Evolving Systems (2024) 15:1115–1136
https://doi.org/10.1007/s12530-023-09543-8

ORIGINAL PAPER

Time series anomaly detection via clustering-based representation

Elham Enayati[1] · Reza Mortazavi[2] · Abdolali Basiri[1] · Javad Ghasemian[1] · Mahmoud Moallem[1]

Received: 16 September 2022 / Accepted: 14 September 2023 / Published online: 17 October 2023

Abstract
Time series anomaly detection is an important field of data science. Statistical, distance-based, clustering-based, or density-based approaches can detect anomalies. Generally, distance-based methods are relatively straightforward, but the method's effectiveness depends on how well they handle the distribution of data points. To address the challenge, a preprocessing step is used to convert the underlying time series into a more useful format. In this paper, a novel clustering-based representation of time series is proposed. This representation is then used to compute anomaly scores and detect anomalies. Experimental studies on synthetic and real datasets show that proposed method outperforms other methods by up to 75% for five standard performance metrics.

Keywords Anomaly detection · Time series · Representation method · Clustering algorithm

1 Introduction

The application of data science has been widely adopted in numerous fields and is considered a powerful tool for scientific investigations. A significant type of data, time series, continues to grow and is used in many areas. Especially, detecting anomalies in time series is a valuable task in many aspects of society, such as economics, environment, astronomy (Hundman et al. 2018), process monitoring (Huang et al. 2020), communication, medicine, climate change (Cheng et al. 2021), industry (Pham et al. 2019), intrusion detection (Azzaoui et al. 2022), cyber-attack detection, fraud detection, etc. Blázquez-García et al. (2021).

✉ Reza Mortazavi
r_mortazavi@du.ac.ir

Elham Enayati
e.enayati@std.du.ac.ir

Abdolali Basiri
basiri@du.ac.ir

Javad Ghasemian
ghasemian@du.ac.ir

Mahmoud Moallem
moallem@du.ac.ir

[1] School of Mathematics and Computer Sciences, Damghan University, Damghan 3671641167, Iran

[2] School of Engineering, Damghan University, Damghan 3671641167, Iran

Generally, there are three types of time series anomalies: point outliers, collective outliers, and contextual outliers. Point outliers are anomalies that occur at different points in the time series. Collective outliers are anomalies that occur in groups of points in the time series. Contextual outliers are points or groups that indicate anomalies based on their context (Lindemann et al. 2021). This paper addresses the problem of anomaly detection in univariate time series in which one value is captured at a time.[1]

Time series anomalies can be detected using both statistical and machine learning approaches. Generally, the former is more straightforward, while the latter is more suitable for larger problems. Machine learning approaches are divided into supervised, semi-supervised, and unsupervised methods. In general, supervised approaches can cover anomaly detection adequately. However, they require sufficiently labeled data, which is difficult to obtain in many applications.

Anomaly detection techniques may use representation techniques (Zhang et al. 2021). Data representation approaches transform input data values into symbols, which can be either numeric values or symbols such as alphabets (Zhou et al. 2021).

Recently, Zhou et al. (2021) introduced interval-based and first-order representation methods that generate alternative forms of time series. First, the input time series is divided into several sliding windows with a fixed length. In the interval-based method, a sliding window is represented

[1] For a more formal definition see Definition 1 in Sect. 2.

by a line. The line is constructed with the interval $[a, b]$. The values of a and b are defined in Eq. (1) (Zhou et al. 2021):

$$\begin{aligned} V(a) &= f_1\left(count\left\{x(k) \in X(n) | a \le x(k) < med(X(n))\right\}\right) \\ &\quad \times f_2(|med(X(n)) - a|) \\ V(b) &= f_1\left(count\left\{x(k) \in X(n) | med(X(n)) \le x(k) \le b\right\}\right) \\ &\quad \times f_2(|med(X(n)) - b|\Big) \end{aligned} \tag{1}$$

where input time series is represented by $X(n)$, which has a length of n. The lower and upper bounds of the interval are denoted by a and b, repectively. The variable k ranges from 1 to n, and $x(k)$ represents the kth point in $X(n)$. Functions f_1 and f_2 are defined by Eq. (2) and $\alpha = 0.5$.

$$\begin{aligned} f_1(u) &= u \\ f_2(u) &= exp(-\alpha u) \end{aligned} \tag{2}$$

Following that, anomalous sliding windows are detected by computing distances between the windows based on distances between intervals. The first-order method shows a sliding window with a rectangle. The method generates the width and length of the rectangle by Eq. (1).

The PAA[2] method is another representation method that uses average values to represent sliding windows (Keogh et al. 2001). The results in (Zhou et al. 2021) indicate that Zhou's methods are more informative than PAA. The two methods effectively represent sliding windows using lines and rectangles.

While Zhou's methods take advantage of the interval representation of time series, they lose some vital information about sliding windows. Sometimes, the methods may lose the original data distributions and fail to generate appropriate representations. For example, Fig. 1 shows two sliding windows, A and B, with different trends: $A(n) = (1, 2, 3, 4, 6, 5, 7, 8, 9, 11, 10, 13, 12, 14, 15, 15, 14, 12, 13, 10, 11, 9, 8, 7, 5, 6, 4, 3, 2, 1)$ (Fig. 1a) and $B(n) = (5, 14, 13, 12, 11, 9, 10, 8, 7, 5, 6, 4, 3, 2, 1, 1, 2, 3, 4, 5, 7, 6, 8, 9, 10, 12, 11, 13, 14, 15)$ (Fig. 1b) in two sliding windows of size 30. In this case, A and B are represented by the same interval $[6, 9]$ in the interval-based method (Eqs. (1) and (2)). More interestingly, the rectangle constructed by the first-order method is $[6, 9]$ and $[-1, 1]$ for both sliding windows, as illustrated in Fig. 1c. The example shows that interval-based and first-order methods suffer from a lack of specificity (Wang et al. 2022).

To address the problem, this paper proposes a method to represent data in a more compact and informative way

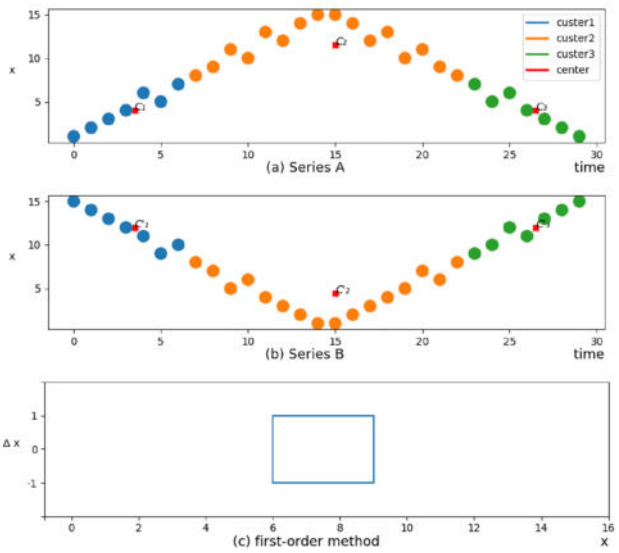

Fig. 1 **a** Time series A and its clustering, **b** time series B and its clustering, **c** first-order representation of A and B

using clustering. In the previous example, the original sliding window is clustered optimally into three groups (Fig. 1a, b where blue, orange, and green points show clusters 1, 2, and 3, respectively). Thus, time series is represented by centroids of groups. In this representation, window A is represented by $c_1 = 4, c_2 = 11.5$, and $c_3 = 4$. Meanwhile, window B is replaced by $c'_1 = 12, c'_2 = 4.5$, and $c'_3 = 12$, where c_i and c'_i are ith centroids in A and B, respectively (red points in Fig. 1a, b).

Additionally, As shown in Fig. 1a, b, a clustering algorithm divides sliding windows into clusters with different sizes. For example, sliding window A is divided into clusters with 7, 16, and 7 points. Using a clustering algorithm presents an adaptive segmentation mechanism for sliding windows which is more flexible than interval-based and first-order methods. Concerning the sequential nature of time series, a clustering algorithm is an appropriate segmentation technique potentially. This motivating example illustrates how a clustering method can provide a more useful transformation than interval-based and first-order techniques.

This paper makes the following major contributions:

1. It introduces a novel clustering-based representation method that segments time series into subsequences of different lengths. The method locates time series cut points that are adaptive, data-oriented, and reflects data behaviors.
2. to improve the effectiveness of the sliding window anomaly detection methods.

[2] Piecewise Aggregate Approximation.

The remainder of the paper is organized as follows. Section 2 provides the background knowledge required for this study. Section 3 briefly introduces the context of the study by providing a review of related literature. Section 4 describes the clustering-based representation method for time series representation and then demonstrates proposed anomaly detection method. Experimental results of the method and discussions are presented in Sect. 5. Finally, conclusions are given in Sect. 6.

2 Preliminary

This section briefly reviews the preliminaries of time series anomaly detection TAD to facilitate understanding of relevant concepts. The definition of a univariate time series is given in Definition 1.

Definition 1 (*Univariate time series*) (Blázquez-García et al. 2021): A univariate time series $X(n) = (x_1, \ldots, x_i, \ldots, x_n)$ is an ordered set of $x_i \in \mathbb{R}$ for $1 \leq i \leq n$.

There is no universally accepted definition of TAD. In this work, time series anomaly detection is the process of identifying records or subsets that do not fit the normal behavior of other records (Geiger et al. 2020). The following is a more formal definition of anomaly detection (Definition 4) based on sliding window (Definition 2) and anomaly score (Definition 3) concepts (Liu et al. 2020).

Definition 2 (*Sliding window*) : Given a time series $X(n) = (x_1, \ldots, x_n)$. A sliding window $W(m)$ of length m is a subsequence of m successive points in $X(n)$ starting from x_i, i.e. $W = (x_i, x_{i+1}, \ldots, x_{i+m-1})$. The window is denoted by $x_{i:i+m-1}$ in this paper.

Definition 3 (*Anomaly score*) : Given a time series $X(n) = (x_1, \ldots, x_n)$, the anomaly score $AS = (as_1, \ldots, as_n)$ is an associated sequence of non-negative real values that shows the degree of anomalies in $X(n)$. The great value of the anomaly score means the point is more likely to be an anomaly.

Definition 4 (*Time series anomaly detection*) : Time series anomaly detection is the process of finding a set of $anomalies = \{x_i \in X(n) | as_i > \tau\}$ where τ is the threshold for anomaly score.

In this paper, a representation method is proposed by using a clustering algorithm. The clustering algorithm attempts to divide the time series $X(n)$ into k-partitions $C = \{c_1, \ldots, c_k\}, k \leq n$, taking into account the order of the time series, so that:

$$\begin{cases} c_i \neq \emptyset, & i = 1, \ldots, k \\ c_i \cap c_j = \emptyset, & i, j = 1, \ldots, k, i \neq j \\ \cup_{i=1}^{k} c_i = X(n) \end{cases} \tag{3}$$

A distance or similarity measure determines the proximity of members and clusters. Let $D(x_i, x_j)$ shows the distance between x_i and x_j. The measure satisfies the following properties (Figueroa et al. 2018).

$$\begin{aligned} &Symmetry : D(x_i, x_j) = D(x_j, x_i) \\ &Positivity : D(x_i, x_j) \geq 0, \forall x_i, x_j \\ &Triangle\ inequality : D(x_i, x_j) \leq D(x_i, x_z) + D(x_z, x_j) \\ &Reflexivity : D(x_i, x_j) = 0, \iff x_i = x_j \end{aligned} \tag{4}$$

3 Related work

Detecting anomalies is not a new area of research in the field. Fox first determined and categorized time series anomalies in 1972 (Fox 1972). A statistical method for detecting anomalies in time series was also proposed by Tukey (1977) as well. Researchers have been active in studying the anomaly detection problem in a wide range of settings, including time series, where they have proposed several statistical and machine learning approaches (Hundman et al. 2018; Cheng et al. 2021; Li et al. 2021). In the following, we discuss TAD and preprocessing methods, and their representations.

3.1 Preprocessing techniques for time series anomaly detection

Preprocessing methods have always been critical to TAD. They include normalization, dimension reduction, segmentation, and data representation tasks (Liang et al. 2021). For instance, data representation techniques are used to reduce input dimensions, transfer data to a new space in which their important properties are shown better, and improve the computational cost of underlying algorithms (Sim et al. 2018; Pérez et al. 2021). Moreover, data representation methods facilitate TAD in the face of memory scarcity and power constraints (Bountrogiannis et al. 2021).

Generally, two types of representation methods are used: pattern and model-based. Pattern representations can preserve the pattern information of time series such as trends, amplitudes, and frequencies. Many methods have been

proposed within this category, including PAA, SAX,[3] DFT,[4] DWT,[5] SVD,[6] and PCA.[7] The model-based representation methods use model parameters to detect time series features, including regression methods, hidden Markov techniques, and neural networks (Ren et al. 2017).

Segmentation is another key preprocessing task in TAD, usually joined with the representation task. Segmentation consists of splitting a time series into subsequences and creating sliding windows. Some representation methods, like PAA and SAX modifications, use fixed or adaptive sliding window techniques (Bountrogiannis et al. 2021; Ghalyan et al. 2021). The right length of sliding windows is computed by different techniques. (Carmona-Poyato et al. 2020) presented an optimal segmentation method based on A^* algorithm. An adaptive sliding window approach is also introduced in Wang et al. (2022) based on time series trends.

3.2 Anomaly detection approaches

Different approaches to anomaly detection have led to the creation of several categories. Some of the most popular categories for anomaly detection approaches include statistical, distance-based, clustering-based, and density-based methods (Singh and Upadhyaya 2012). However, many proposed methods overlap and may be categorized into more than one class.

3.2.1 Statistical methods

Given the statistical nature, statistical methods use the probability or distribution of datasets to fit a model. When points or groups of points do not follow the fitted model, the datasets are confronted with anomalies. Since 1972, the use of statistical methods, e.g., Arumugam and Saranya (2018), SARIMA (Zhang et al. 2022), EWMA (Zhou and Tang 2016), GMM[8] (Reddy et al. 2017), and SOS[9] (Janssens et al. 2012), has been popular due to their potential. In 2023, Fernandes et al. (2019) used autoregressive integrated moving average models for fault detection mechanisms. Identifying anomalous flight by incorporating the GMM with dynamic trajectory pattern classification is investigated in Choi et al. (2023). Statistical methods are mainly based on assumptions about the probability and distribution of the data, which limits their use in practice (Yu and Sun 2020).

