSN 1063-6706.

58. Georg Peters and Richard Weber. Dcc: a framework for dynamic granular clustering. *Granular Computing*, 1 (1):1–11, Mar 2016. ISSN 2364-4974.

59. Kenneth Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of IEEE*, 86(11):2210–2239, November 1998.

60. Stefano Rovetta and Francesco Masulli. Shared farthest neighbor approach to clustering of high dimensionality, low cardinality data. *Pattern Recognition*, 39(12):2415–2425, December 2006. doi: 10.1016/ j.patcog.2006.06.021.

61. Stefano Rovetta and Francesco Masulli. Comparing fuzzy clusterings in high dimensionality. In Francesco Masulli, Alfredo Petrosino, and Stefano Rovetta, editors, *Clustering High–Dimensional Data: First International Workshop, CHDD 2012, Naples, Italy, May 15, 2012, Revised Selected Papers*, pages 50–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

62. A. Schneider. Weighted possibilistic c-means clustering algorithms. In *Ninth IEEE International Conference on Fuzzy Systems. FUZZ- IEEE 2000 (Cat. No.00CH37063)*, volume 1, pages 176–180 vol.1, May 2000. doi: 10.1109/FUZZY.2000.838654.

63. Igor krjanc and Dejan Dovan. Evolving gustafson-kessel possibilistic c-means clustering. *Procedia Computer Science*, 53:191 – 198, 2015. ISSN 1877-0509. doi: http://dx.doi.org/10.1016/j.procs.2015.07.294. INNS Confe-rence on Big Data 2015 Program San Francisco, CA, USA 8-10 August 2015.

64. Jonathan A. Silva, Elaine R. Faria, Rodrigo C. Barros, Eduardo R. Hruschka, André C. P. L. F. de Carvalho, and João Gama. Data stream clustering: A survey. *ACM Comput. Surv.*, 46(1):13:1–13:31, July 2013. ISSN 0360-0300. doi: 10.1145/2522968.2522981.

65. Katharina Tschumitschew and Frank Klawonn. Effects of drift and noise on the optimal sliding window size for data stream regression models. *Communications in Statistics-Theory and Methods*, 46(10):5109–5132, 2017.

66. Vladimir Vovk, Ilia Nouretdinov, and Alexander Gammerman. Testing exchangeability on-line. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 768–775, 2003.

67. R. Wan, X. Yan, and X. Su. A weighted fuzzy clustering algorithm for data stream. In *2008 ISECS International Colloquium on Computing, Communication, Control, and Management*, volume 1, pages 360–364, Aug 2008. doi: 10.1109/CCCM.2008.186.

68. Y. Wang, L. Chen, and J. P. Mei. Incremental fuzzy clustering with multiple medoids for large data. *IEEE Transactions on Fuzzy Systems*, 22(6):1557–1568, Dec 2014. ISSN 1063-6706. doi: 10.1109/TFUZZ.2014.2298244.

69. Roland Winkler, Frank Klawonn, and Rudolf Kruse. Problems of fuzzy c-means clustering and similar algorithms with high dimensional data sets. In *Challenges at the Interface of Data Analysis, Computer Science, and Optimization*, pages 79–87. Springer, 2012.

70. X. L. Xie and G. Beni. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):841–847, Aug 1991. ISSN 0162-8828.

71. Rui Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, May 2005. ISSN 1045-9227. doi: 10.1109/TNN.2005.845141.

72. Miin-Shen Yang and Kuo-Lung Wu. Unsupervised possibilistic clustering. *Pattern Recognition*, 39(1):5 – 21, 2006. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2005.07.005.

73. Jie Yu. A particle filter driven dynamic gaussian mixture model approach for complex process monitoring and fault diagnosis. *Journal of Process Control*, 22(4):778 – 788, 2012. ISSN 0959-1524. doi: https://doi.org/10. 1016/j.jprocont.2012.02.012.

74. Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, SIGMOD '96, pages 103–114, New York, NY, USA, 1996. ACM. ISBN 0-89791-794-4.