Complex & Intelligent Systems (2021) 7:2453–2468
https://doi.org/10.1007/s40747-021-00442-6

ORIGINAL ARTICLE

An edge-cloud collaboration architecture for pattern anomaly detection of time series in wireless sensor networks

Cong Gao[1,2] · Ping Yang[1] · Yanping Chen[1,2] · Zhongmin Wang[1,2] · Yue Wang[1]

Received: 26 October 2020 / Accepted: 10 June 2021 / Published online: 17 June 2021

Abstract
With large deployment of wireless sensor networks, anomaly detection for sensor data is becoming increasingly important in various fields. As a vital data form of sensor data, time series has three main types of anomaly: point anomaly, pattern anomaly, and sequence anomaly. In production environments, the analysis of pattern anomaly is the most rewarding one. However, the traditional processing model cloud computing is crippled in front of large amount of widely distributed data. This paper presents an edge-cloud collaboration architecture for pattern anomaly detection of time series. A task migration algorithm is developed to alleviate the problem of backlogged detection tasks at edge node. Besides, the detection tasks related to long-term correlation and short-term correlation in time series are allocated to cloud and edge node, respectively. A multi-dimensional feature representation scheme is devised to conduct efficient dimension reduction. Two key components of the feature representation trend identification and feature point extraction are elaborated. Based on the result of feature representation, pattern anomaly detection is performed with an improved kernel density estimation method. Finally, extensive experiments are conducted with synthetic data sets and real-world data sets.

Keywords Anomaly detection · Time series · Edge computing · Wireless sensor networks · Feature representation

Introduction

With the innovation and fusion of information technologies such as IoT (Internet of things), cloud computing, and wireless sensor networks, the age of IoE (internet of everything) is around the corner. In this context, there is a remarkable phenomenon that considerable amount of sensors are deployed in various fields, such as environmental monitoring [1], smart manufacturing [2], healthcare [3], and military [4]. Sensors facilitate the human perception of the external world. It is able to conduct effective monitoring in a harsh environment and provide useful information. For a flexible and reconfigurable deployment which is able to collect information within a region of interest, sensors densely located in an area are organized and managed via wireless links, namely wireless sensor networks [5].

A sensor, the fundamental component of wireless sensor networks, is often deployed in a harsh and complicated environment. The common negative factors include high temperature, high humidity, chemical corrosion, electromagnetic interference, and radio frequency interference, etc. Besides, the constrained physical size of a sensor leads to the deficiencies of computational ability, storage space, and power supply, etc. The above external and internal factors compromise the accuracy and reliability of information given by a sensor. In other words, the actual data collecting and transmission of wireless sensor networks contain anomaly [6].

The temporal correlation of data in wireless sensor networks indicates the characteristics of environmental parameter variation. Time series is a vital form of sensor data, which

This work is partly supported by the Science and Technology Project of Shaanxi Province, China (Grant No.2019ZDLGY07-08), the International Science and Technology Cooperation Program of the Science and Technology Department of Shaanxi Province, China (Grant No. 2018KW-049), the Communication Soft Science Program of Ministry of Industry and Information Technology, China (Grant No. 2019-R-29), the Special Scientific Research Program of Education Department of Shaanxi Province, China (Grant No. 17JK0711), and the Special Funds for Construction of Key Disciplines in Universities of Shaanxi Province.

✉ Cong Gao
cgao@xupt.edu.cn

1 School of Computer Science and Technology, Xi’an University of Posts and Telecommunications, Xi’an, China

2 Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi’an University of Posts and Telecommunications, Xi’an, China

is defined as a series of observations with strict sequential order [7]. Unlike traditional types of sensor data, time series in wireless sensor networks possesses temporal continuity, massiveness, and high-dimensionality.

In time series, anomaly indicates the occurrence of abnormal incidents. The abnormal data are defined as the observations which are distinct from majorities and distant from others [8]. Considerable amount of research expects to identify frequent periodical patterns, while neglecting infrequent anomaly. In fact, abnormal data indicate rare events. As rare events are special, they probably contain more value than normal data. Low-quality time series with anomaly requires intensive identification and cleaning for abnormal data. This process reduces productivity and increases the cost of data analysis. Currently, anomaly detection for time series is widely used in IoT [9,10], health monitoring [11,12], financial analysis [13,14], and industrial manufacturing [15,16], etc. The efficient and accurate anomaly detection for time series in wireless sensor networks is essential to subsequent information extraction and data mining.

In [17], the authors focused on the prediction of network failures with machine learning-based anomaly detection techniques. To address the problem of data transferring and computing delay for a large network, edge-cloud computing is introduced to optimize the transferring and computing duration. This model has two drawbacks: (1) machine learning-based techniques require model training and training data, and (2) a detailed procedure for data transferring is missing. In [18], the authors presented a test methodology for the comparison of edge computing architecture and cloud computing architecture for anomaly detection system. The experiments are conducted based on the implementations of deep learning algorithms. The major drawback of deep learning algorithms is that they usually require high computational power. The proposed methodology mainly concentrates on comparisons among deep learning algorithms. The applicability is not wide. In [19], the authors described an autonomous anomaly analysis framework for clustered cloud or edge resources. It aims to find out the cause of a user-aware anomaly in the underlying infrastructure by Hidden Markova Models. Experiments are just conducted in clustered cloud computing resources. Edge resources are not covered. In [20], the authors proposed an intelligence system enabled by edge computing to detect network anomaly. A data-driven method containing four steps is devised to train a learning model. The learning model is used to identify a network anomaly. This proposal just employs the so-called edge intelligence and the discussion about cloud computing is not involved. In [21], the authors depicted a platform concept which combines traditional cloud computing and industrial control empowered by edge computing. Existing self-contained field devices in several dedicated networks are transformed to a unified cloud-assisted control scheme with help of edge devices. For now, the proposed concept does not contain task migration between edge and cloud or anomaly detection.

For pattern anomaly detection in time series, our model proposed in this paper aims to (1) conduct dimension reduction for the purpose of reducing the amount of computation. This motivation is realized by our pattern representation method which contains trend identification and feature point extraction; (2) perform the allocation of detection tasks to the cloud and the edge. Pattern anomaly of both long-term correlation and short-term correlation are handled for subsequent execution of efficient and accurate detection. This motivation is realized by our task migration algorithm; (3) carry out pattern anomaly detection based on kernel density estimation. This motivation is realized by mapping a time series to a five-dimensional feature space. Thus, the traditional anomaly detection based on kernel density estimation which is merely able to handle point anomaly is transformed to detect pattern anomaly.

The remainder of the paper is organized as follows: the next section introduces the classification of anomaly in time series. A comparison is made between cloud-based anomaly detection and edge-based anomaly detection. Both advantages and disadvantages of the two types are analyzed. The following section focuses on the existing models for pattern anomaly detection in time series. The most popular four kinds of feature representation methods are extensively reviewed. The next section describes our edge-cloud collaboration anomaly detection architecture. Five key components task migration algorithm, sliding window, trend identification, feature representation, and kernel density estimation are elaborated. In the following section, the proposed model is evaluated with synthetic data sets and real-world data sets. The next section presents conclusions and directions for future research.

Three kinds of anomaly in time series

Anomaly in time series falls into the following three categories [22].

- Point anomaly [23]: point anomaly refers to a point which is different from other points. Point anomaly is also called *outlier*.
- Pattern anomaly [24]: pattern anomaly refers to a significant difference between a segment pattern and other segment patterns.
- Sequence anomaly [25]: sequence anomaly refers to the non-compliance of a subsequence to other subsequences.

Both point anomaly and pattern anomaly are abnormal behaviors appeared in an individual time series, while sequence anomaly is abnormal behavior appeared between sequences. Existing models for anomaly detection of time series are based on statistics [26,27], distance [28,29], machine learning [30,31], and artificial intelligence [32,33].

For many practical application scenarios, the demand of data analysis focuses on the variation of time series during a period of time, rather than the variation of individual data point. In other words, the benefit of analyzing or data mining for individual point is quite little. While the differences among segments or subsequences are more valuable. For instance, the performance of pattern anomaly detection for hydrologic time series in environmental water quality monitoring [34] is critical to a timely discovery of abnormal water-level variation. This might be very useful for disaster prevention. In [35], the authors pointed out that patterns of continuous time series might reveal contextual and operational conditions of a device. Thus, focusing on segments of time series and identifying a subsequence which is distinct from other subsequences are more rewarding and urgent.

The above anomaly detection methods store and analyze data either locally or remotely. The performance of a local detection approach is limited by constrained resources, such as computational ability and storage space. As the resources of a cloud computing platform are virtually unlimited, users may be inclined to upload data and utilize the super computational power of cloud computing centers to conduct complicated anomaly detection tasks. However, as increasing volume of sensor data demands more bandwidth, data transmission exacerbates the communication bottleneck of cloud computing. Consequently, packet loss and transmission delay get worse. Then, the real-time requirements of anomaly detection cannot be fulfilled.

