Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2020) 28: 331 – 346

doi:10.3906/elk-1906-55

Sensor anomaly detection in the industrial internet of things based on edge computing

Dequan KONG, Desheng LIU*, Lei ZHANG, Lili HE, Qingwu SHI, Xiaojun MA
Department of Electrical Engineering, College of Information and Electronic Technology,
Jiamusi University, Jiamusi, P.R. China

Received: 09.06.2019 • **Accepted/Published Online:** 26.08.2019 • **Final Version:** 27.01.2020

Abstract: In the industrial internet of things (IIoT), because thousands of pieces of hardware, instruments, and various controllers are involved, the core problem is the sensors. Detection using sensors is the bottom line of the IIoT, directly affecting the detection accuracy and control indicators of the IIoT system. However, when a large number of real-time data generated by IIoT devices are transferred to cloud computing centers, large-scale data will inevitably bring computing load, which will affect the computing speed of cloud computing centers and increase the computing load of cloud computing data centers. These factors directly lead to instability and delay in sensor data collected in real time in the IIoT. In this paper, a sensor outlier detection algorithm based on edge calculation is proposed. Firstly, focusing on the problem of the large amount of data in terminal equipment of the IIoT, the edge technology method of data compression is used to optimize the compression of sensor data, and different thresholds are set according to different industrial process requirements, so as to ensure the real-time aspect and authenticity of the data. Then, using the K-means clustering algorithm, the compressed test data sets are analyzed and the abnormal sensor detection values and labels are obtained. Finally, the effectiveness of such an approach is evaluated through a sample case study involving a temperature control system.

Key words: Industrial internet of things, sensors anomaly detection, edge computing

1. Introduction

With the rapid development of internet of things (IoT) technology, a series of national strategies such as Made-in-China 2025, American Advanced Manufacturing Partnership Program, and German Industry 4.0 have been put forward and implemented. The industrial internet of things (IIoT) has emerged as the times require and has become an important promoter of the intelligent transformation of the global industrial system (originated from China Institute of Electronic Technology Standardization). The IIoT is mainly used in the design, production, management, and service of all aspects of the life cycle, which contains tremendous economic value [1]. The IIoT realizes flexible allocation of manufacturing raw materials, on-demand execution of manufacturing processes, rational optimization of manufacturing processes, and rapid adaptation of manufacturing environments through network interconnection, data exchange, and system interoperability of industrial resources, and achieves efficient utilization of resources, thus building a new service-driven industrial ecological system [2, 3]. The IoT involves the information of physical objects (sensors, machines, cars, buildings, and other items) that

*Correspondence: zdhlds@163.com

allows interaction and cooperation of these objects to reach common goals. The so-called IIoT is an advanced production mode to upgrade traditional industry to intelligence by means of a cloud platform.

In recent years, sensor technology, communication network technology, and some advanced computing and analysis technologies have developed rapidly. Artificial intelligence, industrial large data, and edge technologies are gradually applied to the IIoT, promoting the maturity of IIoT technology. At present, many domestic and international scholars have explored the IIoT. Through an incremental clustering algorithm based on k medoids, the peak density and thus collecting cluster dynamic data from the IIoT can be quickly found, effectively reducing the system's sudden failure [4]. Focusing on the energy consumption problem caused by the increasing equipment of the IoT, an energy-saving architecture of the IIoT is adopted that balances the flow load, saves energy consumption, and prolongs the service life of the system [5]. The software definition network (SDN) platform is built, data processing and transmission are carried out by SDN switch, and real-time monitoring and management of data are carried out by SDN controller, improving the reliability of the system [6, 7]. A cloud-assisted IoT with secure data sharing is proposed [8], which protects device data safely and improves processing time to some extent. Using these methods, many difficult problems in the IIoT have been solved. However, with the increasing amount of data generated by sensors, actuators, and other devices in the IoT, the delay of network control is difficult to be solved, which leads to decreasing efficiency of data transmission, thus affecting the control accuracy of the system. Therefore, there is a need to process and transmit the redundant data by reasonably compressing the data size.

With the increase in IoT business, the demand for data storage and computing will put pressure on the ability of cloud computing. To address this challenge, edge computing and cloud computing work together to provide possibilities for the emergence of new ecosystems in the IoT [9]. Edge computing is an open platform with computing, analysis, and storage functions, and is close to the edge of the network or the source of data. It mainly absorbs, stores, and filters the data and sends it to the cloud system. In recent years, the application of edge computing has become more and more widespread. A report [10] proposed edge computing and compressed sensing to eliminate redundant data by reducing bandwidth. Another report [11] designed a real-time large data management framework to upload sensor data in an optimal way. Another report [12] presented an edge algorithm that affected sensor output by controlling bandwidth and filtering local data of mobile devices in energy-constrained systems.

Obviously, these methods were all used to process and compress data in the transport layer of the IoT, which had certain effect in dealing with data redundancy. However, the sensor at the acquisition end is the key to data compression. In the IIoT, because thousands of pieces of hardware, instruments, and various controllers are involved, the core problem is the sensors. Detection using sensors is the bottom line of the IIoT, directly affecting the detection accuracy and control indicators of the IoT system. At present, there is very limited research on the subject of data compression processing at the sensor acquisition end. The data fusion technology can be optimized according to the literature [12, 13]. The sensor data can be processed by state estimation, decision fusion, and classification data fusion, which can process various forms of data, but does not take the transmission cost into account. In practice, it is difficult to achieve optimal results. A report [14] presented a dynamic fusion method for multisensor temperature data that improved the accuracy of temperature data fusion. Another article [15], on Mantri, used the statistical test of data-induced "laggard" tasks to diagnose faults, which has been proved to be effective in fixing faults and network congestion. Moreover, a new decision tree method was proposed to deal with missing data in conjunction with auxiliary regulation control [16]. Obviously, the edge problem of data compression on the sensor side is still not solved. Therefore, a

sensor anomaly detection algorithm based on edge computing is proposed to eliminate redundant data, reduce data transmission, and improve the quality of sensor anomaly detection without affecting the trend of data reflection.

