



# Anomaly and change point detection for time series with concept drift

Jiayi Liu<sup>1</sup> · Donghua Yang<sup>1</sup> · Kaiqi Zhang<sup>1</sup> · Hong Gao<sup>2</sup> · Jianzhong Li<sup>3</sup>

Received: 25 March 2023 / Revised: 6 May 2023 / Accepted: 15 May 2023 /

Published online: 7 July 2023

© The Author(s) 2023

## Abstract

Anomaly detection is one of the most important research contents in time series data analysis, which is widely used in many fields. In real world, the environment is usually dynamically changing, and the distribution of data changes over time, namely concept drift. The accuracy of static anomaly detection methods is bound to be reduced by concept drift. In addition, there is a sudden concept drift, which is manifested as a abrupt variation in a data point that changes the statistical properties of data. Such a point is called a change point, and it has very similar behavior to an anomaly. However, the existing methods cannot distinguish between anomaly and change point, so the existence of change point will affect the result of anomaly detection. In this paper, we propose an unsupervised method to simultaneously detect anomaly and change point for time series with concept drift. The method is based on the fluctuation features of data and converts the original data into the rate of change of data. It not only solves the concept drift, but also effectively detects and distinguishes anomalies and change points. Experiments on both public and synthetic datasets show that compared with the state-of-the-art anomaly detection methods, our method is superior to most of the existing works and significantly superior to existing methods for change point detection. It fully demonstrates the superiority of our method in detecting anomalies and change points simultaneously.

---

✉ Jiayi Liu  
jiayiliu@stu.hit.edu.cn

Donghua Yang  
yang.dh@hit.edu.cn

Kaiqi Zhang  
zhangkaiqi@hit.edu.cn

Hong Gao  
honggao@zjnu.edu.cn

Jianzhong Li  
lijzh@siat.ac.cn

<sup>1</sup> School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150006, China

<sup>2</sup> School of Computer Science and Technology, Zhejiang Normal University, Jinhua 321004, China

<sup>3</sup> Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

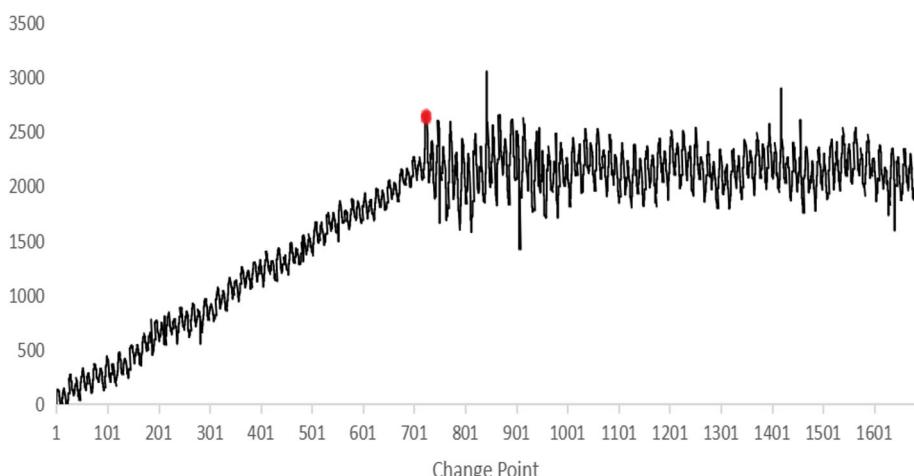
**Keywords** Anomaly detection · Change point · Concept drift · Time series

## 1 Introduction

Nowdays, time series data has been widely produced in financial transactions, Internet applications, system operation and maintenance, industrial equipment and other fields. Therefore, the analysis and research of time series has been a hot spot. Anomaly detection is one of the most important applications. Its importance is reflected in that data anomalies usually reveal important information, according to which decision-makers can take corresponding measures [1]. For example, the IP address of a user logging in to a social network has changed compared with the previous one, which may imply that the account is at risk of theft. If the abnormal account can be accurately detected, appropriate measures can be taken to protect the account security, so that the risk can be avoided effectively. Whether anomaly detection is applied to intrusion detection, fault detection or medical monitoring, the purpose is to ensure the reliability and security of the system, so the accuracy of time series anomaly detection is particularly important.

Usually, we think of anomalies as points that do not obey the distribution. However, in reality, the time series data distribution will change significantly with the passage of time, that is, concept drift. The concept drift will affect the accuracy of anomaly detection [2, 3]. The impact is reflected in two aspects. First, since the static anomaly detection algorithm is not self-adaptive, it may no longer be accurate when applied to the new data distribution. Second, as shown in Figure 1, the red marked point is called change point, at which concept drift happens. Due to the sudden rise of the data at this point and the change of the data distribution before and after this point, the existing detection methods may treat the data at change point and beyond as anomaly output. Thus, it can be seen that the existence of change point will affect the detection results, and it needs to be distinguished from anomalies. To sum up, we believe that it is important to study anomaly detection for time series with concept drift.

There are several challenges to be addressed:



**Figure 1** Change point of Yahoo dataset

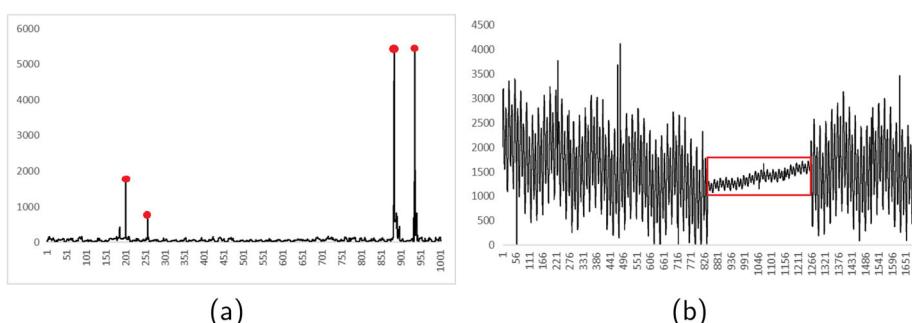
**Challenge 1: Distinguish between anomalies and change points.** The earliest and widely used definition of anomaly was given by Hawkins (1980): “An observation result is so different from other observations that people suspect that it is generated by a different mechanism”. Therefore, if a data point does not follow the expected results or has significant differences with historical observations, it can be considered as anomaly [4]. However, change point also conforms to the above behavior. Since the statistical distribution of the time series will change with time, the concept drift will make the statistical attributes of the data before and after change point become different. There are significant differences [5–7]. For example, wearable devices record human activities, when people run for exercise, the speed record in the device will increase significantly [6, 8]. When more software services are deployed on the server, the PV of each server will decrease significantly [3]. Obviously, this is expected normal behavior. It can be seen that it is difficult to distinguish between anomaly and change point. However, in the actual data, the two kinds of data exist at the same time, so the existence of change point in the time series will reduce the quality of anomaly detection.

**Challenge 2: Generalization.** In different scenarios, there are different types of time series, which may be stationary, non-stationary, periodic, seasonal and so on. Generally, we expect the detection algorithm to perform well in different types. However, the existing methods are not general enough and are not always effective in the face of different types of time series [9]. Therefore, it is difficult to find a method with good generality.

**Challenge 3: Lack of labels.** Since the number of anomalies is small, the cost of manually labeling data is large, and the new anomalies generated over time are unknowable. It is impossible to use the supervision model to learn the feature of all abnormal data.

The existing methods usually do not distinguish between anomalies and change points, and the detection of them is carried out separately, which fundamentally affects the performance of the detection methods [25]. Therefore, to tackle the aforementioned problems, our goal is to design an unsupervised anomaly detection method with generalization to detect both anomalies and change points. Based on the analysis of existing methods and observation of real time series, we have the following two interesting observations. To begin with, we find that when an anomaly occurs, it is usually a large local fluctuation, showing a sudden rise or fall in data, or the amplitude of data fluctuation is obviously inconsistent with the neighboring data, as shown in Figure 2. Both of these can be summarized as the local rate of change of data has changed significantly. Second of all, we note that the data distribution changes caused by change point are permanent changes [5].

