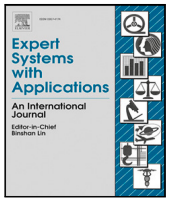
Expert Systems With Applications 233 (2023) 120994

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Anomaly detection in streaming data: A comparison and evaluation study

Félix Iglesias Vázquez [a,*], Alexander Hartl [a], Tanja Zseby [a], Arthur Zimek [b]

[a] TU Wien, Gusshausstraße 25 / E389, 1040, Vienna, Austria
[b] University of Southern Denmark (SDU), Campusvej 55, DK 5230 Odense M, Denmark

ARTICLE INFO

Dataset link: Data for Evaluation of Stream Data Analysis Algorithms (Original data), SWAN-SF (Reference data), Synthetic and real time-series with labeled anomalies (Reference data), Intrusion Detection Evaluation Dataset (CIC-IDS2017) (Reference data)

Keywords:
Anomaly detection
Outlier detection
Streaming data
Concept drift

ABSTRACT

The detection of anomalies in streaming data faces complexities that make traditional static methods unsuitable due to computational costs and nonstationarity. We test and evaluate eight state of the art algorithms against prominent challenges related to streaming data. Results show insights regarding accuracy, memory-dependency, parameterization, and pre-knowledge exploitation, thus revealing the high impact of some data characteristics to establish a most appropriate algorithm, namely: *locality* (i.e., whether outlierness is relative to local contexts), *relativeness* (i.e., if past data defines outlierness), and *concept drift* (if it is expected, its intensity and frequency). In most applied cases, such factors can be inferred in advance through the use of historical data and domain knowledge. Assuming the viability of the studied methods in terms of time efficiency, this work discloses key findings to achieve optimal designs of streaming data anomaly detection in real-life applications.

1. Introduction

Streaming data (aka data streams) refers to the technological challenge in which data are acquired and must be analyzed continuously, resulting in a potentially unlimited and constantly growing dataset (Ramírez-Gallego, Krawczyk, García, Wosfxniak, & Herrera, 2017). Data streams are closely related to multivariate time series, although the latter usually exhibit a stronger time dependence and do not necessarily have to be processed on the fly. We refer the reader to the work by Read, Rios, Nogueira, and de Mello (2020) for further discussion on this topic.

Detecting anomalies in streaming data is required in many domains. Particularly the sequential processing of sensor data is determinant due to the increasing interconnection of devices in the Internet of Things, for instance, in medical, networking or environmental monitoring. A clear definition of the expected anomalies in the specific application is required for a good detection performance. In this respect, "outlier" and "anomaly" are commonly taken as synonyms. A traditional definition of outlier is "an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism" (Hawkins, 1980). This suggests that an "outlier" should stand out from other data due to certain statistical properties, while an "anomaly" is usually rather defined by the application context. But note that such a definition of "outlier" – as well as the common uses of "anomaly" – are inseparable from a certain ambiguity, particularly in time-dependent environments. For example, outliers in a stream can be isolated data points or data points that break temporal patterns, but also isolated small clusters or even big clusters that suddenly burst and later disappear. In this study, we use "anomaly", "outlier" and "novelty" interchangeably, and we do not dismiss the semantic problem; instead, we consider that these terms can be subject to open, application-dependent definitions.

Previous works emphasize that performance of unsupervised classification usually depends on the data and the analysis goals (Campos et al., 2016; Domingues, Filippone, Michiardi, & Zouaoui, 2018). Therefore, although some methods might show clear advantages, establishing a general "best algorithm" is hardly possible. The previous references provide some guidelines for selecting algorithms and parameter settings in static setups, but no similar suggestions exist for outlier detection algorithms in streaming data analysis. In this work, we try to derive the basis for such guidelines by studying the following questions:

1. **How do specific data characteristics affect algorithm accuracy?** We consider: different outlier percentages, variable space and temporal densities, variable distances among normal shapes, and different types of concept drift.
2. **How do algorithms use previous knowledge to fit streaming conditions?** Unsupervised learning algorithms are often designed by abstracting from concrete applications and therefore

* Corresponding author.
E-mail addresses: felix.iglesias@tuwien.ac.at (F. Iglesias Vázquez), alexander.hartl@tuwien.ac.at (A. Hartl), tanja.zseby@tuwien.ac.at (T. Zseby), zimek@imada.sdu.dk (A. Zimek).

https://doi.org/10.1016/j.eswa.2023.120994
Received 24 December 2022; Received in revised form 26 May 2023; Accepted 11 July 2023
Available online 17 July 2023
0957-4174/© 2023 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

ignoring potential pre-knowledge. In practice, pre-knowledge is required to train models and adjust hyperparameters, and ideally such process should be automatizable. Moreover, it is important to evaluate if configurations become suboptimal as data evolve.

3. **How does the variation of time parameters affect algorithms?** The memory span of algorithms (e.g., sliding window sizes) influences detection performance, but also memory demands and computational costs.

We investigate these questions by stress testing eight state-of-the-art outlier detection methods for streaming data. Section 2 introduces the topic and the studied methods. Section 3 discusses the different ways in which algorithms commonly establish the meaning of "anomaly". Section 4 explains the challenges to study, related to space geometries, types of concept drifts, and dependence on memory hyperparameters. Section 5 describes the analysis setup and experimental methodology. Results are shown and discussed in Section 6. We additionally propose two indices to characterize datasets and thus infer the most suitable methods to use. Conclusions are outlined in Section 7. The Appendix includes tables with extended results.

2. Outlier detection in streaming data

Anomaly detection in multidimensional spaces has been widely discussed for non-evolving setups; we have already referred to two popular comparative and evaluation studies (Campos et al., 2016; Domingues et al., 2018). However, anomaly detection in multidimensional streaming data is significantly less explored. Although there are works using static approaches in dynamic environments, and beyond batch-mode setups and hybrid options capable of taking advantage of both batch and incremental analysis (Pishgoo, Akbari Azirani, & Raahemi, 2021), algorithms used in streaming cases should not be the same due to different computational and processing requirements. Moreover, streaming data analysis must face the effect of concept drift, i.e., how the statistic properties of the modeled phenomena change over time in unexpected ways (Gama, Žliobaité, Bifet, Pechenizkiy, & Bouchachia, 2014). In general, the need for advances in streaming data analysis have been claimed by experts several times, very recently by Bezdek and Keller (2021). In the seminal work about concept drift, Gama et al. (2014) remark that "in unsupervised learning over evolving data, (...) validation of change detection and adaptation mechanisms only start to be investigated"[1]. Concept drift is also found as the main missing aspect to tackle by algorithms in one of the most popular surveys in stream clustering (Silva, Faria, Barros, Hruschka, Carvalho et al., 2013). Also Gomes, Read, Bifet, Barddal, and Gama (2019) highlight the need for further discussion of anomaly detection in streaming data.

To date, and to the best of our knowledge, perhaps the most relevant comparison of anomaly detection in streaming data is the work by Tran, Fan, and Shahabi (2016). Here, the authors focus on k-NN-based algorithms and compare them in terms of CPU time and peak memory consumption. Focusing on run-times and memory instead of accuracy is not surprising as many traditional proposals are algorithm variations that consist on techniques for speeding-up and increase computational efficiency. Instead, our study focuses on prototypical models rather than specific algorithms[2]. We assume the general competence of the studied methods (as shown in the original references) and use them in their ground implementations. Note that the scope of this work is to study how each method applies a different interpretation of outlierness and its implications. In addition, we evaluate run-times to obtain insights about the scalability when increasing the memory span[3].

2.1. Methods under study

We have selected methods based on their popularity or recent appearance in reputed publications, demonstrated effectiveness, published code transparency, and aiming to maximize the differences among alternative ways of approaching the same problem. All of them are **unsupervised and specific for streaming data**, except for one case that we use as baseline. They are:

- **SWKNN** (Sliding Window K-Nearest Neighbors) consists of the implementation of the k-NN algorithm proposed by Ramaswamy, Rastogi, and Shim (2000) within a sliding window. This is the most popular approach for outlier detection in streaming data, which underlies algorithms like AbstractC (Yang, Rundensteiner, & Ward, 2009), Exact- and Approx-STORM (Angiulli & Fassetti, 2007), the COD family (Kontaki, Gounaris, Papadopoulos, Tsichlas, & Manolopoulos, 2011), among others. They all share the same definition of outlier, which we cite from (Angiulli & Fassetti, 2007):

  > Let S be a set of objects, obj an object of S, k a positive integer, and R a positive real number. Then, obj is a distance-based outlier (or, simply, an outlier) if less than k objects in S lie within distance R from obj.

  This approach for capturing outlierness goes back to the seminal work by Knorr and Ng (1998), and derived techniques mainly differ in how to reduce the computational load, for instance, using sampling in Approx-STORM or micro-clusters in MCOD. Instead of merely obtaining a binary label, we build a score based on the distance of the kth neighbor to obj (thus allowing fair comparisons with other methods). This implies that our experiments do not use the R hyperparameter for setting a crisp threshold between inliers and outliers. In our implementation, closest neighbors are searched with M-Tree indexing (Ciaccia, Patella, & Zezula, 1997).
- **SWLOF** (Sliding Window Local Outlier Factor) is the implementation of the popular LOF algorithm proposed by Breunig, Kriegel, Ng, and Sander (2000) with a sliding window. LOF is still one of the most reliable options as a general purpose outlier detection technique (Campos et al., 2016); however, it is too computational expensive for streaming data. The incremental LOF algorithm proposed by Pokrajac, Lazarevic, and Latecki (2007) is equivalent to the here implemented SWLOF in terms of accuracy, but makes use of different indexing techniques to reduce run-times.
- **RRCT** (Robust Random Cut Forests) is a model-based approach for outlier detection in dynamic data streams available in AWS Kinesis Analytics. It uses an ensemble of tree graphs and establishes the outlierness of new points based on their differential effect on the forest structure (Guha, Mishra, Roy, & Schrijvers, 2016).
- **RSHash** is a straightforward algorithm based on randomized hashing developed by Sathe and Aggarwal (2016). It uses a hashed representation of the data, an ensemble of weak detectors, and a data sample to create a training model. The method complexity is linear in relation to the size of the data and the space requirement is constant, achieving good accuracy performances.

[1] The authors consider evolving multi-feature spaces (i.e., beyond univariate time series).

[2] Henceforth, we refer to prototypical models as "methods" to differentiate them from their respective internally learned models. Also, when "methods" and "algorithms" are used in the same context, the latter addresses specific implementations of the mentioned methods.

[3] In our experiments, despite being mostly implemented in Python, the core of the studied algorithms is built in C++ and wrapped with SWIG: dSalmon framework in Hartl (2020). This is done to severely reduce run-time demands without detriment to the accuracy, which does not differ from the published in the original sources. There exist other Python frameworks for streaming anomaly detection, like PySAD (Yilmaz & Kozat, 2020), but their computational demands are unfeasible to cover the extensive experimentation here described in realistic times. All experiments in this paper are openly available in Iglesias and Hartl (2021).

- **LODA** is proposed by Pevný (2016), an anomaly detector for streaming data based on combining a set of weak learners that shows low time and space complexity. LODA simplifies the analysis space by means of a set of principal one-dimensional histograms, recalling previous proposals for static environments such as HBOS (Goldstein & Dengel, 2012) or PCA-based detectors (Shyu, Chen, Sarinnapakorn, & Chang, 2006). The algorithms used in our comparative study is implemented with equi-depth histograms.
- **SDO** is a model-based outlier detection technique (Iglesias, Zseby, & Zimek, 2018) and **SDOstream** is its extension for streaming data (Hartl, Iglesias, & Zseby, 2020). SDO is genuinely static, but can operate incrementally without updating internal models. We include it as a baseline and due to its importance to understand how the "outlierness" varies depending on the method. Since SDO is unsupervised and model-based, it is also comparatively useful for obtaining insights about how methods adjust time and memory spans and how pre-knowledge is leveraged. SDOstream adds a hyperparameter T that establishes the model aging and, hence, the memory duration.
- **xStream** is another recent approach with the peculiarity of addressing streaming data scenarios in which the feature space might also vary.[4] It deals with high-dimensionality by projecting data into low-dimensional subspaces and by calculating outlierness scores with half-space chains (Manzoor, Lamba, & Akoglu, 2018).

The introduced methods mainly differ in the way of performing and using density-estimation (Zimek & Filzmoser, 2018), although it is also determinant if a purely geometrical interpretation of "outlierness" is used, or if, instead, algorithms define outliers as instances opposed to a known normality. Particularly in stream environments, a third relevant factor is how past data is remembered.

3. Defining anomalies

Outlier detection was traditionally undertaken in a purely unsupervised way, assuming that the outlierness of a data point must be established by evaluating its location with regard to other data points close in time and space (strict). However, this approach may be inadequate in some cases, e.g., to detect collective anomalies (i.e., clusters of anomalous data points). Hence, more practical approaches define outliers or anomalies as something that diverges from what has been previously defined as normal data points (relative). This second approach is closer to semi-supervised learning, sometimes referred to as one-class classification (Marques, Swersky, Sander, Campello, & Zimek, 2023; Moya & Hush, 1996). The reason for these variable interpretations is that the terms themselves are ambiguous and acquire nuances derived from the applications. In their exhaustive survey about anomaly detection, Ruff et al. (2021) observe handy semantic differences:

> while anomalies are often the data points of interest (e.g., a long-term survivor of a disease), outliers are frequently regarded as "noise" or "measurement error" that should be removed in a data preprocessing step ("outlier removal"), and novelties are new observations that require models to be updated to the "new normal". The methods for detecting points from low probability regions, whether termed "anomaly", "outlier", or "novelty", are essentially the same, however.

[4] This property is not studied in this work.

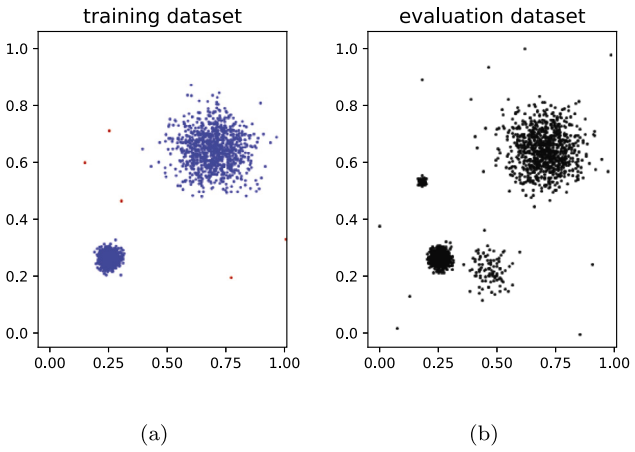


Fig. 1. Two-dimensional example. (a) Training data used for parameter adjustment (blue points: inliers, red points: outliers). (b) Test data used for evaluation.

Therefore, one-class classification, anomaly detection, and outlier detection are commonly seen as equivalent tasks. From the perspective of streaming data application, the question of whether the task should be approached in a supervised or unsupervised manner seems to be more and more irrelevant, since some prior knowledge is almost indispensable (at least to adjust hyperparameters) and there is no reason to obviate historical data. Actually, semisupervised and weakly supervised learning have shown to be promising for anomaly detection (Ruff et al., 2021). Terms become anyway less clear in streaming scenarios; for instance, note that all methods studied in this paper are unsupervised, but some of them learn models progressively, and some of them incorporate training phases in which they use unlabeled data, but can make the most of labeled data if available.

If the discussion is rather oriented to decide between *strict* and *relative* methods, both approaches or even intermediate solutions can be equally correct or incorrect, and the peculiarities of the application are to establish the more suited one. This aspect is rarely tackled in the related literature even in spite of its importance to achieve accurate performances. Furthermore, beyond such dichotomy, our experimental results disclose how each prototypical method internally defines outlierness in a different way. In this section, we show a simple example to empirically evaluate this fact. In the example, a two-dimensional dataset is split into two subsets of the same size: one for known, past data (training), and the second for new data to be processed in a stream (test/evaluation), Fig. 1. Both training and test datasets are very similar to each other, the main differences being new far outliers and two new clusters.

We analyze this example with the methods described in Section 2.1, their hyperparameters adjusted as explained below in Section 5.1 and with a memory parameter (T) equals the size of each split in all cases. Fig. 2 shows performances with two plots for each studied algorithm: a first plot showing outlierness scores with a colorbar and a second plot emphasizing the top 100 data points with highest outlierness. To allow a better comparison among dynamics, scores are normalized ($s = \frac{-1}{1+s}$) and later scaled to the $[0, 1]$ range. Some remarkable differences among methods are:

1. *Variable dynamical range of outlierness*. Compared to other methods, RRCT and LODA stand out due to a higher range variation for scoring inliers, this means a lower difference between the scores of inliers and outliers (i.e., outliers get less extreme scores). Particularly LODA, whose unnormalized scores (ordered from the lowest to the highest) show a nearly linear curve in this example.
2. *Evaluation of the new high-dense cluster*. Depending on the method, it is seen as all inliers, all outliers, or some-in some-out.

