EXPERIENCE: Algorithms and Case Study for Explaining Repairs with Uniform Profiles over IoT Data

ZHICHENG LIU, YANG ZHANG, RUIHONG HUANG, ZHIWEI CHEN, SHAOXU SONG, and JIANMIN WANG, Tsinghua University

IoT data with timestamps are often found with outliers, such as GPS trajectories or sensor readings. While existing systems mostly focus on detecting temporal outliers without explanations and repairs, a decision maker may be more interested in the cause of the outlier appearance such that subsequent actions would be taken, e.g., cleaning unreliable readings or repairing broken devices or adopting a strategy for data repairs. Such outlier detection, explanation, and repairs are expected to be performed in either offline (batch) or online modes (over streaming IoT data with timestamps). In this work, we present TsClean, a new prototype system for detecting and repairing outliers with explanations over IoT data. The framework defines uniform profiles to explain the outliers detected by various algorithms, including the outliers with variant time intervals, and take approaches to repair outliers. Both batch and streaming processing are supported in a uniform framework. In particular, by varying the block size, it provides a tradeoff between computing the accurate results and approximating with efficient incremental computation. In this article, we present several case studies of applying TsClean in industry, e.g., how this framework works in detecting and repairing outliers over excavator water temperature data, and how to get reasonable explanations and repairs for the detected outliers in tracking excavators.

CCS Concepts: • **Information systems** → **Data cleaning**; • **Applied computing** → *Enterprise data management*;

Additional Key Words and Phrases: Outlier explanation, outlier repairs, data profiling, time series

ACM Reference format:
Zhicheng Liu, Yang Zhang, Ruihong Huang, Zhiwei Chen, Shaoxu Song, and Jianmin Wang. 2021. EXPERIENCE: Algorithms and Case Study for Explaining Repairs with Uniform Profiles over IoT Data. *J. Data and Information Quality* 13, 3, Article 18 (April 2021), 17 pages.
https://doi.org/10.1145/3436239

1 INTRODUCTION

Outliers are often found in IoT data with timestamps due to the dirty or imprecise values, such as entity history [9], GPS trajectories or sensor reading sequences [17]. Although a huge body of outlier detection techniques have been proposed over temporal data (see Reference [11] for a survey), a system with temporal explanations about what lead to the outlying data are still missing. The

This work is supported in part by the National Key Research and Development Plan (2019YFB1705301) and the National Natural Science Foundation of China (62072265, 61572272, 71690231).
Authors' addresses: Z. Liu, Y. Zhang, R. Huang, Z. Chen, S. Song (corresponding author), and J. Wang, Tsinghua University; emails: {lzc18, zhang-ya16, hrh16, czw18}@mails.tsinghua.edu.cn, {sxsong, jimwang}@tsinghua.edu.cn.

1936-1955/2021/04-ART18 $15.00
https://doi.org/10.1145/3436239

existing EXAD system [26] proposes to explain the anomalies on big data traces using a decision tree, while the temporal information is not fully explored.

Without a good explanation [22], decision makers only know that something abnormal has just happened, but has limited or no understanding of the reason behind its appearance, and therefore cannot determine what kind of subsequent actions should be taken to react to the situation or to avoid it in the future. For example, outliers appearing repeatedly after a certain time possibly means the device is broken and needs to be repaired.

Besides, existing methods focus more on anomaly detection, rather than repairing the detected anomaly [31]. If outliers are not repaired, then data mining on IoT data with outliers is likely to get wrong conclusions. Unfortunately, there are few systems or tools that realize the integration of IoT data from detection to repair. More importantly, it is very necessary for users to understand why the point is detected as an outlier and how the outlier is repaired to a normal point.

Developing a system for detecting and repairing temporal outliers, however, is nontrivial, given the following three challenges. (1) There are various outlier detection algorithms as aforesaid [11]. It is challenging to devise a uniform framework for explaining the outliers detected by various methods. (2) The engines for big data computing are various, ranging from batch processing to streaming computation. The system should be able to support multiple processing scenarios at the same time, e.g., batch processing and stream processing. (3) On the basis of anomaly detection, the system should also provide anomaly repair function. And the system had better provide multiple repair methods, e.g., repair methods based on prior knowledge or without prior knowledge.