To make up for cloud computing, researchers turn to introduce edge computing. The operating principle of edge computing is designed to reduce bandwidth demands and link cloud computing power to massive sensor data [36]. In [37], the authors proposed a distributed sensor data anomaly detection model based on edge computing. The continuity of time series and the correlation among multi-source sequences are used to conduct anomaly detection. In [38], the authors proposed a data collecting and cleaning method, where edge nodes perform an angle-based outlier detection to obtain training data. The data cleaning model is constructed based on support vector machine. In general, edge nodes are subject to limited storage space in terms of both RAM and disk. Thus, it is unable to store large amount of historical data. However, in most application scenarios, the interplay of observations and historical values of a time series often exhibits long-term correlation. Due to the inaccuracy of anomaly detection, existing approaches based on edge computing fail in identifying anomaly related to long-term correlation in time series. As possessing sufficient storage space, cloud computing shows better performance than edge computing in this regard.

Pattern anomaly detection in time series

Pattern anomaly detection based on raw time series

Time series is high-dimensional with complicated structure. It always contains considerable noise and fluctuates frequently. With the rapid growth of data volume in wireless sensor networks, traditional method [39,40] which performs anomaly detection directly on raw time series suffers from high time complexity and high space complexity.

Thus, for a given time series, the key to enhance accuracy and validity of anomaly detection is rooted in the following ideas: how to achieve satisfactory dimension reduction, eliminate potential noise, and integrate redundant attributes for the purpose of preventing high dimension disaster? Meanwhile, the dimension reduction result retains basic features and main information of the raw time series [41].

Pattern anomaly detection methods based on different feature representations

Feature representation for time series summarizes and rephrases the whole raw time series. The aim is dimension reduction and noise filtration. By analyzing the characteristics of a time series, the raw time series eventually gets transformed appropriately. A good feature representation method is able to accurately show the basic shape and variation trend with as less data as possible. The most popular four strategies of feature representation for time series are as follows.

Domain transform

Discrete Fourier transform (DFT) [42]: In [43], a time series is approximately presented with discrete Fourier transform in time domain and frequency domain. In specific, the mapping from time domain space to frequency domain space is based on spectral analysis. In [44], the authors extracted key frequency features with fast Fourier transform and use approximate entropy to denote the regularity of a time series. The degree of regularity is used to identify anomaly. In [45], an adaptive short-time Fourier transform (ASTFT) method is proposed. This method incorporates analysis window into the traditional discrete Fourier transform. The analysis window improves computational efficiency of the traditional discrete fourier transform. However, discrete Fourier transform utilizes sine function to achieve time–frequency transformation. Thus, there is only frequency domain information. The drawbacks of DFT are: (1) as the transformation from time domain to frequency domain is conducted by mapping a time series to a group of sinusoidal functions, certain important local features might be smoothed by discrete fourier transform; (2) an accurate Fourier coefficient is essential to the performance of anomaly detection. However, the generation of an accurate Fourier coefficient requires extensive tests. Thus, excessive computation cost results in a low user acceptance; (3) discrete Fourier transform exhibits good performance in a steady time series, while an unsteady time series cannot be handled gracefully.

Discrete wavelet transform (DWT): [46,47] Discrete Fourier transform only contains frequency domain information. While discrete wavelet transform is able to simultaneously analyze time domain information and frequency domain information. By a wavelet function $\varphi(x)$ which satisfies $\int_R \varphi(x)\mathrm{d}x$, a raw time series can be approximately presented with a group of shifted and scaled discrete wavelet transform coefficients. In [48], a maximal overlap discrete wavelet transform method is proposed. This method is able to adapt time series with arbitrary length. Alike discrete Fourier transform, discrete wavelet transform can make analysis on both time domain and frequency domain at the same time. Thus, it is faster than discrete Fourier transform. The drawback of DWT is: an accurate transform coefficient should be determined and this is a quite expensive process.

Singular value decomposition (SVD) [49,50]

Singular value decomposition is a common matrix decomposition method. Based on the eigenvalues and eigenvectors of a time series, the most representative k-dimensional orthogonal vector is extracted. Thus, an n-dimensional raw time series can be transformed to a k-dimensional orthogonal vector, where $k < n$. Singular value decomposition is a powerful tool for dimension reduction. However, previously obtained eigenvalues and eigenvectors are inactive for a new time series. Each data update requires a recalculation of orthogonal vector. This is not suitable for dynamics of time series.

Symbolic discretization [51–53]

Symbolic discretization uses a group of abstract symbols with time domain characteristics to represent a time series, namely a raw time series is substituted by a symbol sequence. In [54], the first symbolic aggregate approximation (SAX) method is proposed. This method uses piecewise aggregate approximation to achieve dimension reduction for a time series. Nevertheless, this method is unable to discriminate segments with the same mean value and different trends. In [55], an extended symbolic aggregate approximation (ESAX) method is proposed. This method incorporates max and min points of each segment into the traditional SAX. In [56], a symbolic aggregate approximation standard deviation (SAX-SD) method is proposed. This method incorporates standard deviation into the traditional SAX. A segment is described based on the standard deviation and mean value. In [57], an iterative end point fitting method searches the end points of each segment based on iteration end point fitting (IEPF) algorithm. This method improves the precision of pattern representation and achieves dimension reduction. For an effective discretization, appropriate symbols and similarity measurement should be defined. However, choosing a suitable discretization algorithm which is able to match an actual time series is not easy. In addition, the representation of a time series by symbolic-based approaches often miss the trend of the raw time series.

Piecewise linear representation [58,59]

Piecewise linear representation divides a raw time series into several segments. Each segment is approximately represented by a linear function. Several line segments constitute the approximate representation of the raw time series. In other words, the representations of line segments are based on important points. Compared to the above three kinds of methods, piecewise linear representation possesses small index dimension and low computational overhead. Moreover, this type of representation is more friendly to human visual experience.

Dividing points and number of segments: As piecewise linear representation uses a group of adjacent line segments to represent a time series, the degree/granularity of approximation largely depends on the number of segments. The selections of appropriate dividing points and the number of segments are two vital factors of piecewise linear representation. Existing approaches focus on the following two ideas.

– Segment error e: the adjustment of segment error is based on two indicators. First, the maximum error of an individual segment should be greater than a predetermined threshold. Second, the sum of the maximum errors of all segments should not exceed a predetermined threshold.
– Number of segments k: an optimal value of k should be determined by integrating demands such as compression ratio, computation speed, and searching precision.

Object of division: Piecewise linear representation can be classified as global representation [60,61] and local representation [62,63].

Global representation is based on the whole time series. By comparing the overall fitting error with a predetermined threshold, the optimal set of segments are obtained. In general, the threshold is determined based on the Euclidean distance, orthogonal distance, and perpendicular distance between an observation and a segment.

Local representation focuses on qualified local characteristics of a time series. Based on the selection of dividing points, local representation is classified as the following three types:

- Extremum points [64]: for a segment $\{x_{i-1}, x_i, x_{i+1}\}$ of a time series $X = \{x_1, x_2, \ldots, x_n\}$, if $(x_{i-1} \leq x_i \mid x_{i+1} \leq x_i)$ is true, namely the observations are monotonically increasing or decreasing at extremum x_i, the raw time series can be approximately represented by the set of these extremum points. However, the representation based on extremum points suffers from unfiltered and trivial information. Noise cannot be effectively eliminated.
- Local extremum points [65]: as the above extremum points approach is unable to eliminate noise, local extremum points are introduced to handle details related to noise. For each segment, certain negligible intermediate observations between the maximum and the minimum are filtered. However, to achieve an effective approximate representation of a raw time series, the number, range, and characteristics of negligible observations should be prudently selected.
- Important points [66]: important points are the most influential ones which demonstrate the variation trend of a time series. Traditionally, the selection of important points is conducted by measuring the amplitude variation between observation x_i and its predecessor x_{i-1}. If the amplitude variation is greater than a predetermined threshold, observation x_i is identified as an important point. In specific, if $(|(x_i - x_{i-1})/x_{i-1}| \geq R_1 \mid |(x_i - x_{i-1})| \geq R_2)$ is true, where R_1 and R_2 are application-related values, x_i is identified as an important point. However, identification rate for certain pivot points is low.

In [67], the authors integrated sliding window with Piecewise Aggregate Approximation (PAA). In this model, a raw time series is divided to several segments with equal length. For segment $X_i = x_{i1}, x_{i2}, \ldots, x_{i(n-1)}, x_{in}$, we denote the starting point and the ending point of X_i by x_1 and x_n, respectively. The values of observation x_{ij}, where $2 \leq j \leq n-1$, are substituted by the mean value $\frac{1}{n-2}\sum_{j=2}^{n-1} x_{ij}$. In [68], the authors proposed a temporal correlation diagram model based on piecewise aggregate approximation. This model is designed for multi-dimensional time series. Clusters of time series are obtained based on the degree of correlation. Anomaly detection can be conducted in three ways: within a cluster, among clusters, and within a single dimension. However, substitution with a mean value smooths oscillations of the raw time series, leading to missing of important points. The missing of extremum information results in false negative in anomaly detection.

Based on the idea of piecewise aggregate approximation, the authors proposed adaptive piecewise constant approximation (APCA) in [69]. A raw time series is divided into several segments with unequal lengths based on important points. The observations in each segment are substituted by a mean value. The adaptive piecewise constant approximation is more flexible than the piecewise aggregate approximation. However, there is no commonly accepted method for the selection of important points. For different methods, the obtained important points of the same time series might be quite different.