The main contents of this paper are as follows: Section 2 outlines the application of edge computing in the IoT and the advanced algorithms in sensor outlier detection. Section 3 describes the overall framework of the IIoT. In Section 4, a data compression algorithm based on edge computing and a sensor outlier detection algorithm based on K-means clustering are proposed. In Section 5, the evaluation indicators of the compression algorithm and anomaly detection algorithm are discussed, and the performance of the algorithm is evaluated experimentally using actual data. In Section 6, this study is summarized.

2. Related work

This section summarizes the current work on this topic, which can be divided into two categories: research on edge computing in the IoT and research on sensor outlier detection. The existing research provides some references for this paper.

2.1. Edge computing for the IoT

In the context of the interconnection of all things, application services need to ensure low latency, high reliability, and data security [17]. With the increase in the data volume of IoT equipment and the rapid increase in data volume transmitted through the network, the performance of cloud computing has gradually reached a bottleneck, and it cannot meet these requirements. Therefore, the existing research focuses on how to improve energy efficiency [18, 19]. However, edge computing transfers computing power from the centralized cloud server to the node near the device end of the IoT, and realizes data preprocessing before entry to the cloud server [20]. In the literature [21], Goiuri et al. proposed an edge computing method in industry to improve the energy efficiency of IoT nodes. A new video surveillance system service platform is built by edge computing to improve the intelligent processing ability of the front-end camera of the video surveillance system [17]. A large number of studies have shown that the use of edge computing can effectively reduce the amount of data transmission, computing load, and network traffic load [22, 23]. However, there are limited studies on edge calculation of sensor data preprocessing.

2.2. Sensor abnormal value detection

In different application areas, anomalies have different definitions, usually being defined as nonconforming and abnormal behavior [24]. In any case, managers need to find them in time and make corresponding decisions. Therefore, anomaly detection has been applied in many fields, such as safety systems [25], finance [26], traffic management [27], and industrial control systems [28]. In industrial anomaly detection, the sensor data recording and analysis can determine whether the equipment is malfunctioning or not and whether it can meet the required control effect. In this field, Garces and Sbarbaro proposed an anomaly detection method based on partial least squares regression residual analysis and artificial neural network [29]. The report proposed an outlier mining method based on adaptive clustering for data analysis on an industrial control system OPC server [30]. Another study proposed a multioutlier detection method and robust parameter estimation method to detect discontinuous outliers in industrial time series [31]. A large number of studies have shown that different methods are often different for different problems and data sets, and the anomaly detection is usually related to data worth anomalies. The previous results provided some help for the present study.

3. Edge computing for the IIoT

3.1. Framework overview

In the IIoT, the data acquisition gateway (DTU) has developed very rapidly. The application of gateways used by cloud platforms such as 4G-DTU, GPRS-DTU, and WIFI-DTU to connect lower computer devices is very common. However, when large amounts of real-time data generated by IoT devices are transferred to a cloud computing center, large-scale data will inevitably bring computing load and affect the computing speed of the cloud computing center. In order to degrade the computing load of the cloud computing data center and reduce resource consumption, a solution, i.e. edge industry IoT framework, is proposed. As shown in Figure 1, all components are common infrastructure in industrial control.

The IoT framework of edge industry is mainly composed of user equipment, edge computing, cloud platform server group, and user display terminal. The user equipment is a field instrument consisting of a temperature and humidity sensor, transmitter and other sensor acquisition units, editable logic controller (PLC), digital controller, and other control units. In industrial applications, according to the different service requirements of customers, different field devices are selected. In the new framework, edge calculation is added. A large amount of data collected by sensors are preprocessed before they are transferred to the controller. Only a small amount of data accurately reflecting the change in numerical value is transferred to the controller and cloud platform. The display terminal is a computer web page, mobile web page, and mobile APP, which can realize remote management and data monitoring of the field equipment. The present paper mainly investigates the data processing at the bottom and initial sensor acquisition end of the whole IIoT.

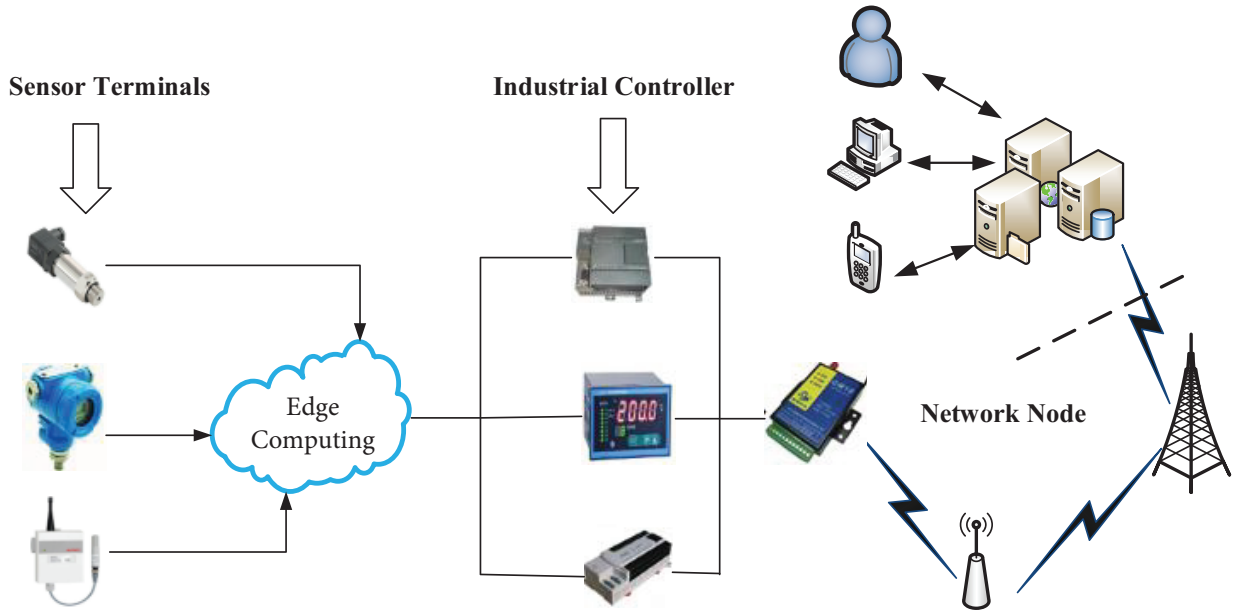

Figure 1. Edge IIoTs architecture.

