IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 8 January 2024, accepted 30 January 2024, date of publication 2 February 2024, date of current version 12 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3361672

RESEARCH ARTICLE

Indicator Fault Detection Method Based on Periodic Self Discovery and Historical Anomaly Filtering

SHENG WU[1,2] AND JIHONG GUAN[1]
[1]College of Electronic Information and Engineering, Tongji University, Shanghai 201804, China
[2]ICBC Data Center, Shanghai 200131, China

Corresponding author: Jihong Guan (jhguan@tongji.edu.cn)

ABSTRACT Data centers' information systems typically encompass a variety of operational objects including applications, systems, networks, and devices, which generate a large volume of indicator data during operation. The traditional threshold-based indicator fault detection method has struggled to adapt to the massive and heterogeneous nature of distributed architectures, resulting in numerous false positives and negatives. Existing research has limitations in terms of the accuracy of fitting indicator operating characteristics, as well as performance. To address the problem of fault detection for indicator data, this article proposes a method based on periodic self-discovery and historical anomaly filtering. It achieves periodic self-discovery based on Fourier transformation, significantly reducing the training cost such as model parameter tuning. Additionally, it introduces a quadratic fuzzy filter based on periodicity to effectively solve possible local misclassification issues while generating a baseband that is more suitable for indicator operating characteristics, improving detection accuracy. Through experimental validation, the fault detection method proposed in this article significantly improves the accuracy of indicator for operational objects. Compared to directly using ARIMA and SARIMA models, this method has improved the MSE by up to 44% and 36%, respectively. This method has also been practically applied in multiple scenarios in a bank's production environment, increasing alert accuracy by 20% and achieving a ratio of 10:1 for alert convergence. Compared to traditional methods, it significantly enhances the ability of fault detection for indicators.

INDEX TERMS Fault detection, periodic self discovery, historical anomaly filtering, dynamic baseline, indicator.

I. INTRODUCTION

With the rapid development of business and the transformation of IT architecture towards distributed, the scale of operation and maintenance objects is showing an explosive growth trend. In the monitoring system [1] of the data center, anomaly detection based on time series indicators covers top-down monitoring of application transactions and monitoring of basic components such as databases, middleware, storage, networks, and operating systems. The demand for refined services in upper-level businesses drives the development of refined operations and maintenance in the lower-level. The granularity of controlled objects is becoming smaller and the sampling intervals of monitoring are becoming shorter, resulting in performance indicator data with characteristics such as massive volume, diversity, personalization, and rapid generation. Realizing fast, accurate, and refined detection of abnormal performance fluctuations is an important guarantee for the stability of business operations and maintenance [2], [3], [4]. Traditional indicator monitoring methods based on fixed thresholds do not support the fine-grained setting of various indicators, resulting in insufficient monitoring accuracy. Conventional machine learning methods [5] are difficult to effectively fit the operational characteristics of various indicators, and there are still shortcomings in accuracy, robustness, and performance. this paper

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Tedesco.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

proposes an indicator fault detection method based on periodic self-discovery and historical anomaly filtering, which significantly improves the algorithm parameter adaptation, robustness, and algorithm performance compared to other methods. This method combines multiple machine learning models to achieve periodic self-discovery based on Fourier transform, and adopts the isolated forest method based on hyperspace partitioning as a preliminary screening method for anomaly detection to quickly identify global anomalies. It has the advantages of low time complexity, minimal dependence on manual tuning parameters, and good algorithm stability. On this basis, a quadratic fuzzy filter based on periodicity is introduced to refine the screening process while avoiding phase noise interference caused by operational deviations, effectively solving possible local misclassifications. This ensures real-time model updates, with good self-learning and self-adaptive capabilities, resulting in a baseband that is more suitable for the operational characteristics of the indicator, improving detection accuracy.

II. RESEARCH BACKGROUND

A. THE CURRENT RESEARCH STATUS

Commonly used indicator fault detection methods can be divided into supervised learning and unsupervised learning methods based on whether the data has abnormal labels. In data center operation and maintenance monitoring, the original indicator data is massive and difficult to manually label for anomalies, so unsupervised learning methods are often used. The indicator fault detection methods for unsupervised learning [6], [7] include statistical methods [8], [9], distance-based methods [10], [11], [12], density-based methods [13], [14], and partition-based methods [15]. In recent years, deep learning-based fault detection methods [16] have become an academic hotspot, with strong abilities in extracting complex features. Common models include RNN [17], LSTM [18], VAE [19], GAN [20], [21], [22], and Transformer [23]. Focusing on the pain point of sparse fault data, literatures [24], [25] applied VAE (Variable AutoEncoder) and its improved network models to fault detection. Literature [25] proposed an anomaly detection method Bagel based on CVAE (Conditional Variable AutoEncoder), which can support time information (timestamps or seasonal characteristics). Aiming at the fault detection tasks of server underlying indicators such as input-output requests, literature [24] proposed a VAE anomaly detection method Buzz based on adversarial training and data partitioning. Literature [26] proposed an unsupervised learning method SaVAE-SR (Self-adversarial Variational Autoencoder with Spectral Residual) to address the pollution problem of normal pattern learning caused by abnormal points and improve the accuracy of anomaly detection. Literature [27] proposed a reinforcement learning-based method PTAD (Policy-based Time series Anomaly Detector) to solve the problem that previous data needs to be based on specific domains and strong assumptions, which cannot be adapted to real data, and improve the recall and accuracy of anomaly detection. The comparison of various detection methods is shown in Table 1.