In [22], the authors proposed piecewise aggregate pattern representation (PAPR). A segment of a raw time series is divided into several areas in amplitude domain. Then, the segment is modeled by a matrix based on statistical information contained in the segment. However, various movement shapes of a time series cannot be accurately presented.

Edge-cloud collaboration anomaly detection architecture

Edge-cloud collaboration architecture

When massive data are transmitted to cloud computing centers for anomaly detection, high bandwidth demands and large network delay are inevitable. In general, as the physical resource of an edge device is constrained, it is unsuitable for the storage of historical data. However, edge computing is capable of detecting anomaly related to short-term correlation in time series. Anomaly in this type of time series is usually caused by noise. As described in the section "Introduction", noise is mainly introduced by external factors. However, when potential anomaly is caused by internal factors, the analysis of historical data is mandatory. Thus, large storage space and powerful computational ability are required. Anomaly detection with edge computing is likely to let the variation related to historical data slip away. Hence, it is unable to detect anomaly related to long-term correlation in time series. As a cloud computing center possesses massive historical data and excellent computing power, it is capable of identifying anomaly related to long-term correlation in time series. In summary, the cloud works on the whole historical data or large part of it. While the edge devices works only on the most recent of data. The amount of data handled by the edge devices in each process is small. This aims to accord real-time response which is not easy for the cloud. For our proposal and experiments in this paper, the number of data points processed by the edge devices each time is about 10^3. While the number of data points processed by the cloud each time is at least 1.5×10^7. For anomaly of time series in wireless sensor networks, a large number of them are related to short-term correlation. Excessive detection tasks waiting

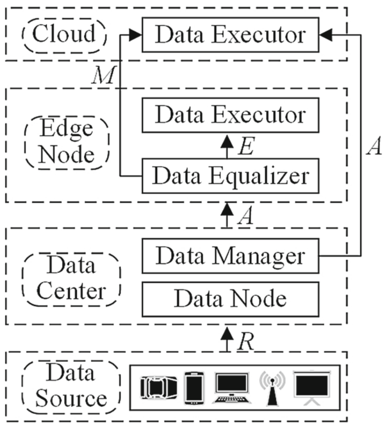


Fig. 1 Edge-cloud collaboration anomaly detection model

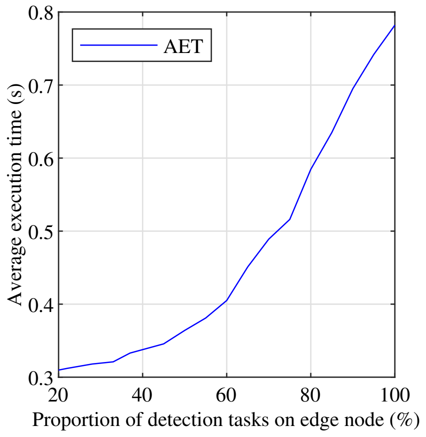


Fig. 2 Variation of execution time with the proportion of detection tasks on edge node

at resource-constrained edge devices bring about congestion and degrade the overall performance of the system. It is necessary that certain computational tasks of edge devices are transferred to a cloud computing center. Thus, a rational task migration mechanism is needed.

Our edge-cloud collaboration architecture is depicted in Fig. 1. There are totally four layers: data source, data center, edge node, and cloud. The four upper case letters A, E, M, and R stand for allocation, execution, migration, and registration, respectively. The data source layer contains various devices (e.g., equipment in a smart factory). Existing data types of data source are registered to data center layer. The data center layer contains several data nodes and a data manager. These data nodes are used to store data. The data manager is responsible for allocating data. Namely, data with long-term correlations are allocated to the cloud, while data with short-term correlations are allocated to the edge. The data equalizer in edge node layer conducts a further scheduling procedure to dispatch short-term correlation computation over the cloud. Certain data are migrated to the cloud. Data executors in the edge node layer and the cloud layer perform specific anomaly detection tasks.

Task migration

For time series $X = \{x_1, x_2, \ldots, x_n\}$ and the corresponding autocorrelation function γ_τ, where τ is the lag order of X, if

$$\lim_{n\to\infty} \sum_{\tau=-n}^{n} |\gamma_\tau| \to \infty \tag{1}$$

holds, X is considered to be a time series with long-term correlation [70]. Data manager in Fig. 1 is responsible for the allocation of anomaly detection tasks. In specific, an anomaly detection task related to long-term correlation in time series is assigned to cloud computing center, while an anomaly detection task related to short-term correlation in time series is carried out at edge nodes.

Compared to cloud computing, the resources of edge computing are quite limited. Thus, it is essential to investigate the relation between the amount of detection tasks for time series with short-term correlation and the execution time of detection tasks. We conduct multiple experiments based on our previous model proposed in [71]. In specific, pattern anomaly detection is performed on 15 edge nodes with respect to ECG data set in UCR [72]. The proportion of detection tasks allocated to edge nodes is in the range of [20%, 100%]. The experimental result is shown in Fig. 2.

With the increase of the detection tasks allocated to edge layer, the average execution time of detection tasks is monotonously increasing. When considerable amount of detection tasks are allocated to edge layer, some of them have to wait. During a given period, the total number of anomaly detection tasks could be considered as fixed. As more tasks introduce more waiting, the overall anomaly detection performance of edge layer is inversely proportional to the number of task allocated to edge nodes. Thus, a task scheduler is needed. When there are excessive detection tasks at edge layer, certain tasks are migrated to remote cloud computing center. Specifically, the tasks to be migrated are selected based on estimated queuing time.

Suppose there are n anomaly detection tasks for time series with short-term correlation at a specific edge node. These tasks are denoted by set $\{\phi_1, \phi_2, \ldots, \phi_n\}$. An individual anomaly detection task is defined as

$$\phi_i = (s_i, r_i, t_{\max i}, \lambda_i), \tag{2}$$

where s_i is the scale of an input time series, r_i is the required resources for detection. In our experiments, s_i is the length of input time series, and r_i is a normalized value represents the capacity of CPU and RAM. $t_{\max i}$ is the maximum acceptable delay, and λ_i denotes the location of anomaly detection. When ϕ_i is processed at edge node, $\lambda_i = 0$. While $\lambda_i = 1$ denotes ϕ_i is handled by cloud. Thus, the processing cost for ϕ_i is

$$T_i = \begin{cases} \frac{s_i}{r_i^e} + \tau_i^e,\ \lambda_i = 0 \\ \frac{s_i}{r_i^c} + \tau_i^c,\ \lambda_i = 1 \end{cases}, \tag{3}$$

where r_i^e is the computing resource edge node could provide, τ_i^e is the estimated queuing time for ϕ_i to get detected, r_i^c is the estimated computing resource consumed for migrating ϕ_i from edge layer to cloud, and τ_i^c is transmission delay for the migration. In (3), both $\frac{s_i}{r_i^e}$ and $\frac{s_i}{r_i^c}$ are values treated as a measure of execution time of the processing procedure. Thus, the objective function for all detection tasks is

$$\min\left(\sum_{i=1}^{N}\left(\lambda_i\ ?\ (\frac{s_i}{r_i^c} + \tau_i^c)\ :\ (\frac{s_i}{r_i^e} + \tau_i^e)\right)\right), \tag{4}$$

where "(conditional expression) ? (expression1) : (expression2)" is ternary operator which is widely used in programming languages. When the result of the conditional expression is true, invoke expression1. When the result of the conditional expression is false, invoke expression2. The calculation of the objective function in (4) omits the processing cost of a task assigned to edge node. when compared to being processed by cloud, queuing time and resource demands of a task assigned to edge node are trivial.

For an application scenario containing m edge nodes and n cloud computing centers, the sets of edge nodes and cloud computing centers are denoted by $N = \{n_1, n_2, \ldots, n_m\}$ and $C = \{c_1, c_2, \ldots, c_n\}$, respectively. The potential anomaly detection tasks of time series with short-term correlation for edge node n_i are set $E_i = \{\phi_{i1}, \phi_{i2}, \ldots, \phi_{ik}, \}$. The anomaly detection tasks for m edge nodes are denoted by set $E = \{E_1, E_2, \ldots, E_m\}$. The detailed task migration algorithm is shown in Algorithm 1.

By default, all anomaly detection tasks in set E are allocated to edge nodes. Thus, set S, which denotes the detection tasks should be migrated to cloud, is initially an empty set. For each edge node, detection tasks initially allocated to it are examined by the following rules: (1) if the estimated queuing time for a detection task is larger than the maximum acceptable delay, the detection task will be migrated to cloud; (2) for a detection task whose estimated queuing time is acceptable, the processing cost for the detection task is further considered. If processing at edge node costs more than that of cloud and the estimated computing resource consumed for migrating the detection task is smaller than the maximum acceptable delay, it is also migrated to cloud.