Inspired by the above observations, we have such an intuitive idea: if we can extract appropriate features to represent the rate of change of data, and make the features locally



**Figure 2** The anomaly types from Yahoo

as extreme as possible, then the problem of concept drift can be well solved. Therefore, in this work, we propose an unsupervised method (algorithm name), which can simultaneously detect anomalies and change points for time series with concept drift. Specifically, we use the absolute second derivative to extract the features of the rate of change of the time series, and enlarge the features locally to eliminate the impact of data drift and obtain candidate anomaly points. Then we can effectively distinguish between anomalies and change points according to the range of data distribution changes. Finally, we propose a volatility similarity to eliminate the impact of inherent patterns. In addition, when extending to multivariate detection, the influence of dependencies between attributes should be taken into account. We use DTW(Dynamic Time Warping) to capture the correlation of fluctuations between attributes, and determine anomalies according to the joint decision of multi-attributes with dependencies.

The contributions of this paper are highlighted as below:

(1) To the best of our knowledge, this paper is the first work to study the problem of detecting anomalies and change points simultaneously for time series with concept drift. To this end, we propose an unsupervised detection algorithm based on fluctuation features, and also distinguish the types of anomalies (point anomalies and collective anomalies), which makes the detection results more explanatory.

(2) We take the fluctuation features as the breakthrough point and use the rate of change of data to characterize the degree of abnormality, which can not only deal with the concept drift well, but also reduce the impact of asynchrony on multivariate time series. Based on the fluctuation direction of data, the fluctuation similarity is proposed to eliminate the influence of inherent patterns such as periodicity.

(3) From the perspective of experiments, we selected different types of datasets to prove that the proposed method has good generality and high effectiveness.

The remaining of the paper is organized as follows. Section 2 discusses the related work. The related definitions and the problem analysis are presented in Section 3. The methodology is introduced in Section 4. Experimental results are analyzed in Section 5. Finally, we conclude our work and put forward future work in Section 6.

## 2 Related work

Anomaly detection has been extensively studied in different research and application fields. Time series anomaly detection can be basically divided into three categories: statistical methods, supervised and unsupervised. The statistics-based detection method requires a large amount of statistics on data, which depends heavily on prior knowledge [1, 5]. Supervised, on the other hand, rely on labels to learn to distinguish anomalies [10, 11]. Therefore, unsupervised time series anomaly detection has been widely studied in recent years. The commonly used unsupervised methods can be roughly categorized into: distance-based, density-based, cluster-based, tree-based, and reconstruction-based methods.

The distance based method detects anomalies by measuring the distance between the current observed data object and its nearest neighbor, and defines the distance between it and its k-th nearest neighbor as the anomaly score [12, 13]. Local Outlier Factor (LOF) [14] is a representative algorithm in density based anomaly detection methods. It is similar to k-nearest neighbor, but the difference is that it is measured by local density deviation relative to its neighbor rather than distance. By extending LOF [15], density based methods such as connectivity based outlier factor (COF), INFLO (influenced outliers), and local outlier

probabilities (LOOP) are proposed. In clustering based methods, such as SVDD [16] and Deep SVDD [17], normal data is treated as a single class to distinguish between normal and abnormal data, so as to detect abnormal values. The author in [18, 19] proposed a tree based method to isolate anomalies as quickly as possible by continuously dividing the data space.

In recent years, the use of deep learning model for anomaly detection has a good performance, especially the model based on reconstruction has attracted more attention. Common models are based on AE, GAN and VAE. The main idea of these three models is to detect anomalies through reconstruction errors. The author [20] proposed a mixed Gaussian model DAGMM based on AE, which optimizes the parameters of AE and mixed model simultaneously in a peer-to-peer manner to reduce reconstruction error. Inspired by GAN, USAD [21] magnified the reconstruction error containing abnormal input by conducting confrontational training on two AEs to make the model more stable. MADGAN [22] uses LSTM as the basic model to capture temporal dependencies and embed them into the GAN framework. OmniAnomaly [23] integrates VAE and GRU together, and uses random variable connection and plane standardization flow, learns the robustness of data to characterize the normal mode of further capturing data. InterFusion [24] models the normal pattern in the data through the hierarchical variable Auto Encoder with two random potential variables, the intra-metric dependency of the simulation sequence and the inter metric dependency between multiple sequences, and finally performs anomaly detection according to the reconstruction information.

However, in reality, the distribution of data will change over time, resulting in concept drift. Since the static detection methods mentioned above are not self-adaptive, its accuracy will inevitably decrease over time.

Some novel methods have been proposed to cope with changing data. In [2], authors first proposed the anomaly detection algorithms SPOT and DSPOT based on the extreme value theory, which do not rely on manually set thresholds and assumed data distribution. However, this method is only sensitive to extreme values and has limited adaptability to concept drift. The authors of [3] proposed a framework, StepWise, which can help any type of anomaly detection algorithm quickly adapt to concept drift. However, this method can only deal with the situation that the data trend is almost the same before and after the concept drift. Microsoft first applied the SR model from the technology of visual saliency detection to time series anomaly detection, and proposed the SR-CNN model based on the Specific Residential and Convolutional Neural Network [9]. This method emphasizes universality and can handle time series with different types, such as seasonal, stable and unstable. However, this method requires a small amount of manual annotation, and its accuracy is not high enough.

Many unsupervised change point detection methods have also been proposed. Utilizing likelihood ratio based on cumulative observations and statistics, subspace model methods such as SST, probabilistic methods, kernel-based methods, and cluster-based methods [7]. Although change point detection is similar to anomaly detection, it is different in essence. Change point detection only focuses on the points that cause the change of data distribution, while the points that cause anomalies do not necessarily cause changes in the data distribution.

Le et al. [25] proposed an anomaly detection algorithm based on invert nearest neighbor in 2020. This algorithm can well distinguish between anomalies and change points, but it is only for univariate time series, and can only obtain high accuracy when active learning is used.

The disadvantages of existing methods motivate us to propose a solution that can not only distinguish anomalies and change points, but also solve concept drift.

### 3 Preliminaries

In this section, we first present related concepts and problem definitions. Then, we analyze the feature of anomaly and find the breakthrough point of the problem, i.e. fluctuation features of the data.

#### 3.1 Problem statement

**Time Series:** A time series is a sequence of data points indexed in time order, defined as  $X = \{x_1, x_2, \dots, x_n\}$  where  $x_i$  is the data recorded at time  $t_i$ .

**Type of Anomaly:** Anomalies can be generally classified into the following two types:

(1) Point Anomalies: If a single data point is significantly different with respect to the rest of data in a specific range, such data is termed a point anomaly, as shown in Figure 2a.

(2) Collective Anomalies: A continuous of data points having a huge deviation from other data in a certain period of time are called collective anomalies, which is shown in Figure 2b.

Understanding anomaly types is helpful to analyze anomalies and make better decisions [5, 26].

**Concept drift:** Concept drift is a phenomenon that the data distribution changes over time in dynamically changing and non-stationary environments [32].

**Change point:** Change point is a point of abrupt variation in time series data, which changes the statistical properties of data [7, 25]. This is a sudden concept drift [3, 5] that causes the data before and after it to follow two different distributions. Let  $\{x_1, x_2, \dots, x_i, \dots, x_n\}$  be a sequence of time series. If  $x_i$  is a change point, the time series will be divided into two disjoint segments  $\{x_1, x_2, \dots, x_{i-1}\}, \{x_i, \dots, x_n\}$  with different statistical properties.

**Problem Definition:** Given a time series  $X = \{x_1, x_2, \dots, x_n\}$ , we aim to assigning a label to each data point,  $Y = \{y_1, y_2, \dots, y_n\}$ ,  $y_i \in \{0, 1, 2, 3\}$ , where “0” indicates normal, “1” indicates pa (abbreviation of point anomaly), “2” indicates ca (abbreviation of collective anomaly), “3” indicates cp (abbreviation of change point).

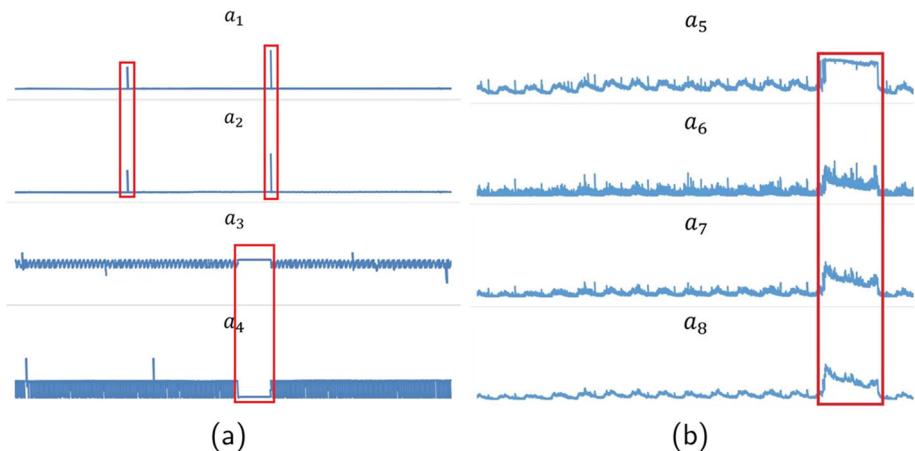
#### 3.2 Motivations

By the observation of the real data, we have some insights that can help solve the difficulties of this paper.