TABLE 1. Comparison of common indicator fault detection methods.

Method Type	Method Core Idea	Problems
Fault Detection Based on Statistics	Assuming that the given data follows a certain distribution, using inconsistency testing can detect outliers	In practical applications, performance data does not conform to any mathematical distribution of ideal states
Fault Detection Based on Distance	Identify abnormal data by searching for relatively isolated data points within the neighborhood	High computational complexity, high algorithm sensitivity, and unstable results
Fault Detection Based on Density	The density around the outlier is significantly different from the density of its neighboring data points	Can effectively identify local outliers, but has high computational complexity and difficulty in joint parameter tuning
Fault Detection Based on Partitioning	Circular partitioning (isolation) of samples isolates normal samples from abnormal ones, making abnormal data more likely to be isolated than normal data	The algorithm has good stability and can quickly detect global anomalies, but is prone to local misjudgments
Fault Detection Based on Deep Learning	Strong ability to extract deep and complex features in time series, skilled in long-term prediction of multi-dimensional time series variables	The training phase has a high resource cost and is too complex for single dimensional indicators, making it more suitable for high-dimensional indicators

It can be seen that there are still gaps in detection accuracy, algorithm performance, and other aspects of the above research that need to be further addressed.

B. ISSUES AND CHALLENGES

The commonly used methods for indicator fault detection include traditional fixed threshold method, deep learning approaches, and unsupervised machine learning methods.

The fixed threshold method lacks the ability to adapt to the varying operational states of different indicators, leading to inaccurate monitoring.

Deep learning methods, a research focus in recent years, are not suitable for the single-dimensional indicator fault detection scenario considered in this article. The reasons are as follows:

High data and model requirements: Deep learning requires a large amount of data for training, making it unsuitable for small data set scenarios. Additionally, the structure and parameter selection of deep learning models have a significant impact on the results, requiring fine parameter tuning and model design.

High computational complexity: The computational complexity of deep learning models is relatively high, requiring high-performance computers or GPUs for training and inference. This makes it unsuitable for real-time monitoring scenarios with high operational requirements.

Poor interpretability: The black-box nature of deep learning models makes their results difficult to interpret.

Difficulty leveraging model advantages in single-dimensional scenarios.

Unsupervised machine learning methods are more suitable for the scenario considered in this article. However, current approaches [28] still suffer from the following issues:

Difficulty in parameter tuning. To extract features specifically for different systems and objects, algorithms require significant parameter adjustment in application.

Sensitivity to noise. Historical outliers can interfere with current detection, reducing algorithm accuracy and leading to false alarms.

Insufficient algorithm performance. It is challenging to support large-scale indicator detection due to the requirement for separate modeling of different indicators in practical applications. This involves substantial model training, becoming a performance bottleneck when using machine learning methods. This leads to difficulties in large-scale promotion and application.

In view of the above technical difficulties, the new model method needs to achieve the following breakthroughs:

1) THE PARAMETERS OF THE ALGORITHM ARE SELF-ADAPTIVE

The algorithm can be automatically adapted according to the operation characteristics of the indicators, without manual parameter adjustment, so as to improve the generalization of the algorithm for different indicators. For example, the indicators reflecting different business operation laws show different periodicity, and the algorithm automatically and accurately identifies and sets the periodicity parameters.

2) THE ROBUSTNESS OF THE ALGORITHM IS IMPROVED

The algorithm can automatically eliminate the interference of noise data such as historical burrs during operation.

3) THE ALGORITHM PERFORMANCE IS IMPROVED

the training cost of single index is very low, and considering the massive nature of the index, the big data technology stack should be used for the engineering implementation of the algorithm.

III. MODEL ARCHITECTURE

Aiming at the problem of pain points in the field of index analysis and detection, this paper proposes an index fault detection method based on periodic self discovery and historical anomaly filtering to achieve parameter adaptation, improve robustness and performance. The model architecture is shown in Figure 1.

It mainly includes three modules:

A. PERIODIC SELF DISCOVERY

This module uses historical transaction monitoring data to automatically learn the cycle parameters of each transaction monitoring data to realize the mining of historical data characteristics. This means that the system can independently discover the cyclical rules contained in different transaction data without relying on prior knowledge or manually setting the cycle parameters. This automatic learning process helps to extract important features of transaction data and provides the basis for subsequent anomaly detection.

B. HISTORICAL ANOMALY FILTERING

In this module, the system automatically labels the abnormal values in the historical transaction data and filters the abnormal fluctuation data. Such preprocessing steps are crucial to improve the accuracy of prediction. By removing the abnormal interference in the historical data, the system can more accurately capture the real abnormal situation in future transactions, so as to improve the reliability of anomaly detection.

C. DYNAMIC BASELINE DETECTION

This module relies on the output of the first two modules, that is, the data after periodic self discovery and historical anomaly filtering. Based on these data and corresponding periodic parameters, the system uses time series algorithm to generate dynamic baseline. Dynamic baseline is a reference line that can be adjusted in real time with time for real-time detection. By comparing with the dynamic baseline, the system can quickly capture abnormal fluctuations in transaction data and respond in time.