Algorithm 1 *migration*(E)

```
1: $S \leftarrow \varnothing$
2: for $i = 1$ to $m$ do
3:   if $|E_i| \neq \varnothing$ then
4:     for $j = 1$ to $|E_i|$ do
5:       if $\tau_{ij}^e > t_{maxij}$ then
6:         $E_i = E_i \setminus \{\phi_{ij}\}$
7:         $S = S \cup \{\phi_{ij}\}$
8:       else
9:         if $T_{ij}\big|_{\lambda_{ij=0}} > T_{ij}\big|_{\lambda_{ij=1}}$ && $\tau_{ij}^c < T_{maxij}$ then
10:          $E_i = E_i \setminus \{\phi_{ij}\}$
11:          $S = S \cup \{\phi_{ij}\}$
12:        end if
13:      end if
14:    end for
15:  end if
16: end for
17: return $S$
```

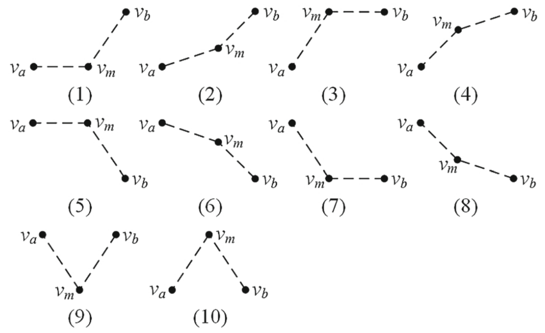


Fig. 3 Ten trends of variation

Multi-dimensional feature representation

Sliding window

As the processing of massive high-dimensional time series data is tricky, dimension reduction is necessary to an efficient feature representation. For a raw time series $X = \{x_1, x_2, \ldots, x_n\}$, local variation and trends at different locations might be radically diverse. We devise a sliding window to extract feature points of the raw time series X. The window size is determined based on trend variation of a raw time series. Thus, the sliding window is able to adapt the variation of data and retain the major characteristics of a raw time series. In brief, the sliding window keeps enlarging until a certain trend is enclosed. To facilitate the identification of a certain trend, we formulate ten trends to accurately reflect feature points (e.g., extreme points and inflection points) of a certain trend. For simplicity and generality, these ten trends are modeled with three key points. Namely, other trivial points are not shown. Here, trivial points refer to points which do not belong to a given trend. An arbitrary time series can be decomposed into a series of trends illustrated in Fig. 3.

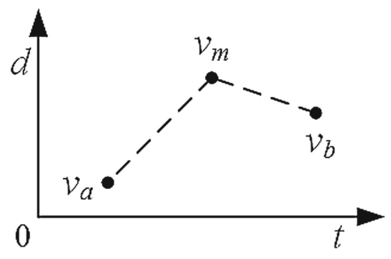


Fig. 4 A trend in the Cartesian coordinate system

Table 1 Description of the ten trends

Trend	Description
Figure 3(1)	$d_a = d_m < d_b$
Figure 3(2)	$d_a < d_m < d_b$ && $k_{am} - k_{mb} < 0$
Figure 3(3)	$d_a < d_m = d_b$
Figure 3(4)	$d_a < d_m < d_b$ && $k_{am} - k_{mb} > 0$
Figure 3(5)	$d_a = d_m > d_b$
Figure 3(6)	$d_a > d_m > d_b$ && $\lvert k_{am}\rvert - \lvert k_{mb}\rvert < 0$
Figure 3(7)	$d_a > d_m = d_b$
Figure 3(8)	$d_a > d_m > d_b$ && $\lvert k_{am}\rvert - \lvert k_{mb}\rvert > 0$
Figure 3(9)	$d_a > d_m < d_b$
Figure 3(10)	$d_a < d_m > d_b$

Moreover, as shown in Fig. 4, the Cartesian coordinates of v_a, v_b, and v_m are denoted by (t_a, d_a), (t_b, d_b), and (t_m, d_m), respectively.

The ten trends shown in Fig. 3 can be described as the ten formulas in Table 1, where k_{xy} $(x, y \in a, m, b)$ denotes the slope of line segment $v_x v_y$ and so forth.

With the enlarging of the sliding window, the number of data points within the window is increasing. Once the overall trend of the data points within the sliding window matches one of the ten trends in Fig. 3, the window stops enlarging. Thus, a specific window size is obtained. The above process keeps looping until an input raw time series ends. The criteria for determining a trend are listed in Table 2. Every time a specific trend is determined, the three points of the trend are added to the set of feature points P.

In Table 2, the parameter ϵ denotes the fluctuation of the raw time series X. In specific, it is calculated as

$$\epsilon = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2}}{g}, \tag{5}$$

where $g \in [g_{\min}, g_{\max}]$. For the raw time series X, there might exist j pairs of extreme values $E = \{\text{emin}_1, \text{emax}_1, \text{emin}_2, \text{emax}_2, \ldots, \text{emin}_j, \text{emax}_j\}$. $g_{\min}$ and $g_{\max}$ are denoted as

Table 2 Determination of the ten trends

Trend	Condition
Figure 3(1)	$\lvert v_m - v_b\rvert > \epsilon \;\Vert\; t_m - t_a + 1 > l$
Figure 3(2)	$v_m - v_a < \epsilon$ && $\lvert v_m - v_b\rvert > \epsilon$
Figure 3(3)	$v_m - v_a > \epsilon \;\Vert\; t_b - t_m + 1 > l$
Figure 3(4)	$v_m - v_a > \epsilon$ && $\lvert v_m - v_b\rvert < \epsilon$
Figure 3(5)	$v_m - v_b > \epsilon \;\Vert\; t_m - t_a + 1 > l$
Figure 3(6)	$\lvert v_m - v_a\rvert < \epsilon$ && $v_m - v_b > \epsilon$
Figure 3(7)	$\lvert v_m - v_a\rvert > \epsilon \;\Vert\; t_b - t_m + 1 > l$
Figure 3(8)	$\lvert v_m - v_a\rvert > \epsilon$ && $v_m - v_b < \epsilon$
Figure 3(9)	$\lvert v_m - v_a\rvert > \epsilon$ && $\lvert v_m - v_b\rvert < \epsilon$
Figure 3(10)	$v_m - v_a > \epsilon$ && $v_m - v_b < \epsilon$

$$g_{\min} = \min_{i=1,2,\ldots,j} \{\text{emax}_i - \text{emin}_i\}, \tag{6}$$

$$g_{\max} = \max_{i=1,2,\ldots,j} \{\text{emax}_i - \text{emin}_i\}, \tag{7}$$

For a given input raw time series X, the numerator in (5) is a constant. Thus, the value of ϵ is inversely proportional to the denominator g. By Table 2, when ϵ is small, a potential trend can be identified more accurate than a large ϵ. In addition, there are more trends identified for the same raw time series X. In general, more trends lead to more feature points. Though more feature points require more amount of computation, the fitting performance is improved. For simplicity, we prefer $g = g_{\max}$ for our experiments in the section "Experiments and analysis".

For the four trends in Fig. 3(1,3,5,7), there is an extra parameter l listed in Table 2. Take the trend in Fig. 3(1) for example, $t_m - a + 1$ is the number of data points contained in the line segment $v_a v_m$. This value is also considered as the width of $v_a v_m$ and it is important to the trend determination and the fitting result. Thus, we introduce the above parameter l as a threshold of the width of $v_a v_m$. For the two trends in Fig. 3(1) and (3), we have

$$l_1 = \frac{\epsilon}{v_b - v_a}, \tag{8}$$

$$l_3 = \frac{\epsilon}{v_m - v_a}. \tag{9}$$

The value of l largely depends on the value of ϵ. Thus, the fitting performance is also largely depending on the value of ϵ. The smaller ϵ is, the smaller the sliding window is. Meanwhile, the better the fitting performance is. In addition, the calculation of l for the other two trends in Fig. 3(5) and (7) is similar to (8) and (9).

For fitting result X, it might contain n trends. To facilitate the subsequent pattern representation and reduce the amount of computation, it is further divided into $\lfloor\sqrt{n}\rfloor$ segments: $X_1, X_2, \ldots, X_i, \ldots, X_{\lfloor\sqrt{n}\rfloor}$. The number of segments is in accordance with the number of trends. For each segment X_i, the following five features of X_i are considered: mean value, kurtosis, oscillation, variation coefficient, and trend coefficient.

Mean value

The mean value of w observations in X_i is

$$M_i = \frac{1}{w}\sum_{k}^{iw} x_k, \tag{10}$$

where $k = (i-1)w + 1$.

Kurtosis

The kurtosis of X_i is a measurement of abruptness or flatness of peak value compared to normal distribution. In general, time series with a large kurtosis tends to contain anomaly, while a small kurtosis indicates that there might be no anomaly. The kurtosis of X_i can be calculated as

$$K_i = \frac{1}{w}\sum_{k=1}^{w} \delta_k^4 - 3, \tag{11}$$

where δ_k denotes the standardized values corresponding to the standard deviation computed with w as the denominator, and the magic number *3* is the kurtosis of normal distribution.

Oscillation

Oscillation refers to the periodic fluctuation of time series. It is an indicator of local variation. In [73], the authors investigated the identification of electroencephalography (EEG) oscillations. In [74], the authors proposed an improved discrete cosine transform (DCT) as a means of feature extraction. Discrete cosine transform is a type of orthogonal transformation. The basis vector of DCT transformation matrix works well for the feature description of human voice signal and image signal. The one-dimensional discrete cosine transform can be defined as

$$F(0) = \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1} x_i, \tag{12}$$

$$F(u) = \sqrt{\frac{2}{N}}\sum_{i=0}^{N-1} x_i \cos\frac{(2i+1)u\pi}{2N}, \tag{13}$$

where u is the generalized frequency, $u = 1, 2, \ldots, N-1$. $F(u)$ is cosine transform coefficient.

