Vietnam J Comput Sci (2017) 4:171–183
DOI 10.1007/s40595-016-0086-9

REGULAR PAPER

Using internal evaluation measures to validate the quality of diverse stream clustering algorithms

Marwan Hassani[1] · **Thomas Seidl**[2]

Received: 23 December 2015 / Accepted: 30 September 2016 / Published online: 14 October 2016

Abstract Measuring the quality of a clustering algorithm has shown to be as important as the algorithm itself. It is a crucial part of choosing the clustering algorithm that performs best for an input data. Streaming input data have many features that make them much more challenging than static ones. They are endless, varying and emerging with high speeds. This raised new challenges for the clustering algorithms as well as for their evaluation measures. Up till now, external evaluation measures were exclusively used for validating stream clustering algorithms. While external validation requires a ground truth which is not provided in most applications, particularly in the streaming case, internal clustering validation is efficient and realistic. In this article, we analyze the properties and performances of eleven internal clustering measures. In particular, we apply these measures to carefully synthesized stream scenarios to reveal how they react to clusterings on evolving data streams using both *k*-means-based and density-based clustering algorithms. A series of experimental results show that different from the case with static data, the *Calinski-Harabasz index* performs the best in coping with common aspects and errors of stream clustering for *k*-means-based algorithms, while the *revised validity index* performs the best for density-based ones.

Keywords Stream clustering · Internal evaluation measures · Clustering · Validation · MOA

✉ Marwan Hassani
m.hassani@tue.nl

Thomas Seidl
seidl@dbs.ifi.lmu.de

[1] Architecture of Information Systems Group, Eindhoven University of Technology, Eindhoven, The Netherlands

[2] Database Systems Group, LMU Munich, Munich, Germany

1 Introduction

Clustering of data objects is a well-established data mining task that aims at grouping these objects. The grouping is made such that *similar* objects are aggregated together in the same group (or cluster) while *dissimilar* ones are grouped in different clusters. In this context, the definition of similarity, and thus the final clustering is highly dependent on the applied distance function between the data objects. Different to classification, clustering does not use a subset of the data objects with known class labels to learn a classification model. As a completely unsupervised task, clustering calculates the similarity between objects without having any information about their correct distribution (also known as the ground truth). The latter fact motivated the research in the field of clustering validation notably more than the field of classification evaluation. It has been even stated that clustering validation is regarded as important as the clustering itself [32].

There are two types of clustering validation [31]. The *external* validation, which compares the clustering result to a reference result which is considered as the ground truth. If the result is somehow similar to the reference, we regard this final output as a "good" clustering. This validation is straightforward when the similarity between two clusterings has been well-defined, however, it has fundamental caveat that the reference result is not provided in most real applications. Therefore, external evaluation is largely used for synthetic data and mostly for tuning clustering algorithms.

Internal validation is the other type clustering evaluation, where the evaluation of the clustering is compared only with the result itself, i.e., the structure of found clusters and their relations to each other. This is much more realistic and efficient in many real-world scenarios as it does not refer to any assumed references from outside which is not always fea-

sible to obtain. Particularly, with the huge increase of the data size and dimensionality as in recent applications with streaming data outputs, one can hardly claim that a complete knowledge of the ground truth is available or always valid.

Obviously, clustering evaluation is a stand-alone process that is not included within clustering task. It is usually performed after the final clustering output is generated. However, internal evaluation methods have been used in the validation phase within some clustering algorithms like k-means [29], k-medoids [26], EM [8] and k-center [19].

Stream clustering deals with evolving input objects where the distribution, the density and the labels of objects are continuously changing [16]. Whether it is high-dimensional stream clustering [14,24], hierarchical stream clustering [15,23] or sensor data clustering [19–21], evaluating the clustering output using external evaluation measures (like SubCMM [17,18]) requires a ground truth that is very difficult to obtain in the above-mentioned scenarios.

For the previous reasons, we focus in this article on the internal clustering validation and study its usability for drifting streaming data. To fairly discuss the ability of internal measures to validate the quality of different types of stream clustering algorithms. We expand the study to cover both a k-means-based stream clustering algorithm [1] as well as a density-based stream clustering one [6]. This is mainly motivated by the fact that those algorithms are good representatives of the two main different categories of stream clustering algorithms.

The remainder of this article is organized as follows: Sect. 2 examines some popular criteria of deciding whether found clusters are valid, and the general procedure we used in this article to evaluate stream clustering. In Sect. 3, we list eleven different mostly used internal evaluation measures and shortly show how they are actually exploited in clustering evaluation. In Sect. 4, we introduce a set of thorough experiments on different kinds of data streams with different errors to show the behaviors of these internal measures in practice with a k-means-based stream clustering algorithm. In addition, we investigate more concretely how the internal measures react to stream-specific properties of data. To do this, several common error scenarios in stream clusterings are simulated and also evaluated with internal clustering validation. In Sect. 5, the internal evaluation measures are again used to validate a density-based stream clustering. This is done by first extracting a "ground truth" of the clustering quality using external evaluation measures and then checking which of the internal measures has the highest correlation with that ground truth. Finally, in Sect. 6, we summarize the contents of this article.

This article further discusses the initial technical results introduced in [22] and extends them by elaborating the algorithmic description in Sect. 2, enriching the results in Sect. 4 and introducing Sect. 5 completely.

2 Internal clustering validation

In this section, we describe our concept of internal clustering validation and how they are realized for existing internal validation measures. Additionally, we will show an abstract procedure to make use of these measures in streaming environments in practice.

2.1 Validation criteria

Contrary to external validation, internal clustering validation is based only on the intrinsic information of the data. Since we can only refer to the input dataset itself, internal validation needs assumptions about a "good" structure of found clusters which are normally given by reference result in external validation. Two main concepts, the compactness and the separation, are the most popular ones. Most other concepts are actually just combinations of variations of these two [34].

The *Compactness* measures how closely data points are grouped in a cluster. Grouped points in the cluster are supposed to be related to each other, by sharing a common feature which reflects a meaningful pattern in practice. Compactness is normally based on distances between in-cluster points. The very popular way of calculating the compactness is through variance, i.e., average distance to the mean, to estimate how objects are bonded together with its mean as its center. A small variance indicates a high compactness (cf. Fig. 1). Quantitatively, one way of calculating the compactness using the average distance is explained in Eq. 1.

The *Separation* measures how different the found clusters are from each other. Users of clustering algorithms are not interested in similar or vague patterns when clusters are not well-separated (cf. Fig. 2). A distinct cluster that is far from the others corresponds to a unique pattern. Similar to the compactness, the distances between objects are widely used to measure separation, e.g., pairwise distances between cluster centers, or pairwise minimum distances between objects in different clusters. Separation is an inter-cluster criterion in the sense of relation between clusters. An example of how to quantitatively calculate the separation using the average distance is explained in Eq. 2.