3.2.2 Distance-based methods

Distance-based methods benefit from distance calculations between points within the same time series or two time series in which one of them is used as a reference normal time series (Mahmoodi et al. 2021). Distance measures play a key role in the success of the methods (Hagemann and Katsarou 2020). The measures in time series can be classified into four categories: shape-based, edit-based, feature-based, and structure-based (Steland et al. 2015). Euclidean and Manhattan distance measures are more common among other measures (Mahmoodi et al. 2021). Although extensive research papers have been written to introduce and utilize more practical distance measures in TAD scope (Aljawarneh and Vangipuram 2020; Yazdi and Douzal-Chouakria 2018). Various attempts fall into this category. In a study by Wahid and Rao (2019), a distance-based outlier detection method is presented in which a particle swarm optimization technique is applied to find outlying subspaces. To improve the efficiency of the TAD method in Tran et al. (2020), the CPOD algorithm is proposed with multi-distance indexing. It must be noted that some weaknesses make these methods vulnerable, and their performance is strictly tied to selecting proper distance measures.

3.2.3 Clustering-based methods

Clustering-based methods mainly focus on the detection of data structures. There are two different strategies for detecting anomalies using clustering-based techniques. In one strategy, fitness values are assigned to data records. The fitness value shows the deviation value of a point from other observations in clusters. Records with the lowest fitness value are considered anomalies (Li et al. 2021). In Akhmedova et al. (2022), a clustering-based method was proposed that computes the deviation and finds anomalous points by a fuzzy and evolutionary algorithm. The second strategy identifies clusters with a few members as anomalies (Mahmoodi et al. 2021). Authors in Pramitarini et al. (2022) used a clustering algorithm with cosine similarity in VANET to find anomalous network packets.

K-means and FCM are among the most common clustering algorithms widely used by CBLOF[10] (Li et al. 2021; Chadha et al. 2021). IForest[11] is another well-known method in this category (Liu et al. 2008; Cook et al. 2019). The underlying principle of the iForest is that anomalous points are far from normal points. The algorithm uses an ensemble method with a decision tree technique (Pham et al. 2019).

[3] Symbolic Aggregate approXimation.
[4] Discrete Fourier Transform.
[5] Discrete Wavelet Transform.
[6] Singular Value Decomposition.
[7] Principal Component Analysis.
[8] Gaussian Mixture Models.
[9] Stochastic Outlier Selection.
[10] Clustering-Based Local Outlier Factor.
[11] Isolation Forest.

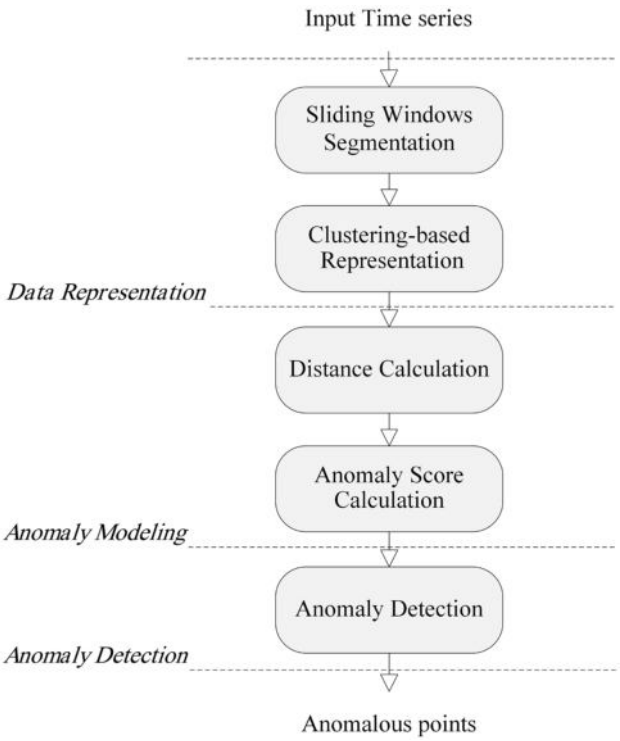

Fig. 2 The main diagram of proposed method (CUBOID)

Despite the popularity of clustering-based methods, they suffer a significant disadvantage: the results completely depend upon selecting the cluster center (Li et al. 2021).

3.2.4 Density-based methods

Density-based methods capture overall data patterns of time series. Patterns are detected by both density and neighborhood factors. Density is defined by two concepts: local density and global density (Mahmoodi et al. 2021). The definition of the neighborhood concept is more complex for time series data than other types of data because time series are ordered data (Blázquez-García et al. 2021). Many studies have applied density-based methods to identify abnormal data points (Ramotsoela et al. 2019; Munir et al. 2018). A density-based method was proposed in Wang and Fan (2022) to address preserving projection process monitoring problems. The method computes the density by the sample distance entropy. LOF (Breunig et al. 2000) is one of the most popular density-based methods in the field that has been extended by many researchers (Yang et al. 2021). Like the methods mentioned above, density-based methods also have their own weaknesses. One shortcoming becomes apparent when some input time series exhibit regular fluctuations due to seasonal variations (Munir et al. 2018).

4 The proposed method

This section describes our proposed anomaly detection method for univariate time series, CUBOID. The algorithm uses a clustering-based representation method to represent

Algorithm 1 CUBOID

Algorithm 1 CUBOID

Require: $X(n)$: time series, k: the number of clusters, w: window size, n: time series length, τ: anomaly threshold
Ensure: *Anomalous points*
$m \leftarrow \lfloor (n-1)/w \rfloor$ ▷ m: The number of sliding windows
while $i = 1 : i < n$ **do**
  $\Delta x_i \leftarrow x_{i+1} - x_i$ ▷ Computing $\Delta X(n-1)$
end while
while $i = 1 : i \leq m$ **do**
  $w_i \leftarrow \Delta x_{(i-1)\times w+1:i\times w}$ ▷ $W = (w_1, \ldots, w_m)$
end while
$C \leftarrow OSC(W, k)$ ▷ $C = (c_1, \ldots, c_m)$
Compute anomaly scores AS by Equation (5) ▷ $AS = (as_1, \ldots, as_m)$
$AS \leftarrow Sort\ (AS)$
$a \leftarrow m \times \tau$ ▷ The number of anomalous sliding windows
$A \leftarrow$ Select the first a sliding windows of AS
return A

time series. The method is categorized as an unsupervised and distance-based technique. As shown in Fig. 2, CUBOID contains three components: a data representation module, an anomaly modeling module, and an anomaly detection module.

The data representation module benefits from proposed clustering-based representation method.[12] This module segments the underlying time series into multiple sliding windows and changes their representations using a sequence clustering algorithm. The anomaly modeling module calculates anomaly scores of represented sliding windows using n-neighbor distance. Finally, the anomaly detection module selects anomalous sliding windows based on anomaly score values. In the following, the algorithm is detailed. The pseudocode of proposed method is presented in Algorithm 1. The inputs of Algorithm 1 are time series, and its outputs are anomalous points.

4.1 Data representation

In this section, we present proposed clustering-based representation method and use it for anomaly detection in the CUBOID algorithm.

According to Eqs. (3) and (4), clustering methods have important properties that make them ideal volunteers for the representation task. The following are some of the key points

1. The clustering algorithm can efficiently locate data change points. The boundaries identified by change points can be used to partition time series.
2. The clustering algorithm can predict anomalous points in time series to be partitioned into isolated clusters.

In the first step, the time series is partitioned into fixed-length windows, and each window is partitioned into clusters. Since preserving the order of data points is critical, a typical clustering algorithm cannot be employed for the task. In this study, OSC[13] (Lin and chan 2002) is applied to cluster data points in each sliding window. It is an optimal univariate clustering algorithm that uses the squared Euclidean distance to determine clustering quality. The algorithm can efficiently achieve optimal clusters with the lowest possible overall cost using a dynamic programming approach.

In the data representation module, the input time series $X(n)$ is transformed into $\Delta X(n-1) = (\Delta x_1, \Delta x_2, \ldots, \Delta x_{n-1})$ where $\Delta x_i = x_{i+1} - x_i, 1 \leq i \leq n-1$ (Lines 2 to 4 of Algorithm 1). It is to capture amplitude changes of time series more easily (Zhu et al. 2016) and to facilitate decision-making of time series pattern changes. Furthermore, $\Delta X(n-1)$ is divided into m sliding windows of length w where $m = \lfloor (n-1)/w \rfloor$. Sliding windows are denoted with $W = \{w_1, \ldots, w_m\}, 1 \leq i \leq m$ where w_i is the ith sliding window (Lines 5 to 7 of Algorithm 1). Sliding windows are clustered by the OSC algorithm into k clusters (Line 8 of Algorithm 1) and a set of cluster centroids $C = \{c_1, \ldots, c_m\}$ is returned as a new representation of the original time series where c_i includes k cluster centers of the ith sliding window.

4.2 Anomaly modeling

Anomaly scores are computed to rank sliding windows in the anomaly modeling module. This module uses the distances between neighboring sliding windows as anomaly scores, $AS = \{as_1, \ldots, as_m\}$ where as_i is the anomaly score of w_i and each w_i is partitioned into k clusters. More formally, to calculate as_i, the average of absolute distances between w_i and two previous sliding windows (w_{i-1} and w_{i-2}) are considered by Eq. (5).

$$as_i = \begin{cases} 0 & i = 1 \\ \sum_{j=1}^{k} |c_{i-1,j} - c_{i,j}| & i = 2 \\ \sum_{j=1}^{k} (|c_{i-2,j} - c_{i,j}| + |c_{i-1,j} - c_{i,j}|)/2 & i = 3, \ldots, m \end{cases} \tag{5}$$

where $c_{i,j}$ is the jth center of the ith sliding window. Cluster centers are calculated by the data representation module, where k is the number of clusters in each sliding window, and m is the number of sliding windows. The pseudocode of the anomaly modeling module is shown in Line 9 of Algorithm 1.

4.3 Anomaly detection

In this module, anomaly scores are sorted in decreasing order (Line 14 of Algorithm 1). The anomaly detection strategy is to set an overall percentage of anomalies (τ) as a threshold and select anomalous sliding windows based on it (Lines 11 and 12 of Algorithm 1). Therefore, τ percent of sliding windows with the highest anomaly scores are marked as anomalies.

4.4 Complexity analysis

In this subsection, the running time complexity of Algorithm 1 is analyzed.

Theorem 1 *The complexity of CUBOID is* $O(\max(nkw, n/w \log(n/w)))$, *where n is the number of data points in the dataset, w is the window size, k is the number of clusters, and m is the number of sliding windows.*

[12] It should be noted that the clustering-based representation mechanism is completely different from the clustering-based anomaly detection approaches discussed in Sect. 3.

[13] Optimal Sequence Clustering algorithm.

Table 1 Overview of datasets

	Yahoo	Synthetic	Sin
# Time series	367	100	1
# Anomalous points	3915	2593	141
# Total data points	572,966	354,656	2000

Table 2 Overview of Yahoo Benchmark Datasets

	A1	A2	A3	A4
# Time series	67	100	100	100
# Anomalous points	1669	466	943	837
# Total data points	94,866	142,100	168,000	168,000

Proof First, the complexity of Algorithm 1 is related to two loops (lines 3 and 6). These loops contain n and m iterations, respectively. The total number of iterations (Line 8 of Algorithm 1) depends on the complexity of the clustering algorithm. Since OSC is used in CUBOID, the complexity of the clustering process for each of m sliding windows is $O(kw^2)$ (Lin and chan 2002). Therefore, this line is completed in $O(mkw^2)$ iterations. In Lines 9 and 10 of Algorithm 1, computing anomaly scores of m sliding windows and sorting them require $O(m\log(m))$ operations. Consequently, the total complexity of the model is determined by Eq. (6).

$$O(n+m+mkw^2+m\log(m)) = O(\max(mkw^2, m\log(m))) \tag{6}$$

Since $m = n/w$, the model's complexity can be rewritten as Eq. (7).

$$O(\max(mkw^2, m\log(m))) = O(\max(nkw, \frac{n}{w}\log(\frac{n}{w})) \tag{7}$$

□

5 Experimental studies

In this section, the performance of proposed method is evaluated using some scenarios for different datasets based on five index measures. More details are given below.

5.1 Datasets

The paper uses several time series datasets containing synthetic and real data to evaluate CUBOID's performance. Generally, the current work is applied to three datasets.

- Yahoo S5 Webescope,[14] is one of the most popular benchmarks for anomaly detection in time series (Hagemann and Katsarou 2020; Ren et al. 2019; Maciąg et al. 2021). Yahoo S5 contains four sub-datasets: A1, A2, A3, and A4. A1 benchmark contains real traffic data on Yahoo systems. In addition, A2, A3, and A4 benchmarks contain synthetic datasets. All repositories in Yahoo S5 are unbalanced or highly unbalanced, so the percentage of input values representing anomalies is less than 1% on average (Maciąg et al. 2021). More details on these repositories can be found in Tables 1 and 2.
- Synthetic dataset includes 100-time series. These time series are generated using agots package[15] and contain four types of anomalies: extreme, shift, trend, and variance.
- Additionally, a synthetic Sin time series is generated by Eq. (8). The time series includes three anomalous subsequences ($e_1(t), e_2(t)$, and $e_3(t)$). This time series has been studied in related works, such as the interval-based method (Zhou et al. 2021; Ren et al. 2018).

$$X(t) = sin(\frac{40\pi t}{K}) + n(t) + e_1(t) + e_2(t) + e_3(t), t \in [0, 2000], k = 1200 \tag{8}$$

where e_1, e_2, and e_3 are three synthetic anomalies computed by Eq. (9). Also, a Gaussian noise $n(t)$ with $\mu = 0, \sigma = 0.1$ is in Eq. (8).

$$\begin{aligned} e_1(t) &= \begin{cases} -0.5 & t \in [550, 600] \\ 0 & otherwise \end{cases} \\ e_2(t) &= \begin{cases} rnorm(0, 0.8) & t \in [1000, 1049] \\ 0 & otherwise \end{cases} \\ e_3(t) &= \begin{cases} 0.2X(t) & t \in [1520, 1559] \\ 0 & otherwise \end{cases} \end{aligned} \tag{9}$$

where in $e_2(t)$ a normal distribution is used.

5.2 Evaluation criteria

In this section, several index measures are introduced to evaluate the method's performance. Anomaly detection performance can be calculated by index measures defined by the confusion matrix. The matrix is a 2×2 matrix and contains indexes such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). This study used five facilitative criteria to assess the quality of underlying methods: *accuracy*, *precision*, *recall*, F-score (Eq. (10)), and *CI* (Eq. (11)).