Thus, for the oscillation detection in univariate time series, the degree of oscillation can be formulated as

$$O_i = \sqrt{\frac{1}{N}}F(0) + \sqrt{\frac{2}{N}}\sum_{u=1}^{N-1} F(u)\cos\frac{(2i+1)u\pi}{2N}. \tag{14}$$

Variation coefficient

Based on (11), when an abrupt peak value appears in a segment of time series, the probability of anomaly is increasing. In this case, variation coefficient is used to measure degree of local abruptness with respect to the whole time series [75]. It can be calculated as

$$V_i = \frac{\delta_i}{\mu}, \tag{15}$$

where σ_i is the standard deviation of segment X_i, μ is the mean value of the whole time series X.

Trend coefficient

Besides the local variation of time series, the trend variation of a segment is also of great importance. Here, we employ the trend variation of ith segment given in [76]

$$T_i = \text{std}(\text{smooth}(X)), \tag{16}$$

where $smooth(X)$ returns the smoothed result of the original time series X, and $std(\cdot)$ returns a standard deviation. For a time series with random trend, a small value of T_i represents that there is no abrupt peak. While, a large value of T_i indicates the existence of abrupt peak.

Based on the above five features, segment X_i can be approximately represented by

$$X_i = [M_i, K_i, O_i, V_i, T_i]. \tag{17}$$

This representation is aimed to retain the major information of a raw time series. The abstract description of basic shape and variation trend of a raw time series not only facilitate the anomaly detection, but also contribute to the improvement of accuracy and effectiveness of subsequent information extraction and data mining.

Pattern anomaly detection based on kernel density estimation

Traditional anomaly detection based on kernel density estimation is targeted at point anomaly. By mapping a raw time series to a five-dimensional feature space, traditional anomaly detection based on kernel density estimation is transformed to detect pattern anomaly in time series. In short, when the local density of a segment is different from its neighborhood, the segment is considered to be a pattern anomaly.

For a raw time series $X = \{X_1, X_2, \ldots, X_n\}$, the Gaussian kernel density distribution of ith segment X_i is

$$f(X_i) = \frac{1}{n} \sum_{n}^{i=1} \frac{1}{\omega} G(X_i, X_j), \tag{18}$$

where $G(X_i, X_j)$ is a Gaussian kernel function with width ω

$$G(X_i, X_j) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{\left\| X_i - X_j \right\|^2}{2\omega^2} \right], \tag{19}$$

where $\left\| X_i - X_j \right\|$ denotes the Euclidean distance between X_i and X_j.

To estimate the density distribution near a segment, the k nearest neighbors of segment X_i are introduced as set

$$D(X_i) = \{N_1(X_i), N_2(X_i), \ldots, N_k(X_i)\}, \tag{20}$$

where $N_r(X_i)$ is the Gaussian kernel distance between X_i and its rth nearest neighbor.

Based on (16) and (18), the anomaly score of segment X_i can be defined as

$$AS(X_i) = \frac{\sum_{X_j \in D(X_i)} f(X_j)}{|D(X_i)|\, f(X_i)}, \tag{21}$$

where $|D(X_i)|$ is the actual number of nearest neighbors. $AS(X_i) > 1$ indicates X_i is distant from its densely distributed neighbors, and thus, X_i is considered to be a pattern anomaly. On the contrary, $AS(X_i) \le 1$ stands for X_i is close to its densely distributed neighbors, hence X_i is considered to be normal.

For an identified pattern anomaly X_i, the corresponding data are considered to be abnormal. In general, this segment of data is not suitable for further analyzing or data mining. It is possible to try to rectify this segment of data by certain techniques. However, in this paper, we focuses on the anomaly detection model and task allocation method. The rectification of abnormal data involves other techniques and is not covered by our proposal.

Experiments and analysis

The performance of our model is evaluated based on synthetic data sets and real-world data sets. Moreover, all the data involved in our experiments are stationary data. Namely, for integer set $\mathbb{Z} = \{0, \pm 1, \pm 2, \ldots\}$, time series $X(t), t \in \mathbb{Z}$ possesses the following three properties.

- $E\,|X(t)|^2 < \infty$, for all $t \in \mathbb{Z}$,
- $EX(t) = m$, for all $t \in \mathbb{Z}$,
- and $\gamma_x(r, x) = \gamma_x(r + t, s + t)$, for all $t \in \mathbb{Z}$,

where $\gamma_x(\cdot, \cdot)$ is the autocovariance function of $X(t)$

$$\begin{aligned} \gamma_x(r, s) &= \mathrm{Cov}(X(r), X(s)) \\ &= E\left[(X(r) - EX(r))(X(s) - EX(s))\right], r, s \in \mathbb{Z}. \end{aligned} \tag{22}$$

Data sets

Synthetic data set

Based on the experiments conducted in [22], the synthetic data set is generated by the following stochastic process:

$$X(t) = \sin\left(\frac{40\pi}{K} t\right) + n(t), \tag{23}$$

$$Y(t) = X(t) + e_1(t) + e_2(t), \tag{24}$$

where $t = 1, 2, \ldots, 1200$, $K = 1200$. $n(t)$ is a Gaussian noise with the mean $\mu = 0$ and the standard deviation $\sigma = 0.1$.

Two abnormal patterns superposed on $X(t)$ are

$$e_1(t) = \begin{cases} n_1(t), & t \in [600, 630] \\ 0, & \text{otherwise} \end{cases}, \tag{25}$$

$$e_2(t) = \begin{cases} 0.4 \cdot \sin\left(\frac{40\pi}{K} t\right), & t \in [800, 830] \\ 0, & \text{otherwise} \end{cases}, \tag{26}$$

where $n_1(t) \sim N(0, 0.55)$. To sum up, time series $Y(t)$ can be considered as a sinusoidal wave with anomaly in [600, 630] and [800, 850].

Real-world data sets

To evaluate our model in production environments, we employ seven real-world data sets: ECG data with the length of 3571 in UCR [77], air quality data with the length of 2190 [78], hydrological data of Yellow River with the length of 4412 [79], individual household electric power consumption data with the length of 9638 [80], traffic data with the length of 7250 in the national road database of Norway [81], temperature data with the length of 16,000, and video surveillance data contain 21,600 samples from ZTE intelligent terminal manufacturing workshop in Xi'an, China. The above seven data sets are denoted as R1, R2, R3, R4, R5, R6, and R7, respectively.

Table 3 Average execution time for three models

Data set	Cloud-based (s)	Edge-based (s)	Edge-cloud collaboration (s)
Y	0.32	0.29	0.27
R1	0.35	0.34	0.31
R2	0.34	0.32	0.29
R3	0.39	0.36	0.30
R4	0.48	0.52	0.39
R5	0.43	0.41	0.37
R6	0.53	0.48	0.42
R7	0.61	0.56	0.44

Experimental parameters and results

In our experiments, we employ 15 edge nodes and 1 cloud node. The 15 edge nodes are implemented on MSP430 single chip computer equipped with nRF905 wireless module. The cloud node is a HP Z6 G4 workstation with 32 2.3 GHz cores and 32 GB RAM. In addition, the cloud node runs a Debian Stretch 9.4.0 [82]. While the 15 MSP430 single chip computers are installed with an open source operating system called FreeRTOS [83]. The MSP430 platform is able to compute in 25MHz and provide 100KB RAM. For simplicity, r_i^e in (3) is computed based on the available RAM of an edge node. The specific values of r_i^e are confined in [10, 100], where 100 denotes that all 100 KB RAM is available. For r_i^c in (3), it is also computed based on the available RAM of the cloud node. In our experiments, we consider it as a fixed value 1.0×10^8.

Since the locations of anomaly detection are different, the summary of the average execution time for cloud-based, edge-based, and edge-cloud collaboration detection models is shown in Table 3. As most data received by cloud-based model are time series with long-term correlation, the cloud-based model demands more bandwidth than the other two models. Similarly, the transmission time of cloud-based model is also more than the other two models. This contributes to a large execution time. For edge-cloud collaboration detection model, long-term/short-term related anomaly, and detailed computational tasks are migrated properly. Thus, the edge-cloud collaboration model is more efficient than cloud-based model and edge-based model.

For anomaly detection task of short-term related time series ϕ_i, the relation between computing resource of edge node r_i^e and average execution time is depicted in Fig. 5. The data illustrated in Fig. 5 are based on the experimental results with respect to data sets R1 $\sim$ R7. As cloud-based model is irrelevant to the resources of edge node, the run time of cloud-based model remains unchanged during the resource variation of edge node. For edge-based model and edge-cloud collaboration model, execution time of a detection task shows a decreasing trend with the increase of resources of edge node. When the resources of edge node are insufficient, the execution time of edge-based model is more than that of the other two models. As certain tasks are migrated to remote cloud computing center, the edge-cloud collaboration model exhibits the best performance in terms of execution time.