As aforementioned that the anomaly data must be significantly different from the surrounding data, which means that there are relatively large local fluctuation of data. This fluctuation can take two forms: In the first form, in terms of value of the data, the value changes significantly compared with the surrounding data, that is, it increases or decreases significantly. In the second form, unlike the former, the value of the data may not change much, but the amplitude of data variation in a certain period of time is significantly different from others. For example, in the Figure 3a,  $a_1$  and  $a_2$  show the first form,  $a_3$  and  $a_4$  show the second form. We find that the two forms of local fluctuations can be summarized as the rate of change of data has changed significantly.

However, in some cases, these fluctuations do not necessarily anomalies. Time series may have periodic, seasonal patterns, etc., and the aforementioned fluctuations may occur frequently at different times, so this is also a normal inherent pattern.

In addition, for multivariate time series, we should also consider the linear or nonlinear relationship between different variables, namely attribute correlation. We have noticed that when an anomaly occurs, there is often a cascade effect between attributes with strong



**Figure 3** Illustration of different forms of fluctuation

correlation which is shown in Figure 3b. This is easy to understand in real world. For example, the environmental monitoring system will judge the potential fire hazard by combining the properties of temperature, humidity, and carbon monoxide concentration. When only one attribute is abnormal, the system may think that the probability of fire is very low, but if several related attributes are abnormal in succession within a period of time, the system will think that the probability of fire is very high and make a fire alarm.

The characteristic of change point is that the data after it obeys the new distribution. Therefore, the key to distinguish between anomalies and change points is that change points change the distribution of data while anomalies do not.

Based on the above analysis, the following issues need to be considered: (1) How to characterize the rate of change of data? (2) It is necessary to determine whether fluctuations occur frequently to eliminate the influence of inherent patterns. It is also necessary to consider attribute correlation to eliminate the impact of noise in multivariate time series. (3) How to decide whether the distribution of data has changed?

## 4 Methodology

In this section, we take univariate time series as an example to briefly outline the overall detection algorithm, and then describe the process of each step in detail. Finally, how to extend the algorithm to multivariable time series will be explained. The main symbols are shown in Table 1.

### 4.1 Overview

Our algorithm focuses on simultaneous detection of anomalies and change points in time series. The main steps of the overall algorithm are as follows.

(1) Select Candidate: In line 3, it is used to filter out normal points and find all possible anomalies and change points. Generate potential candidate points at local extreme points according to the rate of change of data.

**Table 1** Symbols

Symbol	Description
$X$	A time series
$Y$	Detection result series
$\Delta''x_i$	The absolute second derivative of $x_i$
$md(S)$	The maximum deviation is the maximum absolute value of the difference between data set S and its mean value
$FDC(X)$	The sequence fluctuation direction of X
$FDS(X_1, X_2)$	The fluctuation direction similarity of $X_1$ and $X_2$
$FS(X_1, X_2)$	The fluctuation similarity of $X_1$ and $X_2$
$\tau$	Maximum delay tolerance time

**Algorithm 1** ACPD.

---

```

1: Input: Time series  $X = [x_1, x_2, \dots, x_n]$ .
2: Output: Detection result  $Y = [y_1, y_2, \dots, y_n]$ .
3:  $Z = \text{Candidate}(X)$ 
4: for  $x_i \in Z$  do
5:    $y_i = \text{Classification}(x_i, X)$ 
6: end for
7:  $Y = \text{InherentPatterns}(Y, X)$ 
8: Return  $Y$ 

```

---

(2) Classification: In line 4-6, candidate points are identified as point anomalies, collective anomalies or change points. A candidate is classified by using the characteristics of different types in the rate of change and the statistical information obtained by sampling the data before the candidate.

(3) Inherent Patterns: In lines 7-8, eliminate the influence of inherent patterns. By calculating the fluctuation similarity between each candidate, find the frequent fluctuation and get the inherent patterns. For the candidate point belonging to the inherent patterns should be considered normal point.

To extend the algorithm to multivariate time series, the relationship between attributes should also be considered, we will discuss this in Section 4.5.

In the following sections, by analyzing the corresponding algorithms for each step, it can be seen that the time complexity of Algorithm 1 is  $O(n)$ .

## 4.2 Select candidate

Our goal is to identify anomalies and change points. Therefore, compared with the original data, we are more focus on the rate of change of data for it can better describe data fluctuations and reflect data anomalies.

The rate of change of data can be calculated by the second derivative. According to the definition of the second derivative, we can use the second difference approximation. More accurately, we extract the features of data change rate based on absolute second derivative, which can be used to identify the critical points [25].

**Definition 1** (absolute second derivative) The absolute second derivative of  $x_i$  is denoted as

$$\Delta''x_i = |\Delta x_i - \Delta''x_{i-1}|, \quad (1)$$

where

$$\Delta x_i = |x_i - x_{i-1}|, i = 1, 2, \dots, n. \quad (2)$$

Formally, given a time series  $X = \{x_1, x_2, \dots, x_n\}$ , the absolute second derivative of  $X$  is denoted as  $\Delta''X = \{\Delta''x_1, \Delta''x_2, \dots, \Delta''x_{n-1}\}$ .

In order to avoid the impact of concept drift, we locally amplify the fluctuation characteristics of the absolute second derivative obtained from data conversion. In general, time series is kept in a relatively stable state for a period of time. If the fluctuation can be amplified locally as much as possible, even though there is concept drift, anomalies can be well captured. Specifically, the absolute second derivative sequence is split into  $m$  subsequences of equal length, which is denoted as  $\text{seg}_j = \Delta''x_1, \dots, \Delta''x_{i+n/m-1} \subseteq \Delta''X, 1 \leq j \leq m$ . The maximum deviation of each subsequence is calculated to represent the local maximum fluctuation amplitude.

**Definition 2** (maximum deviation) Given a set of data  $S = \{s_1, s_2, \dots, s_i, \dots\}$ , its maximum deviation is  $md(S)$ , defined as

$$md(S) = \max|s_i - \text{average}(S)|. \quad (3)$$

To identify candidate points, we first identify candidate segments. We use the absolute mean deviation (MAD), which is a robust measure of the variability of data. It is more suitable for the anomalies in the data set than the standard deviation. A small number of anomalies will not affect the experimental results [12].

**Definition 3** (MAD) Given a set of data  $S = \{s_1, s_2, \dots, s_i, \dots\}$ , MAD is the median of the absolute value of deviation, defined as

$$MAD(S) = \text{median}|s_i - \text{median}(S)|. \quad (4)$$

If the MAD of the maximum deviation of a subsequence is higher than the MAD of the maximum deviation of all subsequences, it is considered as a candidate subsequence. After finding a candidate subsequence, candidate points are further searched in this sequence.

To identify candidate points, extreme value theory EVT is applied to candidate subsequences. Because the data change rate described by the absolute second derivative will highlight the positions with large fluctuations. EVT is very powerful for peak detection and does not need to manually set the threshold [3].