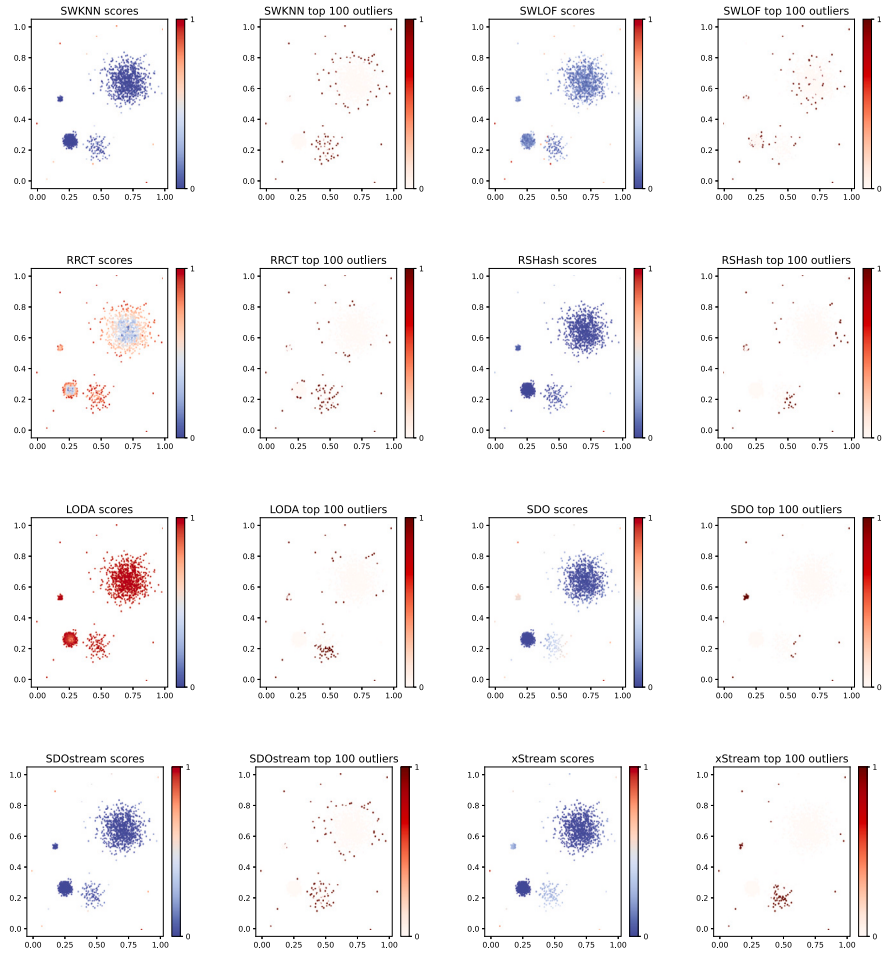


Fig. 2. Performances of the studied methods in a two-dimensional example. Each method shows two plots: normalized outlierness scores and top 100 outliers (dark color).

3. *Evaluation of the new low-dense cluster*. Depending on the method, all data points or only some of them are seen as top outliers.
4. *Evaluation of the external layer of the known medium-dense cluster*. Methods diverge when assessing external data points of known clusters among top outliers or not.

Besides other peculiarities specific to each method, we observe key categories in the way of defining outlierness:

- *Local vs. global* (Schubert, Zimek, & Kriegel, 2014), where local approaches focus on evaluating outlierness in the most immediate neighborhood, whereas global methods tend to see outliers as extreme values and strongly isolated data.
- *Strict vs. relative*, where relative approaches use past data for defining a normality that remains over time in models or parameters, whereas strict methods use a definition of outlierness less dependent on pre-knowledge.

4. Data scenarios and challenges

In this section we present and discuss the streaming data challenges with which we test the methods under study.

4.1. Synthetic data

To evaluate the versatility of the algorithms, we designed nine different multidimensional streaming data scenarios, every one of them posing a specific challenge. They were created with MDCstream (Iglesias, Odjanic, Hartl, & Zseby, 2020) and are available independently in Mendeley (Iglesias, 2021). MDCstream is a streaming data generator able to create different types of concept drift in multidimensional spaces. It adds outliers as noise, close points, extreme values or contextual anomalies depending on the overall configuration, but always as isolated data points (unless small clusters are later relabeled as outliers).

Table 1
MDCstream wrapper configuration for generating synthetic datasets.

Name	Base	Nonstat.	Sequential	Moving	Med.-out.
datasets	20	20	20	20	20
dimensions	several	several	several	several	several
time-behav.	stat.	nonstat.	sequent.	stat.	stat.
outliers	few	few	few	few	medium
clusters	few	few	few	few	few
den.-diffs.	two	two	two	two	two
space	normal	normal	normal	normal	normal
mov.-cls.	no	no	no	all	no
overlap	no	no	no	no	no

Name	Many-out.	Close	Dens.-diff.	Overlap
datasets	20	20	20	20
dimensions	several	several	several	several
time-behav.	stat.	stat.	stat.	stat.
outliers	many	few	few	few
clusters	few	many	many	many
den.-diffs.	two	two	many	many
space	normal	tight	normal	normal
mov.-cls.	no	no	no	no
overlap	no	no	no	yes

Table 2
MDCstream wrapper: meaning of configuration parameters used.

Parameter	Value	Meaning
dimensions	several	Between 3 and 30 features.
time-behav.	stationary	Clusters remain.
	nonstationary	Clusters arbitrarily appear, disappear, and coexist.
	sequential	Clusters replace each other.
outliers	few	Up to 5% outliers.
	medium	Between 5% and 15% outliers.
	many	Between 15% and 40% outliers.
clusters	few	Between 2 and 10 clusters.
	many	Between 11 and 40 clusters.
den.-diffs.	two	Two different distributions[a] and compactness coefficients.
	many	A different distribution[a] and compactness coefficient per cluster.
space	tight	The problem space has a reduced size based on the number of clusters, the number of outliers, and dimensions.
	normal	The problem space is three times bigger per dimension than in the *tight* case.
mov.-clusts.	no	No cluster movement.
	all	All clusters move with arbitrary speeds and directions.
overlap	no	Cluster overlap is avoided.
	yes	Cluster overlap is allowed.

[a]Available distributions are: Uniform, Gaussian, Logistic, Gamma, Triangular, and Ring-Shaped. For each cluster, they can be defined radial or multi-variate.

This means that outliers generated with MDCstream are "statistical anomalies", i.e., data points placed in regions of low spatiotemporal density and not generated by the same distributions used to create the bulk of the data.

Analogously to the insights given by Arbelaitz, Gurrutxaga, Muguerza, Pérez, and Perona (2013) when exploring clustering validation, we find synthetic datasets equally valid as real data for evaluating outlier detection methods, since algorithms can be pushed to the limit for testing specific conditions established by design. Each collection includes 20 different datasets, resulting in a total of 180 datasets.

We have developed a wrapper for MDCstream that simplifies the automatic parameterization and generation of dataset collections according to the aimed test challenges. Table 1 shows the configuration of the wrapper and Table 2 the actual meaning of parameter values.

Collections are:

- **Base**. This group establishes a experiment baseline. It does not implement any specific challenge beyond anomaly detection in multidimensional streaming data per se. It is also devised to set the basic template in which other challenges will be later added. *Base* datasets have between 3 and 30 spatial dimensions, clusters (or classes) remain over time with stable point appearance frequencies, contain few outliers (below 5%), and few clusters (between 2 and 10, each of them with a different cardinality). Clusters do not evolve and cluster-overlap is not allowed; instead, they are placed in a space whose size is relatively consistent with cluster sizes. Therefore, clusters are not too far from and not to close to each other. Fig. 3 shows one of the datasets. Note that it only shows 2 out of 27 dimensions plus time[5]. *Base* datasets are defined allowing two different *cluster densities*, i.e., two different underlying distributions and distribution coefficient sets for the spatial point location in clusters. This is clearly visible in Fig. 3, but note that, since clusters have different cardinalities and they must be stationary, densities should be understood in a spatial–temporal context. This means that cluster-densities are actually more variable than simply two different types.
- **Nonstationary**. The *nonstationary* collection of datasets uses the same configuration as the *base* case, but allows clusters randomly appearing, disappearing, coexisting, and reappearing. Considering the four types of changes over time (or concept drift) discussed by Gama et al. (2014), this scenario implements *sudden* drifts (e.g., context A suddenly disappears and a different context B appears then), *gradual* drifts (e.g., context A progressively disappears and a different context B progressively appears, going through a transition in which both coexist), and *reoccurring contexts* (context A and context B replace each other over time, either abruptly or smoothly). The different types of concept drift can be observed in Fig. 4, which shows one of the *nonstationary* datasets used.
- **Sequential**. This collection is similar to the *nonstationary* one, but here clusters replace each other sequentially, not allowing more than one cluster occurring at the same time. *Sequential* datasets are focused on *sudden* drifts in a much more abrupt manner than *nonstationary* datasets. Fig. 5 shows one of the *sequential* datasets used.
- **Moving**. This dataset collection focuses on the remaining type of concept drift described by Gama et al. (2014), i.e., the *incremental* drift. Here the *base* configuration is taken as baseline, but clusters are allowed to move in any dimension of the space. Fig. 6 shows one of the *moving* datasets used in the evaluation experiments.
- **Medium-outliers**. The remaining data challenges are not related to concept drift, but to geometric properties of data points in spatial dimensions. Specifically, this collection uses the *base* configuration with an increased number of outliers. They are allowed to account for between 5% and 15% of the total number of instances, therefore creating a more noisy space in which clusters are less clearly visible. Hence, their discrimination by analysis algorithms becomes theoretically harder (Gama et al., 2014).
- **Many-outliers**. Here, input spaces are highly corrupted by background noise. Outliers are allowed to be from 15% up to 40% of the total number of instances. Fig. 7 shows one of the *many-outliers* datasets used in the experiments.
- **Close**. The *close* collection uses the *base* configuration, but adds more clusters (between 11 and 40) and force them to coexist in a reduced space, therefore remaining relatively close to each other. Cluster overlap is in any case avoided (or minimized), therefore searching for scenarios in which outliers hover close to clusters, or perhaps in-between neighbor clusters. This scenario is designed to test the finesse of algorithms to perceive local outliers. Fig. 8 shows one of the *close* datasets used in the evaluation experiments.

[5] We are aware of the difficulties and limitations of visualizing time-dependent multidimensional spaces; however, we believe that, although incomplete, plotting the evolution of only two dimensions over time— Figs. 3 to 8 and Figs. 9 to 12—helps to understand the analysis challenges that each studied dataset implies. Note also that the resolution of the plots might sometimes transmit a misleading impression of the density of the space and the proximity of data points (also, outliers are always plotted after inliers, so their prominence is highlighted).

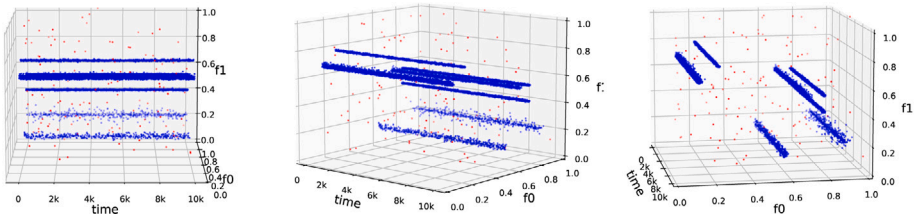


Fig. 3. Visualization of 2 random features (out of 27) throughout time of a *base* dataset. Inliers are shown in blue color and outliers in red color.

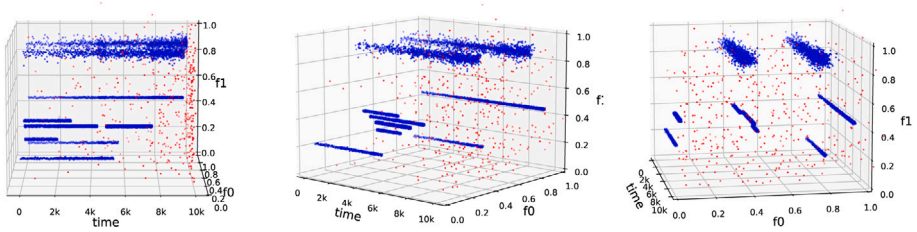


Fig. 4. Visualization of 2 random features (out of 23) throughout time of a *nonstationary* dataset. Inliers are shown in blue color and outliers in red color.

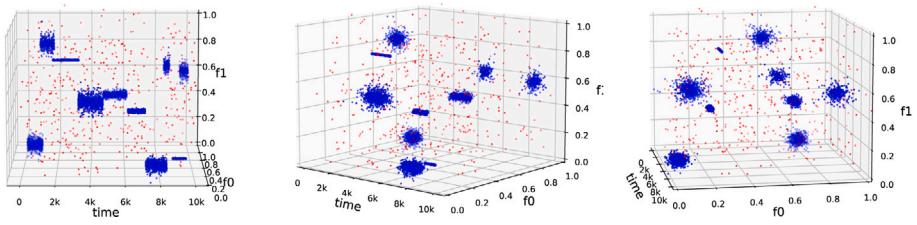


Fig. 5. Visualization of 2 random features (out of 17) throughout time of a *sequential* dataset. Inliers are shown in blue color and outliers in red color.

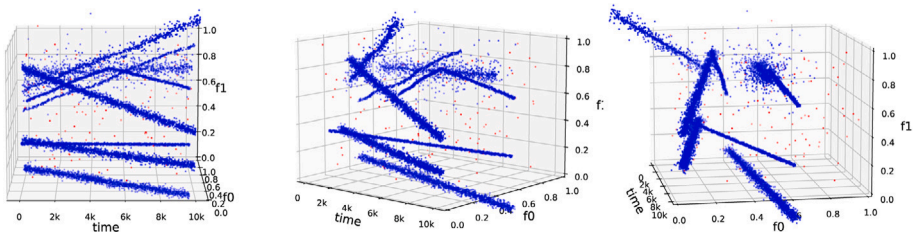


Fig. 6. Visualization of 2 random features (out of 16) throughout time of a *moving* dataset. Inliers are shown in blue color and outliers in red color.

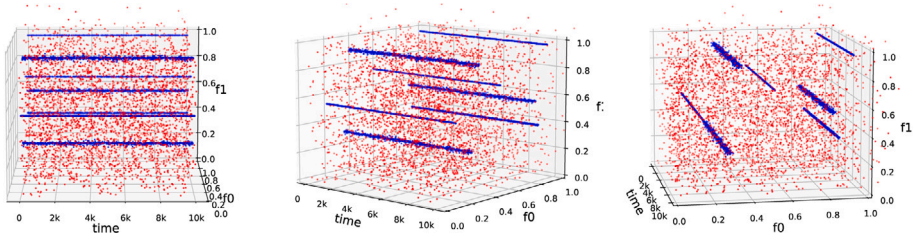


Fig. 7. Visualization of 2 random features (out of 21) throughout time of a *high-outliers* dataset. Inliers are shown in blue color and outliers in red color.

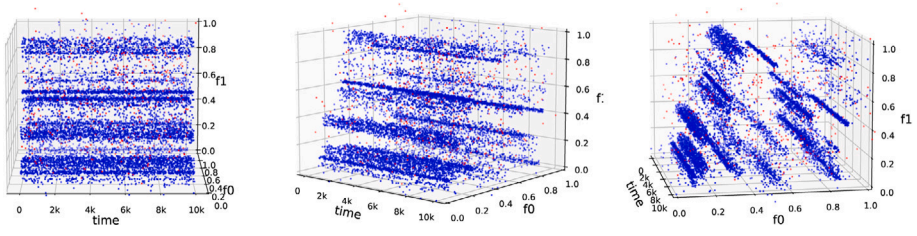


Fig. 8. Visualization of 2 random features (out of 8) throughout time of a *close* dataset. Inliers are shown in blue color and outliers in red color.

- **Density-differences**. Compared to the *base* case, the *density-differences* datasets contain more clusters (between 11 and 40), each of them generated by a different distribution (among 12 possible options), also with different coefficients. Unlike the *close* case, the space is not *reduced*, but *normal*.
- **Overlap**.[6] This last set of datasets is configured as the *density-differences* case, but here cluster overlap is allowed and facilitated.