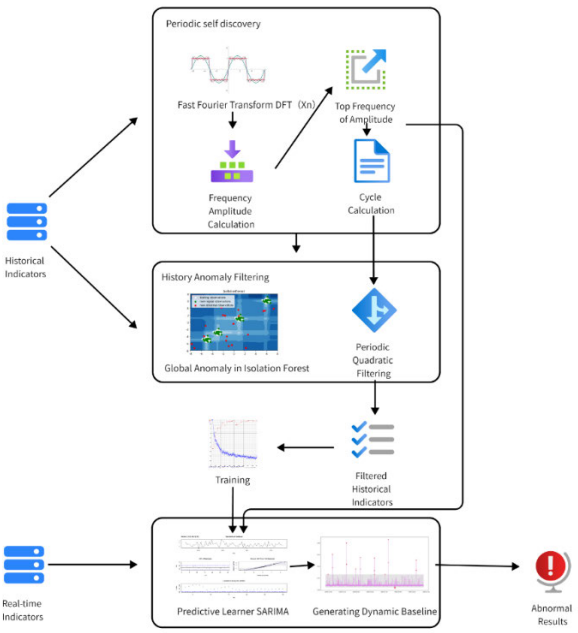

FIGURE 1. Model architecture.

IV. PERIODIC SELF DISCOVERY

Index dynamic baseline detection is a kind of time series prediction. The common methods include autoregression, moving average, ARIMA (auto regression integrated moving

average) [29], prophet [30] and other models. Now there are relatively mature software packages or tools, but the use of these models need to set cycle parameters, such as week, month, year or holiday. Taking the banking system as an example, the different indicators of individual and corporate business systems reflect their business characteristics, such as quarterly interest settlement, monthly credit card repayment, corporate large amount payment, etc., showing different periodicity. When facing the indicators of different business systems, the above algorithm needs to manually set the cycle and a lot of parameter adjustment work, so that the model can adapt to the operating characteristics of the indicators, which is time-consuming and labor-consuming, affecting the generalization of the model. The cycle self discovery module realizes the self discovery and self adaptation of the cycle parameters of the model, greatly reduces the work of parameter adjustment, and compresses the cost of model training.

For performance indicators, the change trend usually has a certain cyclical rule, which is often related to the change of the business load it carries. The operation and maintenance index data is often represented as discrete signals, so discrete Fourier transform (DFT) [31] can be used for processing. DFT can convert time series data from the time domain to the frequency domain. The fundamental idea is to decompose the time series into multiple sinusoidal and cosinusoidal waves of different frequencies, and analyze these waves in the frequency domain to reveal periodic patterns and frequency components in the time series.

The DFT function expression is as follows:

$$X(k) = DFT[x(n)] = \frac{1}{N}\sum_{n=0}^{N-1} x(n)(\cos(\frac{2\pi kn}{N}) - i\sin(\frac{2\pi kn}{N})) = \sum_{n=0}^{N-1} \cdot e^{-i\frac{2\pi kn}{N}} \quad (1)$$

$$ReX\ (k) = \sum_{n=1}^{N-1} x(n)\cos(\frac{2\pi kn}{N}) \quad (2)$$

$$ImX\ (k) = -\sum_{n=0}^{N-1} x(n)\sin(\frac{2\pi kn}{N}) \quad k = 0, 1, n \ldots, N-1 \quad (3)$$

Specifically, each data point in a time series is considered as a vibrational vector, and these vectors accumulate in the time domain to form a complex vibrational shape. By decomposing this vibrational shape into the sum of multiple sinusoidal and cosinusoidal waves, we can obtain the amplitude of each frequency component. These amplitudes represent the contribution degree of each frequency component in the time series in the frequency domain. The result of DFT is usually presented as a spectrum of amplitude, which displays the amplitude of each frequency component. After extracting the amplitude spectrum, we need to perform frequency analysis to identify the frequency components with the largest amplitude and the corresponding periods. These frequencies and periods represent the main periodic patterns in the time series.

In practical applications, to achieve efficient computation, fast Fourier transform (FFT) is commonly used to compute DFT. After converting the time domain signal x(n) into the frequency domain signal via FFT, the amplitude of each frequency component can be calculated and arranged in descending order. Among them, the first few frequencies with the largest amplitude are the fundamental and key harmonic frequencies fi that we are concerned about. By calculating the period Ti = 1/fi, the corresponding period can be obtained.

Through the aforementioned process, the periodic self-discovery module can effectively analyze the periodic characteristics of performance index data, thereby adaptively discovering periodic parameters without manual intervention, reducing adjustment efforts, optimizing model training costs, and improving model generalization performance.

The novel features of this method include:

A. FREQUENCY DOMAIN ANALYSIS

By converting time series data from the time domain to the frequency domain, the periodic patterns and frequency components of the data can be more intuitively revealed, supporting quantitative analysis. By comparing the amplitudes of different frequency components, the main periodic patterns in the time series can be further analyzed and mined.

B. NO PRESET PERIOD

Traditional periodicity detection methods usually require presetting a period value, while Fourier transform does not require preset period. By performing Fourier transform on the entire time series, we can obtain the amplitude spectrum of all frequency components, which contains all periodic information in the time series.

C. EFFICIENT COMPUTATION

Especially in the implementation of FFT, the algorithm complexity is reduced to O(N log N), supporting large-scale data processing.

V. HISTORICAL ANOMALY FILTERING

First, the historical performance data is perused, and the isolated forest [32] method based on division is applied to quickly identify the globally suspected abnormal data points. The isolated forest, an anomaly detection algorithm based on ensemble learning, effectively locates the isolated abnormal data points in the feature space without modeling the normal data, thereby enabling a prompt identification of potential abnormal data for subsequent processing.