Extreme value theory [2] refers to the distribution of extreme events that we may observe by inferring without any distribution assumption based on the original data, which is the extreme value distribution (EVD). Its mathematical expression is as follows:

$$G_\gamma : x \mapsto \exp(-(1 + \gamma x)^{-\frac{1}{\gamma}}), \gamma \in \mathbb{R}, 1 + \gamma x > 1. \quad (5)$$

The  $\gamma$  is the extreme value index, which depends on the extreme value data of different distributions. In order to fit the EVD to the tail of the unknown input distribution, it is necessary to estimate  $\gamma$ . But it is difficult to calculate it effectively. To solve this problem, the Peaks-Over-Threshold (POT) approach is proposed [2]. This method depends on the Pickands-Balkema-de Haan theorem [27, 28] (also called second theorem in EVT) given below.

$$\bar{F}_t(x) = P(X - t > x | X > t) \sim (1 + \frac{\gamma x}{\sigma})^{-\frac{1}{\gamma}} \quad (6)$$

The result shows that the distribution of threshold exceeding part ( $X - t$ ) meets the requirements of Generalized Pareto Distribution(GPD) with parameters  $\gamma$ ,  $\sigma$ . That is, the tail distribution can be fitted by the generalized Pareto distribution(GPD). From this, we can get the threshold value of the overall distribution, and the calculation formula is as follows:

$$z_q \simeq t + \frac{\hat{\sigma}}{\hat{\gamma}} \left( \left( \frac{qn}{N_t} \right)^{-\hat{\gamma}} - 1 \right) \quad (7)$$

where  $t$  is the initial threshold and the empirical value is 98%;  $\sigma$  and  $\gamma$  are parameters of GPD, which are obtained by maximum likelihood;  $q$  is the expected probability;  $N$  is the total number of current samples;  $N_t$  is the peak number, i.e.  $X_i > t$ . The POT provides us with a method to estimate  $z_q$ . The specific algorithm is as follows:

---

**Algorithm 2** POT(Peaks-over-threshold).

---

- 1: **Input:** Time series  $X = [x_1, x_2, \dots, x_n]$ , risk  $q$
  - 2: **Output:**  $z_q$
  - 3:  $t = \text{SetInitialThreshold}(x_1, \dots, x_n)$
  - 4:  $Y_t = \{x_i - t | x_i > t\}$
  - 5:  $\hat{\gamma}, \hat{\sigma} = \text{Grimshaw}(Y_t)$
  - 6:  $z_q = \text{CalcThreshold}(q, \hat{\gamma}, \hat{\sigma}, n, N_t, t)$
  - 7: **Return**  $z_q$
- 

After obtaining  $z_q$  by Algorithm 2, we can find candidate points from candidate subsequences. We summarize the whole method of selecting candidate points as Algorithm 3.

Algorithm 3 illustrates the entire process of identifying candidate points. Firstly, the input time series is preprocessed to convert the original data into absolute second derivatives (Line 4). The converted data is segmented to locally amplify the fluctuation characteristics of the data (Lines 5-6). Find subsequences that may have candidate points based on the amplifying features, and filter out subsequences that do not have anomalies to optimize performance (Lines 7-9). Finally, the candidate subsequences are further detected, using extreme value theory to obtain thresholds (Line 10) and find candidate points (Lines 11-13). The highest time complexity part of Algorithm 3 is calculating  $\Delta''X$  and the time complexity of this part is  $O(n)$ . Therefore, the time complexity of Algorithm 3 is  $O(n)$ .

### 4.3 Classification

Since an anomaly does not change the data distribution, if an anomaly occurs in the time series, the time series data must continue to follow the same data distribution as before the anomaly occurred for a certain period of time. Otherwise, the data distribution is considered to have changed and this is a change point.

In this case, we select data before the candidate points in a period of time and calculate their statistical characteristics to represent the data distribution.

We believe that the data is more likely to have a close neighbor relationship with the data before the anomaly occurred. Therefore, select the data before the candidate points within a certain period of time and calculate their statistical characteristics to represent the data distribution.

**Algorithm 3** Candidate.

---

```

1: Input: Time series  $X = [x_1, x_2, \dots, x_n]$ ,
2: Output: Candidate result  $Y = [y_1, y_2, \dots, y_n]$ 
3: Initialize segment number  $m$ , risk  $q$ 
4: Get  $\Delta''X$  according to Definition 1
5: Divide  $\Delta''X$  into  $m$  subsequence  $seg_1, seg_2, \dots, seg_m$ 
6: Calculate the  $md(seg_j)$  of each subsequence
7:  $Value_{mad} = MAD(md(seg_1), md(seg_2), \dots, md(seg_m))$ 
8: for 1 to  $m$  do
9:   if MAD of  $md(seg_j) > Value_{mad}$  then
10:     $q_z = POT(seg_j, q)$ 
11:    for  $\Delta''x_i \in seg_j$  do
12:      if  $\Delta''x_i > q_z$  then
13:         $y_i = 1$ 
14:      end if
15:    end for
16:   end if
17: end for
18: Return  $Y$ 

```

---

For candidate point  $x_i$ , select  $l$  data in front of it to form  $\hat{X} = \{x_{i-l}, \dots, x_{i-1}\}$ . For  $x_{i-1}, x_j \in \hat{X}$ , find points satisfying the following conditions :

$$x_j \in NNr(x_{i-1}) \text{ iff } x_j \in RNNr(x_{i-1}), (1 \leq r \leq l) \quad (8)$$

where  $NNr(x_{i-1})$  represents the r-nearest neighbor of  $x_{i-1}$  and  $RNNr(x_{i-1})$  represents the reverse r-nearest neighbor of  $x_{i-1}$ .

Calculate the mean value  $\mu$  and variance  $\delta$  of  $x_{i-1}$  and all  $x_j$  satisfying the conditions. We use  $\mu$  and  $\delta$  to represent the distribution characteristics before  $x_i$ .

The key of classification is that the distance for backward searching for data that follow the original data distribution is different after an anomaly or change point occurs. For example, after a point anomaly occurs, the data at its next moment should meet the data interval of the original data distribution. After the occurrence of collective anomalies, the data should be restored to the normal range of the original data distribution within a given time range, otherwise it is considered that the distribution of the data has changed over a long period of time. Algorithm 4 describes the specific details of classification.

In Algorithm 4, take the sudden increase in the value of data as an example. First, sample the data distribution information (Lines 5-17) and obtain the search threshold (Line 18). Then, the search length is calculated based on the threshold condition. The point anomalies and collective anomalies (Lines 21-35) are distinguished based on the search length. If the given length is exceeded, it is a change point (Lines 36-37). If the value decreases, line 18 will be changed to “-”, and the judgment condition of line 23 will be changed to “ $\geq$ ”. The time complexity of Algorithm 4 is  $O(n)$ .

#### 4.4 Inherent patterns

In reality, time series may have inherent patterns that occur frequently, such as periodicity and seasonality. However, the previously obtained detection results may mistake fluctuations with inherent patterns for anomalies. Therefore, in this section, we propose to mine inherent patterns to eliminate their impact on the detection results.

**Algorithm 4** Classification.

---

```

1: Input: Time series  $X = [x_1, x_2, \dots, x_n]$ ,  $Y = [y_1, y_2, \dots, y_n]$ ,
2: Output: Classification result  $Y = [y_1, y_2, \dots, y_n]$ 
3: Initialize length  $l, L$ 
4: for  $i \in [1, n]$  do
5:   if  $y_i \neq 0$  then
6:      $\hat{X} = \{x_{i-1-l}, \dots, x_{i-1}\}$ 
7:      $P = 0$ 
8:   end if
9:   for  $r \in [1, l]$  do
10:    for  $x_j \in \hat{X}$  do
11:      if  $x_j \in NNr(x_{i-1})$  and  $x_j \in RNNr(x_{i-1})$  then
12:         $sum = x_{i-1} + x_i$ 
13:         $P = P + 1$ 
14:      end if
15:    end for
16:   end for
17:   Get  $\mu$  and  $\delta$  according to  $sum$  and  $P$ 
18:    $q = \mu + \delta$ 
19: end for
20:  $t=1$ 
21: while  $t \leq L$  do
22:   flag=1
23:   if  $x_{i+t} \leq q$  then
24:     if  $t \geq 2$  then
25:        $y_i = 2$ 
26:       for  $k \in [1, t]$  do
27:          $y_{i+k} = 2$ 
28:       end for
29:       flag=0
30:     end if
31:      $t = L + 1$ 
32:   else
33:      $t++$ 
34:   end if
35: end while
36: if flag==1 then
37:    $y_i = 3$ 
38: end if
39: Return  $Y$ 

```

---

Due to the existence of concept drift, the fluctuations caused by the same event may not look so similar. It is not difficult to understand in real scenarios, such as traffic flow information. Usually, the data from 6 a.m. to 9 a.m. on weekdays fluctuates greatly, this is not anomalies but an inherent pattern. Now, in order to reduce the traffic congestion in the morning peak, the road will be reconstructed (such as widening the lane, repairing the viaduct, or adjusting the traffic signal) and the recorded traffic volume on each road will significantly decrease, resulting in a concept drift in the traffic volume data. The data fluctuation that originally appeared at 6-9 a.m. has changed to 7-8 a.m.. Although it is the same event, the time for fluctuations to occur has shortened and the range of fluctuations in numerical value has also decreased. Therefore, repeated fluctuations are not completely consistent but similar. To better compare the similarity between fluctuations, a method based on fluctuation similarity is proposed in this paper to detect inherent patterns.