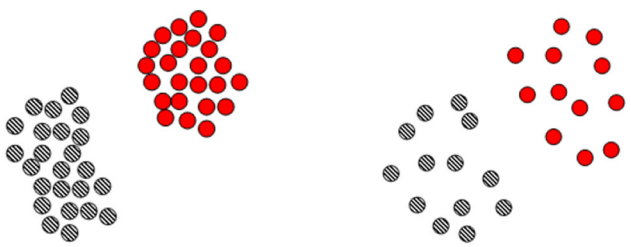

Fig. 1 Clusters on the *left* have better compactness than the ones on the *right*

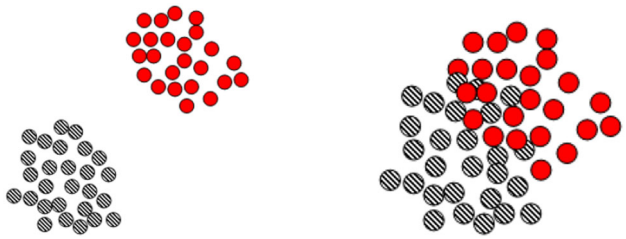

Fig. 2 Clusters on the *left* have better separation than the ones on the *right*

2.2 General procedure

Using a carefully generated synthetic data set, where we know the underlying partitioning and the distribution of the data, we apply the internal validation measures using different parameters of the clustering algorithms. The target is now to observe which of the evaluation measures is reaching its best value when setting the parameters of the selected clustering algorithm to best reflect the distribution of the data set. We collect the values of the internal measures for each batch, and finally average the values of all batches. An abstract procedure of this process is listed in Algorithm 1. The algorithm explains both cases of a k-means-based algorithm and a DBSCAN-based algorithm.

Algorithm 1: InternalValidationProcedure()

```
Prepare the current stream batch from the dataset
initialize the clustering algorithm (a k-means-based or a
DBSCAN-based one);
initialize a set 𝒯 of all combinations of meaningful ranges for
each parameter;
foreach parameter setting ps ∈ 𝒯 do
    Run the selected clustering algorithm with the parameter
    setting ps;
    foreach batch in the stream do
        Compute the corresponding internal validation index of
        the clustering output;
    end
    Average the clustering quality of the validation index over all
    batches from the stream;
end
if the current algorithm is k-means based then
    Check which index is reaching its best values with the correct
    number of generated clusters k in the data set;
end
else
    Check which parameter setting ps_i ∈ 𝒯 causes best values of
    external evaluation measures over the current
    DBSCAN-based algorithm;
    Check which internal index has the highest correlation with
    the external measures w.r.t. ps_i;
end
```

3 Considered existing internal evaluation measures

In this section, we briefly review the most used eleven internal clustering measures in recent works. One can easily figure out of each measure which design criteria is chosen and how they are realized in mathematical form. We will first introduce important notations used in the formula of these measures: D is the input dataset, n is the number of points in D, g is the center of whole dataset D, P is the number of dimensions of D, NC is the number of clusters, Ci is the i-th cluster, n_i is the number of data points in C_i, c_i is the center of cluster C_i, $\sigma(C_i)$ is the variance vector of C_i, and $d(x, y)$ is the distance between points x and y. For the convenience, we will put an abbreviation for each measure and use it through the rest of this article.

First, some measures are designed to evaluate either only one of compactness or separation. The simplest one is the *Root-mean-square standard deviation (RMSSTD)*:

$$RMSSTD = \left(\frac{\sum_i \sum_{x \in C_i} \|x - c_i\|^2}{P \sum_i (n_i - 1)} \right)^{1/2} \tag{1}$$

This measure is the square root of the pooled sample variance of all the attributes, which measures only the compactness of found clusters [10]. Another measure which considers only the separation between clusters is the *R-squared (RS)* [10]:

$$RS = \frac{\sum_{x \in D} \|x - g\|^2 - \sum_i \sum_{x \in C_i} \|x - c_i\|^2}{\sum_{x \in D} \|x - g\|^2} \tag{2}$$

RS is the complement of the ratio of sum of squared distances between objects in different clusters to the total sum of squares. It is an intuitive and simple formulation of measuring the differences between clusters. Another measure considering only separation is the *Modified Hubert* Γ *statistic* (Γ) [25]:

$$\Gamma = \frac{2}{n(n-1)} \sum_{i,j \in \{1 \cdots NC\}, i \neq j} \sum_{x \in C_i} \sum_{y \in C_j} d(x, y) \cdot d(c_i, c_j) \tag{3}$$

Γ calculates the average weighted pairwise distances between data points belonging to different clusters by multiplying them by the distances between the centers of their clusters.

The following measures are designed to reflect both compactness and separation at the same time. Naturally, considering only one of the two criteria is not enough to evaluate complex clusterings. We will introduce first the *Calinski-Harabasz index (CH)* [5]:

$$CH = \frac{\sum_i d^2(c_i, g)/(NC-1)}{\sum_i \sum_{x \in C_i} d^2(x, c_i)/(n-NC)} \quad (4)$$

CH measures the two criteria simultaneously with the help of average between and within cluster sum of squares. The numerator reflects the degree of separation in the way of how much the cluster centers are spread, and the denominator corresponds to compactness, to reflect how close the in-cluster objects are gathered around the cluster center. The following two measures also share this type of formulation, i.e., numerator-separation/denominator-compactness. First, the *I index (I)* [30]:

$$I = \left(\frac{1}{NC} \frac{\sum_{x \in D} d(x, g)}{\sum_i \sum_{x \in C_i} d(x, c_i)} \max_{i,j} d(c_i, c_j) \right)^P \quad (5)$$

To measure separation, I adopts the maximum distance between cluster centers. For compactness, the distance from a data point to its cluster center is used like *CH*. Another famous measure is the *Dunn's indices (D)* [9]:

$$D = \frac{\min_i \min_j \left(\min_{x \in C_i, y \in C_j} d(x, y) \right)}{\max_k \left(\max_{x, y \in C_k} d(x, y) \right)} \quad (6)$$

D uses the minimum pairwise distance between points in different clusters as the inter-cluster separation and the maximum diameter among all clusters as the intra-cluster compactness. As mentioned above, *CH*, *I*, and *D* follow the form $(Separation)/(Compactness)$, though they use different distances and different weights of the two factors. The optimal cluster number can be achieved by maximizing these three indices.

Another commonly used measure is *Silhouette index (S)* [33]:

$$S = \frac{1}{NC} \sum_i \left(\frac{1}{n_i} \sum_{x \in C_i} \frac{b(x) - a(x)}{max[b(x), a(x)]} \right) \quad (7)$$

where $a(x) = \frac{1}{n_i - 1} \sum_{y \in C_i, y \neq x} d(x, y)$
and $b(x) = \min_{j \neq i} \left[\frac{1}{n_j} \sum_{y \in C_j} d(x, y) \right]$.

S does not take c_i or g into account and uses pairwise distance between all the objects in a cluster for numerating compactness $(a(x))$. Here, $b(x)$ measures the separation with the average distance of objects to alternative cluster, i.e., second closest cluster. *Davies-Bouldin index (DB)* [7] is an old but still widely used internal validation measure:

$$DB = \frac{1}{NC} \sum_i \max_{j \neq i} \frac{\frac{1}{n_i} \sum_{x \in C_i} d(x, c_i) + \frac{1}{n_j} \sum_{x \in C_j} d(x, c_j)}{d(c_i, c_j)} \quad (8)$$

DB uses intra-cluster variance and inter-cluster center distance to find the worst partner cluster, i.e., the closest most scattered one for each cluster. Thus, minimizing DB gives us the optimal number of clusters. The *Xie-Beni index (XB)* [35] is defined as:

$$XB = \frac{\sum_i \sum_{x \in C_i} d^2(x, c_i)}{n \cdot \min_{i \neq j} d^2(c_i, c_j)} \quad (9)$$

Apparently, the smaller the values of *XB*, the better the clustering quality. Along with *DB*, *XB* has a form of $(Compactness)/(Separation)$ which is the opposite of *CH*, *I*, and *D*. Therefore, it reaches the optimum clustering by being minimized. It defines the inter-cluster separation as the minimum square distance between cluster centers, and the intra-cluster compactness as the mean square distance between each data object and its cluster center.