The detection of global suspected abnormal data using isolation forest algorithm is mainly divided into three steps. The first step is data preprocessing, including converting time series data into two-dimensional form and normalization. The second step is to build isolation forest, which clusters the data

Algorithm 1 Periodic self discovery algorithm

INPUT: indicator number L;
Indicator set kpiset={kpi_0, kpi_1,…, kpi_l},
where KPI_ L is the L-th index vector;
Index interval ts;
Indicator time length T2 (T2 is suggested to be an integer power of 2, otherwise zero is added to the integer power of 2);
Number of data points in the set n2=t2/ts

OUTPUT: periodic value set tset={t_0, t_1,…, t_l}

```
tset={}
for kpivector in kpiset do
    FFT=Fast Fourier transform (kpivector)
    eng=frequency amplitude energy (kpivector)
    engmax=max (Eng)
    FM=frequency of fundamental wave and key harmonic (kpivoctor, engmax)
    tset.apex (FM)
end for
return tset;
```

through constructing a series of decision trees. When building the decision tree, the algorithm selects a feature to split until the termination condition is reached. Each leaf node of the decision tree represents a data point, and the path length of the leaf node can reflect the abnormality degree of the data point. The third step is to calculate the anomaly score: in isolation forest, the anomaly score of each data point is calculated by averaging the path lengths in all decision trees. The shorter the average path length, the easier it is for the data point to be isolated, so the higher the anomaly score. Typically, data points with anomaly scores greater than 0.5 are considered as outliers.

Subsequently, considering that the above suspected abnormal data points may be misjudged, further analysis and verification are needed to confirm whether they are truly abnormal. for these suspected abnormal data points, the cycle parameters are combined and a cycle-based secondary fuzzy filtering approach is implemented to eliminate periodic data points with significant normal fluctuations, obtaining the final abnormal data result set. The employment of cycle parameters aids in more accurately identifying periodic anomalies and preventing the misclassification of normal periodic fluctuations as anomalies.

Through the aforementioned preprocessing steps, the abnormal data germane to the dynamic baseline prediction of performance indicators is effectively screened out, and appropriate corrections are made to the historical data. This pretreatment process provides a more reliable foundation for subsequent anomaly detection and real-time monitoring, thereby optimizing the accuracy and precision of the model and enhancing the reliability of dynamic baseline predictions.

The novel features of this method include:

A. EFFICIENT AND FAST

Isolation Forest uses a random approach to construct the tree structure, avoiding the problem of traditional clustering or classification algorithms that require scanning all data, thereby improving computational efficiency.

B. HIGH ACCURACY

Isolation Forest uses a random approach to split sample points and detects outliers based on the depth of the sample and the isolation probability, making it more accurate compared to other algorithms. The two-pass filtering method compensates for the local misclassification problem of Isolation Forest and further improves accuracy.

VI. DYNAMIC BASELINE DETECTION

This module is used to generate dynamic baselines for anomaly detection. Considering that the indicator fault detection in this article is mainly for key indicators of operation and maintenance objects, its dimension is single, and deep learning and other indicator fault detection methods are more suitable for high-dimensional and complex feature data. As a classic method, dynamic baseline has the advantages of strong interpretability, low training overhead, and support for large-scale data detection. Therefore, dynamic baseline is selected for detection.

The module provides dynamic baseline detection function to achieve accurate personalized real-time anomaly detection. A classic method for forecasting periodic time series is SARIMA model [33] (seasonal auto regressive integrated moving average model), which can be adjusted by four integer parameters P, D, Q and S.

Where, P represents the auto regressive autoregressive part of the model and is used to represent the lag number of the time series data used in the prediction model; D represents the integrated integration part of the model, which is used to indicate that the time series data needs to be differentiated to be stable; Q represents the moving average part of the model, which is used to represent the lag number of the prediction error used in the prediction model; S stands for the moving average seasonal part of the model, which is used to represent the period of time series data, with month as the dimension. For example, if the period is quarter, the value of S is 4; If the period is year, then the value of S is 12, and so on.

However, the original algorithm cannot be directly used due to the possible problems of missing and abnormal historical data. Moreover, the period parameter must be an integer, that is, the period must be greater than the month. In the automatic discovery of performance data cycle, it is often found that the cycle is in hours and days. Therefore, we need to improve the original SARIMA model, reduce the impact of missing data and abnormal data in the modeling process, and introduce the historical impact attenuation factor to solve the problem of performance drift [34].

VII. EXPERIMENT AND ANALYSIS

A. EXPERIMENTAL ENVIRONMENT

10 servers: Intel (R) Xeon e5-2650v2 CPU@2.60GHz Memory 32GB; Operating system: CentOS 7.3; Three Hadoop standalone clusters are installed; Three deploy Kafka,

Algorithm 2 Historical exception filtering algorithm

INPUT: indicator number L;
Indicator set kpiset={kpi_0, kpi_1,…, kpi_l},
where KPI_ L is the L-th index vector;
The periodic value set tset={t_0, t_1,…, t_l};
Preset height value H;
Preset deviation M

OUTPUT: historical exception filtered set kpiset '={kpi_0', kpi_1',…, kpi_l'}

abnormal POINTSSet={}//the abnormal value data set is initialized to null
for kpivoctor in kpiset, Ti in tset do
  APLS=isolated forest (kpivoctor)
  abnormal points={}
  for aPL in APLS do
    if APL>H do
      make this point KPI_ XK, the period Ti of the indicator
      newkpivoctor=time TJ and neighborhood (TLJ, Tuj) corresponding to the previous P cycles of the point
      calculate KPI according to self discovery rule_ Reference mean MK and reference standard deviation sk of XK
      if KPI_ Difference between XK and MK>m∗sk, do
        abnormal points.append (kpi_xk)
    end if
  end if
  end for
  abnormal points set.apend (abnormal points)
end for
kpisets'=kpiset - abnormal POINTSSet// Calculate the filtered data set of historical exceptions
return kpisets