In this case, TsOutlier [14] is proposed for anomaly detection of IoT data. And TsOutlier has proved to show good explanation of detected outliers in several use cases in real-world industrial scenarios. However, TsOutlier lacks the ability of repairing outliers. Therefore, our new tool TsClean, based on the outlier detecting ability of TsOutlier, has extended the functionality of outlier repairs. We adopt the SCREEN (prior knowledge required) method proposed in the article [29] as time series data anomaly repair method in TsClean. Furthermore, we devise Median Repair (no prior knowledge required) based on speed and acceleration [30] to provide more options of outlier repair in TsClean.

The main contribution of this article is TsClean, a uniform framework for explaining and repairing outliers detected by various algorithms over IoT data, as well as a uniform system supporting both batch and streaming processing. (1) We implement a set of 3-sigma algorithms based on original value, interval, speed, and acceleration in TsClean for outlier detection. Furthermore, TsClean performs distribution visualization on the front end to explain the outliers. (2) We implement several algorithms in TsClean for outlier repair, i.e., SCREEN and median repair (MR). Users are allowed to take appropriate repair methods to repair outliers, e.g., SCREEN with prior knowledge to set the maximum and minimum speeds or MR without prior knowledge. TsClean also combines front-end visualization to explain outlier repair, which means that outliers that fall outside the range of 3-sigma will be repaired within 3-sigma. (3) We take many IoT datasets for verification on TsClean and provide valuable case studies in Section 4. In addition, TsClean supports two processing scenarios, i.e., batch processing and real-time processing, and has been put into use in real production environments.

The remainder of this article is organized as follows: In Section 2, we describe the relevant background and research status of our work. We describe the detailed design of our new system TsClean in Section 3, including the window-based operation, outlier detection module with uniform profiles, and outlier repairs under speed constraints. In Section 4, we present several case studies of applying TsClean in industry and give a brief evaluation of the system in Section 5. The conclusions are given in Section 6.

2 RELATED WORK

Because cheap sensors will cause missed reading and unreliable reading, there will always be many outliers in the temporal data [16]. The existence of these dirty data causes quality problems with temporal data, and people who use these low-quality data can easily get inaccurate or even wrong conclusions. For example, a common phenomenon in temporal data is that part of the data is missing due to a short failure of the sensor, and the missing data may have important information. Therefore, it is particularly important to detect and repair outliers in temporal data.

The initial detection of outliers in temporal data is statistical-based methods [10]. With the development of research, more and more methods are used for outlier detection in temporal data, e.g., some methods in areas of machine learning and deep learning. Related supervised, semi-supervised, and unsupervised algorithms in machine learning area are applied to outlier detection of temporal data. For example, some clustering methods in machine learning, such as DBSCAN [28] and k-means [24], can be used for outlier detection of temporal data. And a sufficient survey on the outlier detection of temporal data has been conducted in Reference [11], where temporal data is divided into multiple types and investigations on outlier detection methods for various temporal data are given. Although there are many outlier detection methods, statistical methods are frequently used due to their computational flexibility [13]. Especially in real-time anomaly detection scenarios, most of the methods used in practice are statistical techniques, which are computationally lightweight [1]. Because our tool TsClean also supports real-time outlier detection, and 3-sigma is computational lightweight, which is an important reason why we choose 3-sigma. Besides, 3-sigma technique is simple but efficient in identifying outliers [25].