In the following, we present more recent clustering validation measures. The *SD validity index (SD)* [12]:

$$SD = NCmax \cdot Scat(NC) + Dis(NC) \quad (10)$$

- NCmax is the maximum number of possible clusters
- $Scat(NC) = \frac{1}{NC} \sum_i \frac{\|\sigma(C_i)\|}{\|\sigma(D)\|}$
- $Dis(NC) = \frac{\max_{i,j} d(c_i, c_j)}{\min_{i,j} d(c_i, c_j)} \sum_i \left(\sum_j d(c_i, c_j) \right)^{-1}$

SD is composed of two terms; $Scat(NC)$ stands for the scattering within clusters and $Dis(NC)$ stands for the dispersion between clusters. Like *DB* and *XB*, *SD* measures the compactness with variance of clustered objects and separation with distance between cluster centers, but uses them in a different way. The smaller the value of *SD*, the better. A revised version of *SD* is *S_Dbw* [11]:

$$S\_Dbw = Scat(NC) + Dens\_bw(NC) \quad (11)$$

- $Dens\_bw(NC) = \frac{1}{NC(NC-1)} \sum_i \times \left(\sum_{j \neq i} \frac{\sum_{x \in C_i \cup C_j} f(x, u_{ij})}{\max\left(\sum_{x \in C_i} f(x, c_i), \sum_{x \in C_j} f(x, c_j) \right)} \right)$
- $f(x, y) = \begin{cases} 0 & \text{if } d(x, y) > \tau, \\ 1 & \text{otherwise.} \end{cases}$

where u_{ij} is the middle point of c_i and c_j, τ is a threshold to determine the neighbors approximated by the average standard deviation of cluster centers: $\tau = \frac{1}{NC} \sqrt{\sum_{i=1}^{NC} ||\sigma(c_i)||}$, and $Scat(NC)$ is the same as that of *SD*. *S_Dbw* takes the density into account to measure the separation between clusters. It assumes that for each pair of cluster centers, at least one of their densities should be larger than the density of their midpoint to be a "good" clustering. Both *SD* and *S_Dbw* indicate the optimal clustering when they are minimized.

4 Internal validation of stream clusterings

In this section, we evaluate the result of stream clustering algorithms with internal validation measures.

4.1 Robustness to conventional clustering aspects

The results on using internal evaluation measures for clustering static data with simple errors in [28] prove that the performance of the internal measures is affected by various aspects of input data, i.e., noise, density of clusters, skewness, and subclusters. Each measure of the discussed 11 evaluation measures reacts differently to those aspects. We perform more complex experiments than the ones in [28], this time on stream clusterings to see how the internal measures behave in real-time continuous data. We run the CluStream [1] clustering algorithm with different parameters, choose the optimal number of clusters according to the evaluation results, and compare it to the true number of clusters. According to [10], *RMSSTD*, *RS* and Γ have the property of monotonicity and their curves will have either an upwards or a downwards tendency towards the optimum when we monotonically increase (or decrease) the number of clusters (or the parameter at hand). The optimal value for each of these measures is at the shift point of their curves which is also known as the "elbow".

Streaming data has usually complex properties that are happening at the same time. The experiments in [28], however, are limited to very simple toy datasets reflecting only one clustering aspect at a time. To make it more realistic, we use a data stream reflecting five conventional clustering aspects at the same time.

4.1.1 Experimental settings

To simulate streaming scenarios, we use *MOA (Massive Online Analysis)* [4] framework. We have chosen *RandomRBFGenerator*, which emits data instances continuously from a set of circular clusters, as the input stream generator (cf. Fig. 3). In this stream, we can specify the size, density, and moving speed of the instance-generating clusters, from which we can simulate the skewness, the different densities, and the subcluster aspect. We set the parameters as follows: number of generating clusters = 5, radius of clusters = 0.11, their dimensionality = 10, varying range of cluster radius = 0.07, varying range of cluster density = 1, cluster speed = 0.01 *per* 200 *points*, noise level = 0.1, noise does not appear inside clusters. The parameters which are not mentioned are not directly related to this experiment and are set to the default values of MOA.

For the clustering algorithm, we have chosen *CluStream* [1] with *k-means* as its macro-clusterer. We vary the parameter k from 2 to 9, where the optimal number of clusters is 5. We set the evaluation frequency to 1000 points and run our stream generator till 30000 points, which gives 30 evaluation results.

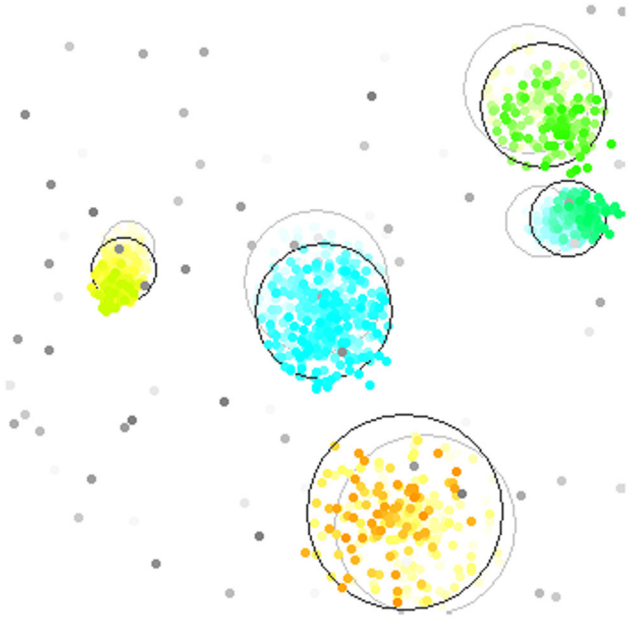

Fig. 3 A screenshot of the Dimensions 1 and 2 of the synthetic data stream used in the experiment. *Colored points* represent the incoming instances, and the *colors* are faded out as the processing time passes. Ground truth cluster boundaries are drawn in *black circle*. *Gray circles*indicate the former state, expressing that the clusters are moving. *Black* (faded out to *gray*) points represent noise points. (color figure online)

4.1.2 Results

Table 1 contains the mean value of 30 evaluation results which we obtained in the whole streaming interval. It shows that *RMSSTD*, *RS*, *CH*, *I*, and *S_Dbw* correctly reach their optimal number of clusters, while the others do not. According to the results in [28], the optimal value of each of *RMSSTD*, *RS*, and Γ is difficult to determine. For this reason, we do not accept their results even if some of them show a good performance.

In the static case in [28], *CH* and *I* were unable to find the right optimal number of clusters. *CH* is shown to be vulnerable to noise, since the noise inclusion (in cases when $k < 5$) makes the found clusters larger and less compact. However, in the streaming case, most clustering algorithms follow the online-offline-phases model. The online phase removes a lot of noise when summarizing the data into microclusters, and the offline phase (k-means in the case of CluStream [1]) deals only with these "cleaned" summaries. Of course, there will be always a chance to get a summary that is completely formed of noise points, but those will have less impact over the final clustering than the static case. Thus, since not all the noise points are integrated into the clusters, the amount of cluster expansion is a bit smaller than the static case.