Algorithm 3 Dynamic baseline algorithm

INPUT: indicator number L;
The filtered set of historical exceptions kpiset '={kpi_0', kpi_1',…, kpi_l'};
The periodic value set tset={t_0, t_1,…, t_l};
Real time indicator set kpiset* ={kpi_0*, kpi_1*,…, kpi_l*}

OUTPUT: faultpointsset={kpif_0, kpif_1,…, kpif_l}

faultpointsset={}
for kpivoctor in kpiset, Ti in tset do
  model=SARIMA (kpivector, Ti)
  predictdata=model predictive value
  baselinedata=(predictdata+3 times standard deviation, predictdata-3 times standard deviation)
  if kpi∗ >baselinedata do//the real-time value exceeds the dynamic baseband range at that time
    make this point kpif_ X
    faultpointsset.append (kpif_x)
  end if
end for
return faultpointsset

two deploy InfluxDB, one deploy MongoDB, and one deploy model programs and front-end applications. See Table 2 for details.

TABLE 2. Deployment environment configuration.

Serial numbe	type	configuration	component deployment	version
1	virtual machine	16c/32g /500g	Hadoop (spark)	1.5
2	virtual machine	16c/32g /500g	Hadoop (spark)	1.5
3	virtual machine	16c/32g /500g	Hadoop (spark)	1.5
4	virtual machine	16c/32g /500g	InfluxDB	1.7.3
5	virtual machine	16c/32g /500g	InfluxDB	1.7.3
6	virtual machine	16c/32g /500g	MongoDB	4.2.8
7	virtual machine	16c/32g /500g	Kafka	2.11
8	virtual machine	16c/32g /500g	Kafka	2.11
9	virtual machine	16c/32g /500g	Kafka	2.11
10	virtual machine	16c/32g /500g	Model and Front-end Application	

In the stage of Engineering landing, the model uses the big data technology stack to adapt to the analysis of massive indicators. The technical architecture is shown in Figure 2 and Figure 3. In the training phase, the algorithm package is loaded from HDFS, the historical data of InfluxDB is read, and the model of each index is output based on batch fragmentation and saved to MongoDB; In the reasoning stage, load the indicator models of MongoDB, read Kafka real-time data, implement dynamic baseline detection, and save the results to InfluxDB.

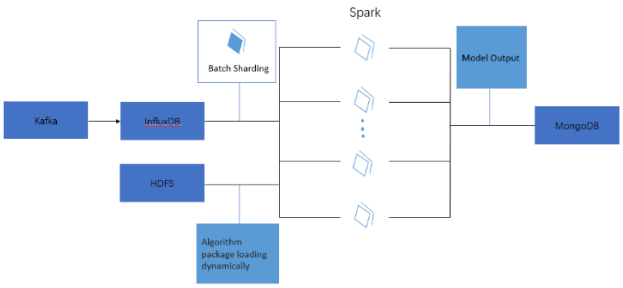

FIGURE 2. Model training based on batch processing.

B. DATA SET

The experiment consists of two data sets.

Dataset 1: QPS dataset of the test database used by a bank. The acquisition frequency is 1 minute. See table 3 for specific fields. The data size is 200, the training set is 150, and the test set is 50.

Dataset 2: is a public dataset [35], which is the golden indicator of business operation under the distributed micro service architecture. The collection frequency is 2 minutes. See Table 4 for specific fields. The data size is 1000, which is

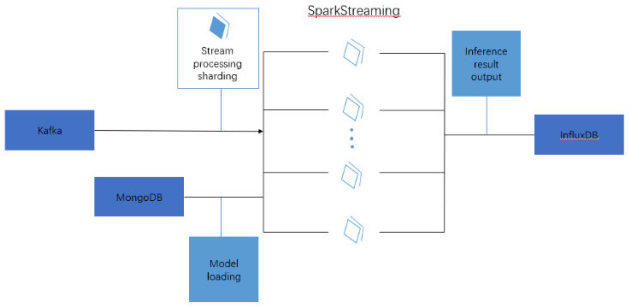

FIGURE 3. Model inference based on streaming processing.

TABLE 3. Database QPS dataset characteristics.

Field name	data type	description
Instance_ Name	varchar (500)	database instance name
Start_ Time	timestamp	start time
QPS	double	average queries per second

divided into training set 900 and test set 100. The accuracy of the model is verified by fitting the average call duration (unit: milliseconds).

TABLE 4. Microservice golden indicator dataset characteristics.

Field name	Data type	Description
serviceName	varchar(500)	Service name
startTime	TIMESTAMP	start time
avg_time	double	Average call duration
num	int	Total calls
succee_num	int	Number of successful calls
succee_rate	varchar(50)	Successful call rate

C. EVALUATION METRICS

Considering that temporal detection is essentially a regression model, MSE is used as the model evaluation metric. MSE stands for Mean Squared Error. It is a metric used in machine learning, statistics, and other related fields to measure the difference between the predicted value of a model and the actual value, as shown in Equation 4.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i hat - y_i)^2 \tag{4}$$

where: Yi is the true value of the ith sample, Yi_ Hat is the predicted value of the ith sample, and N is the number of samples.