Beyond outlier detection, outlier explanation is valuable, which allows users to better understand the problem and take corresponding actions [12]. Research on the explanation of outliers is relatively rare and mostly recent [5, 8, 20–22]. Chalupsky and Lin [20] treat an explanation as a process of classification. For example, they use decision trees to generate outlier explanations, in which a set of decision rules are transformed into outlier explanations. Mejía-Lavalle and Vivar [21] compare the outliers with the points under other clusters and use distance equations to calculate the distance of different attributes to perform outlier explanation. Chen et al. [5] propose outlier detection and interpretation based on prediction. When a data point differs greatly from the value predicted by the model based on neighboring points, it can be considered an outlier. Then they define a template for outlier explanation, i.e., the outlier has a higher or lower value compared with the expected value in a specific attribute. Micenková et al. [22] use attribute subsets to explain outliers where the given outlier shows separability from the inliers. However, the above-mentioned methods on outlier explanation are either too theoretical and not user-friendly, or it is not applicable to IoT data. In our practice, we find that outliers in IoT data usually have large deviations from the average, which is very suitable for using k-sigma to detect and explain. In addition, 3-sigma has the characteristics of computationally lightweight, unsupervised, few parameters, and widely used in the industry (easy for users to understand).

An important aspect after outlier detection is outlier repair. One of the biggest challenges in data repair is that the causes of data errors are many and complicated. Karkouch et al. [18] think that researching the quality of IoT data is very important, and they analyze various factors that affect the data quality of IoT data. In fact, when the causes of the outliers are unknown, it is not convenient to take appropriate and effective data repair methods. Therefore, the explanation of the outliers is very important. Similarly, there are many methods to repair time series data, e.g., smoothing-based algorithm [19], constraint-based algorithm [29], statistics-based algorithm [30].

As mentioned above, there are many methods for detecting and repairing outliers in temporal data. However, it is very important to understand the causes of the outliers and take appropriate

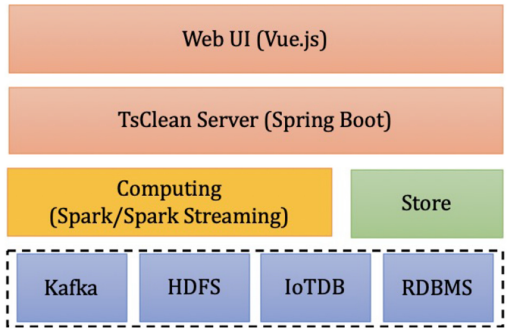

Fig. 1. Technology stack of TsClean.

repair methods to repair the outliers. Therefore, it is very important to explain the cause of the outliers and analyze the events behind the outliers. Unfortunately, few systems provide users with the ability to explain and analyze the causes and accidents of outliers. Even the existing EXAD system [26] uses a decision tree to explain the anomalies on big data traces without exploring temporal information fully.

In this case, we propose an outlier detection system TsOutlier [14] for explaining outliers with uniform profiles over IoT Data. We strive to use data profiling–related technologies to find outliers and statistically analyze the distribution of data and visualize outliers, explaining and analyzing the causes and potential events of outliers. Only after understanding the causes and events of the abnormality can more measures be taken, e.g., repairing the equipment causing the abnormality or repairing the abnormal value. Therefore, on the basis of TsOutlier, we further increase the function of outlier repair and propose a new system TsClean.

3 SYSTEM DESIGN

In this section, we will give a detailed description of the system design of TsClean. Figure 1 shows the technology stack of TsClean. Figure 2 presents the major GUI of the system. Figure 3 shows the processing flow of TsClean in batch and stream processing. Below, we will make a detailed description of these contents.

3.1 System Overview

Figure 1 describes the technology stack of TsClean. It can be seen that TsClean is developed by modules. The function of each module is as the name describes. Web UI Module is used to display the data profile and outlier detecting and repair of time series data. TsClean Server module is used to accept front-end requests and call algorithms. Computing module is the core module of TsClean, which implements batch processing and real-time stream processing. Store module encapsulates read and write operations on data sources. Inside the dotted mine are multiple data sources supported by TsClean, e.g., Kafka, HDFS, RDBMS. In each module, the content in brackets indicates the framework used by the module. For example, we use vue.js in the Web UI Module, Spring Boot framework in the Server Module and Spark for the core module of TsClean, i.e., Computing Module.