Table 1 Evaluation results of internal validation on the stream clusterings

k	*RMSSTD*	*RS*	Γ	*CH*	*I*	*D*	*S*	*DB*	*XB*	*SD*	*S_Dbw*
2	0.0998	0.7992	0.3522	3196	0.4980	0.1921	**0.6535**	**0.5528**	0.1065	**4.5086**	0.2284
3	0.0763	0.8593	0.3724	3619	0.5564	**0.2208**	0.6003	0.5782	0.1561	6.1623	0.1601
4	0.0621	0.9117	0.3834	3860	0.5840	0.0936	0.6143	0.5531	**0.0932**	6.7154	0.1251
5	**0.0538**	**0.9330**	0.3967	**4157**	**0.6134**	0.0669	0.5855	0.5656	0.1143	8.6382	**0.1087**
6	0.0528	0.9355	**0.4007**	3510	0.4945	0.0309	0.5200	0.6360	0.1845	11.1729	0.1319
7	0.0481	0.9464	0.4002	3435	0.4697	0.0042	0.4861	0.6610	0.2580	14.7443	0.1192
8	0.0463	0.9512	0.4007	3095	0.4001	0.0099	0.4617	0.6853	0.2977	16.8715	0.1338
9	0.0430	0.9580	0.4026	3154	0.3943	0.0000	0.4544	0.6913	0.3085	19.5362	0.1355

The best obtained values for each parameter (not necessarily the maximum or the minimum) are in bold. The best values for *RMSSTD*, *RS* and Γ are selected as the first "elbow" in their monotonically increasing or decreasing curves (according to [10])

Therefore, the effect of noise to *CH* is less in the streaming case than the static one.

In the static case, *I* was slightly affected by the different densities of the clusters, and the reasons were not well revealed. Therefore, it is not surprising that *I* performs well as we take average of its evaluation results for the whole streaming interval.

The evaluation of *D* did not result with a very useful output, since it gives unconditional zero values in most evaluation points (before they are averaged as in Table 1). This is because the numerator of Equation (6) could be zero when at least one pair of x and y happens to be equal to each other, i.e., the distance between x and y is zero. This case rises when C_i and C_j are overlapped and the pair (x, y) is elected from the overlapped region. Streaming data has high possibility to have overlapped clusters, and so does the input of this experiment. This drives *D* to produce zero, making it an unstable measure in streaming environments.

Similar to the static data case, *S*, *DB*, *XB*, and *SD* perform bad in the streaming settings. The main reason also lies in the overlapping of clusters. Overlapping clusters are the extreme case of subclusters in the experiments of the static case discussed in [28].

4.2 Sensitivity to common errors of stream clustering

In this section, we perform a more detailed study on the behaviors of internal measures in streaming environments. The previous experiment is more or less a general test on a single data stream, so we use here the internal clustering indices on a series of elaborately designed experiments which well reflects the stream clustering scenarios. *MOA* framework has an interesting tool called *ClusterGenerator*, which can produce a found clustering by manipulating ground truth clusters with a certain error level. It can simulate different kinds of error types and even combine them to construct complicated clustering error scenarios. It is very useful since we can test the sensitivity of evaluation measures to specific errors [27].

Evaluating a variation of the ground truth seems a bit awkward in the sense of internal validation since it actually refers to the predefined result. However, this kind of experiment is absolutely meaningful, because we can watch reactions of internal measures to some errors of interest. [27] used this tool to show the behavior of internal measures, e.g., *S*, *Sum of Squares (SSQ)*, and *C-index*. Although the error types exploited in [27] are limited, those measures are not of our interest and already proved to be bad in the previous experiments.

4.2.1 Experimental Settings

Due to the drifting nature of streaming data, certain errors are common to appear for stream clustering algorithms. These errors are reflected by a wrong grouping of the drifting objects. The "correct" grouping of the objects is reflected in the original data set, where we assume that the real distribution of the objects (and thus the grouping) is previously known. This already known assignment of the drifting objects to their correct clusters is called *the ground truth*. The closer the output of a clustering algorithm to this ground truth, the better its quality. The above-mentioned errors are the deviations of the output of clustering algorithms from the ground truth. A good validation measure should be able to evaluate the amounts of these errors correctly. In the case of internal validation measures, this should be possible even without accessing the ground truth.

To obtain a controlled amount of this error, a simulation of a stream clustering algorithm is embedded in the MOA framework [4]. This previous explained simulation, called *ClusterGenerator*, allows the user to control the amount of deviations from the ground truth using different parameters. The *ClusterGenerator* has six error types as its parameters, and they effectively reflect common errors of stream clusterings. "Radius increase" and "Radius decrease" change the

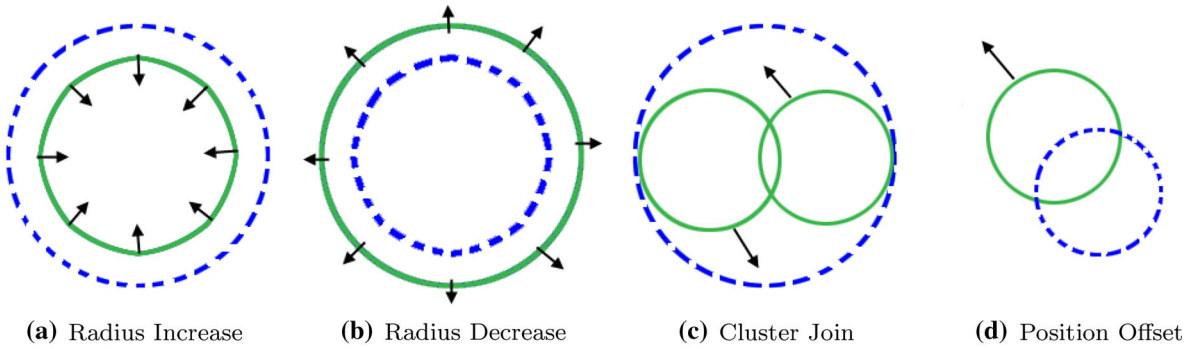

Fig. 4 Common errors of stream clusterings. A *solid circle* represents a true cluster, and a *dashed circle*indicates the corresponding found cluster (error). The cause of the error is the fast evolution of the stream in the direction of the *arrows*.

radius of clusters (Fig. 4a, b), which normally happens in the stream clustering since data points keep fading in and out.

Thus, in Fig. 4a, for instance, the ground truth is represented by the solid line, and the arrows represent the direction of the evolution of the data in the ground truth where the cluster is shrinking. The dashed line represents, however, the output of the simulated clustering algorithm using the *ClusterGenerator* that suffers from the: "Radius Increase" error. The same explanation applies to all other errors depicted in Fig. 4.

"Cluster add" and "Cluster remove" change the number of found clusters, which are caused by grouping noise points or falsely detecting meaningful patterns as a noise cloud. "Cluster join" merges two overlapping clusters as one cluster (Fig. 4c), which is a crucial error in streaming scenarios. Finally, "Position offset" changes the cluster position, and this commonly happens due to the movement of clusters in data streams (Fig. 4d).

We perform the experiments on all the above error types. We increase the level of one error at a time and evaluate its output with *CH*, *I* and *S_Dbw*, which performed well in the previous experiment. For the input stream, we use the same stream settings as in Sect. 4.2.1.