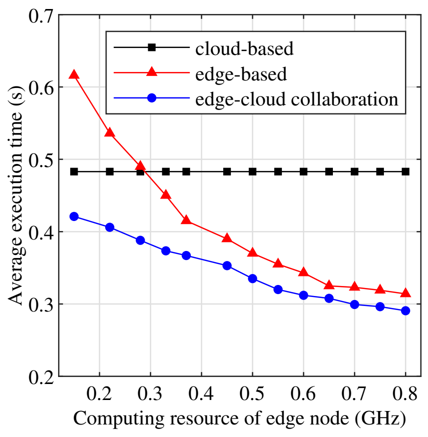


Fig. 5 Variation of average execution time with available computing resource of edge node

To facilitate the presentation, we call our anomaly detection method as multiple dimension feature representation anomaly detection (MDFR-AD). To analyze the relation between anomaly detection performance and the variation of k in kernel density estimation, performance metric area under the curve (AUC) [84] is evaluated with real-world data set R1. As shown in Fig. 6, three methods are depicted. With the increase of k, the three AUC curves initially increase dramatically. In the mid-late stage, the three AUC curves gradually level out. The performance of MDFR-AD is superior to the other two types of methods (PAA-based [68,85] and PLAA-based [67,86]). In particular, a significant performance improvement of MDFR-AD appears around $k = 5$. Thus, parameter $k = 5$ is used for subsequent experiments in this section.

For time series $Y(t)$, the anomaly detection result obtained by MDFR-AD is shown in Fig. 7. As shown in Fig. 7a, the original time series $Y(t)$ is divided to 12 segments with

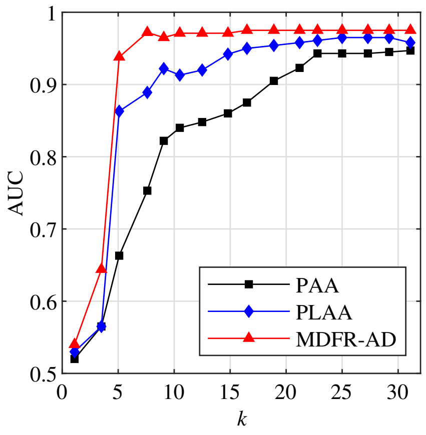
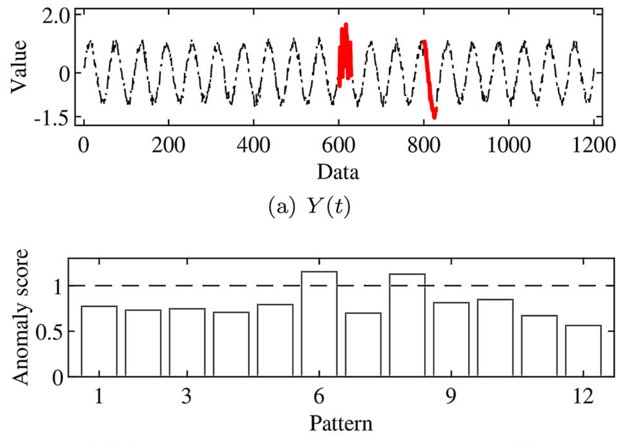


Fig. 6 k vs. AUC



(a) $Y(t)$

(b) Anomaly scores for 12 segments of $Y(t)$

Fig. 7 Anomaly detection of $Y(t)$

equal length. Anomaly can be observed in [600, 630] and [800, 830]. Accordingly, the anomaly scores of 6th and 8th feature patterns in Fig. 7b are larger than 1, while the other 10 anomaly scores are smaller than 1. Namely, the positive detection rate is 100% and the false alarm rate is 0%.

To further investigate the performance of MDFR-AD, additional experiments are conducted with an extended synthetic data set and seven real-world data sets. As described in the section "Real-world data sets", the numbers of data points contained in the seven real-world data sets R1 $\sim$ R7 are 3571, 2190, 4412, 9638, 7250, 16,000, and 21,600, respectively. The original $Y(t)$ specified by (24) contains 1200 data points. To make better use of the above real-world data sets, we extended the length of the original $Y(t)$ specified by (24) to 2190, which is the length of data set R2. For the other six real-world data sets, different segments of data points with the length of 2190 are extracted to conduct multiple experiments for average values. $Y_e(t)$ is directly used as a whole, as well as data set R2. In specific, the extended $Y(t)$ is denoted as

$$Y_e(t) = Y(t) + e_a(t) + e_d(t), \tag{27}$$

where $t = 1, 2, \ldots, 2190$. For $t \in [1, 1200]$, the values of $Y_e(t)$ are the same as $Y(t)$. For $t \in [1201, 2190]$, we introduce eight abnormal patterns and three distracters. Specifically, the eight abnormal patterns are

$$e_a(t) = \begin{cases} \frac{i+1}{20} \cdot \sin\left(\frac{40\pi}{K} t\right), t \in [1200+60(i-1), 1230+60(i-1)], \\ 0, \qquad\qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases}, \tag{28}$$

where $i = 1, 2, \ldots, 8$. The three distracters are

$$e_d(t) = \begin{cases} n_2(t), & t \in [1200+60(i-1), 1230+60(i-1)] \\ 0, & \text{otherwise} \end{cases}, \tag{29}$$

where $n_2(t) \sim N(0, 0.1 + \frac{i-8}{20})$ and $i = 8, 9, 10$.

By (28) and (29), it is obvious that the above eight abnormal patterns and three distracters are not overlapped with each other.

The experimental results are illustrated in terms of precision, recall rate, and F1-measure

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\%, \tag{30}$$

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\%, \tag{31}$$

$$F_1 = \frac{2 \times PR}{P + R} \times 100\%, \tag{32}$$

where TP is the rate of correct identification of anomaly, FN is the rate of missed identification of anomaly, and FP is the rate of false alarm of anomaly.

As shown in Fig. 8a, the performance of PAA is worse than PLAA and MDFR-AD. For PLAA and MDFR-AD, the performance of PLAA is better than that of MDFR-AD for data sets Y, R1, and R2. On the contrary, the performance of MDFR-AD is better than that of PLAA for data sets R3, R4, R5, R6, and R7. As shown in Fig. 8b, the performance of MDFR-AD is superior to PAA and PLAA. The main reason is that MDFR-AD is able to fit the raw time series more accurately than PAA and PLAA. The experimental results of both precision and recall rate indicate that MDFR-AD and PLAA are superior to PAA. In addition, the recall rate of MDFR-AD is far better than PLAA.

The performance of PAA, PLAA, and MDFR-AD, the F1-Measure scores of the above three methods are depicted in Fig. 9. As shown in Fig. 9, it is obvious that our proposal is better than both PAA and PLAA with respect to the eight data sets in our experiments.

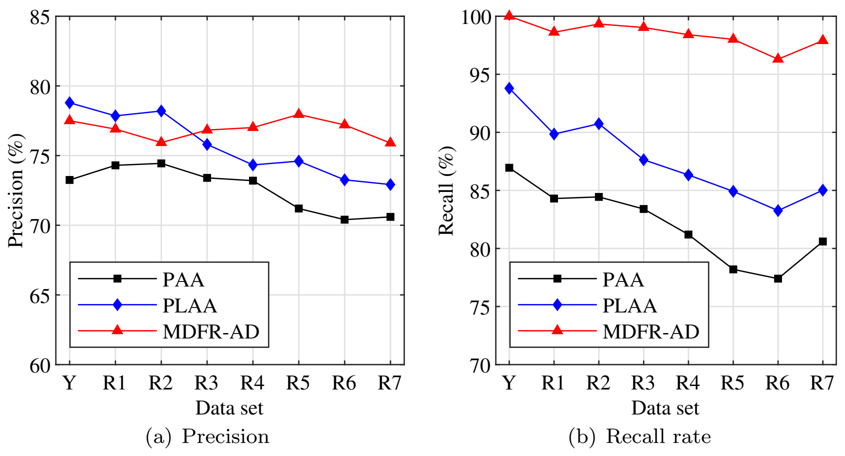


Fig. 8 Comparison of precision and recall rate

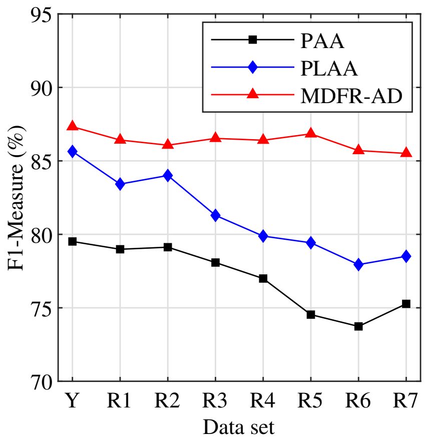


Fig. 9 Comparison of F1-measure score

Conclusion and future work

This paper proposed an edge-cloud collaboration architecture for pattern anomaly detection of time series in wireless sensor networks. A time series with long-term correlation is allocated to the cloud for anomaly detection. On the contrary, a time series with short-term correlation is assigned to edge node for anomaly detection. In addition, when considerable anomaly detection tasks are queuing up at resource-constrained edge nodes, certain tasks which have to wait for a long time are migrated to the cloud. The proposed multi-dimensional feature representation is able to perform an efficient fitting for a raw time series with small amount of computation. Our sliding window achieves accurate trend identification and feature point extraction. The fitting result is used to conduct pattern anomaly detection with an improved kernel density estimation method. Simulation results show that our model possesses satisfactory detection efficiency and quick responsiveness. However, the migration of anomaly detection task reckons without unknown correlation among time series in wireless sensor networks. Further development of our model also deserves testing on more real-world data sets.