3.2. Working principle

The main work flow of this framework is as follows: firstly, each sensor collects real-time data on the scene (generally refers to temperature, humidity, pressure, and liquid level), and transmits the collected data to the

edge calculation model. Secondly, in order to better realize the compression of the collected data, the collected data are uploaded into groups. In this process, the amount of data in the group and the threshold of data separation need to be adjusted and uploaded to the controller after the processing. Thirdly, when the collected data arrive at the controller, the controller will make corresponding control decisions. At the same time, the collected data will be packaged and encrypted by the Modbus master station and uploaded to the cloud platform. Finally, users and managers complete remote management of devices (data viewing and remote control) through the background APP of the cloud platform. Here the application of edge computing can optimize and simplify data, greatly reduce the amount of data uploaded, and reduce the computing load of the cloud platform.

4. Outlier detection algorithm based on edge computation

For the sensor in the IIoT, large amounts of redundant data are uploaded to the cloud computing platform, which greatly downgrades the computing performance of the cloud platform and affects the outlier detection. In this section, a scheme is proposed. Firstly, the edge calculation is used to optimize the compression of sensor data and then the K-means clustering algorithm is used to judge the outliers of the processed data.

4.1. Data compression algorithm

In this section, an effective method is designed to solve the problem of the large amounts of data uploaded to the cloud platform. Increasing data redundancy can improve system stability. Low data redundancy and high data reliability are contradictory in a sense, meaning it is extremely difficult to find the optimal solution of minimum data redundancy and maximum data reliability.

For industrial controls, the algorithm should compress the data without changing the characteristics of the original sensor data and the true reflection of the data. The shorter the processing time, the better the performance. The main processing steps are as follows: firstly, it tests set text. TXT is established and the data to be tested are stored in text, txt, and the number k and error threshold e of each group of data packets are set. Secondly, the sequential data values are tested sequentially. If the average value of the current sequential data t[i+k] and its previous k-1 sequential data is smaller than the error threshold e, the output is not carried out, and then the cycle is carried out. For the sequential data t[i+k], t[i+k+1],..., when t[i+k+10] still satisfies the above conditions, we store t[i+k+10] and the average value of the first k-1 data as the uploaded data in out2.txt and i+k+10 in out1.txt. If the average value of the time series data t[i+k] appears and the average value of the first k-1 time series data in the group is not smaller than the error threshold e, we store t[i+k] as the nonnegligible data in out2.txt. In 2.txt, i+k is stored in out1.txt. The corresponding algorithm is implemented as shown in Algorithm 1 [31].

Algorithm 1 mainly utilizes the characteristics of continuous change in data that can be neglected for enough hours, and compresses the data properly. In order to select the best number of single group data K and error threshold e, because the minimum accuracy of data set change is 0.1, the product of K and E is greater than or equal to 0.1, and it is better to be an integer multiple of 0.1, and the excessive value of e leads to the missing of valid data. Four groups of experiments were conducted with different K and E values. The four groups of test parameters were selected as follows: group 1 K = 10, e = 0.01; group 2 K = 20, e = 0.01; group 3 K = 20, e = 0.02; group 4 k = 10, e = 0.02. Using different parameters, the original data are compressed to varying degrees by adding the edge computing data compression algorithm, and the response time is also different. The specific data processing situation is shown in Table 1.

The most important index to evaluate compression performance is the compression ratio, which represents compression capacity. In Table 1, it can be clearly seen that the compression ratio under Group 3 and Group

```
Algorithm 1 Sensor Data Compression Algorithms
Input:test.txt, k, e//the data and the number of packages and precisions
Output: out1.txt, out2.txt
1)  for i=1 to N
2)    Read the data from “test.txt”, and write them to “data.txt”
3)  if data.text==null
4)    break
5)  end if
6)  for i=1 to N
7)    Read the data from “data.txt” to t[i+1]
8)   aver=sum(t)/i+1;
9)  end
10) if(aver<0)
11)   for i=1 to k
12)     aver=aver+t[i]
13)     aver=aver/k
14)   end
```

```
15) else
16)   for i=2 to n
17)    if(temp<aver)
18)     for j=0 to k-1
19)       if i+j>=n
20)         p<-1
21)         aver=aver+t[i+j]
22)       end if
23)     end
24)    end if
25)   end
26) end if
27)  aver=aver/k
28)  if |aver-temp|>=e
29)   put i+j-1 to “out1.txt”
30)   put t[i+j-1] to “out2.txt”
31)  end if
32)  return “out1.txt”,“out2.txt
```

4 parameters is significantly lower than that under Group 1, and the compression ratio in Group 2 is lower. Secondly, another index for evaluating compression performance is the calculation time, that is, the running time of compression. Obviously, under the parameters of Group 2 and Group 3, the compression ratio is also lower. The algorithm has a short response time and fast data processing speed. In summary, the algorithm performs well when the parameters of Group 2 and Group 3 are used.

In order to truly reflect the actual changes in data, it is not enough to consider only the compression ratio and data processing time. It is necessary to consider an important premise of evaluating compression performance: homogeneity, whether the data after compression can replace the original data and maintain the original data capacity. Therefore, the real-time changes in uploaded data under different parameters are simulated, as shown in Figure 2.

Table 1. Comparisons of data processing using different parameters.