4.2. Real-application data and memory span

Most stream analysis algorithms are defined with at least one hyperparameter that establishes the *memory*, i.e., how long in the past it recalls data points for the analysis (for instance, a sliding window). Time-related hyperparameters should be adjusted based on the scenario dynamics, meaning that the duration of the memory solely depends on the expert estimation, which decides when past data become obsolete and must be forgotten. This is commonly a subjective aspect and therefore difficult to establish automatically.

Apart from that, the memory is expected to be as long as possible, yet this may be not feasible in practice due to computational costs. Additionally, it affects algorithms differently; for instance, doubling the observation period might severely increase complexity in some cases, whereas the effect is negligible in others. On the other hand, in some algorithms, modifications in such time-related hyperparameters affect the adjustment of other (hyper)parameters. Also note that sometimes a long memory can be counterproductive to deal with concept drift.

In addition to the challenges described in the previous section, we also test the impact of different time-related hyperparameters on the accuracy and run-time performances of methods. To do this, real (or application) data is better suited than mathematically-generated data, since applications often imply underlying periodicities to which algorithms might or might not adapt, thus consequently affecting the classification performance. Therefore, different application data will show different time-behaviors for their intrinsic *normal* shapes as well as for their anomalies.

Note also that, in these datasets, anomalies do not correspond to any formal definition, but are determined by the specific application; i.e., they can be either isolated data points, or a high-density micro-cluster, or the extension of a data-class into a new region of space.

Finally, by using application-related data we obtain deeper insight in how unsupervised algorithms can make the most of pre-knowledge to adjust their models, parameters, and the definition of outlierness. Datasets used are:

- The **CICIDS2017** dataset is provided by the Canadian Institute for Cybersecurity. This is one of the most recent datasets for Network Intrusion Detection (Sharafaldin, Lashkari, & Ghorbani, 2018). The CICIDS2017 dataset was designed to meet 11 quality criteria expected for highly reliable IDS testing. We preprocessed the data to obtain binary-labeled flow-instances: "attack related" and "non-attack related". Selected features correspond to the *OptOut vector*, which has proven to maximize the distance between the given classes when using unsupervised analysis (Iglesias, Hartl, Zseby, & Zimek, 2019). "Attack-related" data points (here taken as anomalies) account for 25.25% data. Fig. 9 shows inliers and outliers in two features over time. Attacks happen in bursts and show higher densities than normal data, which is less homogeneous.
- The **Shuttle** dataset is a well-known dataset used for testing machine learning algorithms. It was originally proposed by Catlett (1991) and later donated to the public domain. It collects numerical features about the diagnosis of subsystems in NASA's Space Shuttle, based on hundreds of thousands of training instances from simulator runs and real flight data. We used the popular version published by Rayana (2016), in which training and test data are combined and the minority classes are merged to form the outlier class (1-label), while the majority class is for the inlier class (0-label). Outliers account for 7.15% of the whole dataset instances. Fig. 10 shows inliers and outliers in two features over time.
- The **Swan-SF** dataset has been recently published in the *Scientific Data* journal by Angryk et al. (2020b), being data extracted from solar photospheric vector magnetograms in Spaceweather HMI Active Region Patch (SHARP) series. The dataset is publicly available (Angryk et al., 2020a). For our experiments, we used the whole dataset and extracted features with the tools provided by Ahmadzadeh and Aydin (2020). We extracted the same set of features that the authors use in their examples and combined flaring (FL) and non-flaring (NF) data. Given the strong class-imbalance, we mapped the majority class into a *normal*-class (0-label) and the rest into an *anomalous*-class (1-label), obtaining a dataset with 17.2% outliers. Fig. 11 shows inliers and outliers in two features over time.
- **Yahoo-TSA** is the abbreviation of the *Yahoo! Synthetic and real time-series with labeled anomalies, v1.0* dataset (Yahoo Webscope Program, 2020), provided by the Yahoo! Webscope program to be used for approved non-commercial research. For our experiments we used the 67 *real* time-series in the *A1Benchmark* folder (henceforth called "parts"). We concatenated parts to form a unique time-series, therefore forcing 67 strong context changes. To keep consistency, the first three data points of each part were additionally labeled as outliers. Although these time-series are originally intended to be independently tackled from an univariate time-series perspective, we express them extracting features that draw evolving spaces and make them suitable for the methods under test. Extracted features are:
  - *Feature 1:* normalized difference between the current value and the previous one.
  - *Feature 2:* normalized difference between the current value and the exponential moving average (with a smoothing factor equal to 2).
  - *Feature 3:* normalized difference between the current value and the simple moving average (calculated over the last 20 datapoints).

  Normalization consists of dividing differences by the simple moving average (over the last 20 data points). In total the dataset contains 1.97% outliers. Fig. 12 shows inliers and outliers in two features over time.

5. Evaluation methodology

In this section we describe the analysis setups and experiments conducted for testing and evaluation. Codes and experiments are available to reuse and replicate in our repository (Iglesias & Hartl, 2021). Datasets (introduced in Section 4) are publicly available in the cited references.

Some aspects narrow down the focus of our study and are worth mentioning: we tackle multidimensional numerical feature spaces that change over time, and we test unsupervised algorithms processing data sequentially, in a way that simultaneity is omitted and the temporal distance between two consecutive data points is fixed to one unit. Note the differences of this setup with respect to anomaly detection in univariate time series., e.g., (Fisch, Eckley, & Fearnhead, 2022), anomaly detection in mixed data, e.g., (Davidow & Matteson, 2022), anomaly detection with incremental supervised methods, e.g., deep learning (Du, Li, Zheng, & Srikumar, 2017), or anomaly pattern detection in streaming data, e.g., (Kim & Park, 2020).

[6] Not to be confused with the overlap index between inliers and outliers defined in Section 6.3 Here, "overlap" refers only to different clusters of inliers.

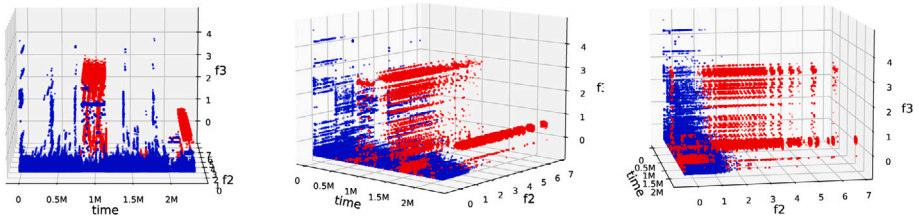


Fig. 9. Visualization of 2 features (out of 5) over time of the CICIDS2017 dataset. Inliers are shown in blue color and outliers in red color.

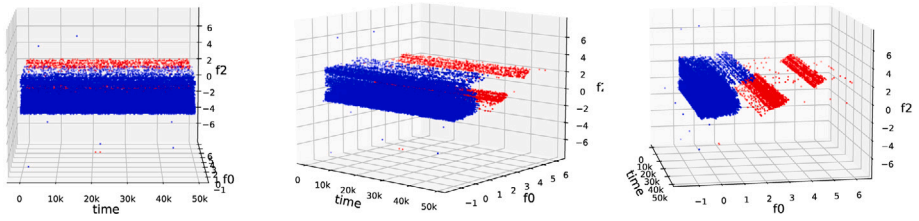


Fig. 10. Visualization of 2 features (out of 9) over time of the Shuttle dataset. Inliers are shown in blue color and outliers in red color.

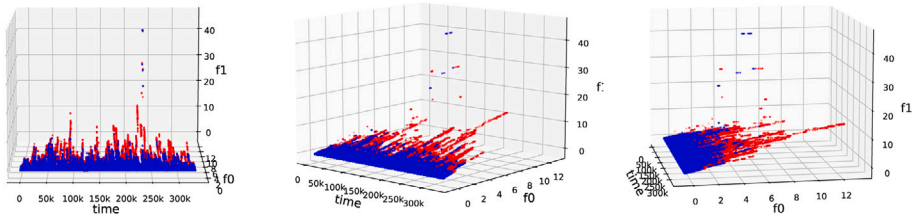


Fig. 11. Visualization of 2 features (out of 12) over time of the Swan dataset. Inliers are shown in blue color and outliers in red color.

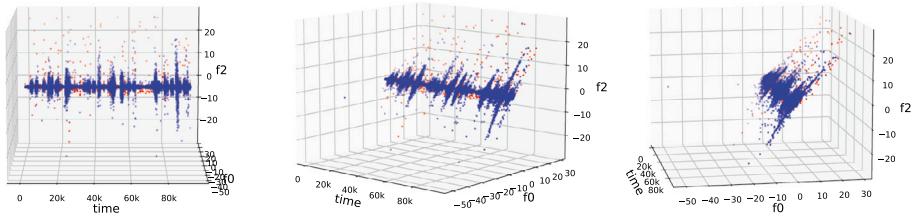


Fig. 12. Visualization of 2 features (out of 3) over time of the Yahoo-TSA dataset. Inliers are shown in blue color and outliers in red color.

5.1. Parameter adjustment

Regardless of supervised or unsupervised, machine learning involves accurate adjustment of learning parameters and hyperparameters. As stated by Snoek, Larochelle, and Adams (2012), this adjustment "is often a 'black art' requiring expert experience, rules of thumb, or sometimes brute-force search". The methods studied in this work show different sensitivity in front of hyperparameter changes; for instance, vanilla SWKNN with a too small or too large R would lead to irrelevant binary classifications, far from its potential capabilities. Keeping default values, using only rules of thumb (which are not always provided), or manually adjusting each algorithm for every experiment would be very costly and still subjective and suboptimal.

Instead, we aimed to reach optimal configurations by automatizing the tuning process. To do this we moved apart the initial 20% data and used it as pre-knowledge, therefore building automated *expert knowledge* with historical data. This takes place in a *training* phase in which algorithms take the pre-knowledge in one batch and are run with different hyperparameter sets. The set that obtains the best performance is later used for the *evaluation* phase. We use labeled data only during the training phase to ensure optimal adjustments, meaning that algorithms adjust to new data as usual during the evaluation phase (i.e., in an unsupervised manner).

This *20%–80% training-evaluation* split (or *past-future* split) in stream analysis is a convention equivalent to the traditional *70%–30% training-test* split used in static supervised classification for obtaining reliable models. Here we assume that the initial 20% of the data should suffice for obtaining hyperparameters that are also suitable for a period four times longer of unseen, future data. The described tuning method guarantees fair comparisons since all algorithms work at their best based on the knowledge obtained from the initial 20% data. However, note that such adjustment could become suboptimal as a result of the emergence of concept drift. Hence, this setup additionally allows to check how robust parameters are when facing such situation. Finally, it is worth mentioning that some of the studied algorithms are able to naturally perform hyperparameter adjustment without requiring labeled data.

5.2. Experimental steps

In the case of **Synthetic Data** (Section 4.1), the experiment goal is to test algorithms when facing different challenges that might be intrinsic to particular streaming data scenarios. We generated the 9 collections of 20 datasets introduced in Section 4, each collection corresponding to a different challenge and one of them being a baseline set (*base*). All datasets have 10,000 data points, from which the first

2,000 were reserved for hyperparameter tuning and algorithm training (Section 5.1), whereas the remaining 8,000 data points formed the evaluation split. The memory hyperparameter T (or length of the time-window) is set to 500 data points for all algorithms and datasets. Steps are:

1. Selection of challenge.
2. Selection of dataset.
3. Selection of algorithm.
4. Training and hyperparameter adjustment. Here we extract best hyperparameters from the initial 2,000 data points (20%). Algorithms were tuned with 30 different random combinations,[7] and later tested in a 3-fold cross-validation setup with stratified sampling. The metric was the *adjusted average precision* (Section 5.3).
5. Evaluation. We applied best hyperparameters and extracted performance indices from the remaining 8,000 data points (80%). In order to remove non-deterministic effects and obtain higher statistical power, each evaluation is repeated 10 times with different random seeds.

For checking the effect of different **Real-application Data and Memory Span** (Section 4.2), experiments use practically the same setup as before, saving the initial 20% data as pre-knowledge. Therefore, the CICIDS2017 dataset is split into 463,584 data points for training and 1,854,338 data points for evaluation; the Yahoo-TSA dataset accounts for 18,975 training and 75,892 evaluation data points; the Swan-SF dataset for 66,237 training and 264,948 evaluation data points; and the Shuttle dataset for 9,820 training and 39,277 evaluation data points.

The main difference when compared with synthetic data experiments is that here we repeat runs with various memory values. Given $T \in \{100, 500, 1\text{k}, 5\text{k}, 100\text{k}, 500\text{k}, 1\text{M}, 5\text{M}\}$, each dataset uses a subset of T with values below its total length. Steps are:

1. Selection of dataset.
2. Selection of T.
3. Selection of algorithm.
4. Training and hyperparameter adjustment.
5. Evaluation with the remaining data points by using best hyperparameters.

5.3. Evaluation metrics

We use the metrics suggested by Campos et al. (2016): precision at n, average precision, and area under the ROC curve. We add the maximum F1 score. In the cited work, *adjusted* indices are introduced for average precision and precision at n, following the procedure introduced by Hubert and Arabie (1985). The adjustment for chance is a standard way of normalizing the metric by the expected value for a random result. Since different outlier ratios in different datasets result in different expected values, adjusted metrics are comparable over results with different outlier ratios.

We show here only *adjusted average precision* to assess accuracy, yet all results and indices are available in our repository, and tables with additional metrics are shown in the Appendix. We select average precision since the Prec–Rec (Precision–Recall) curve is recommended for imbalanced classification (Saito & Rehmsmeier, 2015). Average precision is equivalent to the area under the Prec–Rec curve (Boyd, Eng, & Page, 2013).

Finally, note that methods might not be implemented in their fastest versions, therefore the plain, decontextualized comparison of run-time values would be unfair (Kriegel, Schubert, & Zimek, 2017). Our intention when showing elapsed times is to disclose scalability aspects when the memory hyperparameter is set larger.

[7] Hyperparameter value ranges were set based on recommendations from source references.

6. Results and discussion

In this section we discuss experimental results. The Appendix contains additional performance indices and metrics.

6.1. Streaming data challenges

Fig. 13 shows boxplots for *adjusted average precision* results (abbreviated as `aap`). Critical distance diagrams are also provided in Fig. 14 to compare algorithms by using Wilcoxon–Holm post-hoc analysis tests (Demšar, 2006). We first discuss the results from a global perspective and then approach the challenges one by one.

6.1.1. Accuracy

SWKNN, RSHash and SDOstream tend to show best results in general. The main difference between SWKNN and SDOstream is that, while SWKNN uses the closest points in the observation window, SDOstream uses the closest points in the model (i.e., *observers*). RSHash also shows excellent accuracy, but is more affected by spatial challenges related to clusters (i.e., density differences, close clusters, and cluster overlap) than SWKNN and SDOstream. On the other hand, SDOstream adapts slightly worse to concept drift. In a second level, performances in SWLOF, LODA and RRCT are in general below the methods mentioned before, but still satisfactory. SWLOF is the best at avoiding being mislead in tight spaces in which clusters are too close to each other. SDO drops its performance when concept drift occurs (i.e., *nonstat*, *sequential*, and *moving*). This is expected since it uses fixed learned models, which become obsolete after substantial concept change.

Even in spite of the fact that density differences between inliers and outliers might be clear when the complete timeline is evaluated, the observation window takes comparatively only a few data points and makes such differences less evident. This issue is generally overcome by SWKNN by adjusting a suitable k hyperparameter. On the other hand, RRCT space transformations into tree structures involve an information disruption with regard to point distances that might reduce precision in some cases, requiring higher density differences between inliers and outliers than competitors. Note that the scoring jump between far outliers and close outliers for RRCT is not as pronounced as when using other methods. Also, together with SWLOF, RRCT shows to be the most disturbed by high outlier percentages. LODA shows the best performance among all methods when facing scenarios with high outlier rates, but struggles when facing density differences, close clusters and cluster overlap. As a general rule, methods that resort to some kind of space projection are affected in such scenarios. xStream, while solving relevant issues (e.g., time complexity, high dimensionality, new features), shows hyperparameters strongly sensitive to the adjustments performed during training, thus carrying out space transformations that cannot deal properly with evolving background structures in nonstationary cases.