The smaller the MSE value, the higher the accuracy of the model prediction. When comparing different models, the smaller MSE value indicates that the prediction result of the model is closer to the real value.

D. EXPERIMENTAL PROTOCOL AND ANALYSIS

The above two data sets are used to divide the training set and the test set. Based on the same data, ARIMA model, SARIMA model and the model in this paper are used for comparison.

1) EXPERIMENT OF DATASET 1

See Table 5, figure 4 and figure 5 for test results.

TABLE 5. Model test results about Dataset 1.

	ARIMA	SARIMA	THIS MODEL
MSE TRAIN	278.7997	271.7139	235.0475
MSE TEST	940.1581	941.8095	520.3371

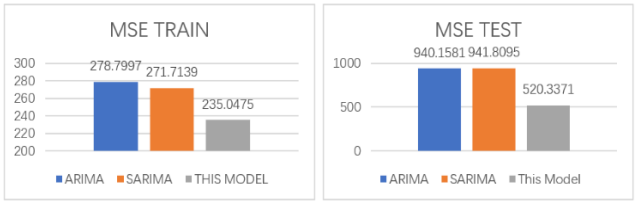

FIGURE 4. Model accuracy comparison about DATASET 1.

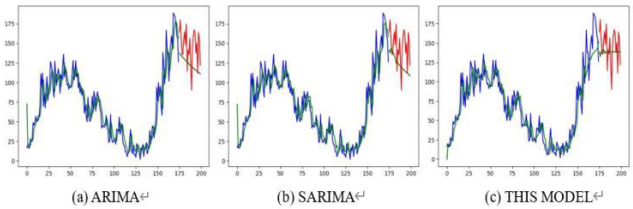

FIGURE 5. Model fitting results comparison about DATASET 1.

2) EXPERIMENT OF DATASET 2

The anomaly filtering of the model in this paper is shown in Figure 6. There are 900 data points in the training set, and 116 global anomalies are detected using the outlier detection method. Based on the periodicity of the second-order fuzzy filter, there are 64 outliers. Therefore, after removing the 64 outliers, the actual training set consists of 836 points.

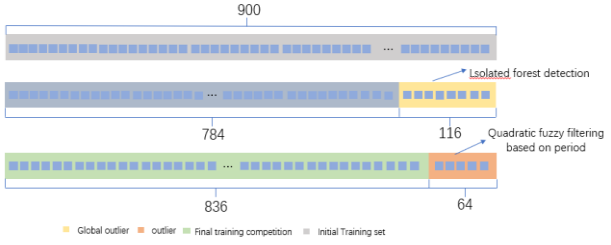

FIGURE 6. Historical anomaly filtering results(reduced misjudgment).

The test results are shown in Table 6, Figure 7 and Figure 8.

TABLE 6. Model test results about Dataset 2.

	ARIMA	SARIMA	THIS MODEL
MSE TRAIN	2095.8762	2089.481	2077.3561
MSE TEST	66.2057	88.6593	56.6196

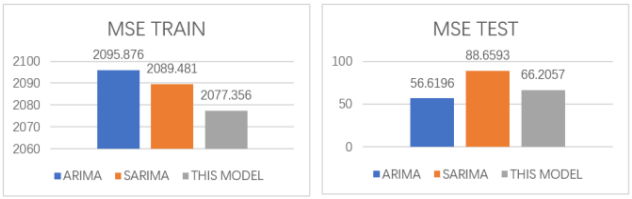

FIGURE 7. Model accuracy comparison about DATASET 2.

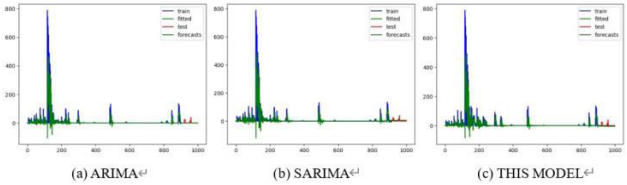

FIGURE 8. Model fitting results comparison about DATASET 2.

In addition, due to the direct use of the SARIMA model, it is necessary to determine the periodic parameters through grid search, which is time-consuming and costly. This model directly obtains the periodic parameters through an algorithm, eliminating this time-consuming step. The following table 7· compares the training time of each model.

TABLE 7. Comparison of the training time of each model.

	ARIMA	SARIMA	THIS MODEL
PERIODIC PARAMETER SETTINGS	-	Grid search	Directly derive
TRAINING TIME	1.1 minutes	85.6 minutes	1.2 minutes

As shown in the table, the training time of this model is close to that of the ARIMA model, and it significantly compresses the training time compared to the SARIMA model, reducing it from 85.6 minutes to 1.2 minutes.

3) CONCLUSION

Based on the test data, the method in this paper has improved the accuracy of the original SARIMA and ARIMA models. The results on Dataset 1 show that the MSE has decreased from about 940 to 520, improving the accuracy by 44%. The results on Dataset 2 show that the MSE has decreased from a maximum of 88 to 56, improving the accuracy by 36%. Additionally, the method automatically generates periodic parameters, greatly reducing the search space for model parameters and effectively reducing the cost of model training time from 85.6 minutes to 1.2 minutes.

E. APPLICATION EFFECT ANALYSIS

The model and system based on this method have been deployed in the actual production environment of a bank's distributed information system, and have performance indicator data from multiple scenarios including servers, networks, and application transactions. They provide real-time alerting for indicator abnormalities and visual querying capabilities, achieving good results.