Group	e	k	Compression ratio	Response time
1	0.01	10	0.0340	7.604 s
2	0.01	20	0.0079	6.392 s
3	0.02	20	0.0043	6.877 s
4	0.02	10	0.0033	7.587 s

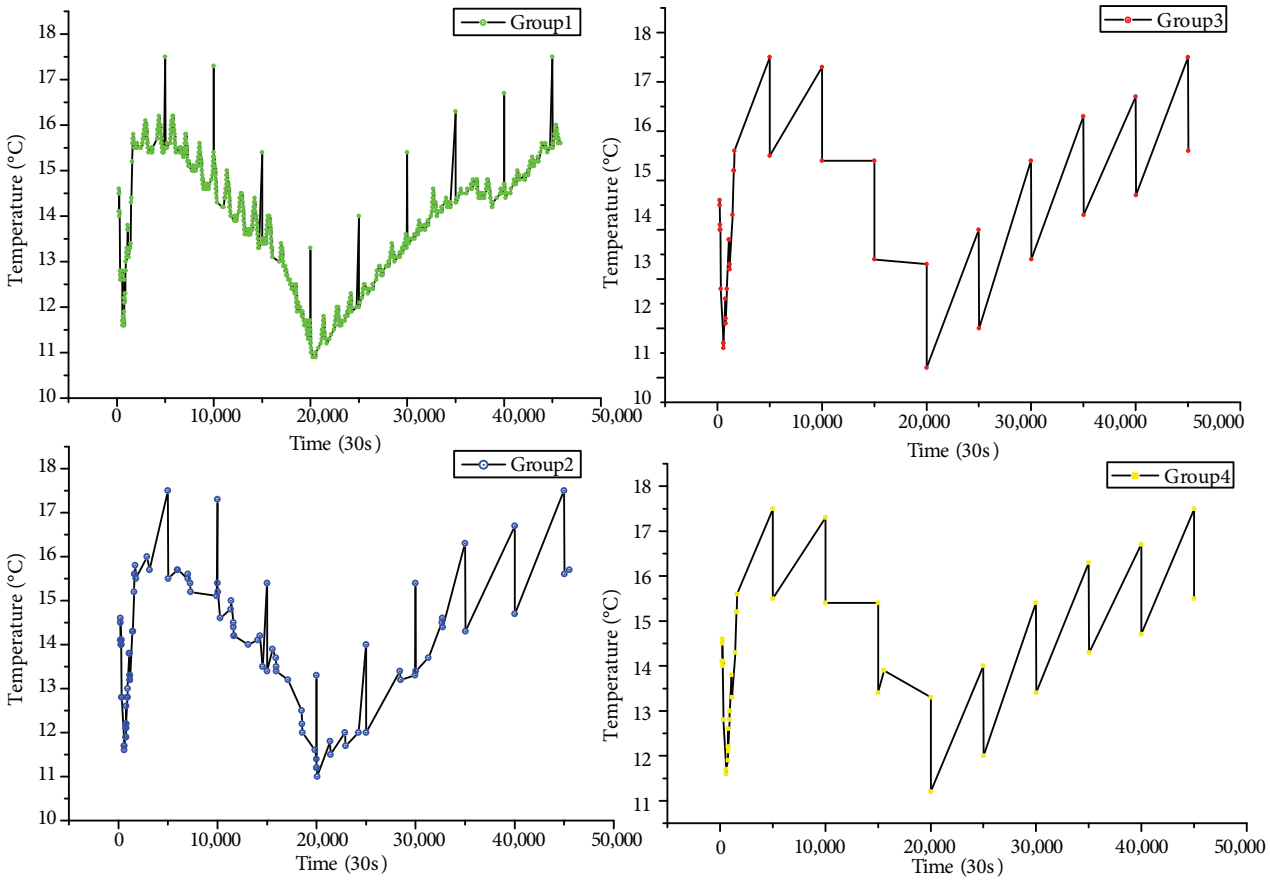

Figure 2. Scatter chart for change trend of data uploads.

Figure 2 shows that the data processed under four different parameters can well reflect the changing trend in the data compared with the original data. It can be revealed that the data compression algorithm can ensure good uniformity of the data.

4.2. Outlier value detection based on a box graph

In this section, the box graph detection method in statistics is designed to detect anomalies in sensor data. The main processing steps are as follows: firstly, the sensor data set S was input, then the normal value interval Nv of the data was calculated using the DataFrame method, finally the abnormal value of the data was detected, and the abnormal value label was set to output the abnormal data set S' and standard data set S'. The corresponding algorithm implementation is shown in Algorithm 2. In Algorithm 2, the total data are

```
Algorithm 2 Sensor Data Anomaly Detection
Input: Sensor Data Set S, Data Normal Value Interval NV
Output: exception data set S', standard data set S"
1)  n[S]= standardized S;
2)  for(i=1;i<=model.length;i++)
3)   if(model[i]!=nv)
4)     S'[i]=S[i];// Get abnormal data
5)   else
6)     S"[i]=S[i];// Get normal data
7)  end if
8) end
```

arranged to a large extent, and the QR and upper and lower bounds (i.e. error tolerance) are calculated by the upper and lower quartiles Q1 and Q3, so as to distinguish the outliers of the data. In order to test the performance of this method, we draw box diagrams of four groups of processed data with different parameters, judge whether there are abnormal values in the original data and four groups of processed data, and analyze them. Nine outliers are added to 45,989 items of data, and the same step anomaly is added to every 5000 data acquisition points. As shown in Figure 3, the outlier detection of the original data and of the data of Groups 1–4 are in turn from left to right, and the average value of all data in this group is indicated in the abscissa. From the boxplot of the original data, it can be seen that the edges of the upper and lower outliers are 10.9 °C and 17.7 °C, respectively; the upper and lower quartiles are 13.0 °C and 14.9 °C, respectively; and the median is 14.2 °C. The distribution of the outliers is right skewness. The outliers are easier to concentrate on the larger value side. It judges the original outliers as mild outliers and so the detection effect is not very good. Obviously, the average values of the first, second, and fourth groups are closer to the original data, the first group of data has not detected abnormal values, and the second, third, and fourth groups have detected abnormal values. It can be seen that the effect of abnormal values under the parameters of the second and fourth groups is relatively good. It can be seen from the anomaly values measured in the second, third, and fourth groups that when the same anomaly values appear many times it is difficult to observe them. It can be seen that the effect of anomaly detection of the box chart in this scenario is insufficient.