Before giving the definition of fluctuation similarity, we first introduce the fluctuation direction and then give the definition of the sequence fluctuation direction. An increase

or decrease in values indicates upward and downward fluctuations in data, which is the fluctuation direction. In this paper, the binary is used to encode the fluctuation direction of data points, and the sequence fluctuation direction is defined as follows.

**Definition 4** (Sequence Fluctuation Direction) Given a time series  $X$ , encode the adjacent points  $x_i$  and  $x_{i+1}$  one by one according to the following rules to obtain the sequence fluctuation direction:

$$\begin{aligned} FDC(X) &= c_1, \dots, c_{n-1}, \\ c_i &= \begin{cases} 1 & \text{if } x_i \leq x_{i+1} \\ 0 & \text{else} \end{cases}. \end{aligned} \quad (9)$$

**Definition 5** (Fluctuation Direction Similarity) Given two time series  $X_1$  and  $X_2$ , the fluctuation direction similarity of  $X_1$  and  $X_2$  is calculated as follows,

$$FDS(X_1, X_2) = \frac{FDC(X_1) \odot FDC(X_2)}{n - 1}, \quad (10)$$

where  $\odot$  is the exclusive NOR,  $n$  is the length of  $X_1$  and  $X_2$ .

**Definition 6** (Fluctuation Similarity) Given two time series  $X_1$  and  $X_2$ , the fluctuation similarity of  $X_1$  and  $X_2$  is calculated as follows,

$$FS(X_1, X_2) = FDS(X_1, X_2) \times \text{Cos}(X_1, X_2), \quad (11)$$

where  $\text{Cos}(\cdot)$  is cosine similarity.

**Example 1:** Consider two time series  $X_1 = \{5, 6, 3, 8, 7, 7, 6\}$  and  $X_2 = \{10, 12, 6, 16, 14, 14, 15\}$ . The sequence fluctuation direction of  $X_1$  and  $X_2$  is  $FDC(X_1)=101010$ ,  $FDC(X_2)=101011$ , then  $FDS(X_1, X_2)=5/6$ ,  $\text{Cos}(X_1, X_2)=0.997$ ,  $FS(X_1, X_2)=5/6 \times 0.997 \approx 0.83$ .

Close fluctuations are allocated to the same set by calculating the fluctuation similarity to find the inherent patterns that occur frequently. Algorithm 5 provides the process of detecting inherent patterns and removing fluctuations that belong to inherent patterns from anomalies.

Due to that the fluctuations occurred frequently are not exactly the same, it is necessary to expand the fluctuation sequence before comparing the fluctuation similarity (Line 6-16). After expanding the sequence, calculate the fluctuation similarity between different fluctuations firstly and put similar fluctuations into the same set (Line 24-25). Then calculate the number of elements contained in each set and find the inherent pattern based on the given threshold (Line 30-31). Finally, the fluctuations with inherent patterns are removed from the anomaly (Line 32-36). The time complexity of Algorithm 5 is  $O(n)$ .

#### 4.5 Correlation measure

Multivariate time series is composed of a group of multiple univariate time series. Each univariate time series describes an attribute and there may be correlation between attributes. In this section, we consider how to extend the univariate algorithm to the multivariate time series using the attribution correlation.

Generally, the correlation between attributions is calculated on the original data. In this paper, the absolute second derivative is applied to calculate attribution correlation. For the reason that some attributes do not show very strong correlation in the raw data, but fluctuations are highly correlated and often occur simultaneously or successively. As a result, the correlation should be analyzed based on fluctuations rather than raw data [29].

**Algorithm 5** Inherent patterns detection.

---

```

1: Input: Time series  $X = [x_1, x_2, \dots, x_n]$ ,  $Y = [y_1, y_2, \dots, y_n]$ 
2: Output:  $Y = [y_1, y_2, \dots, y_n]$ 
3: Initialize confidence  $\sigma$ ,  $\delta$ , length  $l$ 
4:  $S = \{\}$ 
5: for  $i \in [1, n]$  do
6:   if  $y_i == 1$  then
7:      $S = S \cup \{x_{i-l}, \dots, x_i, \dots, x_{i+l}\}$ 
8:   end if
9:   if  $y_i == 2$  then
10:     $j = i + 1, P = \{x_i\}$ 
11:    while  $y_j == 2$  do
12:       $P = P \cup \{x_j\}$ 
13:       $j++$ 
14:    end while
15:     $P = \{x_{i-l}, \dots, x_{i-1}\} \cup P \cup \{x_j, \dots, x_{j+l-1}\}$ 
16:     $S = S \cup \{P\}$ 
17:     $i = j$ 
18:  end if
19: end for
20:  $m = 1, s_i, s_j \in S$ 
21: for  $i \in [1, |S|]$  do
22:    $FS_m = \{s_i\}$ 
23:   for  $s_j \in S - \{s_i\}$  do
24:     if  $FS(s_i, s_j) > \sigma$  then
25:        $FS_m = FS_m \cup \{s_j\}$ 
26:     end if
27:   end for
28:    $m++$ 
29: end for
30: for  $i \in [1, m]$  do
31:   if  $|FS_i| > \delta$  then
32:     while  $FS_i \neq \emptyset$  do
33:        $Q =$  Get first element from  $FS_i$ 
34:       while  $x_r \in Q$  do
35:         if  $y_r == 1$  or  $y_r == 2$  then
36:            $y_r = 0$ 
37:         end if
38:       end while
39:        $FS_i = FS_i - \{Q\}$ 
40:     end while
41:   end if
42: end for
43: Return  $Y$ 

```

---

In addition, most multivariate time series in the real world are asynchronous. For example, the user login volume usually lags behind the checkout volume of the shopping cart, the wireless device transmission signal is delayed, or the sampling rate is inconsistent [30]. Therefore, the impact of the asynchrony of multiple time series on the calculation of correlation should be considered. Due to asynchronism, it is more appropriate for Dynamic Time Warping (DTW) to be chosen to solve such a problem. DTW [31] is an elastic measure, which allows to measure the similarity between time series by determining the best alignment between time series and minimizing the impact of time offset.

Given two time series  $P$  with length of  $n$  and  $Q$  with length of  $m$ , DTW recursion is defined as:

$$DTW(P, Q) = d(p_1, q_1) + \begin{cases} DTW(suf(P), suf(Q)) \\ DTW(suf(P), Q) \\ DTW(P, suf(Q)) \end{cases}, \quad (12)$$

where  $d(\cdot)$  is the distance between two matching points. In this paper, we use Euclidean distance to calculate and  $suf(P)$  represents the suffix subsequence  $(p_2, \dots, p_n)$  of  $P$ .

Determine whether there is correlation between attributions based on the calculation results of DTW and the given confidence  $\varepsilon_1$ . If two or more attributions are related, they are grouped into a cluster.

Since the correlation between attributes is acquired based on fluctuations, the final detection result should be obtained through comprehensive analysis of the univariate time series with attribution correlation. When detecting multiple time series  $X(a_1), \dots, X(a_k), \dots$ , we first detect each time series  $X(a_k)$  using the above method, and then judge whether the number of time series with anomalies in the same cluster within the time interval  $[i, i + \tau]$  is greater than  $\varepsilon_2$  according to the given maximum delay tolerance time  $\tau$  and confidence  $\varepsilon_2$ . The maximum delay tolerance time  $\tau$  refers to the longest delay caused by asynchrony. If the result is yes, the original univariate detection result will be maintained, otherwise, the original result will be modified, that is,  $y_t(X(a_k)) = 0, t \in [i, i + \tau]$ .

## 5 Experiment results

We use precision, recall and F1-score to indicate the performance of our algorithm. In paper [33], the authors propose a point-adjust approach to calculate the performance metrics for anomaly detection. This is such an efficient method that many researchers use it as an evaluation criterion and we will use it as well. To calculate the F1-score, True Positives(TP), False Positives(FP), True Negatives(TN) and False Negative(FN) are need to calculate the precision and recall. The formulas are as follow,

$$precision = \frac{TP}{TP + FP}, \quad (13)$$

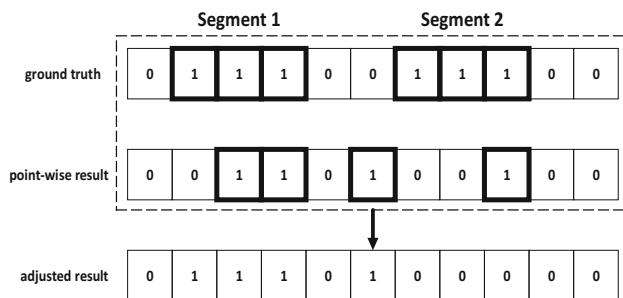
$$recall = \frac{TP}{TP + FN}, \quad (14)$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}. \quad (15)$$

The point adjustment strategy for performance metrics is shown in Figure 4. The frist row is the ground truth with two anomaly segments. The second row is the prediction results of the algorithm and the adjusted results are shown in the third row. We assume that the allowed delay is 1 point and it means that the points in segment1 are adjusted to TP, while the points in segment2 are adjusted to FN.