6.1.2. Challenges

Baseline scenarios (*base*) are satisfactorily managed by all methods. Accuracy drifts are mostly caused either by inlier-labeled data points that were generated by long-tailed distributions or outliers located close to clusters. *Nonstationary* scenarios are also well-solved, but a general performance drop is observed due to cases in which the hyperparameter adjustment becomes suboptimal when data evolves (for example, when clusters in training are considerably denser than new evaluation clusters). LOF and RRCT find *sequential* scenarios slightly easier to solve than *nonstationary* ones because, as clusters appear alone, the inlier/outlier density differences are more pronounced, therefore easier to spot. On the other hand, other methods solve *sequential* datasets worse because they are more sensitive to strong concept changes, which are more radical in *sequential* cases than in *nonstationarity* cases. *Moving* clusters do not imply a particular concept-drift challenge (at least at

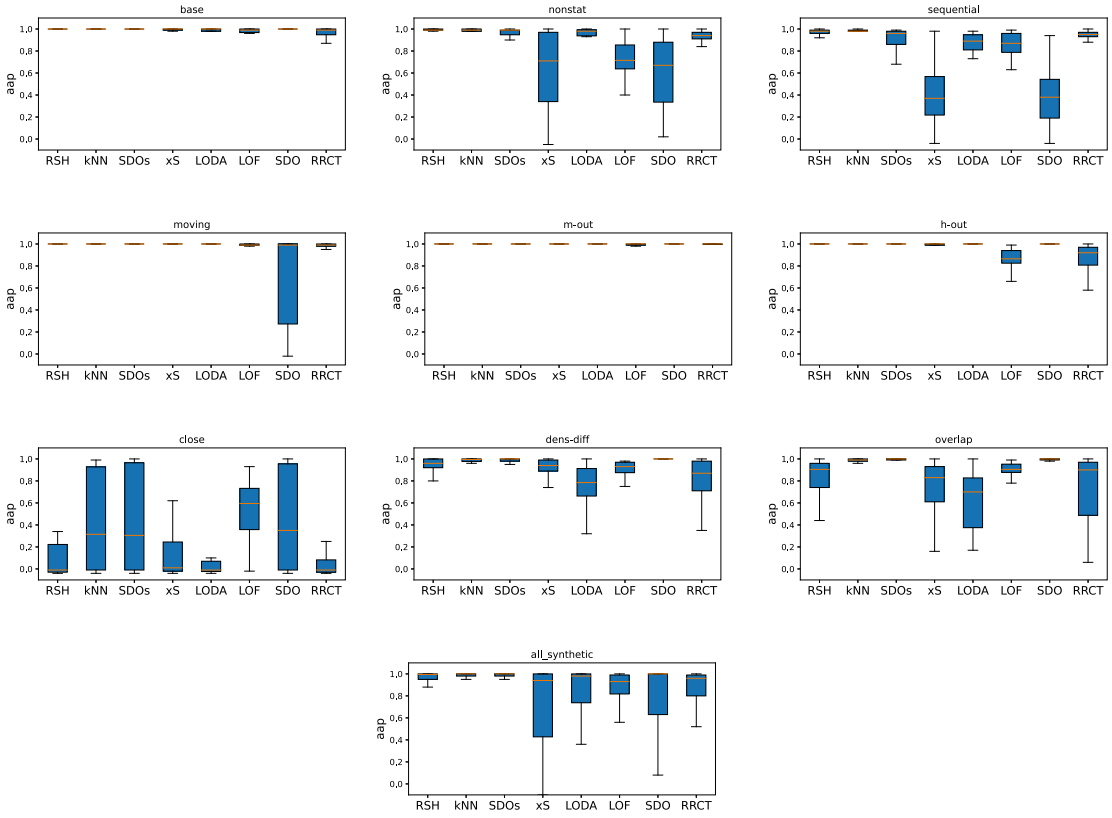


Fig. 13. Boxplots (without fliers) of the *adjusted average precision* (`aap`) obtained for every algorithm and challenge. "RSH" stands for "RSHash", "LOF" for SWLOF, "KNN" for SWKNN, "XS" for xStream, and "SDOs" for SDOstream. The plot at the bottom evaluates all synthetic datasets together.

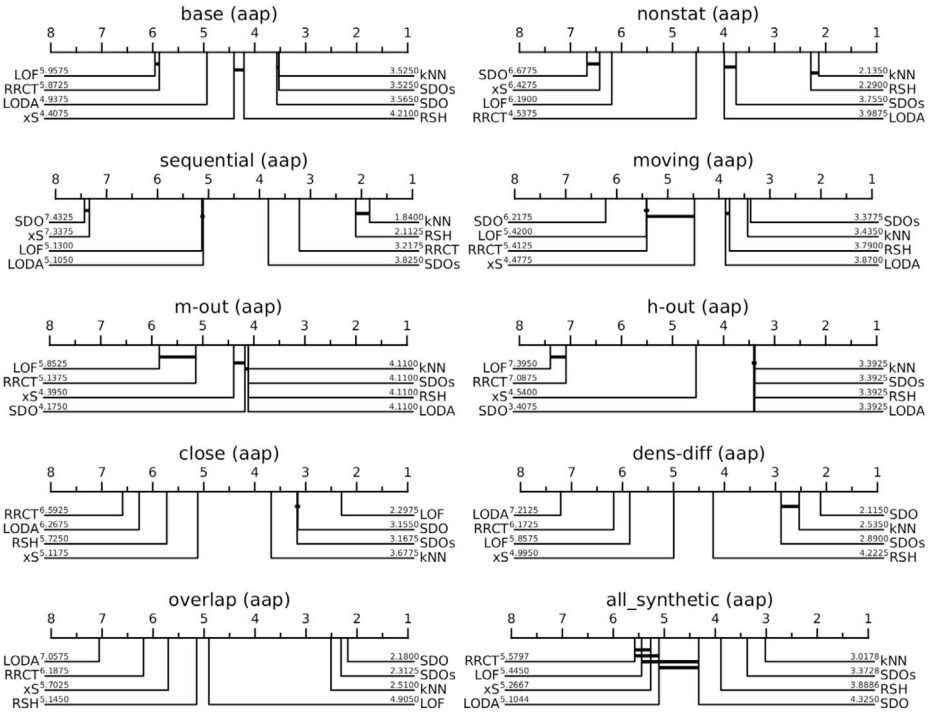


Fig. 14. Critical difference diagrams with the Wilcoxon–Holm post-hoc analysis test on the *adjusted average precision* (`aap`). Adapted from the implementation by Ismail Fawaz, Forestier, Weber, Idoumghar, and Muller (2019). "RSH" stands for "RSHash", "LOF" for SWLOF, "KNN" for SWKNN, "xS" for xStream, and "SDOs" for SDOstream. The plot at the bottom considers all synthetic datasets together.

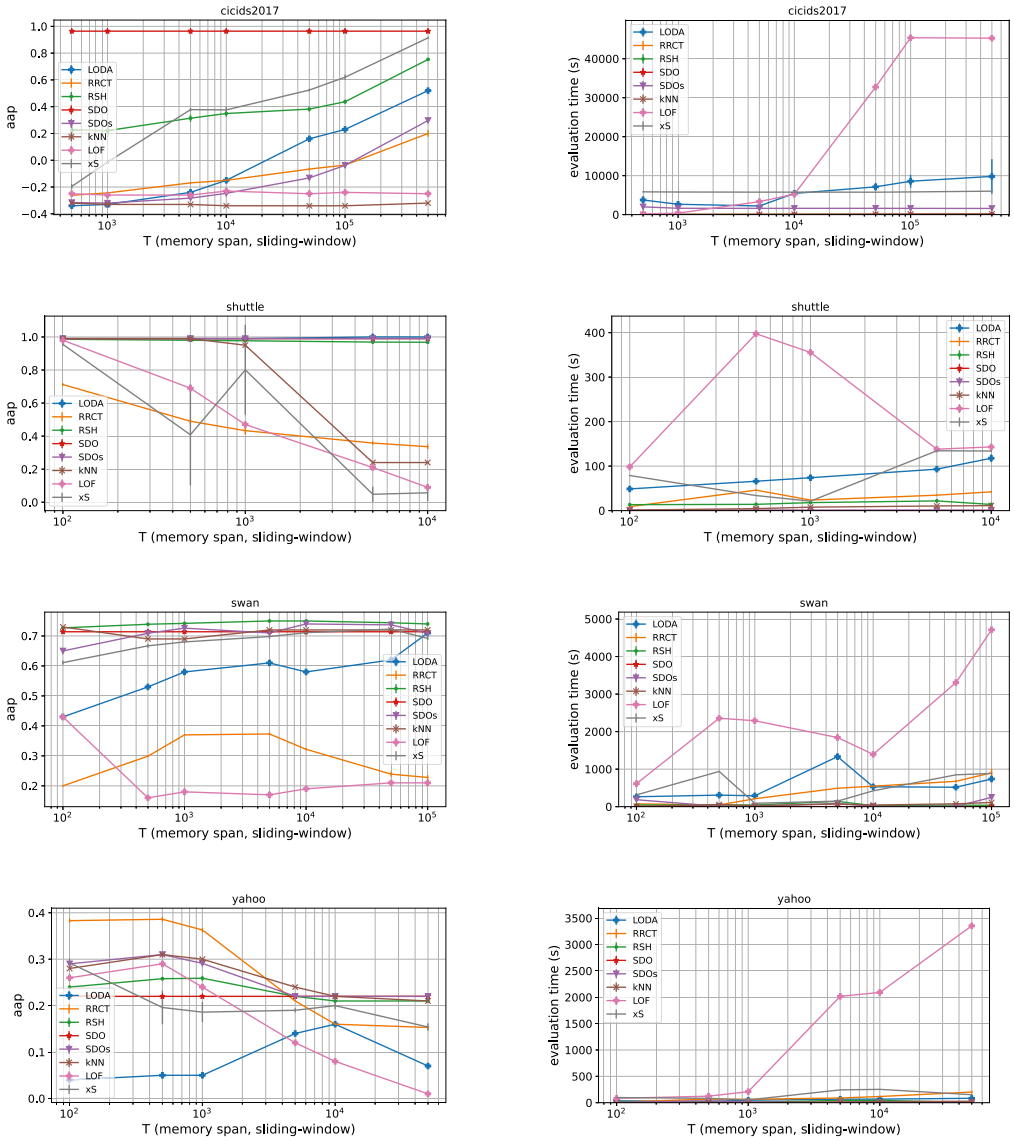


Fig. 15. Left: *Adjusted average precision* (aap) of evaluation data for different memory spans. Right: run-times in seconds of evaluation data for different memory spans.

the given speeds), whereas performances in *medium-outliers* and *many-outliers* experiments are even better than in the baseline cases. This is due to the fact that, by default, MDCstream automatically draws larger data spaces when the outlier rate is higher in order to keep enough separation among data points. Therefore, from an overall perspective, clusters become denser and less data points remain in controversial areas. However, SWLOF and RRCT are affected in the *many-outliers* case because they have a more local understanding of outlierness and are prone to see inliers in the noisy cloud of background points. Implementing *density-differences* and *overlap* in clusters makes setups configured during training more challenging, therefore selected hyperparameters less reliable, except for SDO, SDOstream, and SWKNN, whose parameters are particularly robust. In tight environments (*close*), the problem space contains larger unclear zones, meaning that outliers tend to be closer to inliers, therefore becoming the most challenging cases. The observation window used might be often insufficient to clearly establish inlier/outlier boundaries. In such cases, SWLOF stands out as the best.

6.2. The impact of pre-knowledge and memory span

Fig. 15 shows the results of experiments with real-application datasets and different memory spans. A fact that immediately strikes from the plots is the outstanding performances of SDO in the CICIDS2017 dataset, and SDO, RSHash, and LODA in the Shuttle dataset. This reveals that the main data structures of such datasets, as presented, show stationarity with regard to the data saved for training. This situation happens often in real streaming data, particularly in applications in which normality shows to be stationary, concept drift is minor, and, if it happens, it is controllable or expected by human supervision. In the CICIDS2017 case (related to network traffic anomalies) and the Shuttle case (related to diagnosis of space shuttle systems), non-anomalous shapes are constant throughout all collected data. This means that normality draws a permanent, background profile in the problem space. Such situation is also consistent with the phenomena represented by Swan-SF data (related to sun magnetograms), yet the overlap between normal and anomalous points is stronger here. Instead, in the Yahoo-TS dataset we forced abrupt concept drift by concatenating different time series, therefore any non-updated pre-knowledge learned in a model-based algorithm becomes useless.

We delve into accuracy performances for each dataset and end up with a brief discussion about scalability aspects.

- **CICIDS2017**
  A second insight disclosed by the performance of SDO in this case (also in the Shuttle dataset) is that the meaning of anomaly in such datasets and SDO are consistent to each other. Note that here inliers and outliers have a pragmatical sense. Inliers are shapes or clusters that are dominant (and *harmless* from an application viewpoint), and outliers are a minority of data points (related to *harmful* network activities), which can be isolated, but also appear spontaneously as collective anomalies. SDO is strongly application-oriented and able to identify isolated outliers and anomalous dense clusters likewise, but unable to deal with concept drift.
  On the other hand, the data used for training shows well-represented *normal* shapes and no outliers. Such pre-knowledge contains low information for adjusting hyperparameters in a rigorous way, therefore useless for SWKNN or SWLOF, while it is suitable for defining a relative meaning of anomaly for other methods.
  In this dataset, the larger the memory span is, the more clear overall differences between normal areas and anomalous areas become. Note that SWKNN and SWLOF do not improve their performances with larger memory spans due to their way of measuring outlierness, which only takes into account close areas around the evaluated point. Instead, other methods make the most of more global views of the data context as well as more flexible definitions of outlierness, so they improve their performances as the observation scope increases. Consequently, Fig. 15 (top, left) shows that the longer the memory the better the accuracy, even in spite of the fact that density differences are expected to become more prominent with smaller sample sizes (Zimek, Gaudet, Campello, & Sander, 2013). In this case, larger observation windows help methods like SDOstream, RRCT, RSHash, LODA and xStream not to be tricked by anomalies appearing as bursts or dense small clusters.
- **Shuttle dataset**
  In this dataset the location of the learned normality is key again. Excellent performances of SDO, SDOstream, RSHash and LODA regardless of T attest this fact. Increasing T generally works against the remaining algorithms because anomalies happen to be recurrent, therefore forming clusters that mislead methods that define outlierness locally.
- **Swan-SF dataset**
  Here normality also seems to be stationary and the memory duration does not significantly affect accuracy. The fact that even best `aap` performances are below 0.8 means that a significant number of anomalies are not geometrically separable from inliers in the analyzed feature space (i.e., inliers and outliers overlap). By observing which methods succeed and which ones fail (RRCT, SWLOF), we can infer that outliers are prone to be dense deviations from normal shapes. Fig. 11 corroborates this intuition.
- **Yahoo-TSA dataset**
  This is undoubtedly the most challenging case among the real-application datasets. The overlap between inliers and outliers is considerable and the strong concept drift makes pre-knowledge hardly useful. The evolving shape of this dataset is visible in the pair of features plotted in Fig. 12. Moreover, increasing T is detrimental because algorithms are forced to learn from a diversity of past contexts that are obsolete and do not provide useful information anymore, except for the most recent one. In such situation, it is preferable to reduce the analysis scope to the most immediate context. The method that better deals with the described complexity is RRCT, since its models are less connected to spatial distances, and so its definition of outlierness. In other words, RRCT shows to be the most flexible approach, but still requiring an optimal setting of its memory scope.

As for **Scalability** aspects, larger memory spans cause an increase of the computational effort for instance-based methods like SWLOF (Fig. 15), which might be unaffordable for some applications but, at the very least, is a limiting factor. Also LODA, despite being fast, has a complexity linearly affected by the length of the observation window. In methods like LODA or SWLOF the cost implied by other (hyper)parameters – when adjusted to maximize accuracy – have a noticeable impact in the overall time performance when compared to competitors. Fig. 15 also shows that increasing the memory span does not significantly affect the remaining methods. Particularly fast is SWKNN, revealing that the datasets used in the given format are well suited for M-Tree indexing.

6.3. Key data characteristics

Experiments have revealed that a key aspect that differentiates methods is their way of understanding outlierness according to their *locality* and *relativeness*. This obviously affects the accuracy performance. For estimating the best suited method in a given application, we provide two indices that characterize datasets: outlier-inlier overlap (ϕ) and outlier relative-density (ρ). These indices are inspired by the coefficients defined by Steinbuss and Böhm (2021) for creating realistic synthetic data to compare outlier detection algorithms.