4.3. Abnormal value detection based on k-means clustering

Focusing on the problem that Algorithm 2 is not effective in outlier detection and cannot detect in real time, an effective method is proposed in this section to solve the problem of outliers in sensor data, and a more accurate outlier detection result is obtained through simulation. The main processing steps are as follows: firstly, the parameters are initialized, the sensor data set S is input, the threshold ts of discrete points, the clustering category K, and the number of clustering iterations I are set, and then the data are standardized. Secondly,

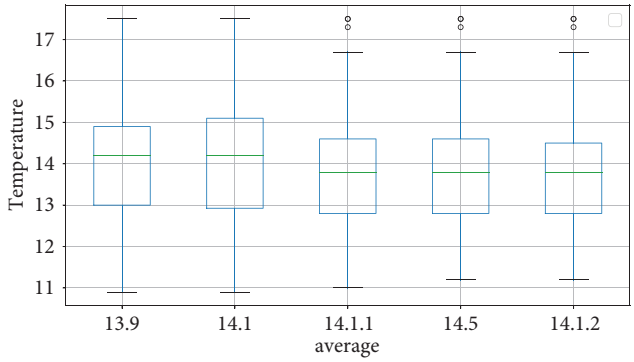

Figure 3. Comparison of outlier detection between all data and different parameters after compression (box chart).

the absolute distance and relative distance are calculated one by one, if the relative distance is larger than the set discrete distance. The point threshold is determined as discrete points, i.e. outliers, and the set of outliers saved DP is output. The corresponding algorithm is implemented as shown in Algorithm 3.

Algorithms 3 K-Means Clustering Sensor Feature Data
Input: sensor data set S, clustering category k, discrete point threshold ts, iteration number i;
Output: clustered data set np, discrete point set DP
1) n [S] = standardized S;
2) Model = KMeans (k, 4, i, n [S]// Clustering model of k-class input data set S obtained by four concurrent iterations of I times
3) for (i = 1; I < = model. length; I ++)
4) TMP = model [i]; / / clustering category
5) if (model[i]>ts)
6) DP = model [i]; / / discrete point
7) else
8) NP = mode [i]; normal point
9) end if
10) end

We add the K-means clustering algorithm to four groups of processed data to test its performance. It is well known that K is worth choosing in the K-means clustering algorithm as it will have a significant impact on the results of the algorithm. However, in practical applications, the value of K needs to be determined beforehand. People often use experience to select values or do it randomly, which makes the clustering effect unstable. To solve this problem, this paper uses the elbow method to select K optimally. With the increase in cluster number K, the sample division will be more precise, and the clustering degree of each cluster will gradually increase, and so the square of error and SSE will naturally become smaller. When K is less than the real clustering number, the aggregation degree of each cluster will be greatly increased with the increase in K. When K reaches the real clustering number, the return of aggregation degree will be rapidly reduced with the increase in K. Therefore, the decrease in SSE will decrease sharply, and will become flat with the increase in K value. That is to say, the relationship graph between SSE and K is the shape of an elbow, and the K value corresponding to this elbow is the optimal clustering number of data [32].

The relationship between SSE and K value is shown in Figure 4. The curvature of the elbow is highest when K is 2. Therefore, for clustering of this data set, the optimal value of K is 2.

As shown in Figure 5-9, the clustering category is 2, the distance threshold is 3, and the maximum number of iterations is 500. The "*" in the figure represents the exceptional data.

Figure 5 shows that 15 abnormal values are detected under the set parameters of Group 1, and 6 normal values are misjudged as abnormal values, with a misjudgment rate of 40%. Figure 6 shows that 9 abnormal values are detected under the set parameters of Group 2, with a misjudgment rate of 0% and all abnormal points are detected. Figure 7 shows that under the set parameters of Group 3, the differences are detected. With 13 constant values, 4 normal values were misjudged as abnormal values, and the rate of misjudgment was 30.8%. According to Figure 9, we can see that 11 abnormal values were detected and 2 normal values were misjudged as abnormal values, with a misjudgment rate of 18.9%. Although there could be some misjudgments in this algorithm when the parameters are different, the false alarm rate is 0%.

Response time is another important indicator of anomaly detection performance. Real-time and continuous monitoring in industrial production process can detect problems as early as possible and solve them in time. It is of great significance for the safety and stability of enterprise production and the improvement of production efficiency. Therefore, the anomaly detection time under different parameters is counted, as shown in Table 2.

Table 2 shows that the time of anomaly detection after data compression is reduced by about half, and the time of anomaly detection under different parameters is almost the same, and so accuracy is the main evaluation index. It can be seen that the K-means clustering algorithm works best in outlier detection when the number of data packets processed is k = 20 and the error threshold is e = 0.01.

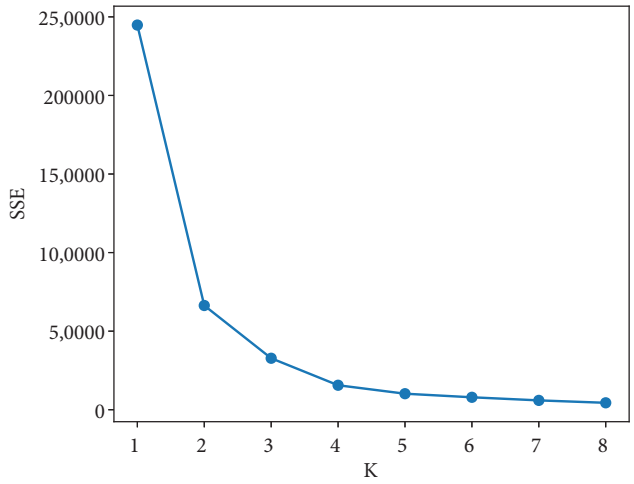

Figure 4. Graph of the relation between SSE and K.

5. Experimental research

5.1. Experimental environment

Hardware environment: All the experiments are based on the Windows 7 operating system. The CPU is Intel Core i54200 U, the graphics card is AMD Radeon HD 8670M, and the memory is 4 GB. The experiment uses Python to compile the algorithm. Test Data Set: The test data in this paper is a test cloud platform of the

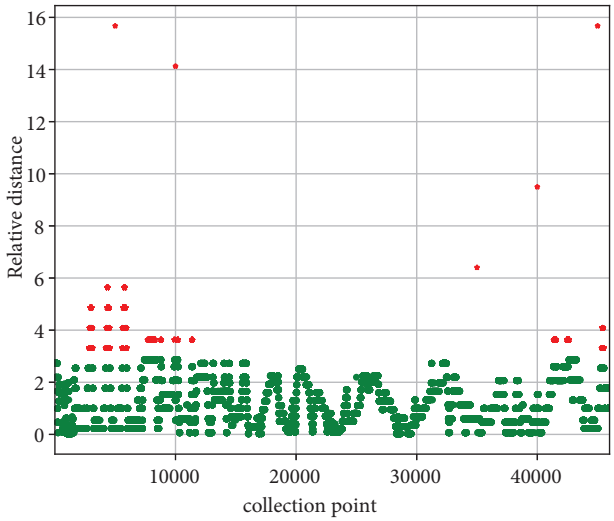

Figure 5. K-means clustering algorithm abnormal value detection results for untreated data.