[14] https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70.

[15] https://github.com/KDD-OpenSource/agots.

Table 3 Details of the experiments

Scenario parameters	Scenario 1	Scenario 2	Scenario 3
Number of clusters	3	3	2–5
Window size	60, 13, 13, 12	10–100	10–100
Dataset	A1 Real-29, A2 Synthetic-62, A3 TS-10, A4 TS-16	Yahoo, Sin, Synthetic	A1 Real-29, A2 Synthetic-62, A3 TS-10, A4 TS-16
Baseline methods	Interval-based (Zhou et al. 2021), PAA (Keogh et al. 2001), first-order (Zhou et al. 2021)	Interval-based, PAA, first-order, iForest (Liu et al. 2008), LOF, SOS (Janssens et al. 2012), CBLOF (He et al. 2003)	–

$$\begin{aligned} Acurracy &= \frac{TP + TN}{TP + TN + FP + FN} \\ Precision &= \frac{TP}{TP + FP} \\ Recall &= \frac{TP}{TP + FN} \\ \text{F-score} &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \end{aligned} \tag{10}$$

Some researchers such as Zhou et al. (2021) have called the recall index as *AR* (Accuracy Rate). The *AR* is employed throughout the paper.

The ability to distinguish abnormal points from normal points is defined by the data anomaly resolution concept. Data anomaly resolution is assessed by *CI* (Confidence Index) (Zhou et al. 2021). Equation (11) formalizes *CI*.

$$CI = \frac{mean(\sum as_{anomaly})}{mean(\sum as_{all})} \tag{11}$$

where the average anomaly score for all anomalous subsequences is $mean(\sum as_{anomaly})$ and the average anomaly score for all subsequences is $mean(\sum as_{all})$.

5.3 Configuration settings

In this section, some experiments were conducted to evaluate the efficiency of clustering-based representation and the performance of CUBOID. Anomaly detection modules use the same threshold in all experiments, i.e., $\tau = 1\%$ is applied to provide fair conditions. Therefore, all data points within one percent of sliding windows with the highest anomaly scores are considered anomalies.

Some anomaly detection methods are used for comparison in the following scenarios, such as interval-based, first-order (Zhou et al. 2021), PAA, and CBLOF (He et al. 2003). The proposed, interval-based, and first-order methods were implemented in Python 3.8.[16] Other method implementations are available in pyts[17] and pycaret[18] packages.

In the following, three scenarios are organized. In the first scenario, the anomaly score assignment process of the CUBOID is visualized. The second scenario compares the CUBOID performance against other methods. The last scenario evaluates how changing the number of clusters (k) affects proposed method's results. Scenario parameters are shown in Table 3.

5.3.1 Scenario 1—Visualization of anomaly score assignment

The first scenario highlights the effectiveness of CUBOID's anomaly score assignment, using multiple competitive methods, such as PAA, interval-based, and first-order methods. These methods were chosen because they share common features with proposed method, including anomaly score calculation and sliding window segmentation. In the scenario, each time series is divided into several sliding windows, and anomaly scores are assigned to each window using the four methods (CUBOID, PAA, interval-based, and first-order). Figures 3, 4, 5, 6, 7 display anomaly scores in bar charts.

Yahoo time series are selected based on attributes and anomalies. Real-29 and TS-16 have no trend. The trend is negative for TS-10 and positive for Synthetic-62. TS-10 has amplitude and shape anomalies. In the scenario, $k = 3$ is passed to the proposed algorithm (Algorithm 1). Each experiment has a fixed window size. For the selected time series of A1, A2, A3, A4, and Sin, set the values of w to 13, 13, 12, 10, and 60, respectively.

Figure 3 presents results obtained for Real-29 time series. In particular, Fig. 3a shows the time series with seven anomalous points. The points were placed in four sliding windows during segmentation process. Among these windows, the second and third contained two and three anomalous points,

[16] https://github.com/ir1979/CUBOID.

[17] https://pyts.readthedocs.io.

[18] https://pycaret.readthedocs.io.

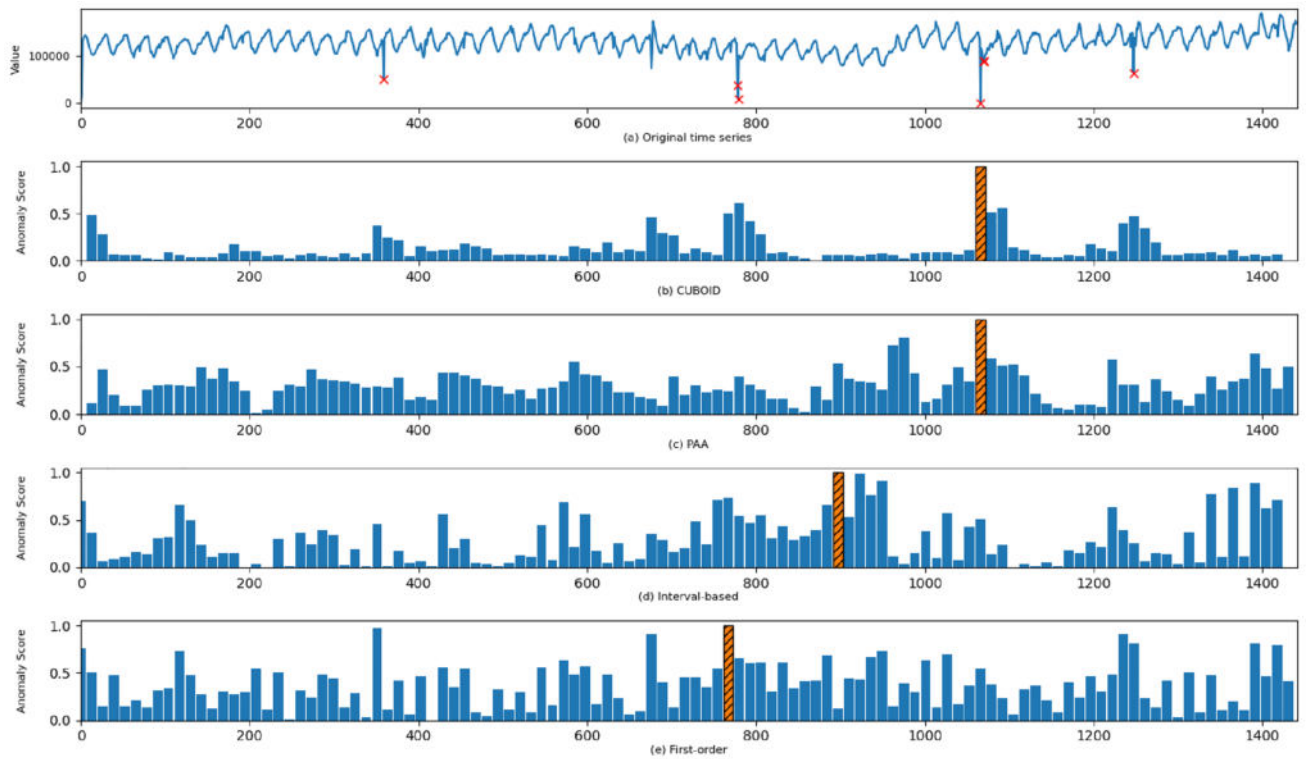

Fig. 3 Scenario 1—Anomaly score assignment on A1 Real-29 dataset. Note that the anomaly score bars are in blue in all scenario figures. Anomalous subsequences are marked with red in the original time series (part (a) of all scenario figures). The dashed bars show anomalous sliding windows detected by these methods (color figure online)