- **Outlier-inlier overlap** (ϕ).
  Given the set of inliers I and the set of outliers O, we define:
  $$\phi = \frac{\int_{-\infty}^{\infty} \min\Big(f_O(x), f_I(x)\Big)\, dx}{\int_{-\infty}^{\infty} \max\Big(f_O(x), f_I(x)\Big)\, dx} \tag{1}$$
  where x is the one-dimensional projection of the problem space driven by Linear Discriminant Analysis (LDA), and $f_O(\cdot)$ and $f_I(\cdot)$ are respectively the probability density functions of (O)utliers and (I)nliers in the LDA projection. Since LDA maximizes class separation, Eq. (1) is a quick estimation of the separation between outlier and inlier subspaces. In our implementation, we use probability histograms as probability density functions. Note that $\phi \in [0, 1]$, with $\phi = 0$ when outlier and inlier spaces are clearly separated, and $\phi = 1$ when they completely overlap.
- **Outlier relative-density** (ρ), defined as:
  $$\rho = \frac{1}{\log_2\Big(1 + \mathrm{IQR}\Big(D_{O,k}\Big)/\mathrm{IQR}\Big(D_{I,k}\Big)\Big)} \tag{2}$$
  where IQR stands for the InterQuartile Range, and $D_{O,k}$ and $D_{I,k}$ are the sets of distances from each outlier and inlier respectively to its k-closest neighbor when considering all points in the dataset. The binary logarithm establishes a reference threshold $\rho = 1$ when the density of outliers equals the density of inliers; it also makes $\rho \in (0, \mathrm{inf})$. $\rho < 1$ is expected for datasets in which outliers are located in low density regions; $\rho > 1$ happens when outliers appear in groups whose density is higher than the normality.

In Fig. 16, ϕ and ρ are calculated for all studied datasets. An approximation to *locality* and *relativeness* can be mapped with such values, but without obviating concept drift and the fact that indices might show dependencies. If the outlier density is foreseen high and the overlap between outliers and inliers is not common ($\rho \uparrow\ \phi \downarrow$), we will opt for high relative methods that are not necessarily local, like LODA, RSHash, or SDO (the last one when concept drift can be discarded); if the overlap increases but outlier density is still high ($\rho \uparrow\ \phi \uparrow$), we need more local methods (e.g., SWLOF); instead, when the density of outliers is not expected to be high ($\rho \downarrow$), methods like SWKNN or SDOstream should be preferred. If the overlap between inliers and outliers areas is predicted to be high, commonly due to strong concept drift ($\phi \uparrow$ concept-drift $\uparrow$), RRCT is the option that stands out. A summary table is shown in Table 3

Table 3
Dataset characteristics contrasted with algorithm performances (aap). Synthetic collections are averaged with the median, whereas for real-application datasets we selected cases with the best performing T.

	SWLOF	SWKNN	xStream	SDO	SDOstream	RRCT	RSHash	LODA	ϕ	ρ	Concept drift
base	0.97	1.00	0.95	0.99	1.00	0.95	0.98	0.96	0.02	0.37	no
nonstat	0.73	0.98	0.63	0.58	0.95	0.92	0.98	0.94	0.05	0.24	yes ↑
seq	0.86	0.96	0.41	0.37	0.89	0.94	0.97	0.83	0.05	0.41	yes ↑↑
mov	0.98	0.99	0.88	0.71	0.99	0.97	0.99	0.99	0.06	0.36	yes ↑
m-out	0.99	1.00	0.93	0.99	1.00	0.98	1.00	0.99	0.04	0.29	no
h-out	0.86	1.00	0.85	0.99	1.00	0.86	1.00	0.99	0.06	0.18	no
close	0.54	0.41	0.19	0.43	0.43	0.08	0.19	0.09	0.34	1.10	no
overlap	0.90	0.97	0.72	0.96	0.97	0.73	0.82	0.65	0.26	0.48	no
dens-diff	0.90	0.98	0.88	0.99	0.98	0.79	0.94	0.70	0.24	0.40	no
shuttle	0.98	0.99	0.95	0.99	0.99	0.71	0.98	0.98	0.02	0.72	no
cicids2017	−0.25	−0.31	0.91	0.96	0.29	0.20	0.75	0.51	0.05	3881.00	no
swan	0.21	0.72	0.69	0.71	0.70	0.22	0.74	0.72	0.20	0.38	no
yahoo	0.25	0.28	0.29	0.22	0.28	0.38	0.24	0.05	0.50	0.52	yes ↑↑

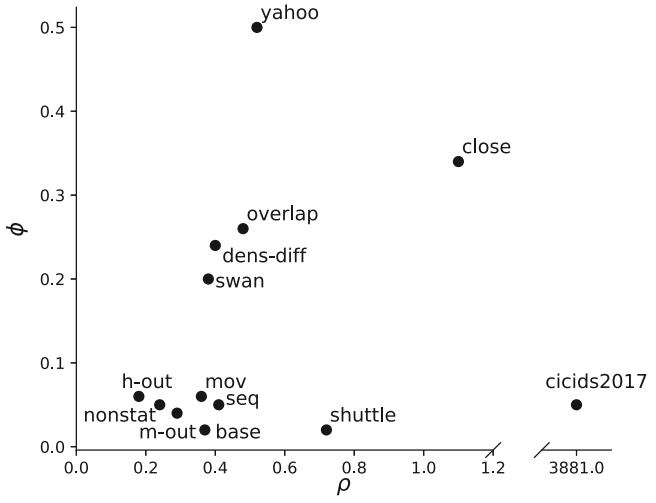


Fig. 16. Outlier-inlier overlap (ϕ) and outlier relative-density (ρ) of datasets used. Indices in dataset collections have been averaged with the median. $k = 3$ for all ρ estimations.

6.4. Method profiles

Finally, from the conducted experiments, we can describe the behavior of the methods in light of the questions formulated in the Introduction, Section 1.

1. **How do specific data characteristics affect algorithm accuracy?** More specifically, what is the response...
   (a) ...in the event of moderate concept drift?
   (b) ...in the event of strong concept drift (high nonstationarity)?
   (c) ...against variable outlier rates?, i.e., if the algorithm is equally suitable when outliers are few or many.
   (d) ...when facing data with different levels of density?
   (e) ...when facing local outliers and overlap between regions of inliers and outliers?
   (f) ...when dealing with high-density, collective or clustered anomalies?
   (g) ...when normality is stationary?
2. **How do algorithms use previous knowledge to fit streaming conditions?** More specifically:
   (a) Is pre-knowledge required for parameter adjustment?
   (b) Is pre-knowledge used to define normality?
   (c) Is the algorithm deterministic or stochastic?
3. **How does the variation of time parameters affect algorithms?** More specifically:
   (a) Does the time window (or memory span) parameter affect accuracy?
   (b) Does the time window (or memory span) parameter affect run-times?
   (c) Are the remaining parameters of the algorithm robust over time? Meaning that, once fixed the memory parameter, should the algorithm be adjusted in the event of data evolution and concept drift?

Table 4 summarizes algorithm profiles according to experiment results and provides a qualitative reference for their selection based on these 13 studied criteria. It is important to keep in mind that there are other factors to consider in the selection of methods; for example, the interpretability of the models (e.g., SDOstream models are easier to interpret than RRCT models), or specific, unique functionalities (e.g., xStream capability to deal with input spaces that vary the number of features).

7. Conclusions

This paper explores state of the art stream outlier detection methods when facing diverse types of challenges that are intrinsic to streaming data. Results disclose insights related to three aspects:

- *Concept drift and data geometries*. Most studied methods deal properly with concept drift and variable outlier ratios, but suffer when facing shrunk spaces with close shapes formed by inliers that overlap or show different densities. SWKNN, RSHash and SDOstream show best performances and seem to cope better with small data point mass than competitors.
- *Use of pre-knowledge*. Algorithms can use training data to properly tune parameters and models. Whereas traditional methods (SWKNN, SWLOF) make the most of strict definitions of outlierness (Schubert et al., 2014), modern proposals (RSHash, RRCT, LODA, SDO, and xStream) build more flexible definitions that further consider the normality imposed by the application. Intermediate approaches are SDOstream and SWKNN (version with non-binary scores).
- *Memory span* is a critical constraint in SWLOF, making it severely shortsighted. To a lesser degree, it is also a limiting factor in LODA. Other methods (SWKNN, SDO, SDOstream, RSHash, RRCT, xStream) can adjust the memory prioritizing application requirements and are less limited by computational costs. However, in practice, the memory span does not only depend on the application dynamics, but also on the available pre-knowledge used for creating models and tune hyperparameters.

Table 4
Summary of method profiles based on the studied questions: (a) *concept drift and data geometries* (seven top rows), (b) *use of pre-knowledge* (three mid rows), (c) *memory span* (three last rows). Keys '−', '+', '++', 'neg.' stand for 'unsuitable', 'suitable', 'excellent', and 'negligible' respectively.

	SWLOF	SWKNN	xStream	SDO	SDOstream	RRCT	RSHash	LODA
Moderate concept drift	+	++	−	−	++	++	++	+
Strong concept drift	+	+	−	−	+	++	+	+
Variable outlier rates	+	++	++	++	++	+	++	++
Variable density levels	+	++	+	++	++	−	+	−
Local outliers and overlap with inliers	++	+	−	+	+	+	−	−
Collective anomalies	−	−	++	++	−	−	+	+
Normality is stationary	−	−	+	++	+	−	+	+
Tuning depends on pre-knowledge	no	low	high	low	low	low	high	low
Pre-knowledge defines normality	no	no	yes	yes	low	low	yes	low
(D)eterministic or (S)tochastic	D	D	S	S	S	S	S	S
Accuracy is affected by the memory span	high	low	high	no	high	high	low	high
Run-time is affected by the memory span	yes	neg.	low	neg.	neg.	low	neg.	low
Parameter robustness in concept drift	high	high	low	high	high	high	high	high

Table 5
Accuracy performances of the **base** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.87 ± 0.10	0.87 ± 0.10	0.97 ± 0.03	0.97 ± 0.03	0.89 ± 0.09	0.89 ± 0.09	0.98 ± 0.01
kNN	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
xS	0.94 ± 0.18	0.94 ± 0.18	0.95 ± 0.17	0.95 ± 0.18	0.95 ± 0.13	0.95 ± 0.13	0.99 ± 0.01
SDO	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.00
SDOs	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
RRCT	0.91 ± 0.09	0.91 ± 0.09	0.95 ± 0.08	0.95 ± 0.08	0.93 ± 0.08	0.92 ± 0.08	0.99 ± 0.00
RSHash	0.97 ± 0.06	0.97 ± 0.06	0.98 ± 0.04	0.98 ± 0.04	0.97 ± 0.05	0.97 ± 0.05	1.00 ± 0.00
LODA	0.93 ± 0.12	0.93 ± 0.12	0.96 ± 0.10	0.96 ± 0.10	0.93 ± 0.11	0.93 ± 0.11	0.99 ± 0.00

Summarizing the remarkable properties of the methods one by one: SDO is the most reliable when concept drift can be neglected and normality is well-defined by historical data. SWLOF is the most suited for overpopulated, narrow spaces in which application goals require a strongly local approach. RRCT is the most adaptive at the expense of losing accuracy. xStream also sacrifices accuracy and adaptability for solving high-dimensionality and gaining the capability of dealing with new dimensions added during application phases. LODA is a relative, global method with greater ability to highlight concentrations of inliers and, therefore, the best method to discriminate clusters within noisy backgrounds populated by outliers. Finally, SWKNN, RSHash, and SDOstream show to be the most accurate in general, and require less data to achieve such accuracy. Deciding between these three options is fundamentally a matter of weighting locality and relativeness: SWKNN is the most local and strict, RSHash the most global and relative, and SDOstream remains in between.

Therefore, figuring out if the problem under analysis requires a **strict** (i.e., shapes in pre-knowledge are not determining to define normality) or a **relative** perspective (i.e., shapes in pre-knowledge are key to define normality) is also an important factor that affects the selection of the methods and the success of the outlier detection task. Many applications make the most of known data shapes that remain over time, then favoring relative methods; on the other hand, relative methods are strongly affected by concept drift whenever new shapes are expected to be automatically seen as the new normality. We have additionally proposed two indices to characterize datasets: outlier-inlier overlap (ϕ) and outlier relative-density (ρ), which help to identify the most suited method for a given data scenario.

Beyond their interpretation of outlierness as local, global, strict or relative, other characteristics must be carefully considered and studied for method selection; for instance, the hyperparameter robustness and ease of adjustment, as well as the inertia of parameters and learned models when facing variations in data contexts.

CRediT authorship contribution statement

Félix Iglesias Vázquez: Conceptualization, Methodology, Investigation, Formal analysis, Writing – original draft. **Alexander Hartl:** Software, Validation, Writing – review & editing. **Tanja Zseby:** Supervision, Writing – review & editing. **Arthur Zimek:** Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data, codes and experiments are available for reproducibility and reuse in open repositories cited in the article.

Data for Evaluation of Stream Data Analysis Algorithms (Original data) (Mendeley Data)
SWAN-SF (Reference data) (Dataverse)
Synthetic and real time-series with labeled anomalies (Reference data) (Webscope Yahoo Labs)
Intrusion Detection Evaluation Dataset (CIC-IDS2017) (Reference data) (UNB IDS Datasets)

Funding

This work was partly supported by the project MALware cOmmunication in cRitical Infrastructures (MALORI), funded by the Austrian security research programme KIRAS of the Federal Ministry for Agriculture, Regions and Tourism (BMLRT) under grant no. 873511.

The authors acknowledge TU Wien Bibliothek for financial support through its Open Access Funding Programme.

Appendix

In Tables 5–17, metrics are abbreviated as: precision at n (`pan`), adjusted precision at n (`apan`), average precision (`ap`), adjusted average precision (`aap`), maximum F1 score (`mf1`), adjusted maximum F1 score (`amf1`), and area under the ROC curve (`roc-auc`). Algorithm abbreviations are: "RSH" for "RSHash", "LOF" for SWLOF, "KNN" for SWKNN, "XS" for xStream, and "SDOs"for SDOstream".

Table 6
Accuracy performances of the **non-stationarity** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.46 ± 0.20	0.44 ± 0.21	0.74 ± 0.15	0.73 ± 0.15	0.57 ± 0.18	0.55 ± 0.19	0.68 ± 0.15
kNN	0.99 ± 0.00	0.99 ± 0.00	0.98 ± 0.01	0.98 ± 0.01	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
xS	0.51 ± 0.32	0.49 ± 0.33	0.64 ± 0.31	0.63 ± 0.32	0.71 ± 0.23	0.70 ± 0.24	0.95 ± 0.09
SDO	0.38 ± 0.30	0.36 ± 0.31	0.59 ± 0.32	0.58 ± 0.33	0.49 ± 0.29	0.48 ± 0.29	0.84 ± 0.10
SDOs	0.96 ± 0.03	0.96 ± 0.03	0.95 ± 0.05	0.95 ± 0.05	0.97 ± 0.02	0.97 ± 0.02	1.00 ± 0.00
RRCT	0.81 ± 0.11	0.80 ± 0.11	0.93 ± 0.05	0.92 ± 0.06	0.83 ± 0.10	0.82 ± 0.11	0.99 ± 0.01
RSHash	0.97 ± 0.03	0.97 ± 0.03	0.98 ± 0.02	0.98 ± 0.02	0.97 ± 0.03	0.97 ± 0.03	1.00 ± 0.00
LODA	0.85 ± 0.14	0.85 ± 0.14	0.94 ± 0.07	0.94 ± 0.07	0.87 ± 0.13	0.86 ± 0.14	0.99 ± 0.02

Table 7
Accuracy performances of the **sequential** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.71 ± 0.16	0.70 ± 0.17	0.86 ± 0.09	0.86 ± 0.09	0.77 ± 0.14	0.76 ± 0.14	0.90 ± 0.12
kNN	0.96 ± 0.04	0.96 ± 0.04	0.97 ± 0.03	0.96 ± 0.03	0.97 ± 0.03	0.97 ± 0.03	1.00 ± 0.00
xS	0.28 ± 0.22	0.25 ± 0.23	0.43 ± 0.26	0.41 ± 0.27	0.42 ± 0.20	0.40 ± 0.21	0.90 ± 0.09
SDO	0.19 ± 0.14	0.17 ± 0.15	0.39 ± 0.23	0.37 ± 0.24	0.26 ± 0.17	0.24 ± 0.17	0.60 ± 0.13
SDOs	0.86 ± 0.19	0.86 ± 0.20	0.89 ± 0.17	0.89 ± 0.17	0.90 ± 0.14	0.90 ± 0.14	0.99 ± 0.00
RRCT	0.88 ± 0.06	0.88 ± 0.06	0.95 ± 0.03	0.94 ± 0.03	0.89 ± 0.06	0.89 ± 0.06	0.99 ± 0.00
RSHash	0.95 ± 0.04	0.95 ± 0.04	0.97 ± 0.02	0.97 ± 0.02	0.96 ± 0.03	0.96 ± 0.03	1.00 ± 0.00
LODA	0.75 ± 0.14	0.74 ± 0.15	0.83 ± 0.12	0.83 ± 0.13	0.80 ± 0.12	0.79 ± 0.12	0.99 ± 0.00