4.2.2 Results

In Fig. 5, the evaluation values are plotted on the y-axis according to the corresponding error level on the x-axis. From Fig. 5a, we can see that *CH* value decreases as the level of "Radius increase", "Cluster add", "Cluster join", and "Position offset" errors increases. *CH* correctly and constantly penalizes the four errors, since smaller *CH* value corresponds to worse clustering. However, it shows completely reversed curves in "Radius decrease" and "Cluster remove" errors. The reason for wrong rewarding of the "Radius decrease" error, is that the reduction of the size of clusters increases their compactness, and thus both *CH* and *I* increase. The "Cluster remove" error detection is a general problem for all internal measures as they compare their clustering result only to its self. Regardless of the "Radius decrease" and the "Cluster remove" errors, *CH* has generally the best performance on streaming data compared to the other measures.

We can see in Fig. 5b that using *I* results in a misinterpretation of the "Radius decrease" and "Cluster remove" error situations. The reason for it is similar to that of *CH*, since the usage of *I* results also in adopting the distance between objects within clusters as the intra-cluster compactness and the distance between cluster centers as the inter-cluster separation. In addition, using *I* wrongly favorites the "Position offset" error instead of penalizing it. If the boundaries of found clusters are moved besides the truth, they often miss the data points, which produces a similar situation to "Radius decrease" which *I* is vulnerable to.

S_Dbw produces high values when it regards a clustering as a bad result, which is opposite to the previous two measures. In Fig. 5c, we can see that it correctly penalizes the three error types "Radius increase", "Cluster add", and "Cluster join". For "Position offset" error, one can say that the value is somehow increasing but the curve is actually fluctuating too much. It also fails to penalize "Cluster remove" correctly.

From these results, we can determine that among the discussed internal evaluation measures, *CH* is the best internal evaluation one which can well handle many stream clustering errors. Even though *S_Dbw* performs very well on the static data (cf. [28]) and on the streaming data in the previous experiments (cf. Sect. 4.2.2), we observed that it has weak capability to capture common errors of stream clustering.

5 Internal evaluation measures of density-based stream clustering algorithms

In this section, we evaluate the performance of internal stream clustering measures using a density-based stream clustering

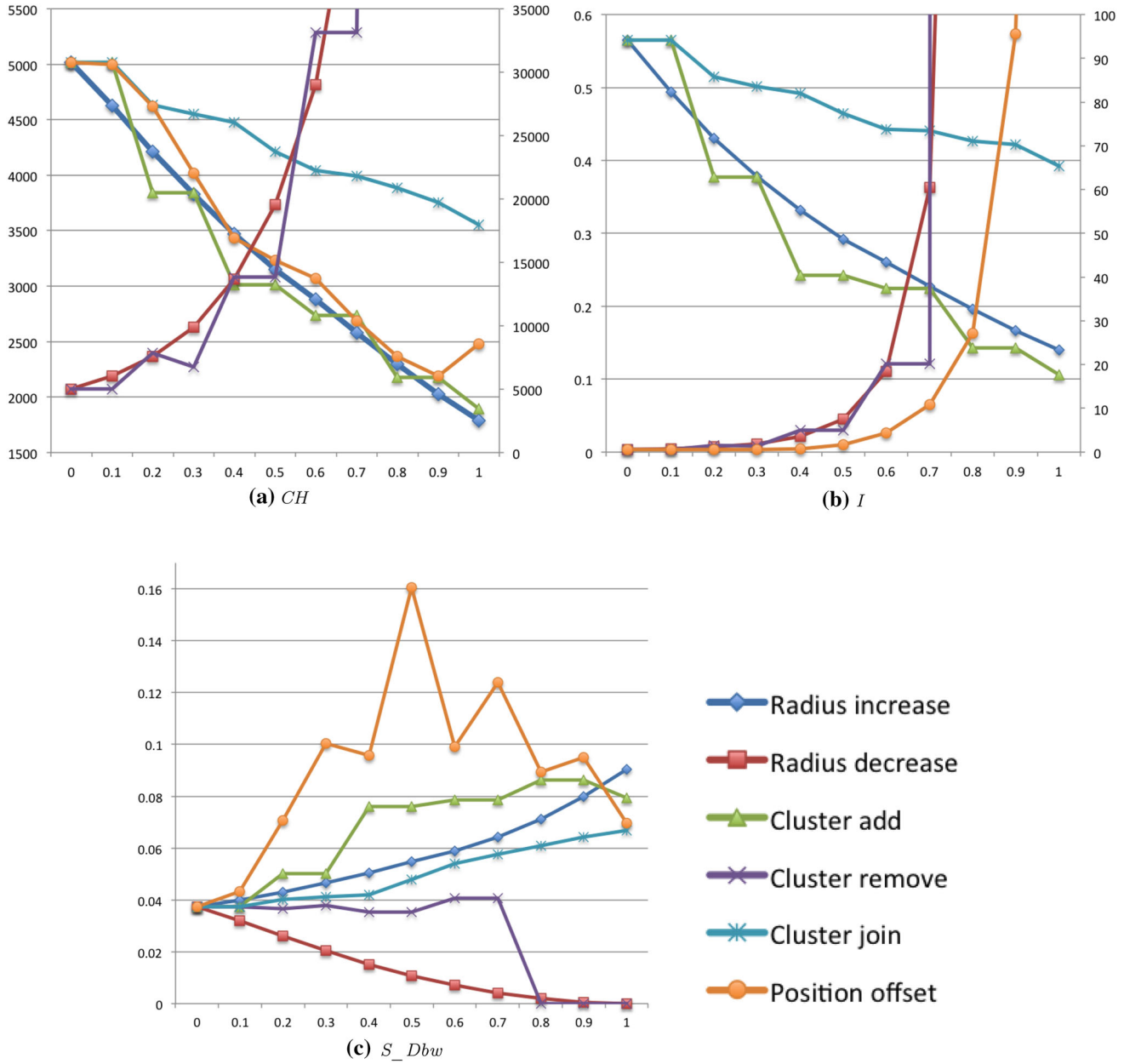

Fig. 5 Experimental results for each error type. Evaluation values (y-axis) are plotted according to each error level (x-axis). Some error curves are drawn on a secondary axis due to its range: **a** "Radius decrease" and "Cluster remove", **b** "Radius decrease", "Cluster remove", and "Position offset".

algorithm, namely DenStream [6]. We will start by experimenting DenStream using external evaluation measures to get some kind of "ground truth", then we will compare the performance of the internal evaluation measures using how close they are to this ground truth. Similar to the previous section, we use *MOA (Massive Online Analysis)* [4] framework for the evaluation. Again we have used the RandomRBFGenerator to create a 10-dimensional dataset of 30,000 objects forming 5 drifting clusters with different and varying densities and sizes. For DenStream [6] and MOA, we set the parameter settings as follows: the evaluation horizon = 1000, the outlier microcluster controlling threshold: $\beta = 0.15$, the initial number of objects initPoints = 1000, the offline factor of ϵ compared to the online one = 2, the decaying factor $\lambda = 0.25$, and the processing speed of the evaluation = 100. The parameters which are not mentioned are not directly related to this experiment and are set to the defaults of MOA.

5.1 Deriving the ground truth using external evaluation measures

Internal evaluation measures do not benefit from the ground truth information provided in the form of cluster label in our dataset. This was not a problem in the case of the k-means-based algorithm CluStream [1] discussed in Sect. 4.1, since the optimal parameter setting was simply $k = 5$ as we have generated 5 clusters. In the case of the density-based stream clustering algorithm DenStream [6] this is not as straightforward. To obtain some kind of ground truth for a density-based stream clustering algorithm like DenStream, we used the results from some external evaluation measures to derive the parameter settings for the best and the worst clustering results. The following external evaluation measures were used.