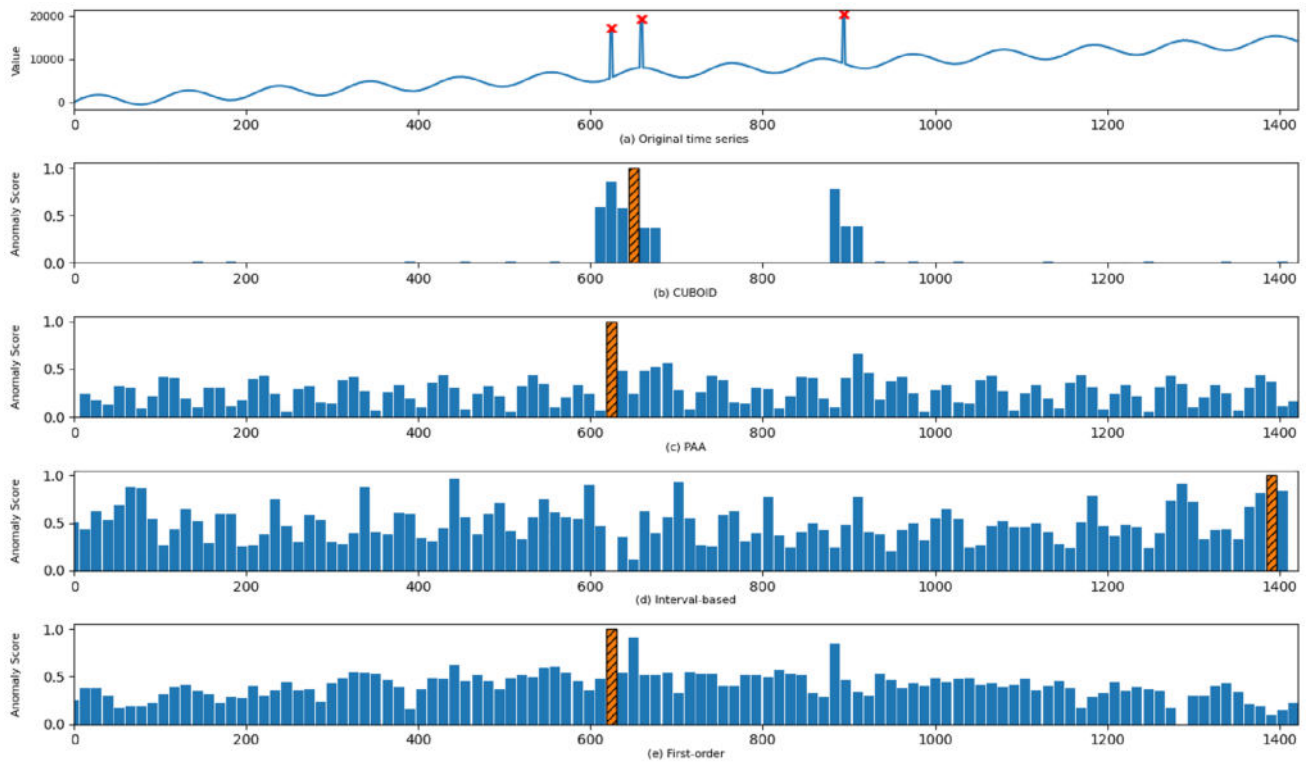

Fig. 4 Scenario 1—anomaly score assignment on A2 Synthetic-62 dataset

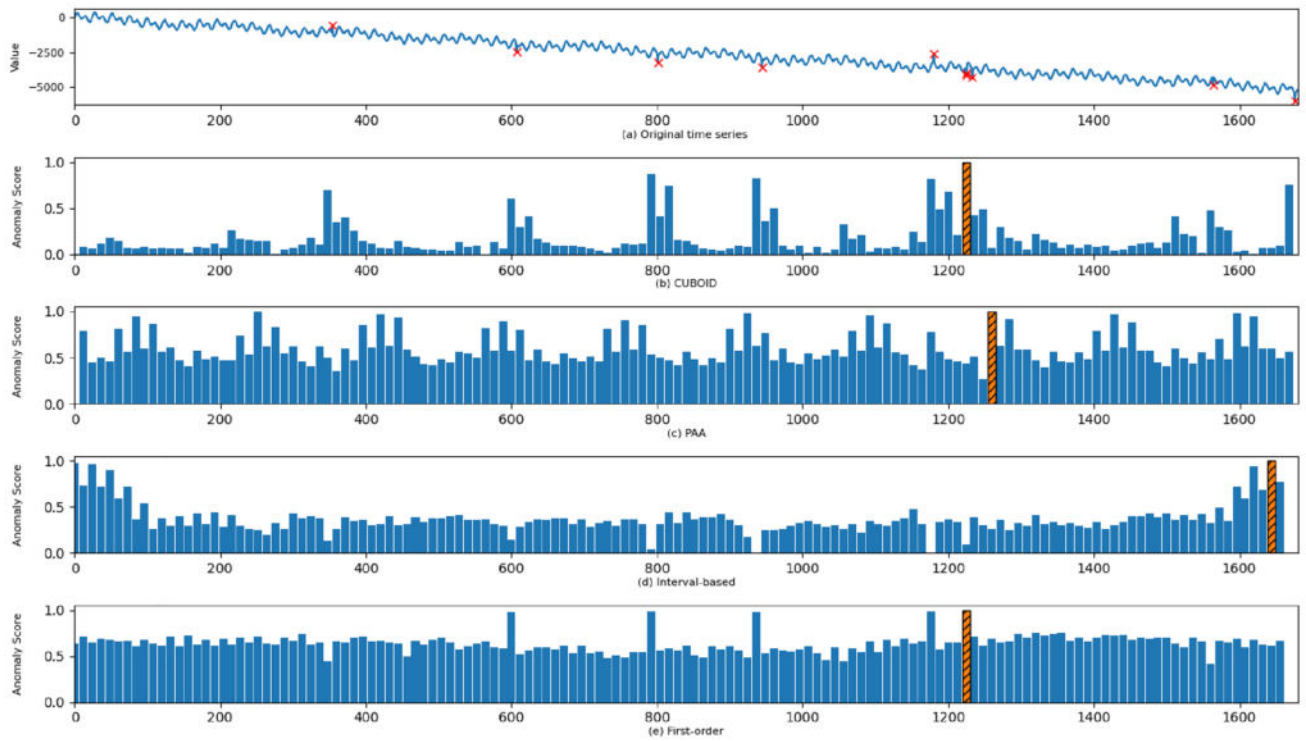

Fig. 5 Scenario 1—anomaly score assignment on A3 TS-10 dataset

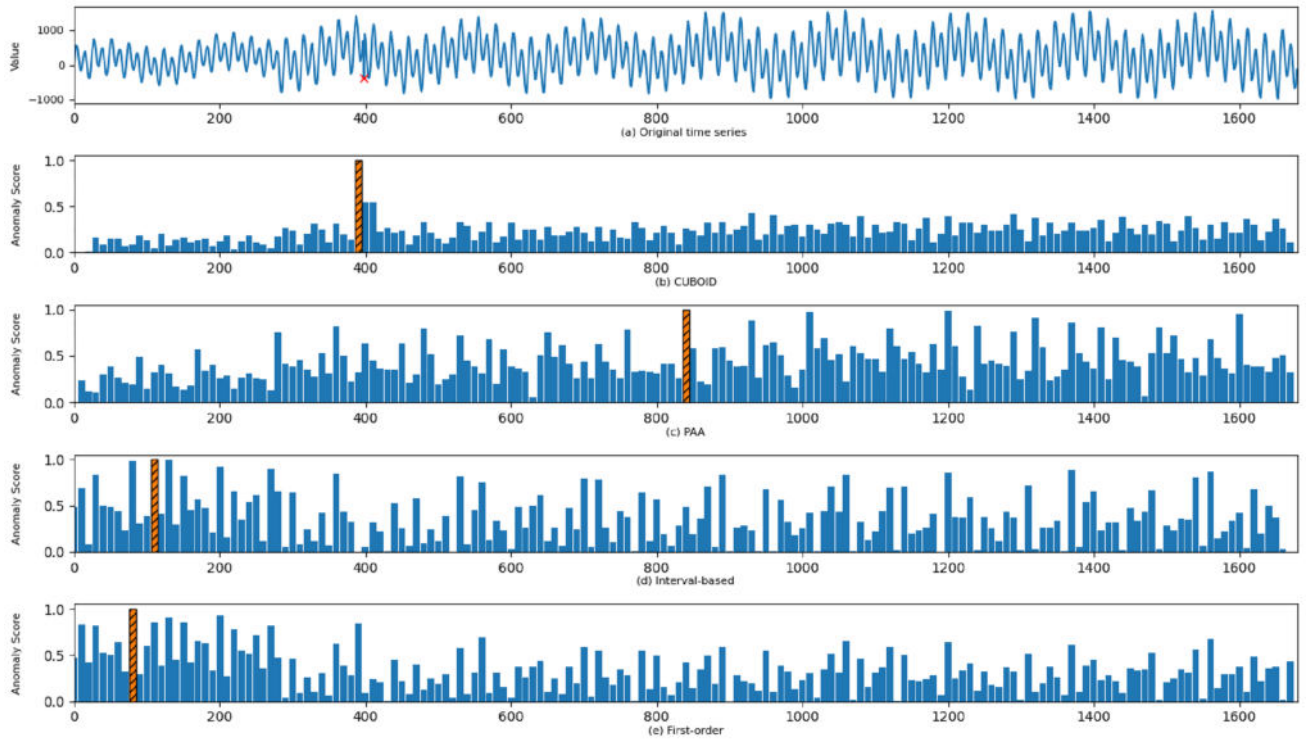

Fig. 6 Scenario 1—anomaly score assignment on A4 TS-16 dataset

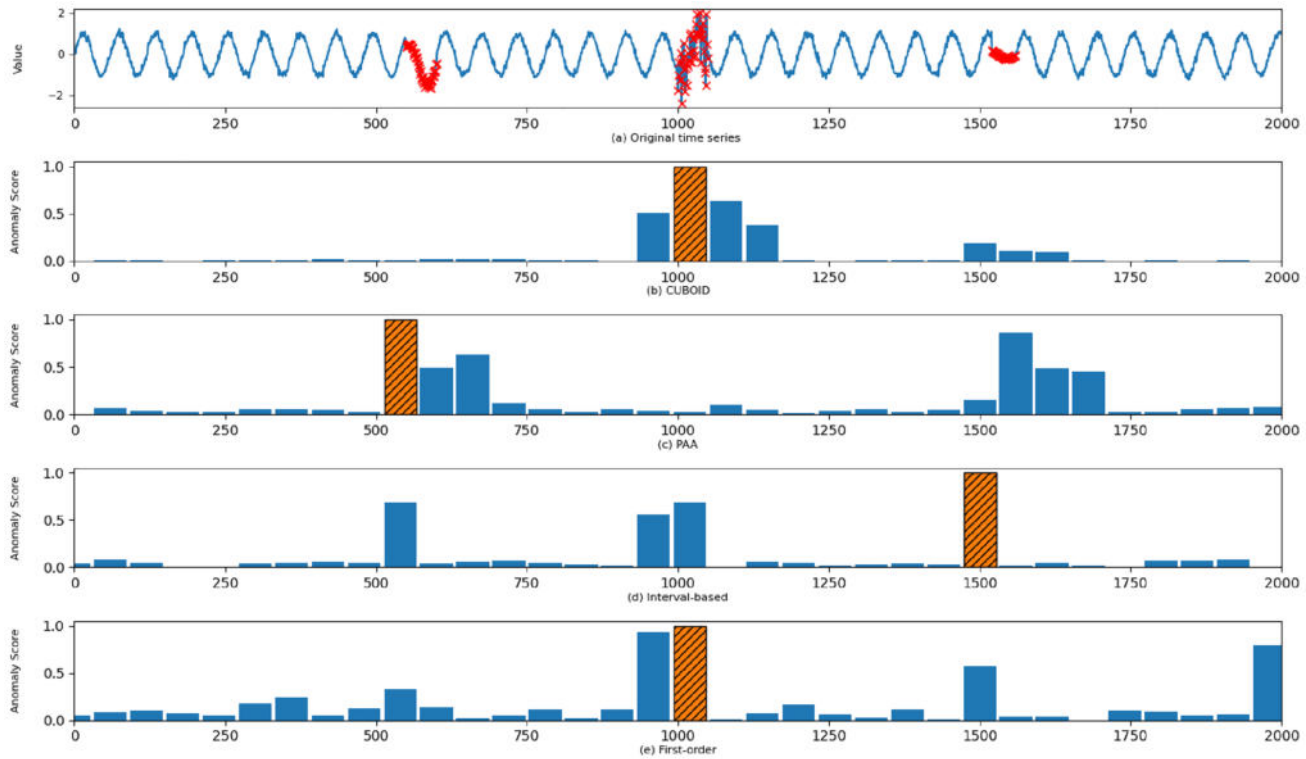

Fig. 7 Scenario 1—anomaly score assignment on Sin dataset

respectively, and were found to be more anomalous than others. Both proposed method, and PAA, as shown in Fig. 3b, c, correctly identified the third anomalous window as an anomaly (dashed bar charts). Other methods could not find any anomalous sliding windows.

Additionally, CUBOID fared better than the other approaches in determining anomaly scores because only the anomaly scores of anomalous windows or their neighbors were 0.5 or higher. Figure 3c–e show high anomaly scores allocated to normal windows, but the anomaly score assignments of the other approaches lacked consistency. Moreover, proposed method's confidence index, which has a value of 7.383, is significantly higher than that of the PAA method (two times).

Synthetic-62 time series, which contains nine amplitude anomalies, is presented in Fig. 4a (As anomalies have appeared in three sequential subsequences, they are not clearly visible in figure). After the segmentation step, these anomalies were grouped into three sliding windows, each containing three sequence point anomalies. The superiority of CUBOID method is clearly shown in Fig. 4b. Unlike other competing approaches that just followed the initial time series changes in the anomaly score assignment procedure, our method successfully recognized all anomalous windows by giving them high anomaly ratings. Moreover, the experiment shows that proposed method outperformed other approaches when applied to a time series showing a positive trend and periodicity, indicating that clustering representation was able to effectively capture the underlying patterns of such a time series.

Figure 5a shows that TS-10 follows a periodic pattern which has a negative trend. The time series contains ten anomalous points and eight anomalous windows. Among these windows, the sixth one stands out as the most anomalous, with three anomalies. As shown in Fig. 5b, e, both CUBOID and first-order methods correctly identified this window as anomalous.

The anomaly score bars in Fig. 5b are highly discriminative, with significantly higher scores observed near anomalous windows compared to normal windows. While the first-order method also performed well in identifying anomalous windows scored half as high, which is not sufficiently discriminative.

The experiment clearly shows that proposed method surpasses other competitive methods when applied to time series data similar to TS-10. Using n-neighboring strategy for anomaly score computation results in producing more discriminative scores, which contributes to CUBOID's superiority.

TS-16 is the third time series included in the experiment. Out of all the time series, it has only one anomaly (as depicted in Fig. 6a). It may be considered the most challenging time

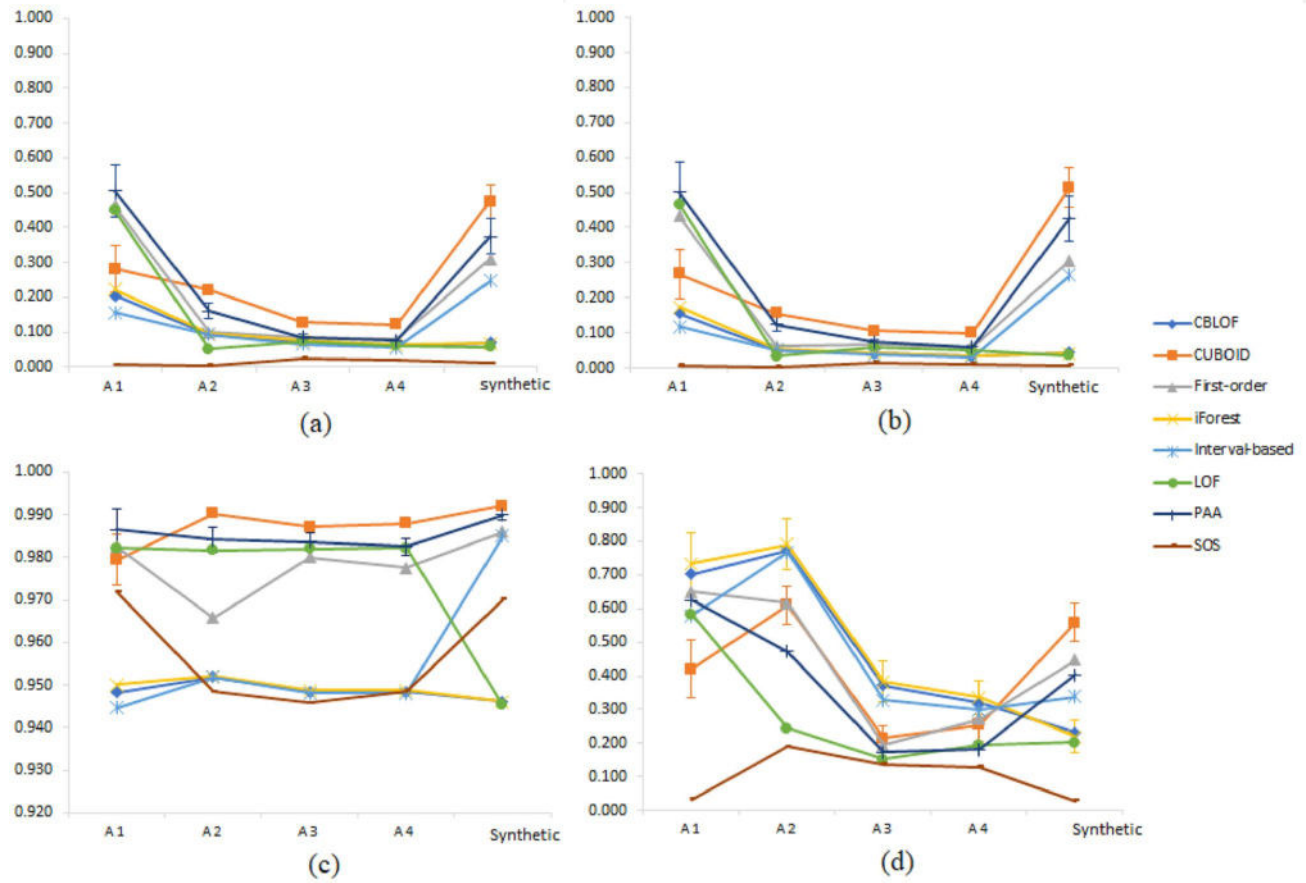

Fig. 8 Scenario 2—**a** F-score, **b** *precision*, **c** *accuracy*, and **d** *AR* measures of different methods. The error bars represent 95% confidence intervals

series to detect the anomalous window. Figure 6 shows that only the proposed technique successfully identifies the anomalous sliding window. The anomaly score assignment for the time series using other methods in the scenario is illustrated in Fig. 6b–e. The stability, and discriminability of proposed method for assigning anomaly scores are evident from the figure. CUBOID method assigns scores of less than 0.25 to the normal points, while other methods assign scores in the wide range of [0,1] to the normal points. The robust dissimilarity approach of the suggested technique, which uses the distance calculation between the n prior neighbors to discover changes in the time series' behavior, is responsible for its efficient performance.

In Sin time series, there are three anomalous windows (as indicated by the red points in Fig. 7a, based on Eq. (8)). While all scenario methods can detect one anomalous window, the interval-based, and PAA methods are more effective in assigning anomaly scores to anomalous windows, with scores of over 0.5 (as shown in Fig. 7c, d). On the other hand, the CUBOID method performs well to assign scores close to zero for normal windows.

In summary, CUBOID usually achieves the most desirable results. However, in some cases, PAA or first-order methods win. A distinguishing feature of proposed method is that it assigns the lowest average of anomaly score values for normal points compared with other techniques.