1) SERVER PERFORMANCE MONITORING

The webpage shown in Figure 9 compares the original production monitoring method using fixed thresholds with the single dynamic baseline provided by this method, which better matches CPU operation patterns, improving detection accuracy and effectively reducing the number of microburst alerts. By correlating with configurations, alerts can be aggregated based on application and server type, increasing alert accuracy by approximately 20% and reducing alert volume by approximately 30%.

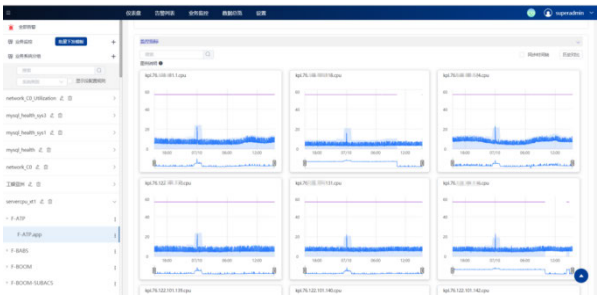

FIGURE 9. Server performance monitoring page.

2) NETWORK TRAFFIC MONITORING

The webpage shown in Figure 10 provides real-time monitoring of core backbone network traffic and bandwidth utilization of some device-level ports, effectively reducing the number of microburst alerts.

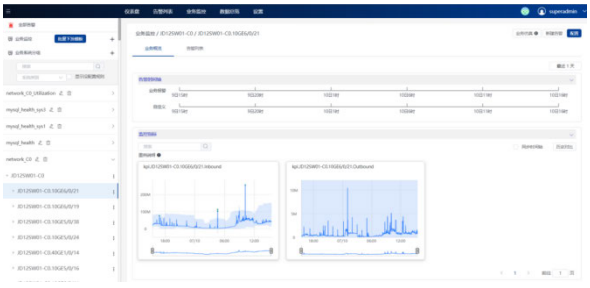

FIGURE 10. Network traffic monitoring page.

3) APPLICATION TRANSACTION MONITORING

The webpage shown in Figure 11 focuses on monitoring key performance indicators of payment applications, with an increase in alert accuracy of approximately 20%.

FIGURE 11. Application transaction monitoring page.

VIII. CONCLUSION

For the existing shortcomings of insufficient accuracy in indicator fault detection, complex training parameter tuning, and high performance overhead, this paper proposes a method based on periodic self-discovery and historical anomaly filtering for indicator fault detection. By combining multiple machine learning models, the method significantly reduces the training cost such as model parameter tuning through the automatic discovery of periodic parameters, ensuring real-time updates of the model and possessing good self-learning and self-adaptive abilities. Using the isolation forest as a primary screening method for anomaly detection, it quickly identifies global outliers with low time complexity and good algorithm stability. On this basis, a quadratic fuzzy filter based on periodicity is introduced to refine the screening process while avoiding phase noise interference caused by operational deviations, effectively solving possible local misclassification problems. The generated baseband is more suitable for indicator operational characteristics, improving detection accuracy. Compared with directly using ARIMA and SARIMA models based on two test datasets, this method improves the MSE by up to 44% and 36%, respectively. Additionally, this method has been practically applied in multiple scenarios in a bank's production environment, increasing alert accuracy by 20% and achieving a ratio of 10:1 for alert convergence. This method significantly enhances the indicator fault detection ability compared to traditional methods.

REFERENCES

[1] J. Kosinska, B. Balis, M. Konieczny, M. Malawski, and S. Zielinski, "Towards the observability of cloud-native applications: The overview of the state-of-the-art," *IEEE Access*, vol. 11, pp. 73036–73052, 2023.

[2] C. Fletcher C. (2016). *Innovation Insight for Algorithmic IT Operations Platforms*. Gartner. [Online]. Available: https://www.gartner.com/en/documents/3263717

[3] P. Prasad and C. Rich. (2016). *Market Guide for AIOps Platforms*. Gartner. [Online]. Available: https://www.gart-ner.com/en/documents/4000217

[4] P. Notaro, J. Cardoso, and M. Gerndt, "A systematic mapping study in AIOps," in *Proc. Int. Conf. Service-Oriented Comput.* Cham, Switzerland: Springer, 2020, pp. 110–123.

[5] P. Notaro, J. Cardoso, and M. Gerndt, "A survey of AIOps methods for failure management," *ACM Trans. Intell. Syst. Technol.*, vol. 12, no. 6, p. 81, Nov. 2021.

[6] D. Sun, M. Fu, L. Zhu, G. Li, and Q. Lu, "Non-intrusive anomaly detection with streaming performance metrics and logs for DevOps in public clouds: A case study in AWS," *IEEE Trans. Emerg. Topics Comput.*, vol. 4, no. 2, pp. 278–289, Apr. 2016.

[7] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009.

[8] M. Goldstein and A. Dengel, "Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm," in *Proc. Poster Demo Track*, 2012, pp. 59–63.

[9] P. Dymora, M. Mazurek, and S. Jaskółka, "VoIP anomaly detection—Selected methods of statistical analysis," *Annales Universitatis Mariae Curie-Sklodowska*, vol. 16, no. 2, p. 14, Dec. 2017.

[10] L. Lopez-Lopez, Á. G. Andrade, and G. Galaviz, "Anomaly detection-based spectrum sensing using the k-nearest neighbors algorithm," in *Proc. IEEE Mex. Int. Conf. Comput. Sci. (ENC)*, Aug. 2022, pp. 1–6.

[11] S. Pasupathipillai and E. D. Valle, "Approximate distance-based anomaly detection at massive scale," in *Proc. IEEE BigData*, Dec. 2020, pp. 1985–1991.