The first one is the **F**1 [3] measure which is a widely used external evaluation that harmonizes the precision and the recall of the clustering output.

The other one is the **purity measure** which is widely used [2,6,24] to evaluate the quality of a clustering. Intuitively, the purity can be seen as the pureness of the final clusters compared to the classes of the ground truth. The average purity is defined as follows:

$$\text{purity} = \frac{\sum_{i=1}^{NC} \frac{n_i^d}{n_i}}{NC} \tag{12}$$

where NC represents the number of clusters, n_i^d denotes the number of objects with the dominant class label in cluster C_i and n_i denotes the number of the objects in the cluster C_i.

The third used external evaluation measure is the **number of clusters** which averages previous numbers of clusters within the H window. Similarly, the $F1$ and the purity are computed over a certain predefined window H from the current time. This is done since the weights of the objects decay over time. Thus, the number of found clusters could be any real value, while $F1$ and purity could be any real value from 0 to 1.

5.1.1 Results

Table 2 contains the mean value of 5 evaluation results which we obtained in the whole streaming interval when considering the external evaluation measures: $F1$, purity and the number of clusters for different settings of the μ and ϵ parameters of DenStream. The bold values of each column represent the best value of the index among the outputs of the used parameter settings. It is the highest value in the case of $F1$ and the purity, and the closest value to 5 in the case of the number of clusters. The worst values in each column are underlined. It can be seen from Table 2 that among the selected 9 parameter settings, $\mu = 2$ and $\epsilon = 0.18$ results in the worst clustering output of DenStream over the current dataset while $\mu = 2$ and $\epsilon = 0.06$ results in the best one. Figure 6 depicts the external evaluation measures values for these settings.

Table 2 Evaluation results of external validation on the stream clusterings

μ	ϵ	F1	Purity	Number of clusters
2	0.06	**0.6454**	**1.00**	**4.7959**
2	0.12	0.6250	0.9999	4.5918
2	0.18	<u>0.5873</u>	<u>0.9095</u>	<u>4.0204</u>
3	0.06	0.6220	**1.00**	4.6531
3	0.12	0.6139	**1.00**	4.5714
3	0.18	0.5857	0.9169	4.0816
4	0.06	0.6166	**1.00**	4.7143
4	0.12	0.6157	**1.00**	4.6735
4	0.18	0.5741	0.9233	4.1020

The best obtained values for each measure are in bold, the worst ones are underlined

Our task now is to get the internal evaluation measure that shows the highest correlation with this result.

5.2 The results of using internal evaluation measures for density-based stream clustering

Figures 7, 8 and 9 show the mean values of 5 evaluation results using all internal evaluation measures over the previous parameter settings.

These results are summarized in Table 3, where *RMSSTD*, *RS* and Γ are directly excluded due to the subjective process of defining the first "elbow" in their monotonically increasing or decreasing curves (according to [10]). We obtained these results for the different selected parameter settings, and the final values are summarized from the measurements in the whole streaming interval.

Table 3 shows that all the internal evaluation measures except for *SD* reach their worst values (underlined values) exactly at the setting ($\mu = 2$ and $\epsilon = 0.18$). This shows that the results of those internal measures are inline with those of the external ones w.r.t. punishing the worst setting.

What is left now is to check which of those measures reaches its best value at the same setting where the external evaluation measures are reaching their best values (i.e., $\mu = 2$ and $\epsilon = 0.06$). It can be seen from Table 3 that none of the internal evaluation measures is reaching the best value (in bold) at that parameter setting.

We have to calculate now which of those internal evaluation measures has the highest (local) correlation between its

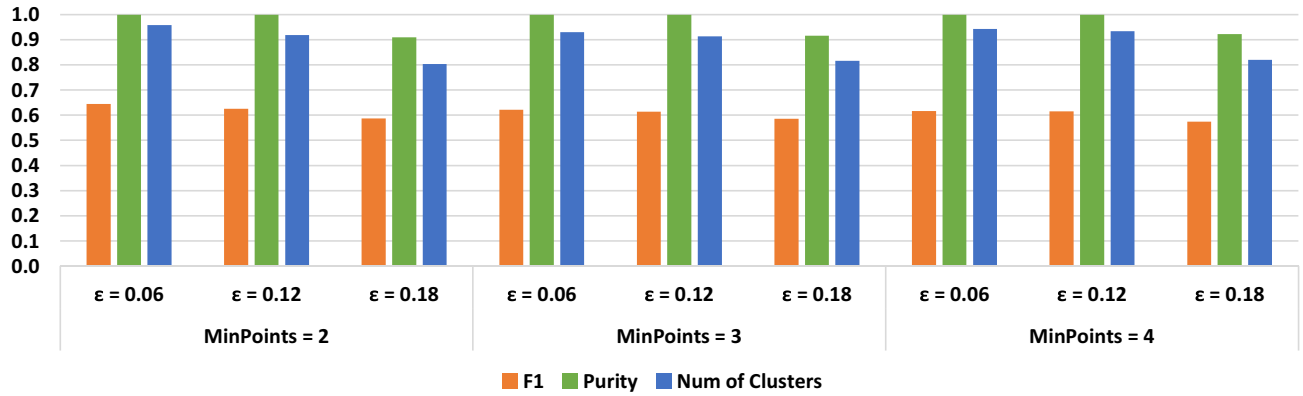

Fig. 6 External evaluation measures on the *y*-axis using different parameter settings of DenStream on the *x*-axis.

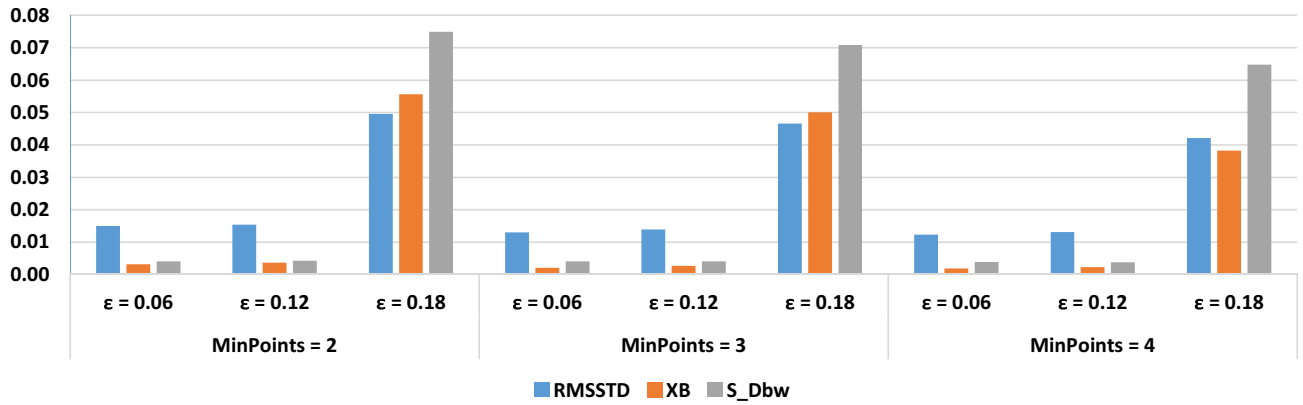

Fig. 7 Performance of the internal evaluation measures: *RMSSTD*, *XB* and *S_Dbw* on the *y*-axis using different parameter settings of DenStreamon the *x*-axis.