Figure 2 corresponds to the Web UI Module, which presents the major GUI of the system. The top left subplot in Figure 2 depicts the distribution of data through statistical histograms. In addition, the data that falls within the two red dotted lines with arrows is the normal data distribution range, and the data that falls outside the two red dotted lines with arrows are the abnormal points. The lower left sub-picture describes the distribution of the outlier and its neighbors and connects these

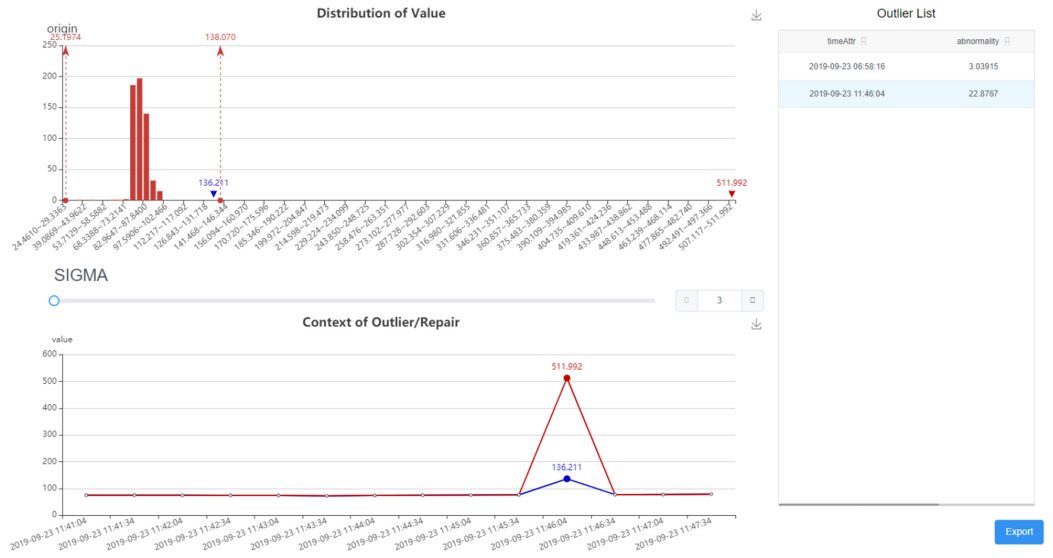

Fig. 2. Graphical user interface of TsClean.

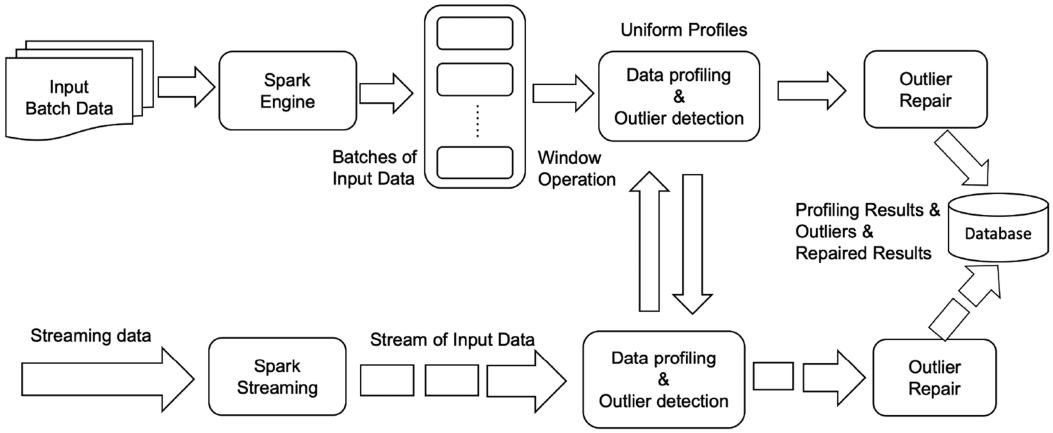

Fig. 3. Overview of the TsClean system in batch and steaming modes, implemented over Spark and Spark Streaming, respectively.

points with red lines. The red solid points represent abnormal points, and the blue solid points represent the values after the abnormal points are repaired. Correspondingly, the blue arrow on the histogram indicates the position of the value after the abnormal point is repaired, and the red arrow indicates the position of the value of the abnormal point. The right subgraph in Figure 2 shows a list of abnormal points that TsClean has found based on an algorithm. Clicking on each item in the list can display the distribution of the abnormal points.