Table 8
Accuracy performances of the **moving** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.90 ± 0.08	0.90 ± 0.08	0.98 ± 0.02	0.98 ± 0.02	0.93 ± 0.05	0.93 ± 0.05	0.96 ± 0.05
kNN	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
xS	0.86 ± 0.30	0.85 ± 0.31	0.88 ± 0.30	0.88 ± 0.30	0.89 ± 0.23	0.88 ± 0.24	0.98 ± 0.05
SDO	0.64 ± 0.40	0.63 ± 0.40	0.71 ± 0.40	0.71 ± 0.41	0.70 ± 0.34	0.70 ± 0.34	0.96 ± 0.04
SDOs	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
RRCT	0.93 ± 0.08	0.93 ± 0.08	0.97 ± 0.06	0.97 ± 0.06	0.94 ± 0.07	0.94 ± 0.07	1.00 ± 0.00
RSHash	0.98 ± 0.03	0.98 ± 0.03	0.99 ± 0.00	0.99 ± 0.00	0.98 ± 0.03	0.98 ± 0.03	0.99 ± 0.00
LODA	0.96 ± 0.05	0.96 ± 0.06	0.99 ± 0.01	0.99 ± 0.01	0.96 ± 0.05	0.96 ± 0.05	1.00 ± 0.00

Table 9
Accuracy performances of the **medium-outliers** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.90 ± 0.05	0.89 ± 0.06	0.99 ± 0.00	0.99 ± 0.01	0.94 ± 0.03	0.94 ± 0.04	0.97 ± 0.03
kNN	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
xS	0.94 ± 0.21	0.93 ± 0.23	0.94 ± 0.21	0.93 ± 0.23	0.96 ± 0.15	0.95 ± 0.17	0.98 ± 0.08
SDO	0.99 ± 0.05	0.99 ± 0.06	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.00
SDOs	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
RRCT	0.97 ± 0.03	0.96 ± 0.04	0.99 ± 0.02	0.98 ± 0.02	0.97 ± 0.03	0.97 ± 0.03	0.99 ± 0.00
RSHash	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
LODA	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.01	1.00 ± 0.00

Table 10
Accuracy performances of the **high-outliers** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.63 ± 0.10	0.49 ± 0.17	0.90 ± 0.05	0.86 ± 0.08	0.75 ± 0.08	0.65 ± 0.13	0.83 ± 0.09
kNN	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
xS	0.90 ± 0.19	0.87 ± 0.26	0.89 ± 0.23	0.85 ± 0.32	0.93 ± 0.13	0.91 ± 0.18	0.95 ± 0.12
SDO	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00
SDOs	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
RRCT	0.86 ± 0.09	0.80 ± 0.14	0.90 ± 0.09	0.86 ± 0.13	0.88 ± 0.07	0.83 ± 0.11	0.96 ± 0.04
RSHash	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
LODA	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	1.00 ± 0.00

Table 11
Accuracy performances of the **close** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.43 ± 0.18	0.41 ± 0.19	0.55 ± 0.25	0.54 ± 0.26	0.45 ± 0.18	0.43 ± 0.18	0.90 ± 0.08
kNN	0.44 ± 0.36	0.43 ± 0.37	0.42 ± 0.41	0.41 ± 0.42	0.59 ± 0.26	0.58 ± 0.26	0.96 ± 0.05
xS	0.19 ± 0.25	0.17 ± 0.25	0.22 ± 0.32	0.19 ± 0.33	0.27 ± 0.22	0.25 ± 0.22	0.78 ± 0.15
SDO	0.47 ± 0.36	0.46 ± 0.37	0.45 ± 0.42	0.43 ± 0.43	0.62 ± 0.25	0.61 ± 0.25	0.96 ± 0.05
SDOs	0.46 ± 0.37	0.45 ± 0.38	0.45 ± 0.42	0.43 ± 0.43	0.62 ± 0.26	0.61 ± 0.26	0.96 ± 0.05
RRCT	0.12 ± 0.20	0.10 ± 0.21	0.11 ± 0.22	0.08 ± 0.22	0.28 ± 0.18	0.26 ± 0.18	0.88 ± 0.08
RSHash	0.19 ± 0.28	0.17 ± 0.28	0.21 ± 0.34	0.19 ± 0.35	0.30 ± 0.23	0.29 ± 0.23	0.86 ± 0.10
LODA	0.10 ± 0.16	0.08 ± 0.16	0.11 ± 0.21	0.09 ± 0.22	0.20 ± 0.13	0.18 ± 0.13	0.82 ± 0.09

Table 12
Accuracy performances of the **density-differences** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.75 ± 0.10	0.75 ± 0.10	0.91 ± 0.06	0.90 ± 0.06	0.78 ± 0.10	0.77 ± 0.10	0.95 ± 0.03
kNN	0.98 ± 0.02	0.98 ± 0.02	0.98 ± 0.01	0.98 ± 0.01	0.99 ± 0.01	0.99 ± 0.01	1.00 ± 0.00
xS	0.79 ± 0.22	0.78 ± 0.22	0.88 ± 0.22	0.88 ± 0.22	0.81 ± 0.18	0.81 ± 0.19	0.98 ± 0.02
SDO	0.98 ± 0.07	0.97 ± 0.07	0.99 ± 0.04	0.99 ± 0.04	0.99 ± 0.03	0.98 ± 0.03	0.99 ± 0.00
SDOs	0.97 ± 0.03	0.97 ± 0.03	0.98 ± 0.02	0.98 ± 0.03	0.98 ± 0.01	0.98 ± 0.01	1.00 ± 0.00
RRCT	0.78 ± 0.19	0.78 ± 0.19	0.80 ± 0.21	0.79 ± 0.21	0.83 ± 0.15	0.82 ± 0.15	0.99 ± 0.00
RSHash	0.88 ± 0.10	0.87 ± 0.10	0.94 ± 0.06	0.94 ± 0.06	0.89 ± 0.09	0.89 ± 0.09	0.99 ± 0.00
LODA	0.61 ± 0.22	0.60 ± 0.23	0.71 ± 0.22	0.70 ± 0.23	0.63 ± 0.21	0.62 ± 0.21	0.97 ± 0.01

Table 13
Accuracy performances of the **overlap** dataset collection.

	pan	apan	ap	aap	mf1	amf1	roc-auc
LOF	0.74 ± 0.09	0.73 ± 0.09	0.91 ± 0.05	0.90 ± 0.05	0.76 ± 0.08	0.76 ± 0.09	0.94 ± 0.05
kNN	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.06	0.97 ± 0.06	0.98 ± 0.03	0.98 ± 0.03	1.00 ± 0.00
xS	0.66 ± 0.26	0.65 ± 0.27	0.73 ± 0.28	0.72 ± 0.29	0.70 ± 0.22	0.69 ± 0.23	0.95 ± 0.10
SDO	0.96 ± 0.09	0.96 ± 0.09	0.96 ± 0.08	0.96 ± 0.08	0.97 ± 0.06	0.97 ± 0.06	0.99 ± 0.00
SDOs	0.97 ± 0.04	0.97 ± 0.04	0.97 ± 0.06	0.97 ± 0.06	0.98 ± 0.03	0.98 ± 0.03	1.00 ± 0.00
RRCT	0.74 ± 0.21	0.74 ± 0.21	0.73 ± 0.27	0.73 ± 0.28	0.80 ± 0.15	0.80 ± 0.15	0.99 ± 0.00
RSHash	0.76 ± 0.16	0.76 ± 0.16	0.82 ± 0.19	0.82 ± 0.19	0.79 ± 0.14	0.79 ± 0.14	0.99 ± 0.01
LODA	0.59 ± 0.21	0.57 ± 0.21	0.65 ± 0.24	0.65 ± 0.24	0.62 ± 0.19	0.61 ± 0.19	0.97 ± 0.01

Table 14
Accuracy performances of the **Shuttle** dataset.

T	alg	pan	apan	ap	aap	mf1	amf1	roc-auc
T100	LOF	0.93 ± 0.00	0.93 ± 0.00	0.98 ± 0.00	0.98 ± 0.00	0.93 ± 0.00	0.93 ± 0.00	0.99 ± 0.00
	kNN	0.93 ± 0.00	0.93 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
	xS	0.90 ± 0.01	0.89 ± 0.01	0.96 ± 0.00	0.95 ± 0.00	0.90 ± 0.01	0.89 ± 0.01	0.99 ± 0.00
	SDO	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	SDOs	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	RRCT	0.64 ± 0.01	0.61 ± 0.01	0.73 ± 0.00	0.71 ± 0.00	0.66 ± 0.00	0.63 ± 0.01	0.96 ± 0.00
	RSH	0.94 ± 0.01	0.94 ± 0.01	0.98 ± 0.00	0.98 ± 0.00	0.95 ± 0.02	0.95 ± 0.02	0.99 ± 0.00
	LODA	0.93 ± 0.00	0.93 ± 0.01	0.98 ± 0.00	0.98 ± 0.00	0.94 ± 0.01	0.94 ± 0.01	0.99 ± 0.00
T500	LOF	0.51 ± 0.00	0.46 ± 0.00	0.72 ± 0.00	0.68 ± 0.00	0.51 ± 0.00	0.48 ± 0.00	0.82 ± 0.00
	kNN	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	xS	0.61 ± 0.21	0.58 ± 0.23	0.45 ± 0.28	0.40 ± 0.30	0.78 ± 0.08	0.76 ± 0.08	0.96 ± 0.01
	SDO	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	SDOs	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	RRCT	0.57 ± 0.01	0.53 ± 0.01	0.52 ± 0.01	0.48 ± 0.01	0.65 ± 0.01	0.62 ± 0.01	0.96 ± 0.00
	RSH	0.96 ± 0.00	0.95 ± 0.00	0.98 ± 0.00	0.98 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
	LODA	0.95 ± 0.00	0.95 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
T1K	LOF	0.28 ± 0.00	0.24 ± 0.00	0.51 ± 0.00	0.46 ± 0.00	0.31 ± 0.00	0.27 ± 0.00	0.57 ± 0.00
	kNN	0.82 ± 0.00	0.81 ± 0.00	0.94 ± 0.00	0.94 ± 0.00	0.82 ± 0.00	0.81 ± 0.00	0.98 ± 0.00
	xS	0.83 ± 0.17	0.82 ± 0.18	0.81 ± 0.25	0.80 ± 0.27	0.88 ± 0.09	0.87 ± 0.10	0.98 ± 0.01
	SDO	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	SDOs	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	RRCT	0.52 ± 0.02	0.49 ± 0.02	0.47 ± 0.02	0.43 ± 0.02	0.64 ± 0.02	0.61 ± 0.02	0.95 ± 0.00
	RSH	0.96 ± 0.00	0.95 ± 0.00	0.98 ± 0.00	0.97 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
	LODA	0.95 ± 0.00	0.95 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
T5K	LOF	0.15 ± 0.00	0.09 ± 0.00	0.25 ± 0.00	0.21 ± 0.00	0.15 ± 0.00	0.10 ± 0.00	0.53 ± 0.00
	kNN	0.19 ± 0.00	0.12 ± 0.00	0.28 ± 0.00	0.24 ± 0.00	0.39 ± 0.00	0.35 ± 0.00	0.86 ± 0.00
	xS	0.08 ± 0.01	0.01 ± 0.01	0.11 ± 0.04	0.04 ± 0.04	0.36 ± 0.07	0.31 ± 0.07	0.78 ± 0.05
	SDO	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	SDOs	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	RRCT	0.44 ± 0.01	0.40 ± 0.01	0.40 ± 0.00	0.35 ± 0.01	0.60 ± 0.02	0.57 ± 0.02	0.94 ± 0.00
	RSH	0.96 ± 0.00	0.95 ± 0.00	0.97 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
	LODA	0.95 ± 0.00	0.95 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
T10K	LOF	0.10 ± 0.00	0.03 ± 0.00	0.15 ± 0.00	0.09 ± 0.00	0.14 ± 0.00	0.09 ± 0.00	0.55 ± 0.00
	kNN	0.14 ± 0.00	0.07 ± 0.00	0.28 ± 0.00	0.24 ± 0.00	0.33 ± 0.00	0.28 ± 0.00	0.81 ± 0.00
	xS	0.09 ± 0.02	0.02 ± 0.02	0.12 ± 0.04	0.05 ± 0.05	0.22 ± 0.04	0.16 ± 0.04	0.64 ± 0.05
	SDO	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	SDOs	0.94 ± 0.00	0.94 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00
	RRCT	0.41 ± 0.02	0.37 ± 0.02	0.38 ± 0.01	0.33 ± 0.01	0.58 ± 0.02	0.55 ± 0.02	0.94 ± 0.00
	RSH	0.96 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	0.97 ± 0.00	0.96 ± 0.00	0.99 ± 0.00
	LODA	0.95 ± 0.00	0.95 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.00

Table 15
Accuracy performances of the **CICIDS2017** dataset.