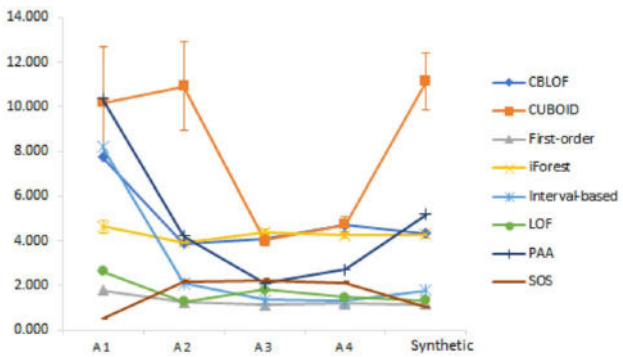

Fig. 9 Scenario 2—the *CI* measures of different methods. The error bars represent 95% confidence intervals

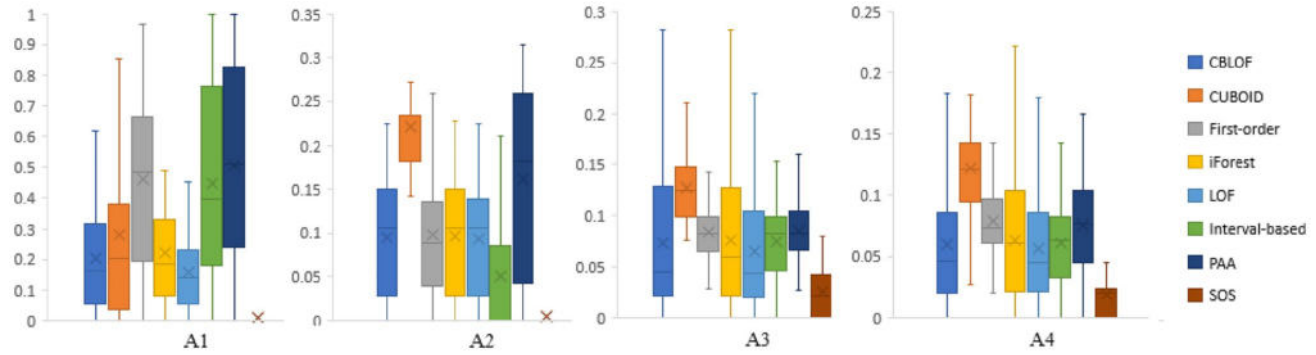

Fig. 10 Scenario 2—box plots of F-score distributions for different methods in Yahoo S5 dataset. The × marker represents the mean of the data, the − marker represents the median, and the colored box indicates the interquartile range (IQR) from the first quartile (Q1) to the third quartile (Q3). The lower and upper whiskers extend to the furthest data points within 1.5 times the IQR in each wing (color figure online)

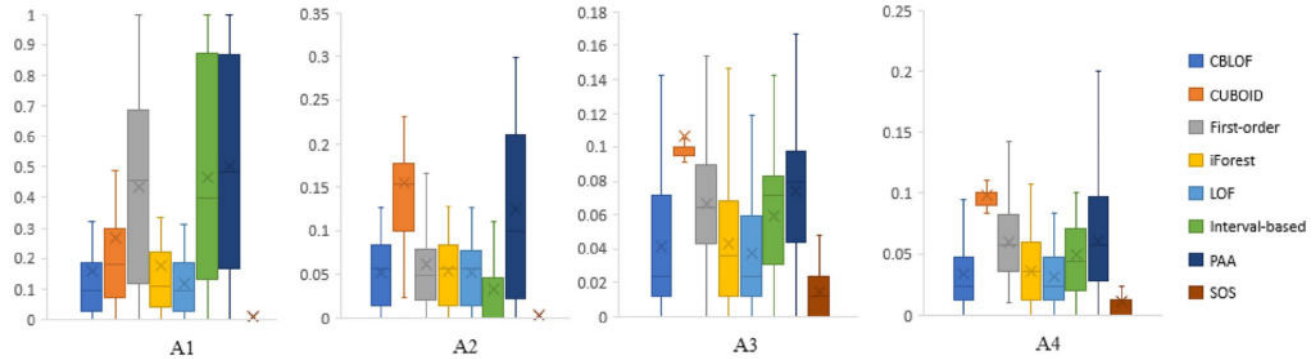

Fig. 11 Scenario 2—*Precision* distributions of different methods for Yahoo S5 dataset. The × marker represents the mean of the data, the − marker represents the median, and the colored box indicates the interquartile range (IQR) from the first quartile (Q1) to the third quartile (Q3). The lower and upper whiskers extend to the furthest data points within 1.5 times the IQR in each wing (color figure online)

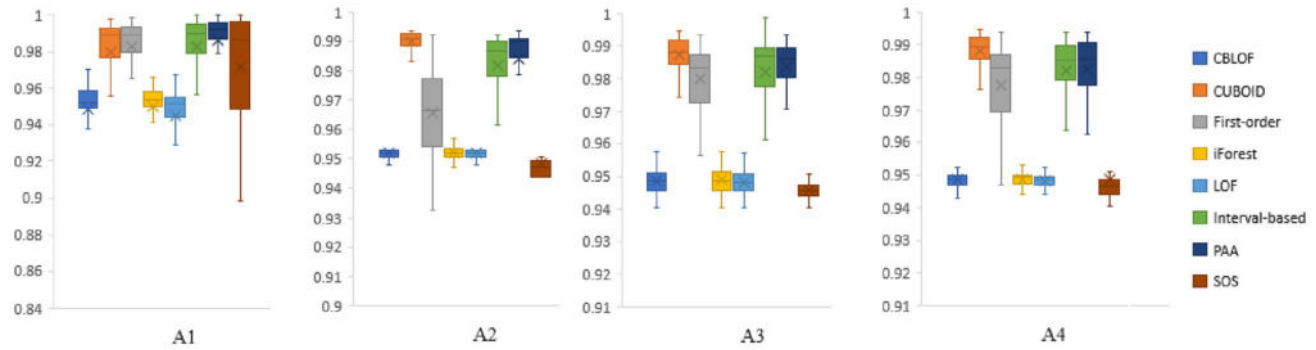

Fig. 12 Scenario 2—*Accuracy* distributions of different methods for Yahoo S5 dataset. The × marker represents the mean of the data, the − marker represents the median, and the colored box indicates the interquartile range (IQR) from the first quartile (Q1) to the third quartile (Q3). The lower and upper whiskers extend to the furthest data points within 1.5 times the IQR in each wing (color figure online)

5.3.2 Scenario 2—Evaluation of the CUBOID Performance

The second scenario is designed to show the comprehensive results of proposed method compared to other anomaly detection methods. Unlike the first scenario, these competitive methods do not necessarily have the same characteristics as proposed method. Several unsupervised methods such as CBLOF, first-order, iForest, LOF, interval-based, PAA, and SOS are selected for fair comparison. These methods are introduced in Sects. 1 and 3. In

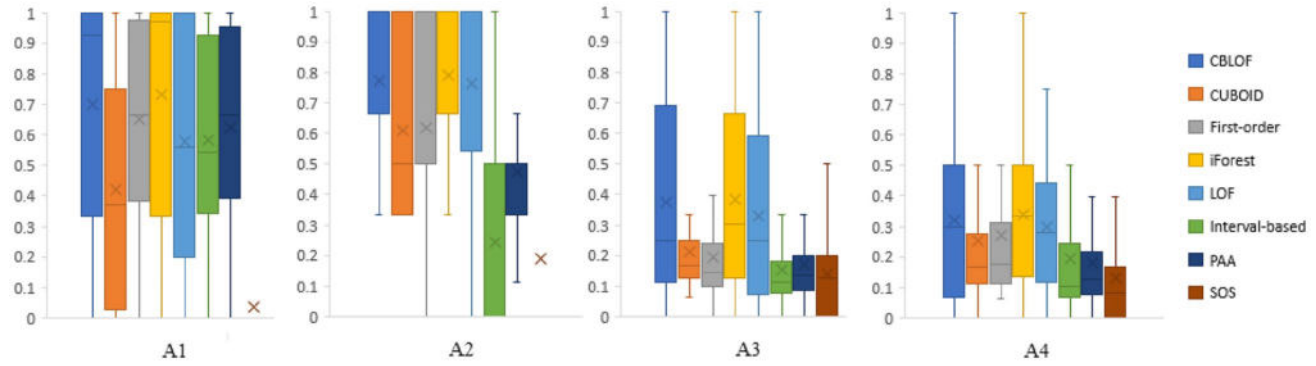

Fig. 13 Scenario 2—*AR* distributions of different methods for Yahoo S5 dataset. The × marker represents the mean of the data, the – marker represents the median, and the colored box indicates the interquartile range (IQR) from the first quartile (Q1) to the third quartile (Q3). The lower and upper whiskers extend to the furthest data points within 1.5 times the IQR in each wing (color figure online)

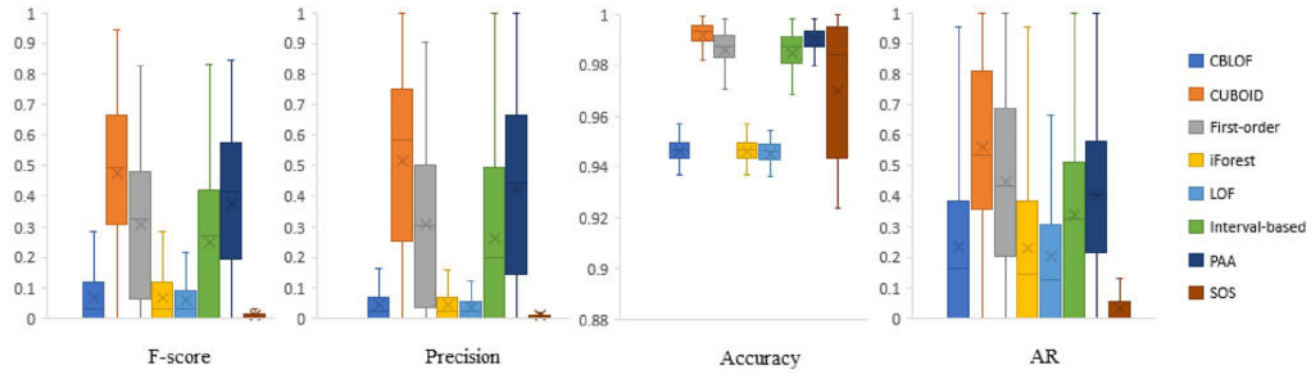

Fig. 14 Scenario 2—Performance index measure distributions of different methods for Synthetic dataset

second scenario, the window size varies between 10 and 100 for sliding window-based methods such as CUBOID, first-order, interval-based, and PAA. The best results of each method are reported in this scenario. The scenario is applied to Yahoo S5 and Synthetic datasets.

Performance evaluation results are presented using different index measures: F-score, *precision*, *accuracy*, *AR*, and *CI*. We used one-way ANOVA, and Post Hoc test to analyze the results. Furthermore, the normalization test was conducted using one-sample Kolmogorov–Smirnov test. The results of the scenario are presented in three different ways. Figures 8 and 9 provide an overall overview. More detailed information about the charts and their index measure distributions is displayed through box plots in Figs. 10, 11, 12, 13, 14. Appendix contains numerical results (Tables 5, 6, 7, 8).

Figure 8 displays the results in line charts based on four indices: F-score and *precision*, *accuracy*, and *AR*. In Fig. 8a–c, it can be observed that orange chart, which corresponds to the F-score, *precision*, and *accuracy* measures of the CUBOID method, is situated at the top of all the charts except A1. It indicates that CUBOID method attains the highest indices across datasets.

The high performance of CUBOID method in F-score index can be attributed to its superior *precision* index. Increasing the number of true positive samples or decreasing the number of false positive samples in the confusion matrix will result in superiority. This improvement is particularly noteworthy in Yahoo datasets, where anomalies are often rare and consist of point anomalies. In these cases, reducing false positives is critical to achieving high *precision*. CUBOID method attains these results using an optimal clustering algorithm that accurately captures the underlying data distribution. By enabling the model to detect changes in the normal data distribution more accurately, false positives can be reduced.

The *AR* measure performs differently from the other measures in Yahoo dataset, as shown in Fig. 8d. The iForest method shows the highest performance for Yahoo dataset, while CUBOID performs the best for the Synthetic dataset. Additionally, CUBOID, and interval-based methods show similar *AR* effectiveness for all datasets.

Figure 9 confirms the findings of the first scenario that the anomaly score assignments of proposed method are significantly discriminative compared to other methods. It is achieved through two critical steps in TAD method: (1) the representation step and (2) the calculation of window similarity or dissimilarity step. A common approach to compute window dissimilarity is to calculate the distance between each window, and all other windows, as is done in Zhou's method. In contrast, proposed method employs the distance between a window, and its n previous neighbors ($n = 2$) as the dissimilarity score.

The charts in Fig. 8 demonstrate that n-neighbor strategy used by proposed method outperforms other solutions. This superiority is achieved because the clustering algorithm used in the time series representation process effectively captures the patterns of subsequences. Furthermore, detecting changes in window patterns using n-neighbor distance strategy is sufficient.

To provide a more precise and statistical discussion of scenario results, the box plots of the indices are presented in Figs. 10, 11, 12, 13. The results for F-score, and *precision* values presented in Fig. 8a, b are supported by the box plots shown in Figs. 10 and 11 for Yahoo datasets. In addition, the short length of the interquartile range (IQR) for the suggested technique demonstrates that the variation of index measures for the CUBOID is minimal, verifying the consistent performance of proposed method across various time series. Moreover, the ANOVA test results showed that there was a significant and meaningful difference in the performance of the algorithms ($p < 0.05$). Further analysis using Tukey HSD test on the F-score results reveals that proposed method outperforms other algorithms on all datasets except A1. Moreover, PAA method achieves better performance than the other baseline methods on the A1 dataset.

There are no significant differences in the *precision* index between proposed method and PAA. Moreover, the Post Hoc test shows that PAA and interval-based methods demonstrate the most significant differences compared to scenario methods for A1 dataset.

In Fig. 12, the distribution of the CUBOID box plot is not only skewed toward the top (one), but it is also positioned above the other approaches with a narrow interquartile range. ANOVA analysis indicated a significant difference in the *accuracy* scores of proposed method compared to the baseline models for A3 and A4 datasets.

The boxplots of *AR* in Fig. 13 do not show a clear superiority of a particular method for Yahoo datasets. The distribution and skewness of box plots and ANOVA test results indicate different winners for each Yahoo subset. For A1, the ANOVA test indicates that CBLOF outperforms the other methods, and the performance of the other methods does not have significant differences. Sheffe test results show that the effectiveness of CBLOF, first-order, iForest, LOF, and interval-based is the same in A2, and the best average is attributed to CBLOF. In A3 dataset, the results of CUBOID, and interval-based tests are homogenous according to homogenous Scheffe test. The p-value for A4 dataset is greater than 0.05, indicating that there are no significant differences in *AR* measures among the methods.

Figure 14 shows the box plot distributions for Synthetic dataset, revealing that the CUBOID method has whiskers that are more skewed towards the top. The p-value for ANOVA test is less than 0.05, indicating a statistically significant difference between the techniques. The Post Hoc test suggests that the suggested CUBOID approach has a higher F-score than the other methods. In terms of *precision* and *accuracy* measures, CUBOID and PAA methods show similar performance levels in the dataset. Furthermore, Scheffe test results show that both the CUBOID and first-order methods outperform the other methods in terms of *AR* measure.

In general, proposed method outperforms other baseline methods in this scenario, except for a few cases in *AR* index measure or A1 dataset. However, the ANOVA and Post Hoc tests show that different methods may have better performance in different datasets, which highlights the importance of selecting appropriate methods for specific anomaly detection tasks.