Acknowledgements This work is partly supported by the Science and Technology Project of Shaanxi Province, China (Grant No. 2019ZDLGY07-08), the International Science and Technology Cooperation Program of the Science and Technology Department of Shaanxi Province, China (Grant No. 2018KW-049), the Communication Soft Science Program of Ministry of Industry and Information Technology, China (Grant No.2019-R-29), the Special Scientific Research Program of Education Department of Shaanxi Province, China (Grant No. 17JK0711), and the Special Funds for Construction of Key Disciplines in Universities of Shaanxi Province. The authors would like to thank the anonymous reviewers whose comments and suggestions greatly helped us to improve the quality and presentation of this paper.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Alam S, De D (2019) Bio-inspired smog sensing model for wireless sensor networks based on intracellular signalling. Inf Fusion 49:100–119
2. Chen S, Zhang S, Zheng X, Ruan X (2019) Layered adaptive compression design for efficient data collection in industrial wireless sensor networks. J Netw Comput Appl 129:37–45

3. Suraiya T, Shaista F (2019) Wireless sensor networks for healthcare monitoring: a review. In: Proceedings of the international conference on inventive computation technologies. Springer, pp 669–676
4. Thomas D, Shankaran R, Orgun M, Hitchens M, Ni W (2019) Energy-efficient military surveillance: coverage meets connectivity. IEEE Sens J 19(10), 3902–3911
5. Shahid N, Naqvi IH, Qaisar SB (2015) Characteristics and classification of outlier detection techniques for wireless sensor networks in harsh environments: a survey. Artif Intell Rev 43(2), 193–228
6. Gil P, Martins H, Januário F (2019) Outliers detection methods in wireless sensor networks. Artif Intell Rev 52(4), 2411–2436
7. Tayeh GB, Makhoul A, Laiymani D, Demerjian J (2018) A distributed real-time data prediction and adaptive sensing approach for wireless sensor networks. Pervasive Mob Comput 49:62–75
8. Habeeb RAA, Nasaruddin F, Gani A, Hashem IAT, Ahmed E, Imran M (2019) Real-time big data processing for anomaly detection: a survey. Int J Inf Manag 45:289–307
9. Chunyong Y, Sun Z, Jin W, Neal NX (2020) Anomaly detection based on convolutional recurrent autoencoder for iot time series. IEEE Trans Syst Man Cybern Syst. DOI: 10.1109/TSMC.2020.2968516
10. Cook AA, Mısırlı G, Fan Z (2019) Anomaly detection for iot time-series data: a survey. IEEE Internet Things J 7(7), 6481–6494
11. Bao Y, Tang Z, Li H, Zhang Y (2019) Computer vision and deep learning-based data anomaly detection method for structural health monitoring. Struct Health Monit 18(2), 401–421
12. Tang Z, Chen Z, Bao Y, Li H (2019) Convolutional neural network-based data anomaly detection method using multiple information for structural health monitoring. Struct Control Health Monit 26(1):e2296
13. Au Jay FK, Yeung ZW, Chan KY, Lau HYK, Yiu K-FC (2020) Jump detection in financial time series using machine learning algorithms. Soft Comput 24(3), 1789–1801
14. Raymond STL (2020) Time series chaotic neural oscillatory networks for financial prediction. In: Quantum finance. Springer, pp 301–337
15. Ankush M, Christian H (2017) Anomaly detection in industrial networks using machine learning: a roadmap. In: Machine learning for cyber physical systems. Springer, pp 65–72
16. Susto GA, Terzi M, Beghi A (2017) Anomaly detection approaches for semiconductor manufacturing. Proc Manuf 11:2018–2024
17. Tajiri K, Ikeda Y, Kawahara R, Shinkuma R (2018) A study on machine learning based anomaly detection for a large network with cooperative edge-cloud computing. In: IEICE Tech. Rep. SR2018-46, vol 118. IEICE, pp 119–120
18. Ferrari P, Rinaldi S, Sisinni E, Colombo F, Ghelfi F, Maffei D, Malara M (2019) Performance evaluation of full-cloud and edge-cloud architectures for industrial iot anomaly detection based on deep learning. In: 2019 II workshop on metrology for Industry 4.0 and IoT (MetroInd4.0 & IoT). IEEE, pp 420–425
19. Samir A, Pahl C (2019) Anomaly detection and analysis for clustered cloud computing reliability. In: 10th international conference on cloud computing, GRIDs, and virtualization. IARIA, pp 110–119
20. Shengjie X, Yi Q, Qingyang HR (2020) Data-driven edge intelligence for robust network anomaly detection. IEEE Trans Netw Sci Eng 7(3), 1481–1492
21. Pallasch C, Wein S, Hoffmann N, Obdenbusch M, Buchner T, Waltl J, Brecher C (2018) Edge powered industrial control: concept for combining cloud and automation technologies. In: 2018 IEEE international conference on edge computing. IEEE, pp 130–134
22. 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
23. Cauteruccio F, Fortino G, Guerrieri A, Liotta A, Mocanu DC, Perra C, Terracina G, Vega MT (2019) Short-long term anomaly detection in wireless sensor networks based on machine learning and multi-parameterized edit distance. Inf Fusion 52:13–30
24. Hansheng R, Bixiong X, Yujing W, Chao Y, Congrui H, Xiaoyu K, Tony X, Mao Y, Jie T, Qi Z (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
25. Li J, Pedrycz W, Jamal I (2017) Multivariate time series anomaly detection: a framework of hidden Markov models. Appl Soft Comput 60:229–240
26. Talagala PD, Hyndman RJ, Smith-Miles K, Kandanaarachchi S, Mu noz MA, (2020) Anomaly detection in streaming nonstationary temporal data. J Comput Graph Stat 29(1):13–27
27. Mou W, Tan L (2017) An adaptive distributed parameter estimation approach in incremental cooperative wireless sensor networks. AEU-Int J Electron Commun 79:307–316
28. Liu M, Huang M, Tang W (2017) A hybrid algorithm for mining local outliers in categorical data. Int J Wirel Mob Comput 13(1), 78–85
29. Chandel K, Kunwar V, Sabitha S, Choudhury T, Mukherjee S (2016) A comparative study on thyroid disease detection using k-nearest neighbor and naive Bayes classification techniques. CSI Trans ICT 4(2–4), 313–319
30. Chuxu Z, Dongjin S, Yuncong C, Xinyang F, Cristian L, Wei C, Jingchao N, Bo Z, Haifeng C, Nitesh VC (2019) A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. Proceedings of the AAAI conference on artificial intelligence 33:1409–1416
31. Watson J, Raj MS, Shamik S (2019) Anomaly detection using supervised learning and multiple statistical methods. In: Proceedings of the 2019 18th IEEE international conference on machine learning and applications (ICMLA). IEEE, pp 1291–1297
32. Erica R, Markus W, Martin A (2019) A computational framework for interpretable anomaly detection and classification of multivariate time series with application to human gait data analysis. In: Proceedings of the artificial intelligence in medicine: knowledge representation and transparent and explainable systems. Springer, pp 132–147
33. Adam K, Paweł K (2019) Fuzzy approach for detection of anomalies in time series. In: Proceedings of the international conference on artificial intelligence and soft computing. Springer, pp 397–406
34. Karunasingha DSK, Liong S-Y (2018) Enhancement of chaotic hydrological time series prediction with real-time noise reduction using extended Kalman filter. J Hydrol 565:737–746
35. Len F, Vincent V, Boris C, Wannes M, Bart G (2019) Pattern-based anomaly detection in mixed-type time series. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, pp 240–256
36. Shi W, Sun H, Cao J, Zhang Q, Liu W (2017) Edge computing-an emerging computing model for the internet of everything era. J Comput Res Dev 54(5), 907–924
37. Zhang Q, Yupeng H, Ji C, Zhan P, Li X (2018) Edge computing application: real-time anomaly detection algorithm for sensing data. J Comput Res Dev 55(3), 524–536
38. Wang T, Ke H, Zheng X, Wang K, Sangaiah AK, Liu A (2019) Big data cleaning based on mobile edge computing in industrial sensor-cloud. IEEE Trans Ind Inform 16(2), 1321–1329
39. Aymen A, Abdennaceur K, Adel M (2016) Anomaly detection through outlier and neighborhood data in wireless sensor networks. In: Proceedings of the 2016 2nd international conference on advanced technologies for signal and image processing (ATSIP). IEEE, pp 26–30
40. Kim PT, Truong TH et al (2017) Data driven hyperparameter optimization of one-class support vector machines for anomaly detection in wireless sensor networks. In: Proceedings of the 2017