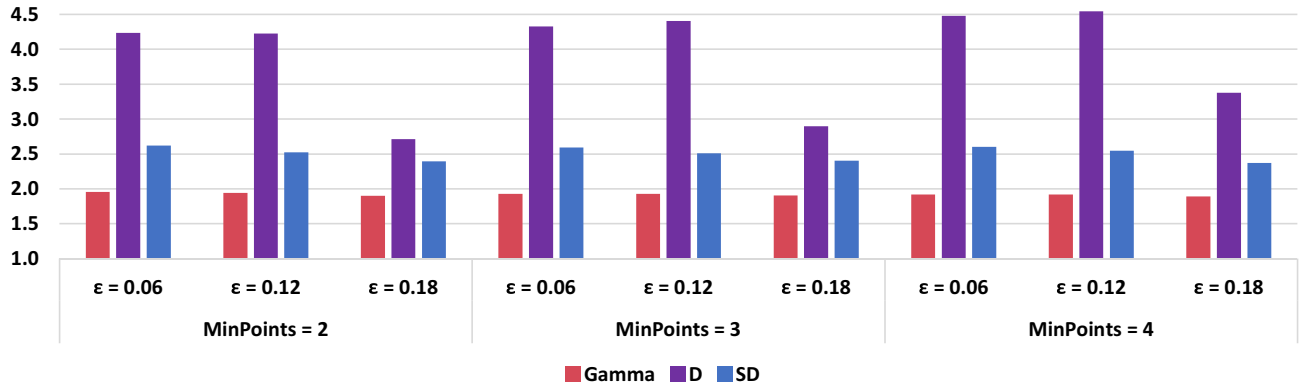

Fig. 8 Performance of the internal evaluation measures: Γ, *D* and *SD* on the *y*-axis using different parameter settings of DenStream on the *x*-axis.

best value V^i_{best}, and the value calculated at the best ground truth setting ($\mu = 2$ and $\epsilon = 0.06$), we call this value V^i_{truth}.

Let:

$$V^i_{avg} = \frac{\sum_{s=1}^{9} V^i_s}{9} \tag{13}$$

be the average of the values taken for each internal measure i over the each setting s of the 9 considered parameter settings. Our target is to get out of the 7 winning internal measures in Table 3, the internal evaluation measure i that achieves:

$$min_i \left(\frac{V^i_{best} - V^i_{truth}}{V^i_{best} - V^i_{avg}} \right) \tag{14}$$

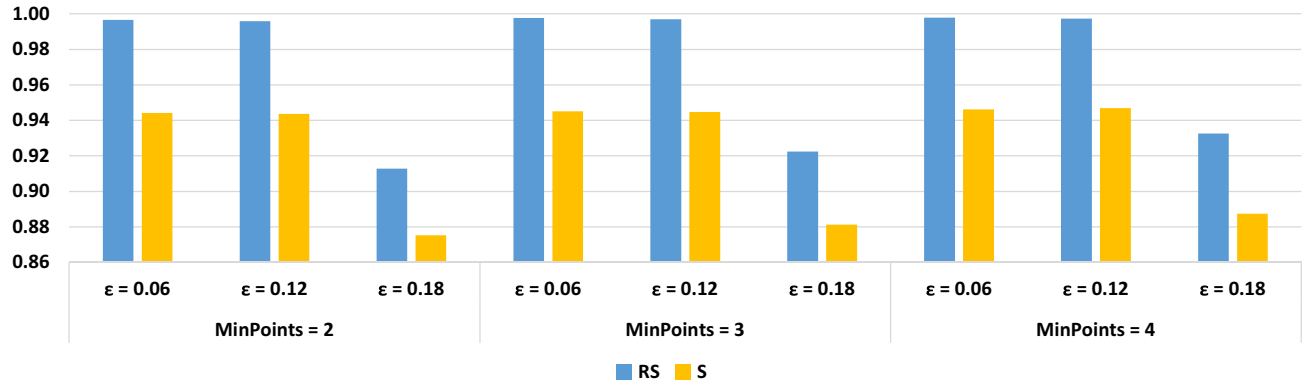

Fig. 9 Performance of the internal evaluation measures: *RS* and *S* on the *y*-axis using different parameter settings of DenStream on the *x*-axis.

Table 3 Evaluation results of internal validation on the stream clusterings

μ	ϵ	*CH*	*I*	*D*	*S*	*DB*	*XB*	*SD*	*S_Dbw*
2	0.06	50911	58.711	4.2364	0.9441	0.1240	0.0032	2.6188	0.0041
2	0.12	54495	65.131	4.2261	0.9436	0.1297	0.0037	2.5212	0.0042
2	0.18	34451	37.131	2.7138	0.8753	0.2859	0.0557	2.3928	0.0749
3	0.06	66717	88.312	4.3258	0.9451	0.1038	0.0021	2.5936	0.0040
3	0.12	65504	84.805	4.4055	0.9447	0.1132	0.0027	2.5111	0.0041
3	0.18	40237	45.474	2.8965	0.8812	0.2640	0.0501	2.4014	0.0708
4	0.06	**74205**	**102.57**	4.4786	0.9462	**0.0974**	**0.0019**	2.6039	0.0038
4	0.12	72501	100.93	**4.5413**	**0.9469**	0.1059	0.0023	2.5455	**0.0038**
4	0.18	48553	60.397	3.3772	0.8873	0.2368	0.0382	**2.3713**	0.0648

The best obtained values (not necessarily the maximum or the minimum) are in bold

Table 4 Testing the winning internal measures (i.e., those whose worst value in Table 3 matched the worst ground truth)

Internal measure i	*CH*	*I*	*D*	*S*	*DB*	*XB*	*S_Dbw*
$\frac{V^i_{best}-V^i_{truth}}{V^i_{best}-V^i_{avg}}$	1.30807	1.41513	0.48392	0.11849	0.40945	0.08125	**0.01208**

The test is aiming to find which i of those has the highest correlation between its best value V^i_{best} and its value at the best ground truth setting V^i_{truth}

In other words, we are seeking for the measure whose V^i_{best} has the smallest relative deviation from V^i_{truth} compared to its deviation from the mean.

It should be noted that the simple tendency check mentioned in Eq. 14 is reliable. For a specific measure i, minimizing the fraction mentioned in Eq. 14 implies that the numerator is considerably smaller than the denominator. Thus, the deviation of the measure i from the ground truth V^i_{truth} is considerably smaller than the deviation from its own mean. Thus, we can get some kind of guaranty that this correlation is strong enough. As we are unable to find an internal measure i whose $V^i_{best} = V^i_{truth}$, we perform this approximation to find the one with the closest tendency to make V^i_{truth} its V^i_{best}.

Table 4 shows that *S_Dbw* has the highest correlation between its $V^{S\_Dbw}_{best}$ value and the ground truth $V^{S\_Dbw}_{truth}$ value. This is because it has the smallest $\frac{V_{best}-V_{truth}}{V_{best}-V_{avg}}$ value highlighted in bold.

This means that the among the tested internal evaluation measures, *S_Dbw* has shown the best results when considering the density-based stream clustering algorithm DenStream [6]. Similar to the static data case, *CH*, *I*, *D S*, *DB*, and *SD* perform bad in the streaming settings. This is different to the *k*-means stream clustering case, where *CH* performed the best. On the other hand, *S_Dbw* performed the best which is similar to the static case results reported in [28]. *XB* worked also well.