Moreover, the computational cost of the interval-based and first-order methods, according to Sect. 4.4, is about the same as that of proposed method. On the one hand, these methods use all sliding windows to calculate anomaly score values. This contrasts with CUBOID, in which only n previous neighbors are considered for calculating anomaly scores. This results in a more efficient algorithm. This is supported in experiments where CUBOID was much faster than interval-based and first-order methods. However, even though the simple implementation of the algorithm in this study was not optimized for achieving the fastest running time in the first place, the experiments show promising improvements in this direction when compared with similar techniques of interval-based and first-order algorithms.

5.3.3 Scenario 3—The effect of the number of clusters on the CUBOID performance

The third scenario evaluates the effect of changing the number of clusters (k) in proposed method. The scenario is divided into two parts. The first part contains the anomaly score visualization of the scenario experiments on selected datasets (Figs. 15, 16, 17, 18, and 19). The second part includes the evaluation results of the scenario on all datasets (Table 4).

The method is run on datasets from the first scenario in first part of the scenario. The window sizes are also similar

Table 4 Scenario 3-Performance results of Yahoo S5, Synthetic, and Sin datasets for different values of $k \in \{2, 3, 4, 5\}$

	Clusters	*Precision*	*Accuracy*	F-score	*AR*	*CI*
A1	2	0.282 ± 0.08	0.982 ± 0.01	0.290 ± 0.07	0.415 ± 0.09	10.512 ± 2.04
	3	0.267 ± 0.06	0.979 ± 0.00	0.282 ± 0.05	0.420 ± 0.07	10.060 ± 2.07
	4	0.275 ± 0.06	0.978 ± 0.07	0.289 ± 0.00	0.465 ± 0.05	10.106 ± 2.03
	5	0.272 ± 0.06	0.979 ± 0.00	0.292 ± 0.07	0.468 ± 0.07	9.901 ± 1.97
A2	2	0.169 ± 0.02	0.991 ± 0.00	0.229 ± 0.01	0.610 ± 0.07	10.609 ± 2.09
	3	0.155 ± 0.01	0.990 ± 0.00	0.221 ± 0.01	0.610 ± 0.06	10.923 ± 2.00
	4	0.178 ± 0.01	0.991 ± 0.00	0.240 ± 0.01	0.610 ± 0.06	11.425 ± 1.98
	5	0.196 ± 0.02	0.992 ± 0.00	0.256 ± 0.01	0.610 ± 0.06	11.812 ± 1.91
A3	2	0.100 ± 0.01	0.987 ± 0.00	0.121 ± 0.01	0.211 ± 0.04	4.216 ± 0.24
	3	0.106 ± 0.01	0.987 ± 0.00	0.127 ± 0.13	0.214 ± 0.04	4.019 ± 0.21
	4	0.108 ± 0.01	0.988 ± 0.00	0.126 ± 0.01	0.207 ± 0.04	3.816 ± 0.11
	5	0.122 ± 0.01	0.989 ± 0.00	0.139 ± 0.01	0.214 ± 0.04	3.850 ± 0.10
A4	2	0.090 ± 0.01	0.986 ± 0.00	0.114 ± 0.01	0.259 ± 0.06	4.503 ± 0.38
	3	0.099 ± 0.01	0.988 ± 0.00	0.122 ± 0.01	0.255 ± 0.05	4.742 ± 0.36
	4	0.098 ± 0.01	0.988 ± 0.00	0.120 ± 0.01	0.242 ± 0.05	4.214 ± 0.23
	5	0.107 ± 0.01	0.988 ± 0.00	0.128 ± 0.01	0.259 ± 0.05	4.114 ± 0.22
Synthetic	2	0.518 ± 0.12	0.991 ± 0.00	0.476 ± 0.06	0.577 ± 0.07	8.417 ± 1.17
	3	0.515 ± 0.06	0.992 ± 0.00	0.476 ± 0.06	0.559 ± 0.06	11.132 ± 1.30
	4	0.519 ± 0.06	0.991 ± 0.00	0.480 ± 0.05	0.577 ± 0.06	11.168 ± 1.14
	5	0.520 ± 0.06	0.991 ± 0.00	0.474 ± 0.05	0.581 ± 0.06	11.759 ± 1.12
Sin	2	1.000	0.955	0.523	0.355	5.114
	3	1.000	0.955	0.523	0.355	8.166
	4	1.000	0.955	0.523	0.355	6.632
	5	1.000	0.955	0.523	0.355	8.479

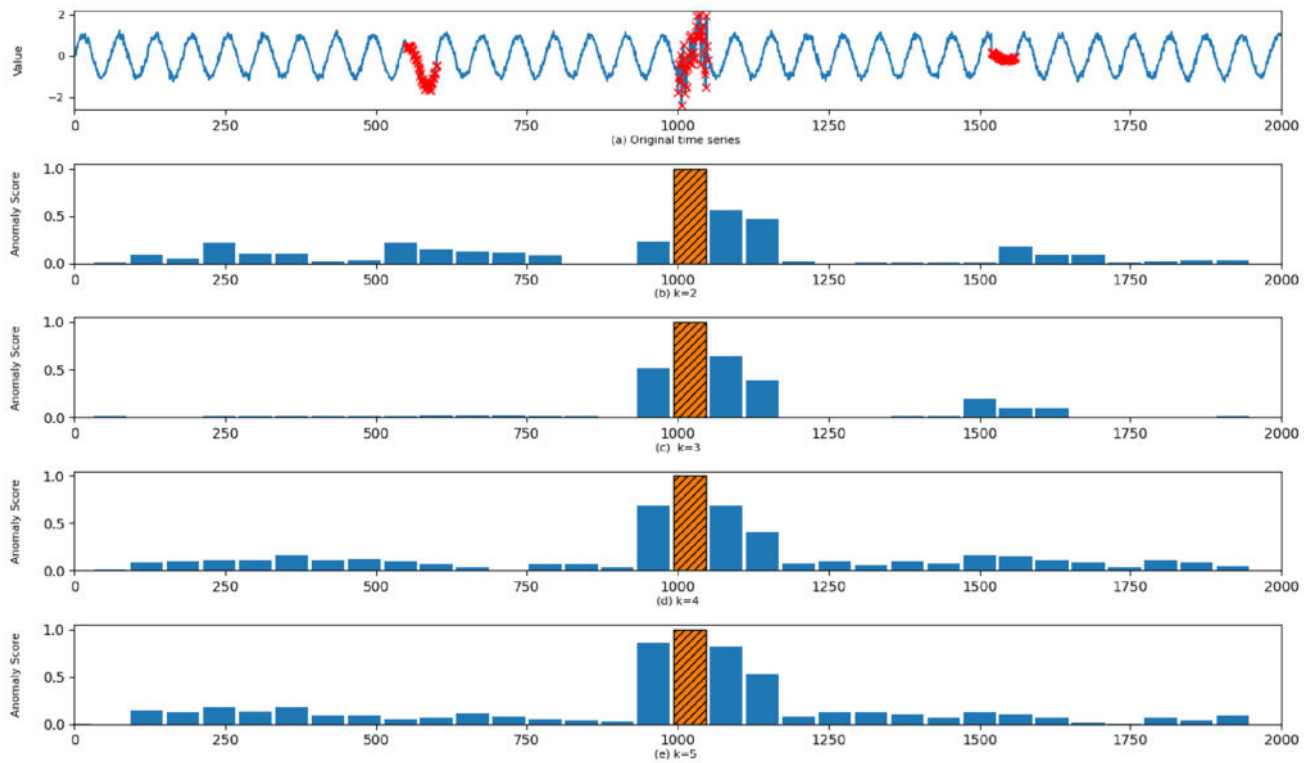

Fig. 15 Scenario 3—results of the CUBOID experiments on Sin dataset, the number of clusters $k \in \{2, 3, 4, 5\}$

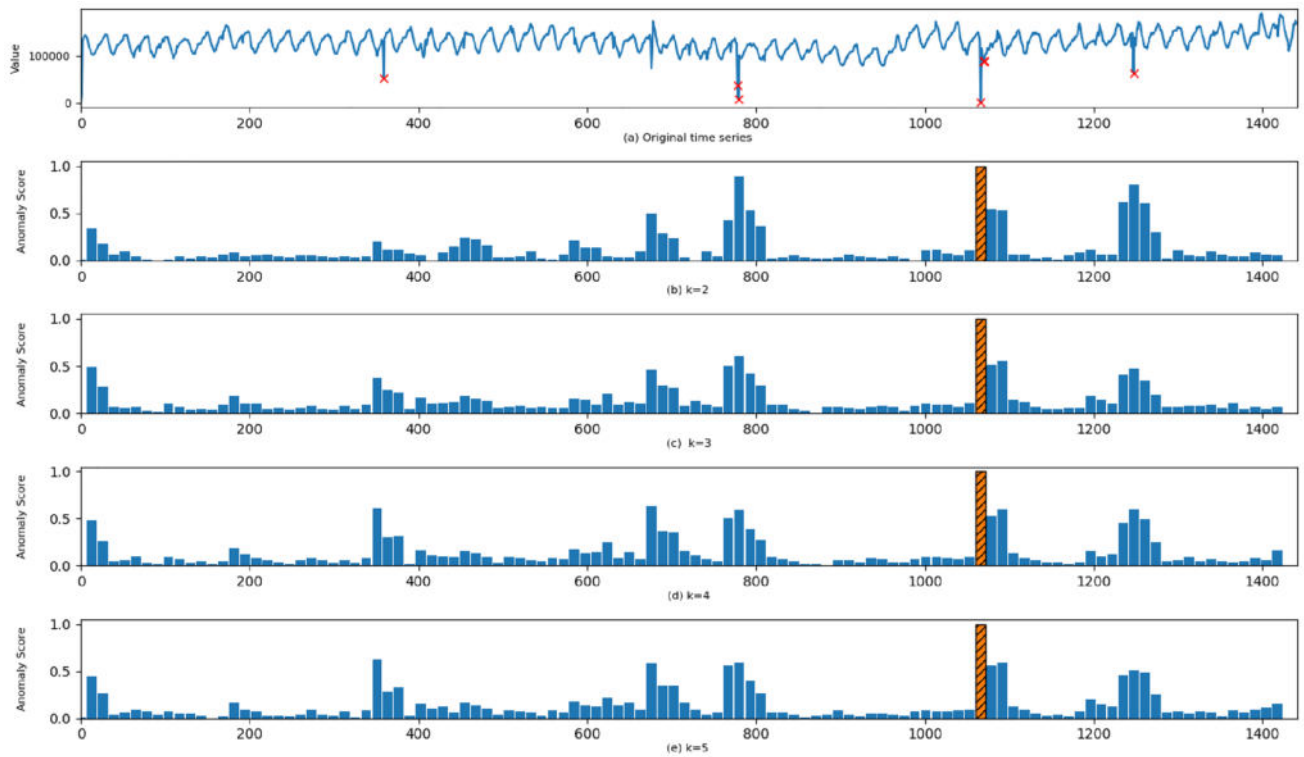

Fig. 16 Scenario 3—results of the CUBOID experiments on A1 Real-29 dataset with the number of clusters $k \in \{2, 3, 4, 5\}$

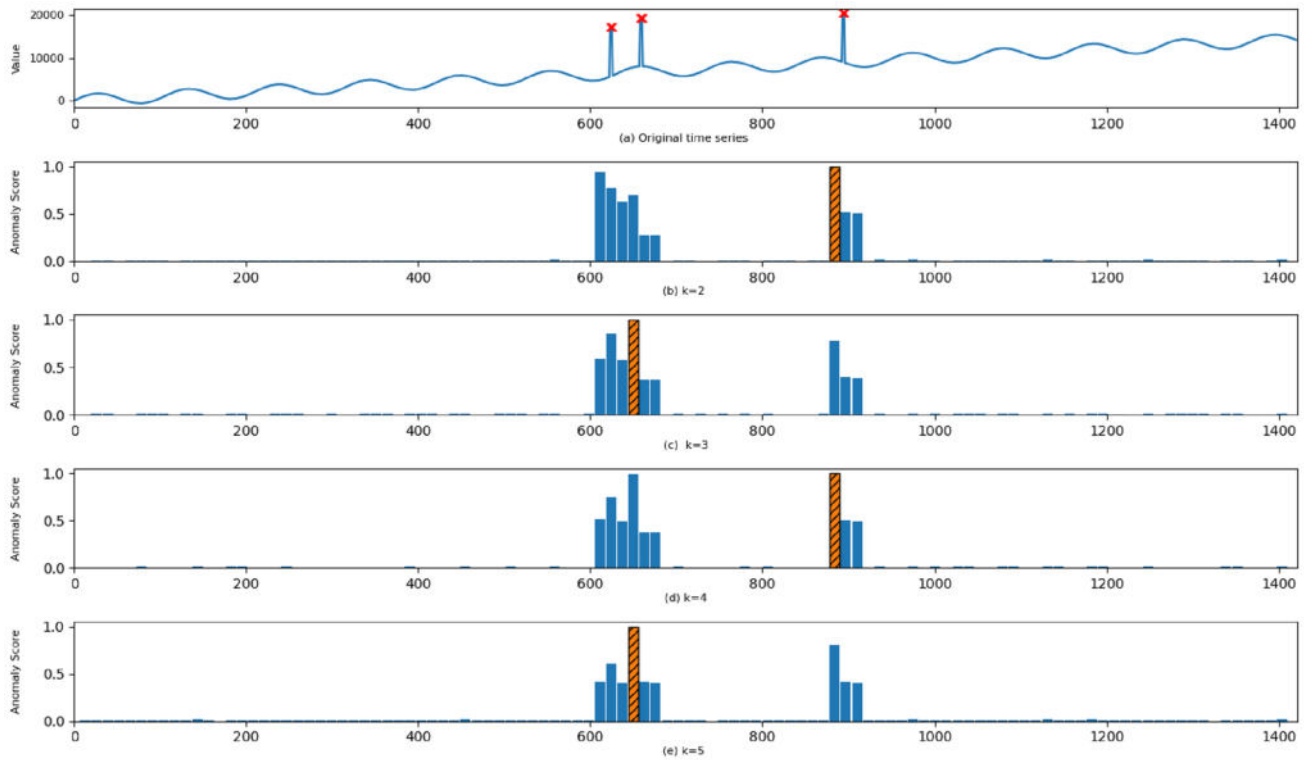

Fig. 17 Scenario 3—results of the CUBOID experiments on A2 Synthetic-62 dataset with the number of clusters $k \in \{2, 3, 4, 5\}$