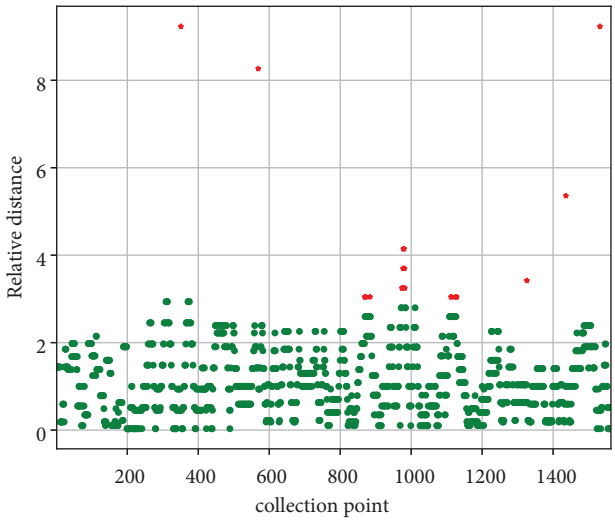

Figure 6. K-means clustering algorithm abnormal value detection results for compressed data in Group 1.

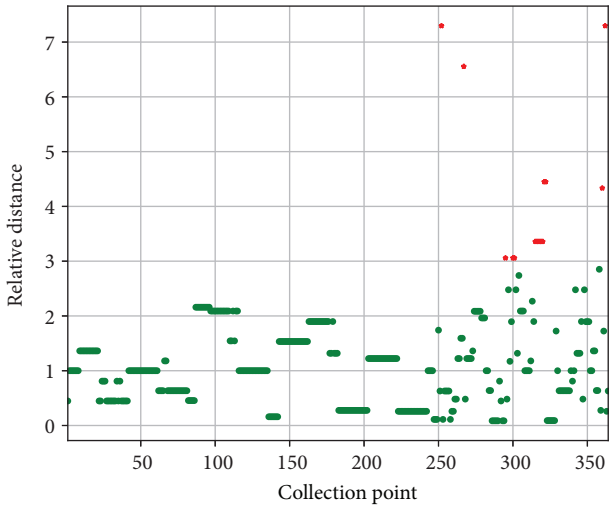

Figure 7. K-means clustering algorithm abnormal value detection results for compressed data in Group 2.

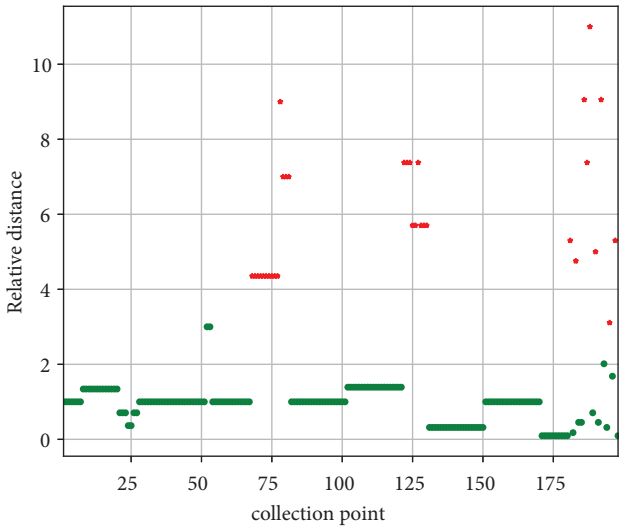

Figure 8. K-means clustering algorithm abnormal value detection results for compressed data in Group 3.

Table 2. Comparisons of data processing using different parameters.

e	k	Response time (s)	Total response time (s)
0.01	10	0.0000004462	7.6040004462
0.01	20	0.0000004462	6.3920004462
0.02	20	0.0000004462	6.8770004462
0.02	10	0.0000004462	7.5870004462
0	0	0.0000008925	0.0000008925

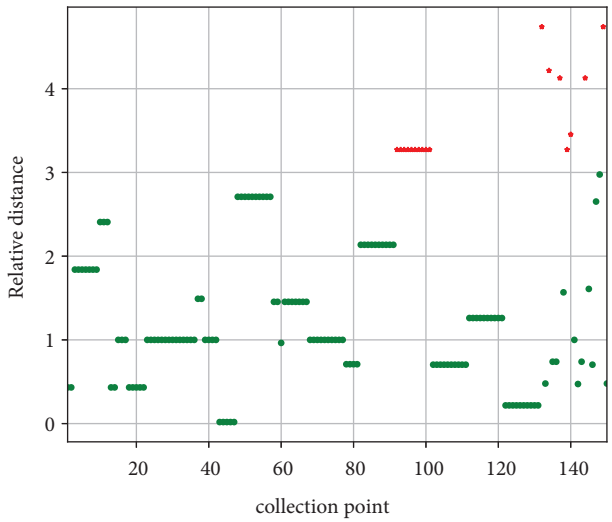

Figure 9. K-means clustering algorithm abnormal value detection results for compressed data in Group 4.

IIOT. The experimental platform is shown in Figure 10, which mainly includes a programmable logic controller (PLC), IIOT module, touch screen, host computer, temperature sensor, fan, switch, etc. The system uses Siemens S7-200 Smart PLC as the main controller. The CPU of the controller controls the output electric signal to control the action of each device. At the same time, the corresponding Smart 1000IE V3 touch screen and the host computer are matched. The interactive input touch screen technology, remote host computer control technology, and cloud-based platform movement are used. The mobile terminal monitoring technology realizes the real-time monitoring of the whole system.

5.2. Analysis of experimental results

A total of 1200 sensor temperature values were collected from 7:00 to 17:00 on the day of 20 November 2018. The time interval of data acquisition was 30 s. There were two anomalies added by hand-held sensor probe. The two anomalies increased temperature data at 9:00 and 11:10, respectively. The anomalies were added for 2 min, including the anomalies caused by events, mutations, etc.