### 5.1 Dataset

The real datasets with three multidimensional time series and two one-dimensional time series data are used to verify the generality and effectiveness of our proposed algorithm.



**Figure 4** Point adjustment strategy

Besides, three synthetic datasets are created to verify the effect of our proposed algorithm in detecting change points. The details of the real datasets and synthetic datasets are as follows.

**Synthetic Datasets:** These datasets aims to evaluate that our algorithm can identify change points in the presence of various types of anomalies such as point anomalies, collective anomalies, and change points. We first generate the true distribution of data points so that there is no anomaly. Then, we randomly inject anomalies, including point anomalies, collective anomalies and change points. In order to ensure that the algorithm can find change points in complex data, these anomalies exist in multiple and mixed.

#### Real Datasets:

(1) Secure Water Treatment(SWaT) [22]: The SWaT dataset is obtained from 51 sensors of the critical infrastructure system under continuous operations. The collected dataset consists of 7 days collected under normal operations and 4 days collected with attack scenarios.

(2) Server Machine Dataset(SMD) [23]: The SMD is a dataset from a large Internet company collected and made publicly available. It is a 5-week-long dataset containing the data from 28 server machines. Each machine can collect 33 attributes.

(3) Pooled Server Metrics (PSM) [30]: The PSM dataset is collected from multiple application server nodes at eBay with 26 attributes. These characteristics describe server machine metrics such as CPU utilization and memory. The dataset consists of a training set of 13 weeks and a test set of 8 weeks. There are anomalies in both training and testing, and only labels are prepared for the latter. Labels are manually created by engineers and application experts, which may include planned and unplanned anomalies.

(4) Yahoo [25]: The Yahoo dataset is collected from real products traffic, relations and has from 1.5K to 20K records, which 50 time series are annotated anomalies.

(5) IoT [25]: The IoT dataset is collected from 2 ultrasonic sensors deployed at the top of the tank every hour to monitor the liquid level.

As shown in Table 2, the PSD and SWaT datasets have only one entity. Therefore, the result of our algorithm running on the PSD and SWaT datasets is the F1 score. However, the SMD dataset has 12 entities and the result of our algorithm running on the SMD dataset is

**Table 2** Dataset Statistics

Dataset	Entities	Metrics	Test	Anomaly(%)
<i>SMD</i>	12	38	304174	5.84
<i>SWaT</i>	1	51	449919	12.13
<i>PSM</i>	1	25	87841	27.76

the average of the F1 scores for these 12 entities. Details about the number of test points and anomaly rates can be found in this table.

## 5.2 Detection results

**Baselines:** Several methods for detecting anomalies are used as baselines and be compared with our algorithm. An introduction to the baseline is shown below.

(1) CL-MPPCA is a data-driven anomaly detection algorithm that has a Multivariate Convolution LSTM with Mixtures of Probabilistic Principal Component Analyzers. It uses both neural networks and probabilistic clustering to improve the anomaly detection performance.

(2) DAGMM is a Deep Autoencoding Gaussian Mixture Model (DAGMM) for unsupervised anomaly detection. The model utilizes a deep autoencoder to generate a low-dimensional representation and reconstruction error for each input data point, which is further fed into a Gaussian Mixture Model.

(3) Deep-SVDD is a new anomaly detection method-Deep Support Vector Data Description, which is trained on an anomaly detection based objective. The adaptation to the deep regime necessitates that neural network and training procedure satisfy certain properties.

(4) LSTM-VAE is a long short-term memory-based variational autoencoder (LSTM-VAE) that fuses signals and reconstructs their expected distribution by introducing a progress-based varying prior. The LSTM-VAE-based detector reports an anomaly when a reconstruction-based anomaly score is higher than a state-based threshold.

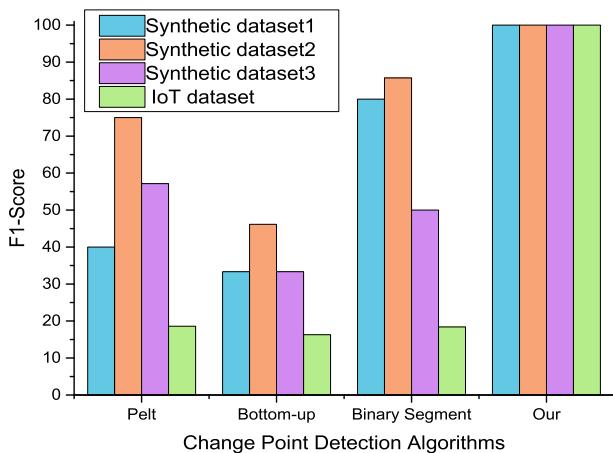
(5) InterFusion is an unsupervised method that simultaneously models the inter-metric and temporal dependency for MTS. Its core idea is to model the normal patterns inside MTS data through hierarchical Variational Auto-Encoder with two stochastic latent variables, each of which learns low-dimensional inter-metric or temporal embeddings. Furthermore, the authors propose an MCMC-based method to obtain reasonable embeddings and reconstructions at anomalous parts for MTS anomaly interpretation.

(6) OmniAnomaly is a stochastic recurrent neural network for multivariate time series anomaly detection that works well robustly for various devices. Its core idea is to capture the normal patterns of multivariate time series by learning their robust representations with key techniques such as stochastic variable connection and planar normalizing flow, reconstruct input data by the representations, and use the reconstruction probabilities to determine anomalies. Moreover, for a detected entity anomaly, OmniAnomaly can provide interpretations based on the reconstruction probabilities of its constituent univariate time series.

(7) USAD is based on adversely trained autoencoders. Its autoencoder architecture makes it capable of learning in an unsupervised way. The use of adversarial training and its architecture allows it to isolate anomalies while providing fast training.

**Change Point Detection:** As can be seen from Figure 5, our proposed algorithm has a higher F1 score than other existing algorithms such as Pruned Exact Linear Time (PELT) [19], Bottom-up [12], and Binary segmentation [13], which are implemented by using the ruptures library. These algorithms all use some parameters such as “penalty value” and the number of data segments. Our algorithm is still the best performer when the parameters are adjusted so that these algorithms achieve optimal results. The main reason is that our algorithm can distinguish point anomalies, collective anomalies and change points well, while other algorithms suffer from the interference of point anomalies and collective anomalies.

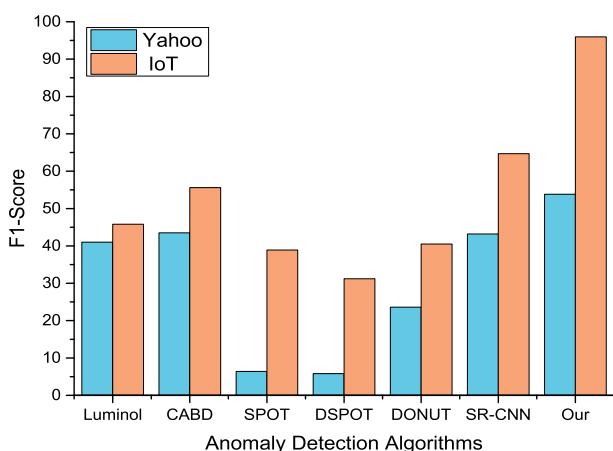
**Anomaly point detection:** To verify the effectiveness of our proposed algorithm, different types of datasets are used to test our algorithm. IoT and Yahoo datasets have only one attribute. Change point detection and experiments based on Yahoo and IoT datasets do not apply



**Figure 5** Change point detection on synthetic datasets and IoT dataset

point adjustment technology. Luminol[1] is a light weight python library for time series data analysis. The two major functionalities it supports are anomaly detection and correlation. It can be used to investigate possible causes of anomaly. CABD[2] is an algorithm that accurately detects and labels anomalies with a non-parametric concept of neighborhood and probabilistic classification. Given a desired quality, the confidence of the classification is then used as termination condition for the active learning algorithm. As Shown in Figure 6, compared with these two algorithms, our algorithm can achieve a higher F1 score, which means that our algorithm can not only process single-dimensional data but also achieve better performance compared with the existing work dealing with single-dimensional data. In Yahoo dataset, there are some concept drift. Since our proposed algorithm can solve the impact of concept drift, our algorithm has better robustness.