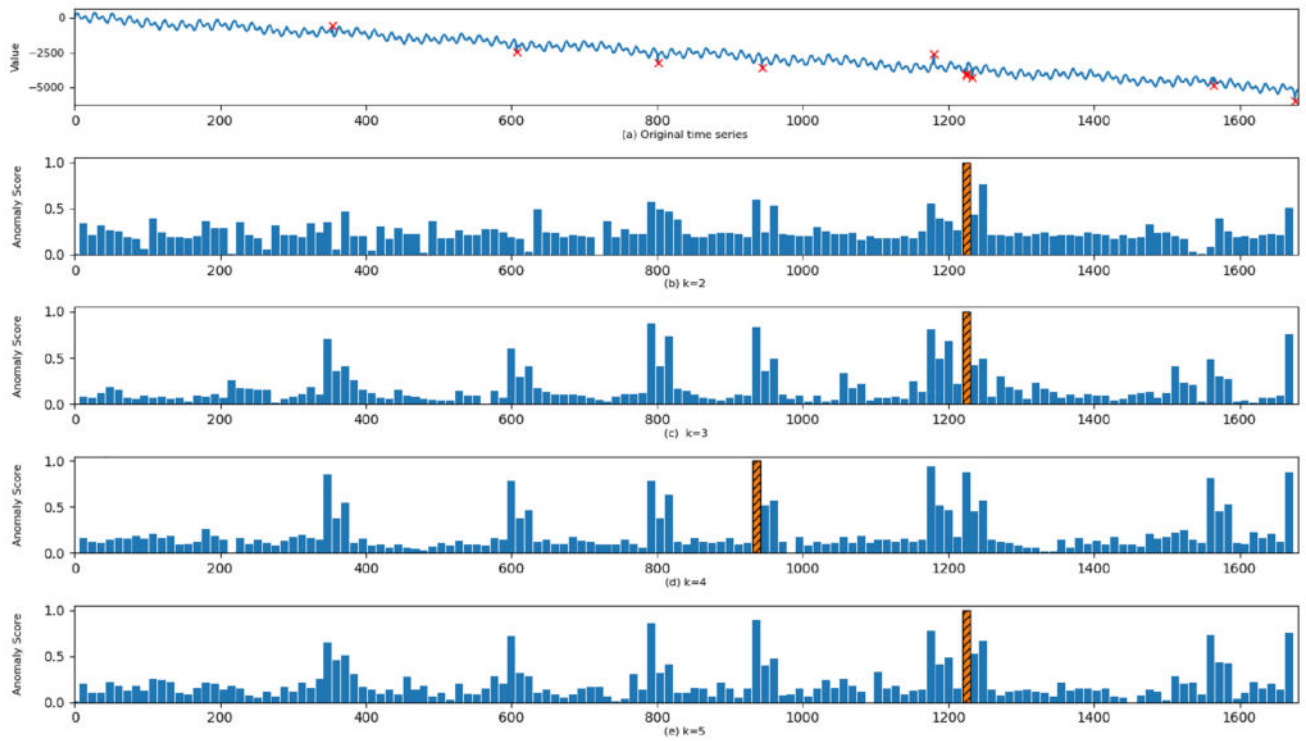

Fig. 18 Scenario 3—results of the CUBOID experiments on A3 TS-10 dataset with the number of clusters $k \in \{2, 3, 4, 5\}$

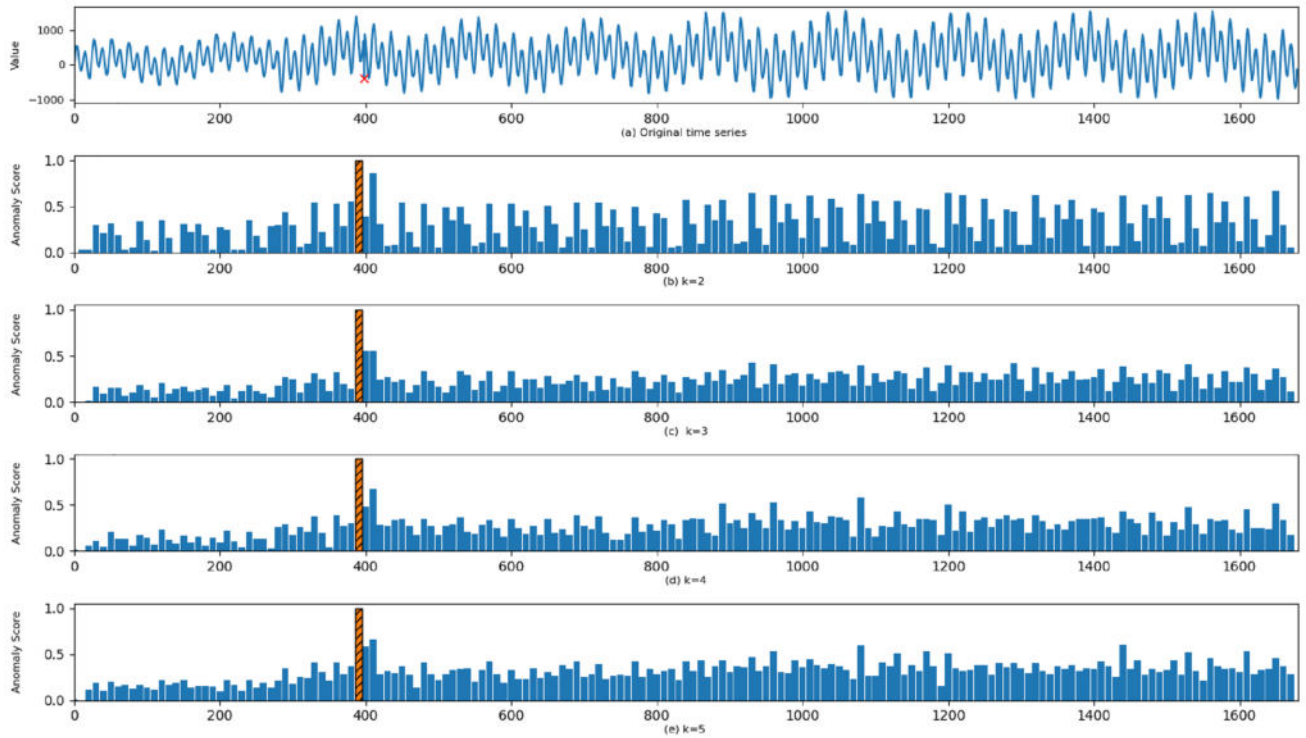

Fig. 19 Scenario 3—results of the CUBOID experiments on A4 TS-16 dataset with the number of clusters $k \in \{2, 3, 4, 5\}$

to the first scenario while the number of clusters (k) varies between 2 and 5.

Figure 15 shows the results of implementing the scenario on the Sin dataset with $k = 2, 3, 4$, and 5. The bar charts in the figure illustrate that the detected anomalous window is the same across all experiments for the dataset. However, the experiment with $k = 3$ yielded the highest *CI* index, indicating that setting the number of clusters to 3 results in the CUBOID producing the most discriminative anomaly scores, as shown in Fig. 15c.

The evaluation results of proposed method for Real-29 in Fig. 16 and TS-16 in Fig. 19 show that the model performance is quite independent of the number of clusters. Moreover, all experiment indices, except *CI*, were almost the same. In all experiments of Real-29 and TS-16, a certain anomalous sliding window is chosen.

Figures 17 and 18 for Synthetic-62 and TS-10 show that the results of the method are the same in most cases. Two out of four experiments with Synthetic-62 ($k = 2, 4$ and $k = 3, 5$) and three with TS-10 ($k = 2, 3, 5$) show the same anomalous sliding window, and their performance indices are the same.

In the second part of the scenario, overall evaluation results with Yahoo S5, Synthetic, and Sin datasets are presented in Table 4. The table shows the performance results of the model using various index measures such as *precision*, *accuracy*, F-score, *AR*, and *CI* with a confidence interval of 95%. The table confirms that variations in the number of clusters have small effects on index measures.

The table shows some indices are approximately constant. For example, *precision* in Synthetic, *AR* in A2, *accuracy* in A2, A3, A4, and Synthetic, and F-score in A1 and A2 have no significant changes. For Sin, all index measures remained completely constant (Table 4). In other cases, index measures show very small amplitude changes. For example, in A1, when k varied between 2 and 5, the F-score changed slightly between 0.290 and 0.292.

The table generally illustrates that *accuracy* changes by 0.004 for different cluster numbers. Changes for other indices ranged from 0.001 to 0.01 (Table 4). In summary, the results confirm that proposed model is very insensitive to the number of clusters. Therefore, $k = 3$ is recommended for practical applications based on the experiment.

6 Conclusion

This paper presents a novel clustering-based time series representation technique and a method for anomaly detection called CUBOID. Proposed representation method transforms the original time series into cluster centroids. The suggested representation method has two advantages: (1) the primary time series is transformed into a modified form that can better capture changes in time series, and (2) the input length is reduced, enabling a faster algorithm. The results show that proposed anomaly detection method achieved almost the highest performance among the other anomaly detection methods.

For future work, some components of CUBOID can be further developed. For example, an adaptive window size algorithm may be an interesting extension to the representation module. In addition, a weighted distance measure can be considered in the anomaly score computation phase. An adaptive threshold mechanism can improve the performance of an anomaly detection algorithm by dynamically adjusting the threshold based on the data. Setting the threshold to a predetermined value may not be appropriate for all types of data. In contrast, an adaptive threshold mechanism may modify the threshold depending on the properties of the data. This strategy allows the algorithm to be more adaptable and sensitive to changes in the data, which may lead to improved performance. Also, the work could extend to multivariate TAD.

Appendix: Tables of Scenario 2

The numerical results in Tables 5, 6, 7, 8 are presented by a confidence interval of 95%.

Table 5 Scenario 2—Performance indices of different methods for Sin dataset

	Precision	*Accuracy*	F-score	*AR*	*CI*
CBLOF	0.350	0.915	0.290	0.249	4.161
CUBOID	**1.000**	**0.955**	**0.524**	*0.355*	**8.166**
First-order	**1.000**	**0.955**	*0.523*	*0.355*	6.011
iForest	0.320	0.912	0.265	0.227	4.153
Interval-based	0.714	0.945	0.473	*0.355*	7.376
LOF	0.330	0.913	0.273	0.234	*7.965*
PAA	*0.927*	*0.953*	0.520	**0.362**	4.493
SOS	0.130	0.893	0.108	0.092	2.659

Values in bold and italics indicate the first and second-best results, respectively

Table 6 Scenario 2—Performance indices of different methods for Yahoo dataset

		Precision	*Accuracy*	F-score	*AR*	*CI*
CBLOF	A1	0.156 ± 0.05	0.948 ± 0.00	0.204 ± 0.05	*0.702* ± 0.09	7.744 ± 0.98
	A2	0.052 ± 0.01	0.952 ± 0.00	0.094 ± 0.02	*0.775* ± 0.08	3.871 ± 0.16
	A3	0.041 ± 0.01	0.949 ± 0.00	0.073 ± 0.01	*0.373* ± 0.06	**4.088** ± 0.10
	A4	0.034 ± 0.01	0.948 ± 0.00	0.060 ± 0.01	*0.319* ± 0.05	4.712 ± 0.18
CUBOID	A1	0.267 ± 0.07	0.979 ± 0.01	0.282 ± 0.07	0.420 ± 0.09	*10.181* ± 2.03
	A2	**0.155** ± 0.01	**0.990** ± 0.00	**0.221** ± 0.01	0.610 ± 0.06	**10.925** ± 2.00
	A3	**0.106** ± 0.01	**0.987** ± 0.00	**0.127** ± 0.01	0.214 ± 0.04	*4.019* ± 0.21
	A4	**0.099** ± 0.01	**0.988** ± 0.00	**0.122** ± 0.01	0.255 ± 0.05	**4.742** ±0.36
First-order	A1	0.434 ± 0.07	0.982 ± 0.00	0.461 ± 0.07	0.651 ± 0.07	1.787 ± 0.24
	A2	0.061 ± 0.01	0.966 ± 0.00	0.099 ± 0.01	0.617 ± 0.05	1.247 ± 0.03
	A3	0.067 ± 0.01	0.980 ± 0.00	0.084 ± 0.01	0.193 ± 0.03	1.146 ± 0.02
	A4	*0.060* ± 0.01	0.978 ± 0.00	*0.079*± 0.01	0.271 ± 0.05	1.187 ± 0.03
iForest	A1	0.174 ± 0.05	0.950 ± 0.00	0.223 ± 0.05	**0.733** ± 0.09	4.644 ± 0.26
	A2	0.053 ± 0.01	0.952 ± 0.00	0.096 ± 0.02	**0.791** ± 0.07	3.933 ± 0.13
	A3	0.043 ± 0.01	0.949 ± 0.00	0.076 ± 0.01	**0.382** ± 0.06	4.406 ± 0.09
	A4	0.036 ± 0.01	0.949 ± 0.00	0.063 ± 0.01	**0.337** ± 0.05	*4.243* ± 0.10
Interval-based	A1	*0.467* ± 0.09	0.982 ± 0.01	0.450 ± 0.08	0.583 ± 0.08	2.624 ± 0.39
	A2	0.032 ± 0.01	*0.982* ± 0.00	0.051 ± 0.01	0.244 ± 0.06	1.271 ± 0.03
	A3	0.060 ± 0.01	0.982 ± 0.00	0.074 ± 0.01	0.153 ± 0.03	1.823 ± 0.11
	A4	0.050 ± 0.01	*0.982* ± 0.00	0.061 ± 0.01	0.193 ± 0.05	1.471 ± 0.05
LOF	A1	0.117 ± 0.03	0.945 ± 0.01	0.157 ± 0.03	0.579 ± 0.09	8.181 ± 2.01
	A2	0.052 ± 0.01	0.952 ± 0.00	0.094 ± 0.02	0.766 ± 0.08	2.107 ± 0.34
	A3	0.037 ± 0.01	0.948 ± 0.00	0.065 ± 0.01	0.328 ± 0.06	1.369 ± 0.08
	A4	0.032 ± 0.01	0.948± 0.00	0.056 ± 0.01	0.298± 0.04	1.317 ± 0.03
PAA	A1	**0.502** ± 0.08	**0.986** ± 0.00	**0.505** ± 0.08	0.626 ± 0.08	**10.344** ± 2.13
	A2	*0.124* ± 0.02	0.984 ± 0.00	*0.161* ± 0.02	0.473 ± 0.07	*4.221* ± 0.72
	A3	*0.074* ± 0.01	*0.984* ± 0.00	*0.086* ± 0.01	0.173 ± 0.03	2.090 ± 0.07
	A4	*0.060*± 0.01	*0.982* ± 0.00	0.076± 0.01	0.181 ± 0.04	2.711 ± 0.13
SOS	A1	0.007 ± 0.01	0.972 ± 0.01	0.007 ± 0.00	0.033 ± 0.02	0.507 ± 0.22
	A2	0.003 ± 0.00	0.949 ± 0.00	0.005 ± 0.00	0.190 ± 0.08	2.175 ± 0.08
	A3	0.014± 0.00	0.946 ± 0.00	0.025 ± 0.00	0.138 ± 0.03	2.254 ± 0.01
	A4	0.011 ± 0.00	0.949 ± 0.00	0.019 ± 0.00	0.129 ± 0.04	2.137 ± 0.10