Using the edge data compression algorithm and K-means clustering algorithm proposed in the present paper, the outlier detection of data centralized temperature sensor is carried out and the detection results of the two algorithms are analyzed in detail. Detailed parameters: setting the number of data packet processing $k = 20$, error threshold $e = 0.01$, discrete point threshold $TS = 2$, clustering category $k = 4$, and clustering iteration number $I = 500$. The test results are shown in Figure 11.

As can be seen in Figure 11 and 12, the abnormal data occur at 241 and 741 sampling points, which coincides with the time when the abnormal data begin to appear in the original data. It can be seen that the algorithm can accurately detect the abnormal sensor data. While 31 outliers were detected in the original data test, 30 were detected in the processed data test. In Figure 11, the 731st acquisition point is misjudged and the normal value is judged as an abnormal value. By looking at the original data, we can see that the values of 732 and 733 acquisition points tend to be close. Because the period of data acquisition is 30 s, there could be a certain data bias. However, due to the large amount of data, there is only one misjudgment in this test and so we can consider that the number is close. The actual abnormal detection results are shown in Table 3. According to the compression of the outlier detection algorithm, the detection effect is very good.

After the data compression, the data set is compressed from 1200 to 408, the compression ratio is 0.34, and the response time is 6.756 s, which can meet the real-time requirement of industrial fields. It is not difficult to find from the graph that the compressed data do not deviate from the track of change of the original data. When the data are flat, the compressed data can replace the original data with fewer values. When the data are abnormal, the compressed data can be retained to detect the abnormal values.

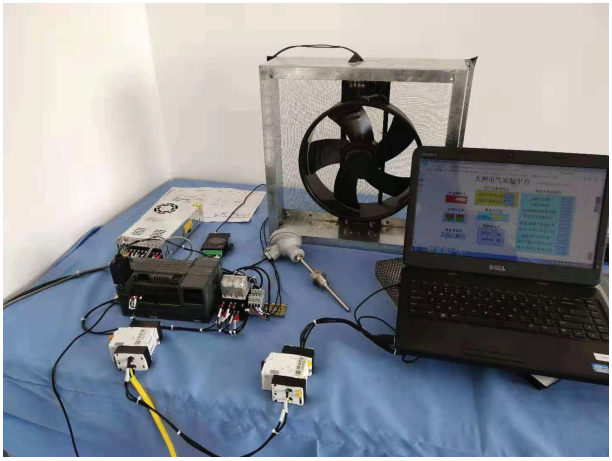

Figure 10. Appearance of the experimental platform.

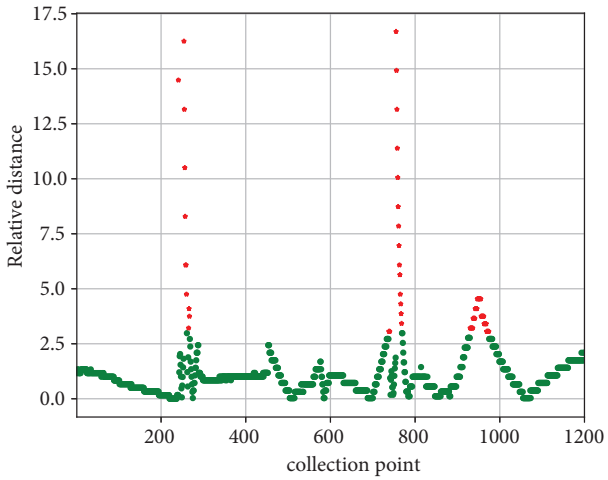

Figure 11. Detection of outliers after data processing.

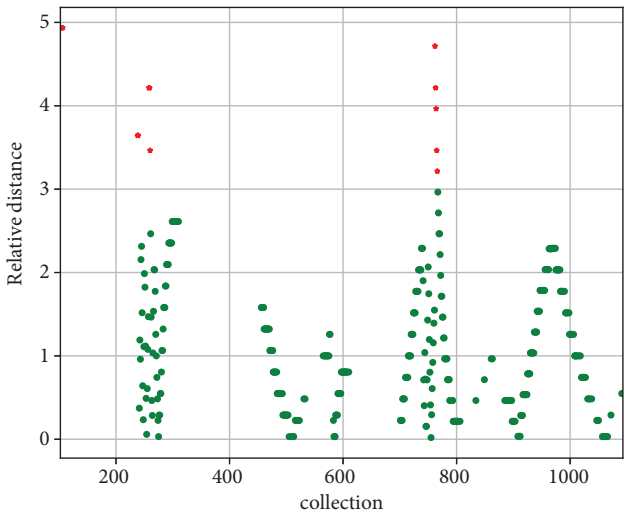

Figure 12. Abnormal value detection without data processing.

Table 3. Comparison of abnormal detection results.

k	e	Compression ratio	Response time (s)	Total response time (s)	Misinterpretation
20	0.01	0.34	0.0000004462	6.7560004462	1
0	0	1	0.0000004463	0.0000004463	0

6. Conclusions

The IIoT represents the foundation of industry 4.0. The entry of massive data into a cloud computing platform will inevitably result in a bottleneck of cloud computing capability, which will directly affect the performance of the whole system. Due to the continuous progress of SDN and edge computing, the development of the IIoT has found new opportunities. In order to effectively solve the problems of large amounts of data transmission and computing bottlenecks, this paper proposes a data compression algorithm based on edge computing. At the same time, focusing on the situation that the field equipment in industrial system is prone to abnormal data for a long period of continuous operation, a data anomaly detection algorithm based on K-means clustering is proposed. The simulation and experiment results show that the compression ratio of sensor data can be kept low and the continuity of data can be guaranteed. At the same time, the detection of abnormal values of sensor data can ensure high accuracy, and the processing time of abnormal values detection for the whole sensor data is less than 7 s. It can almost satisfy the real-time performance, solve the problem of large amounts of data to a great extent, and improve the system performance. However, the parameters of the algorithm need to be modified in different situations.

Acknowledgments

This research was funded by Heilongjiang Province science fund for returnees (grant number: LC2017027) and Jiamusi University Science and Technology Innovation Team Construction Project (grant number: CXTDPY-2016-3).