T	alg	pan	apan	ap	aap	mf1	amf1	roc-auc
T500	LOF	0.19 ± 0.00	−0.18 ± 0.00	0.14 ± 0.00	−0.25 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.34 ± 0.00
	kNN	0.18 ± 0.00	−0.20 ± 0.00	0.10 ± 0.00	−0.31 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.37 ± 0.00
	xS	0.13 ± 0.00	−0.26 ± 0.01	0.18 ± 0.01	−0.19 ± 0.01	0.48 ± 0.00	0.24 ± 0.00	0.23 ± 0.02
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.11 ± 0.00	−0.29 ± 0.00	0.10 ± 0.00	−0.31 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.31 ± 0.00
	RRCT	0.14 ± 0.00	−0.24 ± 0.00	0.13 ± 0.00	−0.26 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.23 ± 0.00
	RSH	0.50 ± 0.03	0.27 ± 0.04	0.47 ± 0.03	0.22 ± 0.04	0.52 ± 0.02	0.30 ± 0.03	0.52 ± 0.01
	LODA	0.09 ± 0.00	−0.33 ± 0.00	0.07 ± 0.00	−0.34 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.25 ± 0.00
T1K	LOF	0.19 ± 0.00	−0.19 ± 0.00	0.12 ± 0.00	−0.25 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.34 ± 0.00
	kNN	0.17 ± 0.00	−0.22 ± 0.00	0.09 ± 0.00	−0.33 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.37 ± 0.00
	xS	0.21 ± 0.00	−0.15 ± 0.01	0.30 ± 0.01	−0.01 ± 0.01	0.48 ± 0.00	0.24 ± 0.00	0.31 ± 0.01
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.11 ± 0.00	−0.28 ± 0.00	0.10 ± 0.00	−0.31 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.31 ± 0.00
	RRCT	0.15 ± 0.00	−0.23 ± 0.00	0.14 ± 0.00	−0.24 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.24 ± 0.00
	RSH	0.48 ± 0.03	0.25 ± 0.05	0.46 ± 0.02	0.22 ± 0.04	0.52 ± 0.02	0.30 ± 0.02	0.52 ± 0.01
	LODA	0.11 ± 0.00	−0.31 ± 0.00	0.09 ± 0.00	−0.33 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.25 ± 0.00
T5K	LOF	0.18 ± 0.00	−0.19 ± 0.00	0.14 ± 0.00	−0.25 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.34 ± 0.00
	kNN	0.15 ± 0.00	−0.23 ± 0.00	0.09 ± 0.00	−0.33 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.36 ± 0.00
	xS	0.46 ± 0.01	0.21 ± 0.01	0.57 ± 0.01	0.37 ± 0.01	0.49 ± 0.00	0.26 ± 0.01	0.61 ± 0.01
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.09 ± 0.00	−0.33 ± 0.00	0.12 ± 0.00	−0.28 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.29 ± 0.00
	RRCT	0.18 ± 0.00	−0.19 ± 0.00	0.20 ± 0.00	−0.16 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.28 ± 0.00
	RSH	0.53 ± 0.02	0.32 ± 0.04	0.53 ± 0.01	0.31 ± 0.02	0.55 ± 0.01	0.34 ± 0.02	0.58 ± 0.01
	LODA	0.18 ± 0.00	−0.20 ± 0.00	0.14 ± 0.00	−0.24 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.31 ± 0.00
T10K	LOF	0.18 ± 0.00	−0.19 ± 0.00	0.15 ± 0.00	−0.23 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.34 ± 0.00
	kNN	0.15 ± 0.00	−0.22 ± 0.00	0.09 ± 0.00	−0.34 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.36 ± 0.00
	xS	0.52 ± 0.02	0.30 ± 0.03	0.57 ± 0.01	0.37 ± 0.02	0.57 ± 0.02	0.37 ± 0.03	0.66 ± 0.01
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.12 ± 0.00	−0.28 ± 0.00	0.14 ± 0.00	−0.24 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.28 ± 0.00
	RRCT	0.19 ± 0.00	−0.17 ± 0.00	0.21 ± 0.00	−0.15 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.29 ± 0.00
	RSH	0.55 ± 0.02	0.35 ± 0.03	0.55 ± 0.01	0.34 ± 0.02	0.56 ± 0.01	0.36 ± 0.03	0.61 ± 0.01
	LODA	0.24 ± 0.00	−0.11 ± 0.00	0.21 ± 0.00	−0.14 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.39 ± 0.00
T50K	LOF	0.19 ± 0.00	−0.19 ± 0.00	0.14 ± 0.00	−0.25 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.34 ± 0.00
	kNN	0.17 ± 0.00	−0.21 ± 0.00	0.07 ± 0.00	−0.34 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.37 ± 0.00
	xS	0.55 ± 0.01	0.34 ± 0.02	0.67 ± 0.00	0.52 ± 0.01	0.64 ± 0.01	0.47 ± 0.01	0.77 ± 0.00
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.17 ± 0.00	−0.20 ± 0.00	0.22 ± 0.00	−0.13 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.28 ± 0.00
	RRCT	0.24 ± 0.00	−0.10 ± 0.00	0.27 ± 0.00	−0.06 ± 0.01	0.48 ± 0.00	0.24 ± 0.00	0.36 ± 0.00
	RSH	0.57 ± 0.02	0.37 ± 0.02	0.57 ± 0.01	0.38 ± 0.01	0.59 ± 0.02	0.40 ± 0.03	0.70 ± 0.02
	LODA	0.37 ± 0.00	0.07 ± 0.00	0.43 ± 0.00	0.15 ± 0.00	0.50 ± 0.00	0.27 ± 0.00	0.59 ± 0.00
T100K	LOF	0.19 ± 0.00	−0.18 ± 0.00	0.14 ± 0.00	−0.24 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.35 ± 0.00
	kNN	0.17 ± 0.00	−0.21 ± 0.00	0.07 ± 0.00	−0.34 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.37 ± 0.00
	xS	0.61 ± 0.00	0.44 ± 0.01	0.73 ± 0.01	0.62 ± 0.01	0.68 ± 0.01	0.53 ± 0.02	0.83 ± 0.01
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.20 ± 0.00	−0.15 ± 0.00	0.28 ± 0.00	−0.03 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.28 ± 0.00
	RRCT	0.26 ± 0.00	−0.08 ± 0.01	0.29 ± 0.01	−0.03 ± 0.01	0.48 ± 0.00	0.24 ± 0.00	0.39 ± 0.00
	RSH	0.57 ± 0.02	0.38 ± 0.02	0.61 ± 0.00	0.43 ± 0.01	0.63 ± 0.01	0.45 ± 0.02	0.77 ± 0.01
	LODA	0.38 ± 0.00	0.09 ± 0.00	0.46 ± 0.00	0.23 ± 0.00	0.53 ± 0.00	0.31 ± 0.00	0.62 ± 0.00
T500K	LOF	0.19 ± 0.00	−0.18 ± 0.00	0.14 ± 0.00	−0.25 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.36 ± 0.00
	kNN	0.19 ± 0.00	−0.19 ± 0.00	0.10 ± 0.00	−0.31 ± 0.00	0.49 ± 0.00	0.25 ± 0.00	0.38 ± 0.00
	xS	0.87 ± 0.01	0.81 ± 0.02	0.94 ± 0.00	0.91 ± 0.00	0.88 ± 0.00	0.83 ± 0.01	0.93 ± 0.00
	SDO	0.89 ± 0.02	0.84 ± 0.02	0.97 ± 0.01	0.96 ± 0.01	0.91 ± 0.00	0.87 ± 0.01	0.97 ± 0.00
	SDOs	0.33 ± 0.00	0.03 ± 0.00	0.51 ± 0.00	0.29 ± 0.00	0.48 ± 0.00	0.24 ± 0.00	0.41 ± 0.00
	RRCT	0.39 ± 0.01	0.11 ± 0.01	0.45 ± 0.01	0.20 ± 0.02	0.48 ± 0.00	0.24 ± 0.00	0.57 ± 0.01
	RSH	0.72 ± 0.01	0.60 ± 0.02	0.82 ± 0.00	0.75 ± 0.01	0.73 ± 0.01	0.61 ± 0.02	0.86 ± 0.00
	LODA	0.43 ± 0.00	0.17 ± 0.00	0.67 ± 0.00	0.51 ± 0.00	0.55 ± 0.00	0.34 ± 0.00	0.68 ± 0.00

Table 16
Accuracy performances of the **Swan-SF** dataset.

T	alg	pan	apan	ap	aap	mf1	amf1	roc-auc
T100	LOF	0.41 ± 0.00	0.28 ± 0.00	0.53 ± 0.00	0.43 ± 0.00	0.43 ± 0.00	0.31 ± 0.00	0.75 ± 0.00
	kNN	0.61 ± 0.00	0.53 ± 0.00	0.78 ± 0.00	0.73 ± 0.00	0.61 ± 0.00	0.53 ± 0.00	0.88 ± 0.00
	xS	0.57 ± 0.00	0.49 ± 0.00	0.67 ± 0.00	0.61 ± 0.00	0.60 ± 0.00	0.52 ± 0.00	0.89 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.59 ± 0.00	0.51 ± 0.00	0.71 ± 0.00	0.65 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	RRCT	0.33 ± 0.00	0.20 ± 0.00	0.33 ± 0.00	0.20 ± 0.00	0.39 ± 0.00	0.26 ± 0.00	0.71 ± 0.00
	RSH	0.59 ± 0.00	0.50 ± 0.01	0.77 ± 0.00	0.72 ± 0.00	0.59 ± 0.00	0.51 ± 0.01	0.84 ± 0.01
	LODA	0.48 ± 0.00	0.37 ± 0.00	0.52 ± 0.00	0.42 ± 0.01	0.51 ± 0.00	0.42 ± 0.00	0.83 ± 0.00
T500	LOF	0.24 ± 0.00	0.09 ± 0.00	0.29 ± 0.00	0.15 ± 0.00	0.29 ± 0.00	0.15 ± 0.00	0.59 ± 0.00
	kNN	0.59 ± 0.00	0.51 ± 0.00	0.74 ± 0.00	0.68 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	xS	0.60 ± 0.00	0.52 ± 0.00	0.72 ± 0.00	0.66 ± 0.00	0.61 ± 0.00	0.53 ± 0.00	0.90 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.60 ± 0.00	0.52 ± 0.00	0.75 ± 0.00	0.70 ± 0.00	0.61 ± 0.00	0.53 ± 0.00	0.90 ± 0.00
	RRCT	0.34 ± 0.00	0.21 ± 0.00	0.41 ± 0.00	0.29 ± 0.00	0.38 ± 0.00	0.25 ± 0.00	0.70 ± 0.00
	RSH	0.62 ± 0.00	0.54 ± 0.00	0.78 ± 0.00	0.73 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.87 ± 0.01
	LODA	0.53 ± 0.00	0.44 ± 0.00	0.59 ± 0.01	0.51 ± 0.01	0.57 ± 0.00	0.48 ± 0.00	0.86 ± 0.00
T1K	LOF	0.25 ± 0.00	0.11 ± 0.00	0.31 ± 0.00	0.18 ± 0.00	0.31 ± 0.00	0.17 ± 0.00	0.59 ± 0.00
	kNN	0.61 ± 0.00	0.53 ± 0.00	0.74 ± 0.00	0.68 ± 0.00	0.62 ± 0.00	0.55 ± 0.00	0.90 ± 0.00
	xS	0.59 ± 0.00	0.50 ± 0.00	0.73 ± 0.00	0.68 ± 0.00	0.61 ± 0.00	0.53 ± 0.00	0.89 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.61 ± 0.00	0.53 ± 0.00	0.77 ± 0.00	0.72 ± 0.00	0.61 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	RRCT	0.38 ± 0.00	0.25 ± 0.00	0.47 ± 0.00	0.37 ± 0.00	0.41 ± 0.00	0.28 ± 0.00	0.73 ± 0.00
	RSH	0.63 ± 0.00	0.55 ± 0.00	0.78 ± 0.00	0.74 ± 0.00	0.63 ± 0.00	0.56 ± 0.00	0.88 ± 0.01
	LODA	0.55 ± 0.01	0.46 ± 0.01	0.63 ± 0.04	0.56 ± 0.05	0.58 ± 0.00	0.50 ± 0.00	0.88 ± 0.00
T5K	LOF	0.27 ± 0.00	0.12 ± 0.00	0.31 ± 0.00	0.17 ± 0.00	0.31 ± 0.00	0.17 ± 0.00	0.59 ± 0.00
	kNN	0.62 ± 0.00	0.55 ± 0.00	0.76 ± 0.00	0.72 ± 0.00	0.63 ± 0.00	0.55 ± 0.00	0.90 ± 0.00
	xS	0.60 ± 0.00	0.52 ± 0.00	0.74 ± 0.00	0.69 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.89 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.61 ± 0.00	0.54 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.55 ± 0.00	0.90 ± 0.00
	RRCT	0.39 ± 0.00	0.27 ± 0.00	0.47 ± 0.00	0.37 ± 0.00	0.42 ± 0.00	0.3 ± 0.00	0.75 ± 0.00
	RSH	0.63 ± 0.00	0.56 ± 0.00	0.79 ± 0.00	0.75 ± 0.00	0.63 ± 0.00	0.56 ± 0.00	0.89 ± 0.00
	LODA	0.59 ± 0.00	0.51 ± 0.00	0.66 ± 0.01	0.59 ± 0.02	0.60 ± 0.00	0.52 ± 0.00	0.89 ± 0.00
T10K	LOF	0.28 ± 0.00	0.14 ± 0.00	0.33 ± 0.00	0.19 ± 0.00	0.31 ± 0.00	0.18 ± 0.00	0.59 ± 0.00
	kNN	0.62 ± 0.00	0.55 ± 0.00	0.76 ± 0.00	0.72 ± 0.00	0.63 ± 0.00	0.56 ± 0.00	0.90 ± 0.00
	xS	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.63 ± 0.00	0.55 ± 0.00	0.89 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.61 ± 0.00	0.53 ± 0.00	0.78 ± 0.00	0.74 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	RRCT	0.37 ± 0.00	0.24 ± 0.00	0.43 ± 0.00	0.32 ± 0.00	0.41 ± 0.00	0.3 ± 0.00	0.74 ± 0.00
	RSH	0.63 ± 0.00	0.56 ± 0.00	0.79 ± 0.00	0.75 ± 0.00	0.63 ± 0.00	0.56 ± 0.00	0.89 ± 0.00
	LODA	0.59 ± 0.01	0.51 ± 0.01	0.65 ± 0.03	0.58 ± 0.04	0.61 ± 0.00	0.53 ± 0.01	0.89 ± 0.00
T50K	LOF	0.28 ± 0.00	0.14 ± 0.00	0.34 ± 0.00	0.21 ± 0.00	0.31 ± 0.00	0.18 ± 0.00	0.59 ± 0.00
	kNN	0.63 ± 0.00	0.56 ± 0.00	0.76 ± 0.00	0.72 ± 0.00	0.63 ± 0.00	0.57 ± 0.00	0.90 ± 0.00
	xS	0.62 ± 0.00	0.55 ± 0.00	0.76 ± 0.00	0.72 ± 0.00	0.63 ± 0.00	0.56 ± 0.00	0.89 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.62 ± 0.00	0.54 ± 0.00	0.78 ± 0.00	0.73 ± 0.00	0.63 ± 0.00	0.55 ± 0.00	0.90 ± 0.00
	RRCT	0.32 ± 0.00	0.19 ± 0.00	0.36 ± 0.00	0.23 ± 0.00	0.38 ± 0.00	0.26 ± 0.00	0.71 ± 0.00
	RSH	0.63 ± 0.00	0.55 ± 0.00	0.79 ± 0.00	0.74 ± 0.00	0.64 ± 0.00	0.57 ± 0.00	0.90 ± 0.00
	LODA	0.57 ± 0.04	0.48 ± 0.05	0.71 ± 0.05	0.65 ± 0.06	0.59 ± 0.03	0.51 ± 0.03	0.88 ± 0.01
T100K	LOF	0.28 ± 0.00	0.14 ± 0.00	0.34 ± 0.00	0.21 ± 0.00	0.31 ± 0.00	0.18 ± 0.00	0.59 ± 0.00
	kNN	0.63 ± 0.00	0.56 ± 0.00	0.76 ± 0.00	0.72 ± 0.00	0.63 ± 0.00	0.57 ± 0.00	0.90 ± 0.00
	xS	0.58 ± 0.00	0.50 ± 0.00	0.74 ± 0.00	0.69 ± 0.00	0.59 ± 0.00	0.50 ± 0.00	0.78 ± 0.00
	SDO	0.61 ± 0.00	0.53 ± 0.00	0.76 ± 0.00	0.71 ± 0.00	0.62 ± 0.00	0.54 ± 0.00	0.90 ± 0.00
	SDOs	0.62 ± 0.00	0.54 ± 0.00	0.75 ± 0.00	0.70 ± 0.00	0.63 ± 0.00	0.56 ± 0.00	0.90 ± 0.00
	RRCT	0.31 ± 0.00	0.17 ± 0.00	0.35 ± 0.00	0.22 ± 0.00	0.37 ± 0.00	0.25 ± 0.00	0.70 ± 0.00
	RSH	0.63 ± 0.00	0.56 ± 0.00	0.78 ± 0.00	0.74 ± 0.00	0.64 ± 0.00	0.57 ± 0.00	0.89 ± 0.00
	LODA	0.62 ± 0.00	0.54 ± 0.00	0.77 ± 0.00	0.72 ± 0.00	0.62 ± 0.00	0.55 ± 0.01	0.90 ± 0.00

Table 17
Accuracy performances of the **Yahoo-TSA** dataset.