6 Conclusions and outlook

Evaluating clustering results is very important to the success of clustering tasks. In this article, we discussed the internal clustering validation scheme in both *k*-means and density-based stream clustering scenarios. This is much more efficient and easier to apply in the absence of any previ-

ous knowledge about the data than the external validation. We explained fundamental theories of internal validation measures and its examples. In the k-means-based case, we performed a set of clustering validation experiments that well reflect the properties of streaming environment with five common clustering aspects at the same time. These aspects reflect monotonicity, noise, different densities of clusters, skewness and the existence of subclusters in the underlying streaming data. The three winners from the first experimental evaluation were then further evaluated in the second phase of experiments. The sensitivity of each of those three measures was tested w.r.t. six stream clustering errors. Different to the results gained in a recent work on static data, our final experimental results on streaming data showed that *Calinski-Harabasz index (CH)* [5] has, in general, the best performance in k-means-based streaming environments. It is robust to the combination of the five conventional aspects of clustering, and also correctly penalizes the common errors in stream clustering.

In the density-based case, we performed a set of experiments over different parameter settings using the DenStream [6] algorithm. We used external evaluation measures to extract some ground truth. We used the ground truth to define the best, and the worst parameter settings. Then, we tested which of the internal measures has the highest correlation with the ground truth. Our results showed that the *revised validity index*: *S_Dbw* [11] shows the best performance under density-based stream clustering algorithms. This is inline with the results reported over static data in [28]. Additionally, the *Xie-Beni index (XB)* [35] has shown also a good performance.

In the future, we want to test those measures on different categories of advanced stream clustering algorithms like adaptive hierarchical density-based ones (e.g., HAStream [15]) or projected/subspace ones (e.g., PreDeConStream [24] and SubClusTree [14]). Additionally, we want to evaluate the measures when streams of clusters available in subspaces [13] are processed by the above algorithms.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB, pp. 81–92 (2003)
2. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for projected clustering of high dimensional data streams. In: VLDB, pp. 852–863 (2004)
3. Assent, I., Krieger, R., Müller, E., Seidl, T.: INSCY: Indexing subspace clusters with in-process-removal of redundancy. In: Proceedings of the 8th IEEE International Conference on Data Mining, ICDM '08, pp. 719–724. IEEE (2008)
4. Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., Seidl, T.: MOA: Massive online analysis, a framework for stream classification and clustering. JMLR **11**, 44–50 (2010)
5. Calinski, T., Harabasz, J.: A dendrite method for cluster analysis. Comm. Stat. **3**(1), 1–27 (1974)
6. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: SIAM SDM, pp. 328–339 (2006)
7. Davies, D., Bouldin, D.: A cluster separation measure. IEEE PAMI **1**(2), 224–227 (1979)
8. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B. **39**(1), 1–38 (1977)
9. Dunn, J.: Well separated clusters and optimal fuzzy partitions. J. Cybern. **4**(1), 95–104 (1974)
10. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On clustering validation techniques. J. Intell. Inf. Syst. **17**(2), 107–145 (2001)
11. Halkidi, M., Vazirgiannis, M.: Clustering validity assessment: Finding the optimal partitioning of a data set. In: IEEE ICDM, pp. 187–194 (2001)
12. Halkidi, M., Vazirgiannis, M., Batistakis, Y.: Quality scheme assessment in the clustering process. In: PKDD, pp. 265–276 (2000)
13. Hassani, M., Kim, Y., Seidl, T.: Subspace MOA: subspace stream clustering evaluation using the MOA framework. In: DASFAA, pp. 446–449 (2013)
14. Hassani, M., Kranen, P., Saini, R., Seidl, T.: Subspace anytime stream clustering. In: SSDBM, p. 37 (2014)
15. Hassani, M., Spaus, P., Seidl, T.: Adaptive multiple-resolution stream clustering. In: MLDM, MLDM'14, pp. 134–148 (2014)
16. Hassani, M.: Efficient clustering of big data streams. PhD thesis, RWTH Aachen University (2015)
17. Hassani, M., Kim, Y., Choi, S., Seidl, T.: Effective evaluation measures for subspace clustering of data streams. In: Trends and Applications in Knowledge Discovery and Data Mining—PAKDD 2013 International Workshops, pp. 342–353 (2013)
18. Hassani, M., Kim, Y., Choi, S., Seidl, T.: Subspace clustering of data streams: new algorithms and effective evaluation measures. J. Intell. Inf. Syst. **45**(3), 319–335 (2015)
19. Hassani, M., Müller, E., Seidl, T.: EDISKCO: Energy Efficient Distributed In-Sensor-Network K-center Clustering with Outliers. In: Proceedings of the 3rd International Workshop on Knowledge Discovery from Sensor Data, SensorKDD '09 @KDD '09, pp. 39–48. ACM (2009)
20. Hassani, M., Müller, E., Spaus, P., Faqolli, A., Palpanas, T., Seidl, T.: Self-organizing energy aware clustering of nodes in sensor networks using relevant attributes. In: Proceedings of the 4th International Workshop on Knowledge Discovery from Sensor Data, SensorKDD '10 @KDD '10, pp.39–48. ACM (2010)
21. Hassani, M., Seidl, T.: Distributed weighted clustering of evolving sensor data streams with noise. J. Dig. Inf. Manag. (JDIM) **10**(6), 410–420 (2012)
22. Hassani, M., Seidl, T.: Internal clustering evaluation of data streams. In: Trends and Applications in Knowledge Discovery and Data Mining—PAKDD 2015 Workshop: QIMIE, 2015. Revised Selected Papers, pp. 198–209 (2015)
23. Hassani, M., Spaus, P., Cuzzocrea, A., Seidl, T.: Adaptive stream clustering using incremental graph maintenance. In: Proceedings of the 4th International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, BigMine 2015 at KDD'15, pp. 49–64 (2015)

24. Hassani, M., Spaus, P., Gaber, M.M., Seidl, T.: Density-based projected clustering of data streams. In: Proceedings of the 6th International Conference on Scalable Uncertainty Management, SUM '12, pp. 311–324 (2012)
25. Hubert, L., Arabie, P.: Comparing partitions. J. Intell. Inf. Syst. **2**(1), 193–218 (1985)
26. Kaufman, L., Rousseeuw, P.: Clustering by means of medoids. Statistical Data Analysis Based on the L_1 *Norm*, pp. 405–416 (1987)
27. Kremer, H., Kranen, P., Jansen, T., Seidl, T., Bifet, A., Holmes, G., Pfahringer, B.: An effective evaluation measure for clustering on evolving data streams. In: ACM SIGKDD, pp. 868–876 (2011)
28. Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J.: Understanding of internal clustering validation measures. In: ICDM, pp. 911–916 (2010)
29. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, volume 1, pp. 281–297. University of California Press (1967)
30. Maulik, U., Bandyopadhyay, S.: Performance evaluation of some clustering algorithms and validity indices. IEEE PAMI **24**, 1650–1654 (2002)
31. Rendón, E., Abundez, I., Arizmendi, A., Quiroz, E.M.: Internal versus external cluster validation indexes. Int. J. Comp. Comm. **5**(1), 27–34 (2011)
32. Ramze Rezaee, M., Lelieveldt, B.B.F., Reiber, J.H.C.: A new cluster validity index for the fuzzy c-mean. Pattern Recogn. Lett. **19**(3–4):237–246 (1998)
33. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. **20**(1), 53–65 (1987)
34. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley Longman, Inc. Boston (2005)
35. Xie, X.L., Beni, G.: A validity measure for fuzzy clustering. IEEE PAMI **13**(8), 841–847 (1991)