References

[1] Boyes H, Hallaq B, Cunningham J. The industrial internet of things (IIoT): an analysis framework. Computers in Industry 2018; 101: 1-12.

[2] Atzori L, Iera A, Morabito G. The internet of things: a survey. Computer Networks 2010; 54 (15): 2787-2805.

[3] Huang Z, Chen J, Lin Y, You P, Peng Y. Minimizing data redundancy for high reliable cloud storage systems. Computer Networks the International Journal of Computer & Telecommunications Networking 2015; 81: 164-177.

[4] Zhang Q, Zhu C, Yang LT, Chen Z, Zhao L et al. An incremental CFS algorithm for clustering large data in industrial internet of things. IEEE Transactions on Industrial Informatics 2017; 99: 1-8.

[5] Wang K, Wang Y, Sun Y, Guo S, Wu J. Green industrial internet of things architecture: an energy-efficient perspective. IEEE Communications Magazine 2016; 54 (12): 48-54.

[6] Cui L, Yu F R, Yan Q. When big data meets software-defined networking: SDN for big data and big data for SDN. IEEE Network 2016; 30 (1): 58-65.

[7] Al-Rubaye S, Kadhum E, Ni Q, Anpalagan A. Industrial internet of things driven by SDN platform for smart grid resiliency. IEEE Internet of Things Journal 2017; 99: 1-1.

[8] Mollah MB, Azad MAK, Vasilakos A. Secure data sharing and searching at the edge of cloud-assisted internet of things. IEEE Cloud Computing 2017; 4 (1): 34-42.

[9] Pan J, McElhannon J. Future edge cloud and edge computing for internet of things applications. IEEE Internet of Things Journal 2018; 5 (1): 439-449.

[10] Chen CY, Hasan M, Mohan S. Securing real-time internet of things. Sensors 2018; 18 (12): 4356.

[11] Kang KD, Chen L, Yi H, Wang B, Sha M. Real-time information derivation from big sensor data via edge computing. Big Data Cognitive Computing 2017; 1 (1): 5.

[12] Khaleghi B, Khamis A, Karray F, Razavi SN. Multisensor data fusion: a review of the state-of-the-art. Information Fusion 2013; 14 (1): 28-44.

[13] Castanedo F. A review of data fusion techniques. The Scientific World Journal 2013; 2013 (2): 704504.

[14] Li S. The approach of dynamic data fusion based on multi-sensor temperature data. Bulletin of Science & Technology 2015; 31 (1): 146-149.

[15] Ananthanarayanan G, Kandula S, Greenberg A, Stocia L, Lu Y et al. Reining in the outliers in map-reduce clusters using Mantri. Usenix Conference on Operating Systems Design and Implementation. USENIX Association 2010; 64 (2): 265-278.

[16] Yang H, Fong S, Sun G, Wong R. A very fast decision tree algorithm for real-time data mining of imperfect data streams in a distributed wireless sensor network. International Journal of Distributed Sensor Networks 2014; 2012 (4): 16.

[17] Shi W, Sun H, Cao J, Zhang Q. Edge computing—an emerging computing model for the internet of everything era. Journal of Computer Research & Development 2017; 54 (5): 907-924.

[18] Raufi B, Ismaili F, Ajdari J, Zenuni X. Web personalization issues in big data and Semantic Web: challenges and opportunities. Turkish Journal of Electrical Engineering and Computer Science 2019; 27 (4): 2379-2394.

[19] Sharma M, Arunachalam K, Sharma D. Analyzing the data center efficiency by using PUE to make the data centers more energy efficient by reducing the electrical consumption and exploring new strategies. Procedia Computer Science 2015; 48: 142-148.

[20] Li H, Ota K, Dong M. Learning IoT in edge: deep learning for the internet of things with edge computing. IEEE Network 2018; 32 (1): 96-101.

[21] Peralta G, Iglesiasurkia M, Barcelo M, Gomez R, Moran A et al. Fog computing based efficient IoT scheme for the Industry4.0. In: IEEE International Workshop of Electronics, Control, Measurement, Signals and Their Application to Mechatronics 2017; 24 (3): 1-6.

[22] Sun X, Ansari N. Edge IoT: mobile edge computing for the internet of things. IEEE Communications Magazine 2016; 54 (12): 22-29.

[23] Cho Y, Paek Y, Ahmed E, Ko K. A survey and design of a scalable mobile edge cloud platform for the Smart IoT Devices and it's applications. In: Park J, Pan Y, Yi G, Loia V (editors). Advances in Computer Science and Ubiquitous Computing, Lecture Notes in Electrical Engineering. Singapore: Springer 2016; pp. 694-698.

[24] Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Computing Survey (CSUR) 2009; 41 (3): 1-72.

[25] King S, King D, Astley K, Tarassenko L, Hayton P et al. The use of novelty detection techniques for monitoring high-integrity plant. In: Proceedings of the 2002 International Conference on Control Applications; Glasgow, UK 2002; pp. 221-226.

[26] Borrajo ML, Baruque B, Corchado E, Bajo J, Corchado JM. Hybrid neural intelligent system to predict business failure in small-to-medium-size enterprises. International Journal of Neural Systems 2011; 21: 277-296.

[27] Zheng Y, Rajasegarar S, Leckie C, Palaniswami M. Smart car parking: temporal clustering and anomaly detection in urban car parking. In: IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing; Singapore 2014; pp. 1-6.

[28] Kiss I, Genge B, Haller P, Gheorghe S. Data clustering-based anomaly detection in industrial control systems. IEEE International Conference on Intelligent Computer Communication and Processing. IEEE 2014; 5: 275-281.

[29] Garces H, Sbarbaro D. Outliers detection in environmental monitoring databases. Engineering Applications of Artificial Intelligence 2011; 24 (2): 341-349.

[30] Chen Z, Huang Y, Zou H. Anomaly detection of industrial control system based on outlier mining. Computer Science 2014; 41 (5): 178-203.

[31] Luceno A. Detecting possibly non-consecutive outliers in industrial time series. Journal of the Royal Statistical Society 2010; 60 (2): 295-310.

[32] Han LB, Wang Q, Jiang ZF, Hao ZQ. Improved k-means initial clustering center selection algorithm. Computer Engineering and Applications 2010; 46 (17): 150-153.