T	alg	pan	apan	ap	aap	mf1	amf1	roc-auc
T100	LOF	0.24 ± 0.00	0.23 ± 0.00	0.28 ± 0.00	0.25 ± 0.00	0.24 ± 0.00	0.23 ± 0.00	0.75 ± 0.00
	kNN	0.18 ± 0.00	0.15 ± 0.00	0.28 ± 0.00	0.28 ± 0.00	0.19 ± 0.00	0.18 ± 0.00	0.78 ± 0.00
	xS	0.19 ± 0.00	0.18 ± 0.00	0.30 ± 0.00	0.29 ± 0.00	0.20 ± 0.00	0.19 ± 0.01	0.84 ± 0.00
	SDO	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.73 ± 0.00
	SDOs	0.20 ± 0.00	0.19 ± 0.00	0.30 ± 0.00	0.28 ± 0.00	0.21 ± 0.00	0.20 ± 0.00	0.81 ± 0.00
	RRCT	0.24 ± 0.00	0.23 ± 0.00	0.39 ± 0.00	0.38 ± 0.00	0.25 ± 0.00	0.24 ± 0.00	0.76 ± 0.00
	RSH	0.14 ± 0.00	0.12 ± 0.00	0.25 ± 0.00	0.24 ± 0.00	0.14 ± 0.00	0.13 ± 0.00	0.79 ± 0.00
	LODA	0.09 ± 0.00	0.07 ± 0.00	0.07 ± 0.01	0.05 ± 0.01	0.14 ± 0.00	0.12 ± 0.00	0.76 ± 0.00
T500	LOF	0.25 ± 0.00	0.25 ± 0.00	0.29 ± 0.00	0.28 ± 0.00	0.27 ± 0.00	0.25 ± 0.00	0.76 ± 0.00
	kNN	0.21 ± 0.00	0.20 ± 0.00	0.31 ± 0.00	0.31 ± 0.00	0.21 ± 0.00	0.20 ± 0.00	0.81 ± 0.00
	xS	0.14 ± 0.01	0.13 ± 0.01	0.20 ± 0.03	0.19 ± 0.03	0.18 ± 0.01	0.17 ± 0.01	0.79 ± 0.02
	SDO	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.73 ± 0.00
	SDOs	0.21 ± 0.00	0.19 ± 0.00	0.31 ± 0.00	0.31 ± 0.00	0.21 ± 0.00	0.20 ± 0.00	0.81 ± 0.00
	RRCT	0.26 ± 0.00	0.25 ± 0.00	0.39 ± 0.00	0.38 ± 0.00	0.27 ± 0.00	0.25 ± 0.00	0.81 ± 0.00
	RSH	0.14 ± 0.00	0.13 ± 0.00	0.27 ± 0.00	0.25 ± 0.00	0.15 ± 0.00	0.13 ± 0.00	0.80 ± 0.00
	LODA	0.07 ± 0.00	0.06 ± 0.00	0.07 ± 0.02	0.05 ± 0.01	0.14 ± 0.00	0.12 ± 0.00	0.76 ± 0.00
T1K	LOF	0.23 ± 0.00	0.22 ± 0.00	0.25 ± 0.00	0.24 ± 0.00	0.24 ± 0.00	0.22 ± 0.00	0.76 ± 0.00
	kNN	0.20 ± 0.00	0.19 ± 0.00	0.31 ± 0.00	0.29 ± 0.00	0.21 ± 0.00	0.19 ± 0.00	0.81 ± 0.00
	xS	0.11 ± 0.01	0.10 ± 0.01	0.19 ± 0.01	0.18 ± 0.02	0.12 ± 0.01	0.10 ± 0.00	0.74 ± 0.01
	SDO	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.73 ± 0.00
	SDOs	0.19 ± 0.00	0.17 ± 0.00	0.30 ± 0.00	0.29 ± 0.00	0.20 ± 0.00	0.18 ± 0.00	0.80 ± 0.00
	RRCT	0.25 ± 0.00	0.24 ± 0.00	0.37 ± 0.00	0.36 ± 0.00	0.25 ± 0.00	0.24 ± 0.00	0.80 ± 0.00
	RSH	0.15 ± 0.00	0.13 ± 0.00	0.27 ± 0.00	0.25 ± 0.00	0.15 ± 0.00	0.13 ± 0.00	0.79 ± 0.01
	LODA	0.06 ± 0.01	0.04 ± 0.01	0.06 ± 0.01	0.05 ± 0.00	0.11 ± 0.00	0.09 ± 0.00	0.75 ± 0.01
T5K	LOF	0.12 ± 0.00	0.10 ± 0.00	0.12 ± 0.00	0.12 ± 0.00	0.12 ± 0.00	0.12 ± 0.00	0.68 ± 0.00
	kNN	0.14 ± 0.00	0.12 ± 0.00	0.25 ± 0.00	0.24 ± 0.00	0.14 ± 0.00	0.12 ± 0.00	0.73 ± 0.00
	xS	0.11 ± 0.00	0.10 ± 0.00	0.20 ± 0.00	0.19 ± 0.00	0.12 ± 0.00	0.10 ± 0.00	0.72 ± 0.00
	SDO	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.73 ± 0.00
	SDOs	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.72 ± 0.00
	RRCT	0.14 ± 0.00	0.13 ± 0.00	0.22 ± 0.00	0.21 ± 0.00	0.14 ± 0.00	0.13 ± 0.00	0.73 ± 0.00
	RSH	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.74 ± 0.00
	LODA	0.09 ± 0.02	0.08 ± 0.03	0.13 ± 0.04	0.12 ± 0.04	0.10 ± 0.01	0.09 ± 0.01	0.72 ± 0.01
T10K	LOF	0.07 ± 0.00	0.06 ± 0.00	0.10 ± 0.00	0.07 ± 0.00	0.07 ± 0.00	0.06 ± 0.00	0.65 ± 0.00
	kNN	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.14 ± 0.00	0.12 ± 0.00	0.71 ± 0.00
	xS	0.11 ± 0.00	0.10 ± 0.00	0.21 ± 0.00	0.20 ± 0.00	0.12 ± 0.00	0.10 ± 0.00	0.72 ± 0.00
	SDO	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.73 ± 0.00
	SDOs	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.72 ± 0.00
	RRCT	0.11 ± 0.00	0.09 ± 0.00	0.17 ± 0.00	0.15 ± 0.00	0.11 ± 0.00	0.10 ± 0.00	0.69 ± 0.00
	RSH	0.11 ± 0.00	0.10 ± 0.00	0.22 ± 0.00	0.21 ± 0.00	0.12 ± 0.00	0.10 ± 0.00	0.74 ± 0.00
	LODA	0.09 ± 0.00	0.07 ± 0.00	0.13 ± 0.01	0.12 ± 0.01	0.10 ± 0.00	0.09 ± 0.00	0.69 ± 0.00
T50K	LOF	0.03 ± 0.00	0.00 ± 0.00	0.03 ± 0.00	0.00 ± 0.00	0.03 ± 0.00	0.03 ± 0.00	0.57 ± 0.00
	kNN	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.21 ± 0.00	0.12 ± 0.00	0.12 ± 0.00	0.71 ± 0.00
	xS	0.09 ± 0.00	0.07 ± 0.00	0.16 ± 0.00	0.15 ± 0.00	0.09 ± 0.00	0.07 ± 0.00	0.52 ± 0.00
	SDO	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.73 ± 0.00
	SDOs	0.12 ± 0.00	0.11 ± 0.00	0.23 ± 0.00	0.22 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.74 ± 0.00
	RRCT	0.10 ± 0.00	0.08 ± 0.00	0.16 ± 0.00	0.15 ± 0.00	0.10 ± 0.00	0.09 ± 0.00	0.69 ± 0.00
	RSH	0.11 ± 0.00	0.10 ± 0.00	0.22 ± 0.00	0.21 ± 0.00	0.12 ± 0.00	0.11 ± 0.00	0.72 ± 0.00
	LODA	0.09 ± 0.00	0.08 ± 0.00	0.08 ± 0.01	0.06 ± 0.01	0.10 ± 0.00	0.09 ± 0.00	0.71 ± 0.00

References

Ahmadzadeh, A., & Aydin, B (2020). Multivariate timeseries feature extraction on SWAN data benchmark (swan_Features). URL https://github.com/charlespwd/project-title. GSU Data Mining Lab, Bitbucket repository.

Angiulli, F., & Fassetti, F (2007). Detecting distance-based outliers in streams of data. In *Proc. of the 16th ACM conf. on information and knowledge management* (pp. 811–820). New York, NY, USA: ACM.

Angryk, R., Martens, P., Aydin, B., Kempton, D., Mahajan, S., Basodi, S., Ahmadzadeh, A., Cai, X., Boubrahimi, S. Filali, Hamdi, S. M., Schuh, M., & Georgoulis, M (2020a). SWAN-SF. http://dx.doi.org/10.7910/DVN/EBCFKM, Harvard Dataverse.

Angryk, R. A., Martens, P. C., Aydin, B., Kempton, D., Mahajan, S. S., Basodi, S., Ahmadzadeh, A., Cai, X., Boubrahimi, S. Filali, Hamdi, S. M., Schuh, M. A., & Georgoulis, M. K (2020b). Multivariate time series dataset for space weather data analytics. *Scientific Data*, *7*.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., & Perona, I (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, *46*, 243–256.

Bezdek, J. C., & Keller, J. M. (2021). Streaming data analysis: Clustering or classification? *IEEE Transactions on Systems, Man & Cybernetics*, *51*, 91–102.

Boyd, K., Eng, K. H., & Page, C. D. (2013). Area under the precision–recall curve: Point estimates and confidence intervals. In *Joint Eur. conf. on mach. learn. and knowl. disc. in databases* (pp. 451–466). Springer.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. In *ACM SIGMOD int. conf. on management of data* (pp. 93–104).

Campos, G. O., Zimek, A., Sander, J., Campello, R. J., Micenková, B., Schubert, E., Assent, I., & Houle, M. E. (2016). On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, *30*, 891–927.

Catlett, J. (1991). Mega induction: A test flight. In L. A. Birnbaum, & G. C. Collins (Eds.), *Mach. learn. proc* (pp. 596–599). San Francisco (CA): Morgan Kaufmann.

Ciaccia, P., Patella, M., & Zezula, P (1997). M-tree: An efficient access method for similarity search in metric spaces. In *Int. conf. on very large data bases*, (pp. 426–435). San Francisco, CA, USA: Morgan Kaufmann Pub. Inc..

Davidow, M., & Matteson, D. S. (2022). Factor analysis of mixed data for anomaly detection. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *15*, 480–493.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Domingues, R., Filippone, M., Michiardi, P., & Zouaoui, J. (2018). A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition*, *74*, 406–421.

Du, M., Li, F., Zheng, G., & Srikumar, V. (2017). Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security* (pp. 1285–1298). New York, NY, USA: Association for Computing Machinery.

Fisch, A. T. M., Eckley, I. A., & Fearnhead, P. (2022). A linear time method for the detection of collective and point anomalies. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *15*, 494–508.

Gama, J. a., Žliobaité, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, *46*.

Goldstein, M., & Dengel, A. (2012). Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm. In *Poster and demo track* (pp. 59–63).

Gomes, H. M., Read, J., Bifet, A., Barddal, J. P., & Gama, J. (2019). Machine learning for streaming data: State of the art, challenges, and opportunities. *SIGKDD Explorations*, *21*, 6–22.

Guha, S., Mishra, N., Roy, G., & Schrijvers, O (2016). Robust random cut forest based anomaly detection on streams. In *Int. conf. on mach. learn* (pp. 2712–2721).

Hartl, A. (2020). Dsalmon (Data stream analysis algorithms for the impatient). Github: https://github.com/CN-TU/dSalmon. TU Wien CN Group.

Hartl, A., Iglesias, F., & Zseby, T. (2020). SDOstream: Low-density models for streaming outlier detection. In *Eur. symp. on artificial neural networks, comp. int. and mach. learn* (pp. 661–666).

Hawkins, D. M. (1980). *Identification of outliers*: *Vol. 11*, New York: Chapman and Hall London.

Hubert, L., & Arabie, P (1985). Comparing partitions. *Journal of Classification*, *2*, 193–218.

Iglesias, F. (2021). Data for evaluation of stream data analysis algorithms. In *Mendeley data, V1*. URL https://data.mendeley.com/datasets/c43kr4t7h8/1.

Iglesias, F., & Hartl, A. (2021). Comparison and evaluation of outlier detection in streaming data. Github: https://github.com/CN-TU/py-outlier-detection-stream-data. TU Wien CN Group.

Iglesias, F., Hartl, A., Zseby, T., & Zimek, A. (2019). Are network attacks outliers? A study of space representations and unsupervised algorithms. In *MLCS work. on mach. learn. for cybersecurity* (pp. 1–16).

Iglesias, F., Odjanic, D., Hartl, A., & Zseby, T. (2020). Mdcstream: Stream data generator for testing analysis algorithms. In *EAI valuetools* (pp. 1–8). Tsukuba, Japan.

Iglesias, F., Zseby, T., & Zimek, A. (2018). Outlier detection based on low density models. In *2018 IEEE int. conf. on data mining workshops* (pp. 970–979).

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2019). Deep learning for time series classification: A review. *Data Mining and Knowledge Discovery*, *33*, 917–963.

Kim, T., & Park, C. H. (2020). Anomaly pattern detection for streaming data. *Expert Systems with Applications*, *149*, Article 113252.

Knorr, E. M., & Ng, R. T. (1998). Algorithms for mining distance-based outliers in large datasets. In *VLDB* (pp. 392–403).

Kontaki, M., Gounaris, A., Papadopoulos, A. N., Tsichlas, K., & Manolopoulos, Y. (2011). Continuous monitoring of distance-based outliers over data streams. In *IEEE 27th int. conf. on data engineering* (pp. 135–146).

Kriegel, H., Schubert, E., & Zimek, A. (2017). The (black) art of runtime evaluation: Are we comparing algorithms or implementations? *Knowledge and Information Systems*, *52*, 341–378.

Manzoor, E., Lamba, H., & Akoglu, L. (2018). xStream: Outlier detection in feature-evolving data streams. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1963–1972). Association for Computing Machinery.

Marques, H. O., Swersky, L., Sander, J., Campello, R. J., & Zimek, A. (2023). On the evaluation of outlier detection and one-class classification: A comparative study of algorithms, model selection, and ensembles. *Data Mining and Knowledge Discovery*, 1–45. http://dx.doi.org/10.1007/s10618-023-00931-x.

Moya, M. M., & Hush, D. R. (1996). Network constraints and multi-objective optimization for one-class classification. *Neural Networks*, *9*, 463–474.

Pevný, T. (2016). Loda: Lightweight on-line detector of anomalies. *Machine Learning*, *102*, 275–304.

Pishgoo, B., Akbari Azirani, A., & Raahemi, B. (2021). A hybrid distributed batch-stream processing approach for anomaly detection. *Information Sciences*, *543*, 309–327. http://dx.doi.org/10.1016/j.ins.2020.07.026.

Pokrajac, D., Lazarevic, A., & Latecki, L. J. (2007). Incremental local outlier detection for data streams. In *IEEE symp. on comp. int. and data mining* (pp. 504–515).

Ramaswamy, S., Rastogi, R., & Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD int. conf. on management of data* (pp. 427–438).

Ramírez-Gallego, S., Krawczyk, B., García, S., Wosfxniak, M., & Herrera, F. (2017). A survey on data preprocessing for data stream mining: Current status and future directions. *Neurocomputing*, *239*, 39–57. http://dx.doi.org/10.1016/j.neucom.2017.01.078.

Rayana, S. (2016). ODDS library. http://odds.cs.stonybrook.edu.

Read, J., Rios, R. A., Nogueira, T., & de Mello, R. F. (2020). Data streams are time series: Challenging assumptions. In *Intelligent systems: 9th Brazilian conference, proceedings, part ii* (pp. 529–543). Berlin, Heidelberg: Springer-Verlag, http://dx.doi.org/10.1007/978-3-030-61380-8_36.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., & Müller, K.-R. (2021). A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, *109*, 756–795.

Saito, T., & Rehmsmeier, M. (2015). The precision–recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLoS One*, *10*, Article e0118432.

Sathe, S., & Aggarwal, C. C. (2016). Subspace outlier detection in linear time with randomized hashing. In *IEEE 16th int. conf. on data mining* (pp. 459–468).

Schubert, E., Zimek, A., & Kriegel, H. (2014). Local outlier detection reconsidered: A generalized view on locality with applications to spatial, video, and network outlier detection. *Data Mining and Knowledge Discovery*, *28*, 190–237.

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP* (pp. 108–116).

Shyu, M.-L., Chen, S.-C., Sarinnapakorn, K., & Chang, L. (2006). Principal component-based anomaly detection scheme. In T. Young Lin, S. Ohsuga, C.-J. Liau, & X. Hu (Eds.), *Foundations and novel approaches in data mining* (pp. 311–329). Springer Berlin Heidelberg.

Silva, J. A., Faria, E. R., Barros, R. C., Hruschka, E. R., Carvalho, A. C. P. L. F. d., & Gama, J. a. (2013). Data stream clustering: A survey. *ACM Computing Surveys*, *46*.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems*, *25*, 2951–2959.

Steinbuss, G., & Böhm, K (2021). Benchmarking unsupervised outlier detection with realistic synthetic data. *ACM Transactions on Knowledge Discovery from Data*, *15*.

Tran, L., Fan, L., & Shahabi, C (2016). Distance-based outlier detection in data streams. *Proceedings of the VLDB Endowment*, *9*, 1089–1100.

Yahoo Webscope Program (2020). Synthetic and real time-series with labeled anomalies, v1.0 (ydata-labeled-time-series-anomalies-v1_0). URL https://webscope.sandbox.yahoo.com/.

Yang, D., Rundensteiner, E., & Ward, M. O. (2009). Neighbor-based pattern detection for windows over streaming data. In *Int. conf. on extending database tech.: Advances in Database Tech* (pp. 529–540). ACM.

Yilmaz, S. F., & Kozat, S. S. (2020). Pysad: A streaming anomaly detection framework in python. Github repo: https://github.com/selimfirat/pysad arXiv preprint arXiv:2009.02572.

Zimek, A., & Filzmoser, P. (2018). There and back again: Outlier detection between statistical reasoning and data mining algorithms. *WIREs Data Mining and Knowledge Discovery*, *8*, Article e1280.

Zimek, A., Gaudet, M., Campello, R. J. G. B., & Sander, J. (2013). Subsampling for efficient and effective unsupervised outlier detection ensembles. In *KDD* (pp. 428–436).