Table 3 and Figure 7 show the performance of different methods on multiple multidimensional datasets. It can be found that our algorithm achieves the best results on both SMD and



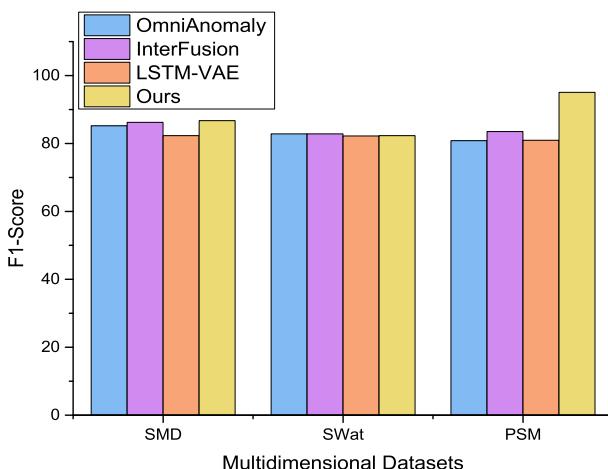
**Figure 6** Anomaly detection on Yahoo and IoT

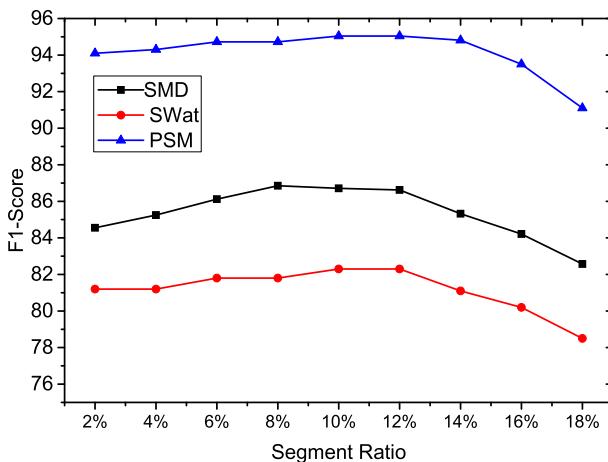
**Table 3** Performance Comparison

Dataset Metric(%)	SWaT			SMD			PSM		
	P	R	F1	P	R	F1	P	R	F1
<i>CL – MPPCA</i>	76.78	81.50	79.07	82.36	76.07	79.09	56.02	99.93	71.80
<i>IsolationForest</i>	49.29	44.95	47.02	42.31	73.29	53.64	76.09	92.45	83.48
<i>DAGMM</i>	89.92	57.84	70.40	67.30	49.89	57.30	93.49	70.03	80.08
<i>Deep – SVDD</i>	80.42	84.45	82.39	78.54	79.67	79.10	95.41	79.10	79.10
<i>LSTM – VAE</i>	76.00	89.50	82.20	75.76	90.08	82.30	73.62	89.92	80.96
<i>InterFusion</i>	80.59	85.58	<b>83.01</b>	87.02	85.43	86.22	83.61	83.45	83.52
<i>Omni Anomaly</i>	81.42	84.30	82.83	83.68	86.82	85.22	88.39	74.46	80.83
<i>USAD</i>	63	74	68	55	72	50	92	58	71
<i>Ours</i>	81.04	83.52	82.26	84.20	89.37	<b>86.71</b>	91.95	98.35	<b>95.04</b>

PSM datasets. Although the F1 score achieved by our algorithm on the SWaT dataset is 82.26, which is smaller than the 83.01 of the InterFusion algorithm, the effect is very close. This means that compared with these efficient existing works mainly based on machine learning, our algorithm has more advantages and good performance in dealing with the problem of anomaly detection in multi-dimensional time series data.

After dividing the data, the size of the segment may have a little impact on the data anomaly detection. In order to analyze the impact, we set the data segment to account for 2% - 18% of the total value. Too small a data segment will cause misjudgment. If the data segment is too large, it will affect the location of the anomalies. We study the impact of segment ratio on three datasets: SMD, SWaT and PSM. The experimental results in Figure 8 show that the algorithm can achieve good performance when the size of segment is 6% - 10% of the data, and the difference is not large. Too large segments do not make the anomaly detection result better. In the same way, too small segments will make F1-Score unable to reach the maximum value.

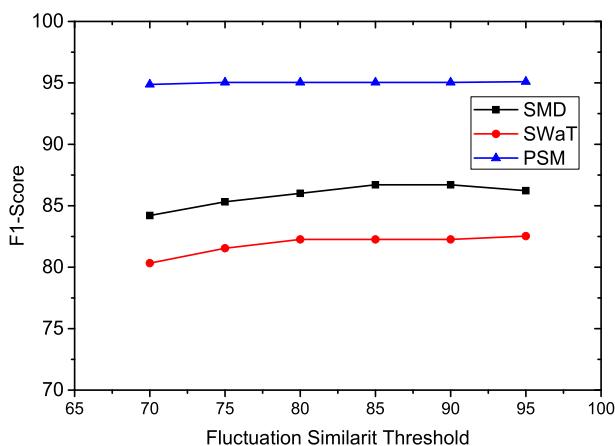
**Figure 7** Anomaly detection on multidimensional datasets



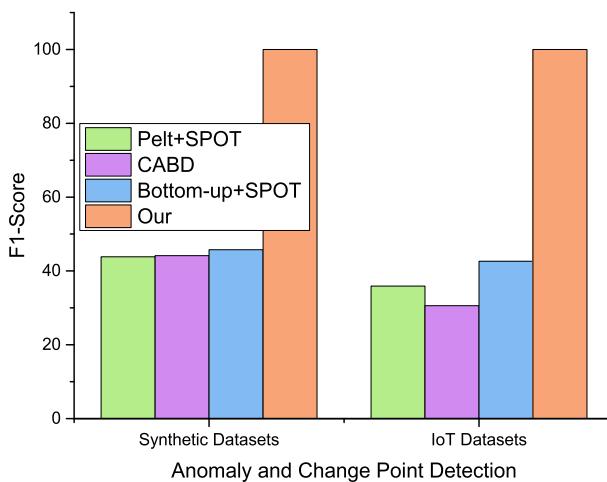
**Figure 8** The influence of segment ratio on anomaly detection

When the fluctuation similarity threshold is changed, it may affect the result of anomaly detection. An experiment on the effect of fluctuation similarity threshold on F1-Score was performed on three datasets SMD, SWat and PSM. The experimental results are shown in Figure 9. In fact, too small a fluctuation similarity threshold can easily lead to many sequences without fluctuation correlation being judged to have the same inherent pattern. We study the impact on the judgment of inherent patterns when the fluctuation similarity threshold is between 0.7 and 0.95. The experimental results show that when the threshold value is greater than 0.7, the effect of natural mode detection is very close. When the fluctuation similarity threshold reaches 0.95, a strict threshold will cause a small part of the inherent pattern to be omitted, but the impact is not serious.

An experiment on the simultaneous detection of anomaly and change point is performed on the synthetic datasets and IoT dataset. Since few methods in the existing work can detect both anomalies and change points at the same time, we use the combination of two algorithms



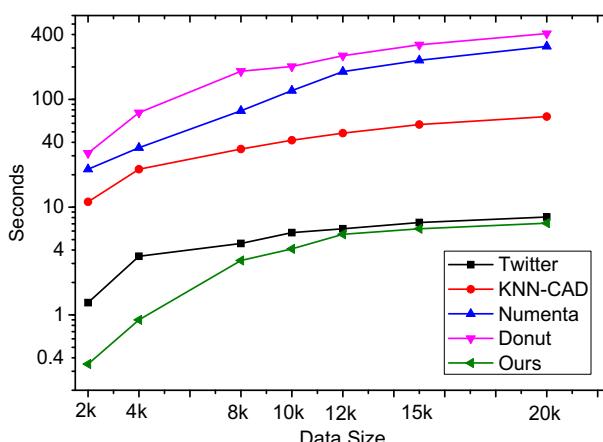
**Figure 9** The influence of fluctuation similarity threshold on anomaly detection



**Figure 10** Anomaly and change point detection

to detect anomalies and change points as much as possible. Pelt and Bottom-up are algorithms used for change point detection. Combined with these two algorithms is the anomaly detection algorithm SPOT. The CABD is an algorithm that can detect anomalies and change points at the same time. The experimental results in Figure 10 show that our proposed algorithm has obvious advantages in detecting anomalies and change points at the same time compared with the existing work.