Values in bold and italic indicate the first and second best results, respectively

Table 7 Scenario 2—Performance indices of different methods for Synthetic dataset

	Precision	*Accuracy*	F-score	*AR*	*CI*
CBLOF	0.043 ± 0.01	0.946 ± 0.00	0.069 ± 0.02	0.232 ± 0.05	4.326 ± 0.16
CUBOID	**0.515** ± 0.06	**0.992** ± 0.00	**0.476** ± 0.05	**0.559** ± 0.06	**11.132** ± 1.30
First-order	0.307 ± 0.00	0.986 ± 0.00	0.309 ± 0.04	*0.447* ± 0.06	1.159 ± 0.02
iForest	0.042 ± 0.01	0.946 ± 0.00	0.068 ± 0.02	0.220 ± 0.05	4.288 ± 0.16
Interval-based	0.264 ± 0.05	0.985 ± 0.00	0.248 ± 0.04	0.338 ± 0.06	1.796 ± 0.04
LOF	0.035 ± 0.01	0.945 ± 0.00	0.056 ± 0.01	0.203 ± 0.04	1.346 ± 0.05
PAA	*0.426* ± 0.06	*0.990* ± 0.00	*0.375* ± 0.05	0.404 ± 0.06	*5.191* ± 0.46
SOS	0.007 ± 0.00	0.970 ± 0.01	0.010 ± 0.00	0.029 ± 0.01	1.007 ± 0.22

Values in bold and italic indicate the first and second best results, respectively

Table 8 Scenario 2—F-score indices for all datasets

	Yahoo				
	A1	A2	A3	A4	Synthetic
CBLOF	0.204 ± 0.05	0.094 ± 0.02	0.073 ± 0.01	0.060 ± 0.01	0.069 ± 0.02
CUBOID	0.282 ± 0.07	**0.221** ± 0.01	**0.127** ± 0.01	**0.122** ± 0.01	**0.476** ± 0.05
First-order	*0.461* ± 0.07	0.099 ± 0.01	0.084 ± 0.01	*0.079* ± 0.01	0.309 ± 0.04
iForest	0.223 ± 0.05	0.096 ± 0.02	0.076 ± 0.01	0.063 ± 0.01	0.068 ± 0.02
Interval-based	0.450 ± 0.08	0.051 ± 0.01	0.074 ± 0.01	0.061 ± 0.01	0.248 ± 0.04
LOF	0.157 ± 0.03	0.094 ± 0.02	0.065 ± 0.01	0.056 ± 0.01	0.056 ± 0.01
PAA	**0.505** ± 0.08	*0.161* ± 0.02	*0.086* ± 0.01	0.076 ± 0.01	*0.375* ± 0.05
SOS	0.007 ± 0.00	0.005 ± 0.00	0.025 ± 0.00	0.019 ± 0.00	0.010 ± 0.00

Values in bold and italic indicate the first and second best results, respectively

Data availibility Yahoo S5 datasets analyzed during the current study are available at https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70, and the Synthetic datasets generated by https://github.com/KDD-OpenSource/agots repository. The Sin dataset is also generated by Eq. (8).

References

Akhmedova S, Stanovov V, Kamiya Y (2022) A hybrid clustering approach based on fuzzy logic and evolutionary computation for anomaly detection. Algorithms 15(10):342

Aljawarneh SA, Vangipuram R (2020) GARUDA: Gaussian dissimilarity measure for feature Representation and anomaly Detection in internet of things. J Supercomput 76(6):4376–4413

Arumugam P, Saranya R (2018) Outlier detection and missing value in seasonal ARIMA model using rainfall data. Mater Today Proc 5(1):1791–1799

Azzaoui H, Boukhamla AZE, Arroyo D, Bensayah A (2022) Developing new deep-learning model to enhance network intrusion classification. Evol Syst 13(1):17–25

Blázquez-García A, Conde A, Mori U, Lozano JA (2021) A review on outlier/anomaly detection in time series data. ACM Comput Surv (CSUR) 54(3):1–33

Bountrogiannis K, Tzagkarakis G, Tsakalides P (2021) Anomaly detection for symbolic time series representations of reduced dimensionality. In: 28th European signal processing conference (EUSIPCO), pp 2398–2402

Breunig MM, Kriegel HP, Ng RT, Sander J (2000) LOF: identifying density-based local outliers. In: Proceedings of the ACM SIGMOD international conference on management of data, pp 93–104

Carmona-Poyato Á, Fernández-García NL, Madrid-Cuevas FJ, Durán-Rosal AM (2020) A new approach for optimal time-series segmentation. Pattern Recogn Lett 135:153–159

Chadha GS, Islam I, Schwung A, Ding SX (2021) Deep convolutional clustering-based time series anomaly detection. Sensors 21(16):5488

Cheng X, Wang Z, Yang X, Xu L, Liu Y (2021) Multi-scale detection and interpretation of spatio-temporal anomalies of human activities represented by time-series. Comput Environ Urban Syst 88:101627

Choi H-C, Deng C, Park H, Hwang I (2023) Gaussian Mixture Model-Based online anomaly detection for vectored area navigation arrivals. J Aerosp Inf Syst 20(1):37–52

Cook AA, Mısırlı G, Fan Z (2019) Anomaly detection for IoT time-series data: a survey. IEEE Internet Things J 7(7):6481–6494

Fernandes M, Canito A, Corchado JM, Marreiros G (2019) Fault detection mechanism of a predictive maintenance system based on Autoregressive Integrated Moving Average models. In: Distributed computing and artificial intelligence, 16th international conference, pp 171–180

Figueroa K, Paredes R, Reyes N (2018) New permutation is similarity measures for proximity searching. In: International conference on similarity search and applications, pp 122–133

Fox AJ (1972) Outliers in time series. J R Stat Soc Ser B (Methodol) 34(3):350–363

Geiger A, Liu D, Alnegheimish S, Cuesta-Infante A, Veeramachaneni K (2020) Tadgan: Time series anomaly detection using generative adversarial networks. In: IEEE international conference on big data (Big Data), pp 33–43

Ghalyan IF, Ghalyan NF, Ray A (2021) Optimal window-symbolic time series analysis for pattern classification and anomaly detection. IEEE Trans Industr Inf 18(4):2614–2621

Hagemann T, Katsarou K (2020) Reconstruction-based anomaly detection for the cloud: a comparison on the Yahoo! Webscope S5 dataset. In: Proceedings of the 4th international conference on cloud and big data computing, pp 68–75

He Z, Xu X, Deng S (2003) Discovering cluster-based local outliers. Pattern Recogn Lett 24(9–10):1641–1650

Huang K, Wu Y, Wen H, Liu Y, Yang C, Gui W (2020) Distributed dictionary learning for high-dimensional process monitoring. Control Eng Pract 98:104386

Hundman K, Constantinou V, Laporte C, Colwell I, Soderstrom T (2018) Detecting spacecraft anomalies using LSTM and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp 387–395

Janssens J, Huszár F, Postma E, van den Herik H (2012) Stochastic outlier selection. Tilburg centre for Creative Computing, techreport 2012-001

Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Dimensionality reduction for fast similarity search in large time series databases. Knowl Inf Syst 3:263–286

Li J, Izakian H, Pedrycz W, Jamal I (2021) Clustering-based anomaly detection in multivariate time series data. Appl Soft Comput 100:106919

Liang H, Song L, Wang J, Guo L, Li X, Liang J (2021) Robust unsupervised anomaly detection via multi-time scale DCGANs with forgetting mechanism for industrial multivariate time series. Neurocomputing 423:444–462

Lin CR, Chen MS (2002) On the optimal clustering of sequential data. In: Proceedings of the SIAM international conference on data mining, pp 141–157

Lindemann B, Maschler B, Sahlab N, Weyrich M (2021) A survey on anomaly detection for technical systems using LSTM networks. Comput Ind 131:103498

Liu FT, Ting KM, Zhou ZH (2008) Isolation forest. In: Eighth IEEE international conference on data mining, pp 413–422

Liu Y, Garg S, Nie J, Zhang Y, Xiong Z, Kang J, Hossain MS (2020) Deep anomaly detection for time-series data in industrial IoT: a communication-efficient on-device federated learning approach. IEEE Internet Things J 8(8):6348–6358

Maciąg PS, Kryszkiewicz M, Bembenik R, Lobo JL, Del Ser J (2021) Unsupervised anomaly detection in stream data with online evolving spiking neural networks. Neural Netw 139:118–139

Mahmoodi K, Ketabdari MJ, Vaghefi M (2021) Proposing a new local density estimation outlier detection algorithm: an empirical case study on flow pattern experiments. Pattern Anal Appl 24:1859–1872

Munir M, Siddiqui SA, Dengel A, Ahmed S (2018) DeepAnT: a deep learning approach for unsupervised anomaly detection in time series. IEEE Access 7:1991–2005

Pérez D, Alonso S, Morán A, Prada MA, Fuertes JJ, Domínguez M (2021) Evaluation of feature learning for anomaly detection in network traffic. Evol Syst 12(1):79–90

Pham V, Nguyen N, Li J, Hass J, Chen Y, Dang T (2019) MTSAD: multivariate time series abnormality detection and visualization. In: 2019 IEEE international conference on big data (Big Data), pp 3267–3276

Pramitarini Y, Perdana RHY, Tran T-N, Shim K, An B (2022) A hybrid price auction-based secure routing protocol using advanced speed and cosine similarity-based clustering against sinkhole attack in VANETs. Sensors 22(15):5811

Ramotsoela DT, Hancke GP, Abu-Mahfouz AM (2019) Attack detection in water distribution systems using machine learning. HCIS 9(1):1–22

Reddy A, Ordway-West M, Lee M, Dugan M, Whitney J, Kahana R, Ford B, Muedsam J, Henslee A, Rao M (2017) Using Gaussian Mixture Models to detect outliers in seasonal univariate network traffic. In: IEEE security and privacy workshops (SPW). IEEE, San Jose, CA, USA, pp 229–234

Ren H, Liu M, Li Z, Pedrycz W (2017) A Piecewise Aggregate pattern representation Approach for anomaly detection in time series. Knowl-Based Syst 135:29–39

Ren H, Li X, Li Z, Pedrycz W (2018) Data representation based on interval-sets for anomaly detection in time series. IEEE Access 6:27473–27479

Ren H, Xu B, Wang Y, Yi C, Huang C, Kou X, Xing T, Yang M, Tong J, Zhang Q (2019) Time-series anomaly detection service at Microsoft. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp 3009–3017

Sim KH, Sim KY, Bong N (2018) Dynamic time interval data representation in scalable financial time series pattern recognition. In: ACM international conference proceeding series, pp 120–125

Singh K, Upadhyaya S (2012) Outlier detection: applications and techniques. Int J Comput Sci Issues (IJCSI) 9(1):307

Steland A, Rafajłowicz E, Szajowski K (2015) Stochastic models. Statistics and their applications. Springer, Wrocław

Tran L, Mun MY, Shahabi C (2020) Real-time distance-based outlier detection in data streams. Proc VLDB Endowm 14(2):141–153

Tukey JW (1977) Exploratory data analysis. Addison-Wesley, Reading

Wahid A, Rao ACS (2019) A distance-based outlier detection using particle swarm optimization technique. In: Information and communication technology for competitive strategies: proceedings of third international conference on ICTCS, pp 633–643

Wang Z, Fan Y (2022) Density-based structure preserving projections process monitoring model for fused magnesia smelting process. In: IEEE transactions on industrial informatics, pp 1–12

Wang D, Liu H, Pedrycz W, Song W, Li H (2022) Design Gaussian information granule based on the principle of justifiable granularity: a multi-dimensional perspective. Expert Syst Appl 197:116763

Wang Z, Wang Y, Gao C, Wang F, Lin T, Chen Y (2022) An adaptive sliding window for anomaly detection of time series in wireless sensor networks. Wirel Netw:1–19

Yang Y, Chen L, Fan C (2021) ELOF: fast and memory-efficient anomaly detection algorithm in data streams. Soft Comput 25(6):4283–4294

Yazdi SV, Douzal-Chouakria A (2018) Time warp invariant kSVD: sparse coding and dictionary learning for time series under time warp. Pattern Recogn Lett 112:1–8

Yu M, Sun S (2020) Policy-based reinforcement learning for time series anomaly detection. Eng Appl Artif Intell 95:103919

Zhang C, Zuo W, Yin A, Wang X, Liu C (2021) ADET: Anomaly DEtection in time series with linear Time. Int J Mach Learn Cybern 12(1):271–280

Zhang W, Lin Z, Liu X (2022) Short-term offshore wind power forecasting-a hybrid model based on Discrete Wavelet Transform (DWT), Seasonal Autoregressive Integrated Moving Average (SARIMA), and deep-learning-based Long Short-Term Memory (LSTM). Renew Energy 185:611–628

Zhou ZG, Tang P (2016) Improving time series anomaly detection based on Exponentially Weighted Moving Average (EWMA) of season-trend model residuals. In: IEEE international geoscience and remote sensing symposium (IGARSS), pp 3414–3417

Zhou Y, Ren H, Li Z, Pedrycz W (2021) An anomaly detection framework for time series data: an interval-based approach. Knowl-Based Syst 288:107153

Zhou Y, Ren H, Li Z, Wu N, Al-Ahmari AM (2021) Anomaly detection via a combination model in time series data. Appl Intell 51(7):4874–4887

Zhu X, Pedrycz W, Li Z (2016) Granular encoders and decoders: a study in processing information granules. IEEE Trans Fuzzy Syst 25(5):1115–1126