[12] C. Ji, X. Zou, S. Liu, and L. Pan, "ADARC: An anomaly detection algorithm based on relative outlier distance and biseries correlation," *Softw., Pract. Exp.*, vol. 50, no. 11, pp. 2065–2081, Nov. 2020.

[13] W. Lu and I. Traore, "Unsupervised anomaly detection using an evolutionary extension of k-means algorithm," *Int. J. Inf. Comput. Secur.*, vol. 2, no. 2, p. 107, 2008.

[14] Z. Chen and Y. F. Li, "Anomaly detection based on enhanced DBScan algorithm," *Proc. Eng.*, vol. 15, pp. 178–182, Jan. 2011.

[15] M. Chater, A. Borgi, M. T. Slama, K. S. Gandoura, and M. I. Landoulsi, "Fuzzy isolation forest for anomaly detection," *Proc. Comput. Sci.*, vol. 207, pp. 916–925, Jan. 2022.

[16] G. Pang, C. Shen, L. Cao, and A. van den Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, vol. 54, no. 2, pp. 1–38, 2021.

[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

[18] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 387–395.

[19] H. Xu, Y. Feng, J. Chen, Z. Wang, H. Qiao, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, and D. Pei, "Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications," in *Proc. World Wide Web Conf. World Wide Web*, 2018, pp. 187–196.

[20] M. A. Bashar and R. Nayak, "TAnoGAN: Time series anomaly detection with generative adversarial networks," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2020, pp. 1778–1785.

[21] A. Geiger, D. Liu, S. Alnegheimish, A. Cuesta-Infante, and K. Veeramachaneni, "TadGAN: Time series anomaly detection using generative adversarial networks," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 33–43.

[22] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," in *Proc. ICANN*, 2019, pp. 703–716.

[23] X. Wang, D. Pi, X. Zhang, H. Liu, and C. Guo, "Variational transformer-based anomaly detection approach for multivariate time series," *Measurement*, vol. 191, Mar. 2022, Art. no. 110791.

[24] W. Chen, H. Xu, Z. Li, D. Pei, J. Chen, H. Qiao, Y. Feng, and Z. Wang, "Unsupervised anomaly detection for intricate KPIs via adversarial training of VAE," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 1891–1899.

[25] Z. Li, W. Chen, and D. Pei, "Robust and unsupervised KPI anomaly detection based on conditional variational autoencoder," in *Proc. IEEE 37th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Nov. 2018, pp. 1–9.

[26] Y. Liu, Y. Lin, Q. Xiao, G. Hu, and J. Wang, "Self-adversarial variational autoencoder with spectral residual for time series anomaly detection," *Neurocomputing*, vol. 458, pp. 349–363, Oct. 2021.

[27] M. Yu and S. Sun, "Policy-based reinforcement learning for time series anomaly detection," *Eng. Appl. Artif. Intell.*, vol. 95, Oct. 2020, Art. no. 103919.

[28] M. Goswami, C. I. Challu, L. Callot, L. Minorics, and A. Kan, "Unsupervised model selection for time series anomaly detection," in *Proc. ICLR*, 2023.

[29] A. Jastrzebska, W. Homenda, and W. Pedrycz, "ARIMA feature-based approach to time series classification," in *Proc. ICCS*, vol. 1, 2022, pp. 192–199.

[30] S. J. Taylor and B. Letham, "Forecasting at scale," vol. 5, p. e3190, Sep. 2017.
[31] H. Musbah and M. El-Hawary, "SARIMA model forecasting of short-term electrical load data augmented by fast Fourier transform seasonality detection," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, May 2019, pp. 1–4.
[32] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 413–422.
[33] J. Arlt and P. Trcka, "Automatic SARIMA modeling and forecast accuracy," *Commun. Statist. Simul. Comput.*, vol. 50, no. 10, pp. 2949–2970, Oct. 2021.
[34] L. Girish and S. K. N. Rao, "Anomaly detection in cloud environment using artificial intelligence techniques," *Computing*, vol. 105, no. 3, pp. 675–688, Mar. 2023.
[35] Z. Li, N. Zhao, S. Zhang, Y. Sun, P. Chen, X. Wen, M. Ma, and D. Pei, "Constructing large-scale real-world benchmark datasets for AIOps," 2022, *arXiv:2208.03938*.

SHENG WU received the B.S. and M.S. degrees in computer science and technology from Tongji University, Shanghai, China, in 2004 and 2007, respectively, where he is currently pursuing the Ph.D. degree in computer science and technology. From April 2007 to March 2019, he was with the System Department, ICBC Data Center, where he engaged in system operation and maintenance-related work. Since April 2019, he has been with the Basic Technology Laboratory of the Financial Technology, Research Institute of ICBC (affiliated with ICBC Data Center), where he engaged in research and application related to AIOps.

JIHONG GUAN received the bachelor's degree from Huazhong Normal University, in 1991, the master's degree from the Wuhan Technical University of Surveying and Mapping (merged into Wuhan University, in 2000), in 1998, and the Ph.D. degree from Wuhan University, in 2002. She is currently a Professor with the Department of Computer Science and Technology, Tongji University, Shanghai, China. She has extensively published more than 300 papers in domestic and international journals (including *Nature Communications*, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, *Nucleic Acids Research*, and *Bioinformatics*) and conferences (including AAAI, ICDE, VLDB, SIGIR, RECOMB, and DASFAA). Her research interests include databases, data mining, distributed computing, bioinformatics, and geographic information systems.