We compare the runtime with algorithms on data series that have points from 2k to 20K. Experiments are performed on a computer with Intel i5-1135G7 @ 2.40GHz, 2 Cores, 16GB of RAM and Windows 10. For 2K points, Twitter, KNN-CAD, Numenta and Donut take 1.3, 11.2, 22.5, and 31.7 seconds, respectively, while our algorithm takes 0.35 seconds. For 20K data points, Twitter, KNN-CAD, Numenta and Donut take 8.1, 69.3, 310.2 and 408.2 seconds, respectively, while our algorithm takes 7.1 seconds. The results are shown in



**Figure 11** Rrunning time of anomaly detection algorithms over different data sizes

Figure 11. Compared with other algorithms, our algorithm is faster and more efficient because most of the scores are filtered and ignored, thus reducing the amount of computation.

## 6 Conclusion

In this paper, we propose an unsupervised anomaly and change point detection method for time series with concept drift. Different from most of the previous detection methods, we pay more attention to the rate of change of data than the original data, because it can better reflect the fluctuation features of data. Our method not only effectively detects anomalies and change points, but also distinguishes anomalies into point anomalies and collective anomalies. Experimentally, we compared our method with the state-of-the-art methods on several different types of public and synthetic datasets. The results show that the proposed method has good generality and high effectiveness.

**Acknowledgements** This work was supported in part by the National Natural Science Foundation of China under grants U19A2059, U22A2025.

**Author Contributions** Jiayi Liu wrote the main manuscript text, Hong Gao gave suggestions to revise the paper. All authors reviewed the manuscript.

**Funding** This work was supported in part by the National Natural Science Foundation of China under grants U19A2059, U22A2025.

**Availability of data and materials** All of the material is owned by the authors and/or no permissions are required.

## Declarations

**Competing interests** The authors declare that they have no competing interests.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Comput Surv (CSUR). **41**(3), 1–58 (2009)
2. Siffer, A., Fouque, P., Termier, A., Largouët, C.: Anomaly detection in streams with extreme value theory. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, Halifax, NS, Canada, August 13–17, 2017. ACM, pp. 1067–1075
3. Ma, M., Zhang, S., Pei, D., Huang, X., Dai, H.: Robust and rapid adaption for concept drift in software system anomaly detection. In: 2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)s, pp. 13–24. IEEE (2018)
4. Blázquez-García, A., Conde, A., Mori, U., Lozano, J.A.: A review on outlier/anomaly detection in time series data. ACM Comput Surv (CSUR). **54**(3), 1–33 (2021)

5. Cook, A.A., Misirlı, G., Fan, Z.: Anomaly detection for IoT time-series data: A survey. *IEEE Internet Things J.* **7**(7), 6481–6494 (2019)
6. Zamani, M., Sadri, A., Ghafoori, Z., Moshtaghi, M., Salim, F.D., Leckie, C., Ramamohanarao, K.: Unsupervised online change point detection in high-dimensional time series. *Knowl Inf Syst.* **62**, 719–750 (2020)
7. Aminikhanghahi, S., Cook, D.J.: A survey of methods for time series change point detection. *Knowl Inf Syst.* **51**(2), 339–367 (2017)
8. Zamani, M., Ghafoori, Z., Sadri, A., Leckie, C., Ramamohanarao, K.: Change point detection for streaming high-dimensional time series. In: Database Systems for Advanced Applications: DASFAA 2019 International Workshops: BDMS, BDQM, and GDMA, Chiang Mai, Thailand, April 22–25, 2019, pp. 515–519. Proceedings 24. Springer (2019)
9. Ren, H., Xu, B., Wang, Y., Yi, C., Huang, C., Kou, X., Xing, T., Yang, M., Tong, J., Zhang, Q.: Time-series anomaly detection service at microsoft. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, KDD 2019, Anchorage, AK, USA, 4–18 Aug 2019. ACM, pp. 3009–3017 (2019)
10. Liu, D., Zhao, Y., Xu, H., Sun, Y., Pei, D., Luo, J., Jing, X., Feng, M.: Opprendice: Towards practical and automatic anomaly detection through machine learning. In: Proceedings of the 2015 ACM internet measurement conference, IMC 2015, Tokyo, Japan, 28–30 Oct 2015. pp. 211–224 (2015)
11. Yamada, M., Kimura, A., Naya, F., Sawada, H.: Change-point detection with feature selection in high-dimensional time-series data. In: IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, 3–9 Aug 2013. pp. 1827–1833. IJCAI/AAAI (2013)
12. Mehrotra, K.G., Mohan, C.K., Huang, H.: Anomaly detection principles and algorithms, vol. 1. Springer (2017)
13. Burnaev, E., Ishmitsev, V.: Conformalized density-and distance-based anomaly detection in time-series data. arXiv preprint [arXiv:1608.04585](https://arxiv.org/abs/1608.04585) (2016)
14. Breunig, M.M., Kriegel, H., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on management of data, 16–18 May 2000. Dallas, Texas, USA, pp. 93–104. ACM (2000)
15. Gupta, M., Gao, J., Aggarwal, C.C., Han, J.: Outlier detection for temporal data: A survey. *IEEE Trans Knowl Data Eng.* **26**(9), 2250–2267 (2013)
16. Tax, D.M., Duin, R.P.: Support vector data description. *Mach Learn.* **54**, 45–66 (2004)
17. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: International conference on machine learning, pp. 4393–4402. PMLR (2018)
18. Liu, F.T., Ting, K.M., Zhou, Z.-H.: Isolation forest. In: 2008 eighth IEEE international conference on data mining, pp. 413–422. IEEE (2008)
19. Guha, S., Mishra, N., Roy, G., Schrijvers, O.: Robust random cut forest based anomaly detection on streams. In: International conference on machine learning, pp. 2712–2721. PMLR (2016)
20. Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D., Chen, H.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 – May 3 2018, Conference Track Proceedings. OpenReview.net (2018)
21. Audibert, J., Michiardi, P., Guyard, F., Marti, S., Zuluaga, M.A.: USAD: unsupervised anomaly detection on multivariate time series. In: KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual Event, CA, USA, 23–27 Aug 2020. pp. 3395–3404. ACM (2020)
22. Li, D., Chen, D., Jin, B., Shi, L., Goh, J., Ng, S.-K.: Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In: Artificial Neural Networks and Machine Learning–ICANN 2019: Text and Time Series: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part IV, pp. 703–716. Springer (2019)
23. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D.: Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, 4–8 Aug 2019. ACM, pp. 2828–2837 (2019)
24. Li, Z., Zhao, Y., Han, J., Su, Y., Jiao, R., Wen, X., Pei, D.: Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In: KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual Event, Singapore, 14–18 Aug 2021. ACM, pp. 3220–3230 (2021)
25. Le, K.-H., Papotti, P.: User-driven error detection for time series with events. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE), pp. 745–757. IEEE (2020)

26. Braei, M., Wagner, S.: Anomaly detection in univariate time-series: A survey on the state-of-the-art. arXiv preprint [arXiv:2004.00433](https://arxiv.org/abs/2004.00433) (2020)
27. Pickands III, J.: Statistical inference using extreme order statistics. *Ann Stat.* **3**(1), 119–131 (1975)
28. Beirlant, J., Goegebeur, Y., Segers, J., Teugels, J.L.: Statistics of extremes: theory and applications. John Wiley & Sons (2006)
29. Su, Y., Zhao, Y., Xia, W., Liu, R., Bu, J., Zhu, J., Cao, Y., Li, H., Niu, C., Zhang, Y., et al: Coflux: robustly correlating kpis by fluctuations for service troubleshooting. In: Proceedings of the International Symposium on Quality of Service, IWQoS 2019, Phoenix, AZ, USA, 24–25 June 2019. ACM, pp. 1–10 (2019)
30. Abdulalal, A., Liu, Z., Lancewicki, T.: Practical approach to asynchronous multivariate time series anomaly detection and localization. In: KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual Event, Singapore, 14–18 Aug 2021. ACM, pp. 2485–2494 (2021)
31. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. *Knowl Inf Syst.* **7**, 358–386 (2005)
32. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. *ACM Comput Surv (CSUR)*. **46**(4), 1–37 (2014)
33. Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In: Proceedings of the 2018 World Wide Web Conference, WWW 2018, Lyon, France, 23–27 Apr 2018. ACM, pp. 187–196 (2018)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.