international conference on advanced technologies for communications (ATC). IEEE, pp 6–10
41. Miaomiao Z, Dechang PI (2017) A novel method for fast and accurate similarity measure in time series field. In: Proceedings of the 2017 IEEE international conference on data mining workshops (ICDMW). IEEE, pp 569–576
42. Walden AT, Leong Z (2018) Tapering promotes propriety for Fourier transforms of real-valued time series. IEEE Trans Signal Process 66(17), 4585–4597
43. Xia Y, He Y, Wang K, Pei W, Blazic Z, Mandic DP (2015) A complex least squares enhanced smart dft technique for power system frequency estimation. IEEE Trans Power Deliv 32(3), 1270–1278
44. Ni M, Zhao BW, Fusheng FP (2018) Crebad: chip radio emission based anomaly detection scheme of iot devices. J Comput Res Dev 55(7):1451
45. Zhong J, Huang Y (2010) Time-frequency representation based on an adaptive short-time Fourier transform. IEEE Trans Signal Process 58(10), 5118–5128
46. Kongchang D, Zhao Y, Lei J (2017) The incorrect usage of singular spectral analysis and discrete wavelet transform in hybrid models to predict hydrological time series. J Hydrol 552:44–51
47. Pannakkong W, Sriboonchitta S, Huynh V-N (2018) An ensemble model of arima and ann with restricted Boltzmann machine based on decomposition of discrete wavelet transform for time series forecasting. J Syst Sci Syst Eng 27(5), 690–708
48. Walden AT, Contreras CA (1998) Matching pursuit by undecimated discrete wavelet transform for non-stationary time series of arbitrary length. Stat Comput 8(3), 205–219
49. Sidratul M, Andreas G, Michael S, Sascha S, von Spiczak S, Ulrich S, Theodor M, Thomas M (2019) Svd square-root iterated extended Kalman filter for modeling of epileptic seizure count time series with external inputs. In: Proceedings of the 2019 41st annual international conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE, pp 616–619
50. Peipei X, Wenjie R, Quan ZS, Tao G, Lina Y (2017) Interpolating the missing values for multi-dimensional spatial-temporal sensor data: a tensor svd approach. In: Proceedings of the 14th EAI international conference on mobile and ubiquitous systems: computing, networking and services, pp 442–451
51. Márquez-Grajales A, Acosta-Mesa H-G, Mezura-Montes E, Graff M (2020) A multi-breakpoints approach for symbolic discretization of time series. Knowl Inf Syst 62(7), 2795–2834
52. Le Nguyen T, Gsponer S, Ilie I, O'Reilly M, Ifrim G (2019) Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations. Data Min Knowl Discov 33(4), 1183–1222
53. Virani N, Jha DK, Ray A, Phoha S (2019) Sequential hypothesis tests for streaming data via symbolic time-series analysis. Eng Appl Artif Intell 81:234–246
54. Jessica L, Eamonn K, Stefano L, Bill C (2003) A symbolic representation of time series, with implications for streaming algorithms. In: Proceedings of the 8th ACM SIGMOD workshop on research issues in data mining and knowledge discovery. ACM, pp 2–11
55. Battuguldur L, Yu S, Kyoji K (2006) New time series data representation esax for financial applications. In: Proceedings of the 22nd international conference on data engineering workshops (ICDEW'06). IEEE, pp 17–22
56. Chaw TZ, Hayato Y (2016) An improved symbolic aggregate approximation distance measure based on its statistical features. In: Proceedings of the 18th international conference on information integration and web-based applications and services. ACM, pp 72–80
57. Chen H, Jinghan D, Zhang W, Li B (2020) An iterative end point fitting based trend segmentation representation of time series and its distance measure. Multimed Tools Appl 79(19), 13481–13499
58. Ge L, Ke Y, Siu-Wing C, Zhenguo L, Wei F, Cheng H, Yadong M (2015) Piecewise linear approximation of streaming time series data with max-error guarantees. In: Proceedings of the 2015 IEEE 31st international conference on data engineering. IEEE, pp 173–184
59. Gang H, Xiaofeng Z (2016) A piecewise linear representation method of hydrological time series based on curve feature. In: Proceedings of the 2016 8th international conference on intelligent human-machine systems and cybernetics (IHMSC), vol 2. IEEE, pp 203–207
60. Jing W, Haibin Y, Qicai W, Rong L, Juan S (2015) A piecewise linear representation based on compression ratio. In: Proceedings of the 2015 prognostics and system health management conference (PHM). IEEE, pp 1–5
61. Jianshu S, Yuansheng L, Feng Y (2017) Research on anomaly pattern detection in hydrological time series. In: Proceedings of the 2017 14th web information systems and applications conference (WISA). IEEE, pp 38–43
62. Mai VH, Luong CM et al (2016) Pattern discovery in the financial time series based on local trend. In: Proceedings of the international conference on advances in information and communication technology. Springer, pp 442–451
63. Fei H, Alexandre S, Kondo HA, Zhouhang W (2018) Bearings degradation monitoring indicator based on segmented hotelling t square and piecewise linear representation. In: Proceedings of the 2018 IEEE international conference on mechatronics and automation (ICMA). IEEE, pp 1389–1394
64. Yupeng H, Cun J, Ming J, Yiming D, Shuo K, Xueqing L (2016) A continuous segmentation algorithm for streaming time series. In: Proceedings of the international conference on collaborative computing: networking, applications and worksharing. Springer, pp 140–151
65. Sun Y, Li Z (2015) Clustering algorithm for time series based on locally extreme point. Comput Eng 41(5), 33–37
66. Cun J, Shijun L, Chenglei Y, Lei W, Li P, Xiangxu M (2016) A piecewise linear representation method based on importance data points for time series data. In: Proceedings of the 2016 IEEE 20th international conference on computer supported cooperative work in design (CSCWD). IEEE, pp 111–116
67. ChaoHong M, XiaoQing W, ZhongNan S (2017) Early classification of multivariate time series based on piecewise aggregate approximation. In: Proceedings of the international conference on health information science. Springer, pp 81–88
68. Ding X, Shengjian Y, Wang M, Wang H, Gao H, Yang D (2020) Anomaly detection on industrial time series based on correlation analysis. J Softw 32(3), 726–747
69. Octavian LH, Rodica P (2017) Time series—a taxonomy based survey. In: Proceedings of the 2017 13th IEEE international conference on intelligent computer communication and processing (ICCP). IEEE, pp 231–238
70. McLeod AI, Hipel KW (1978) Preservation of the rescaled adjusted range: 1. A reassessment of the Hurst phenomenon. Water Resour Res 14(3):491–508
71. Yanping C, Ping Y, Cong G, Zhongmin W, Zhong Y (2019) A skeleton pattern representation method for anomaly detection in wireless sensor networks. In: Proceedings of the 2019 IEEE 21st international conference on high performance computing and communications; IEEE 17th international conference on smart city; IEEE 5th international conference on data science and systems (HPCC/SmartCity/DSS). IEEE, pp 1833–1838
72. Dau HA, Bagnall A, Kamgar K, Yeh C-CM, Zhu Y, Gharghabi S, Ratanamahatana CA, Keogh E (2019) The ucr time series archive. IEEE/CAA J Autom Sin 6(6), 1293–1305
73. Jäncke L, Kühnis J, Rogenmoser L, Elmer S (2015) Time course of eeg oscillations during repeated listening of a well-known aria. Front Hum Neurosci 9:401

74. Rodrigo DT, Arturo SB, Aquiles R (2015) Nontechnical losses detection: a discrete cosine transform and optimum-path forest based approach. In: Proceedings of the 2015 North American power symposium (NAPS). IEEE, pp 1–6
75. Rajesh KN, Dhuli R (2017) Classification of ecg heartbeats using nonlinear decomposition methods and support vector machine. Comput Biol Med 87:271–284
76. Min H, Ji Z, Yan K, Guo Y, Feng X, Gong J, Zhao X, Dong L (2018) Detecting anomalies in time series data via a meta-feature based approach. IEEE Access 6:27760–27776
77. Ucr time series classification archive. http://www.cs.ucr.edu/eamonn/time_series_data/. Online. Accessed 18 Mar 2021
78. Uc irvine machine learning repository. https://archive.ics.uci.edu/ml/datasets/Air+Quality/. Online. Accessed 18 Mar 2021
79. Heqing H, Xiaofang L, Gary JB, Carola C (2016) Hydrology of the yellow river source zone. In: Proceedings of the landscape and ecosystem diversity, dynamics and management in the Yellow River Source Zone. Springer, pp 79–99
80. Uc irvine machine learning repository. https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption. Online. Accessed 18 Mar 2021
81. National road database. https://www.vegvesen.no/en/professional/roads/national-road-database. Online. Accessed 18 Mar 2021
82. https://www.debian.org/. Online. Accessed 18 Mar 2021
83. https://www.freertos.org/. Online. Accessed 18 Mar 2021
84. Jiménez-Valverde A (2012) Insights into the area under the receiver operating characteristic curve (auc) as a discrimination measure in species distribution modelling. Glob Ecol Biogeogr 21(4), 498–507
85. Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Dimensionality reduction for fast similarity search in large time series databases. Knowl Inf Syst 3(3), 263–286
86. Nguyen QVH, Duong TA (2008) An improvement of paa for dimensionality reduction in large time series databases. In: Proceedings of the Pacific Rim international conference on artificial intelligence. Springer, pp 698–707