Figure 3 corresponds to the core algorithm module Computing module. The system mainly consists of two parts of processes, i.e., first calculating profile information, and then detecting outliers according to the information as well as repairing them. As shown in Figure 3, the system is designed to handle data in both batch and streaming scenario, which can be implemented with different compute engines. For example, Spark can be used for batch processing, while Spark

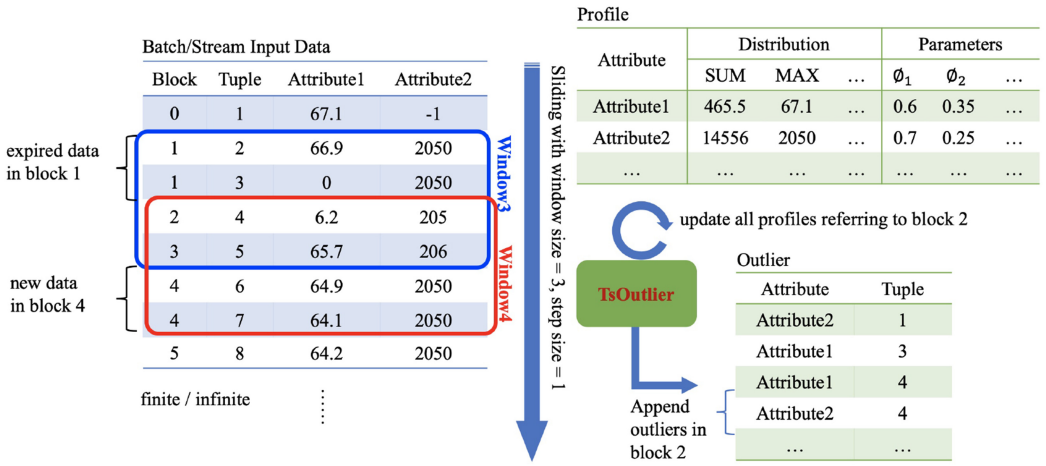

Fig. 4. Window-based operation.

Streaming and Flink can be applied for online streaming computing. The profiling and outlier detecting algorithms are composed of many window-based batch algorithms that can easily be adapted to batch or stream mode compute engines. In consideration of the inestimable large volume of batch data that cannot be put into the memory in the meantime, we divide them according to certain regulations into small batches and perform the profiling and detection work in parallel among them. While for stream data, it is natural to process data in different time windows. Additionally, many compute engines already have built-in windowed mechanism for stream data. After the outliers are found, we visualize the distribution of the outliers in the foreground and repair the outliers with speed constraints set by domain knowledge. Of course, if there is no domain knowledge or business knowledge, our MR method can be used to repair outliers. Assuredly, all the profiling results, outlier detection results, and repair results are stored into the database. In next part, we will describe the details of the window-based processing framework.

3.2 Window-based Processing Framework

Figure 4 shows the design of TsClean's window-based operation. The basic unit of the window is **block**. A **block** is the small batch of data divided by line number or timestamp (for batch data), or the batch of data at certain time (for stream data). In particular, by varying the block size, it provides a tradeoff between computing the accurate results and approximating with efficient incremental computation sometimes. It is required to specify two parameters (**window size** and **step size**) to define how window flows on the original data. The **window size** determines the number of blocks in each window, while the **step size** decides the number of blocks that are removed from the window when sliding to the next one.

Each window-based operation is expected to update some global states and to output several local results. For example, to calculate the sum of all the data, the sum of the data in each window should be added to the global SUM state. In addition, the outliers detected in each window should be outputted as local results and appended to the global results. However, the updater function may be different for batch and stream mode. Sometimes, it is required to traverse data more than once to get accurate result in batch mode, which is inapplicable in streaming scenario.

Table 1. Example of Uniform Profiles

Distribution statistics					
	Count	Sum	Quadratic sum	Max	Min
Original value	3,630	388,017.96	1.5055e11	106.899	105.877
Interval on time	3,629	20,838.988	4.3426e8	258	5
Speed on time	3,629	−2.6302e-4	6.91847	0.00112	−0.00121
Acceleration on time	3,628	2.6298e-6	6.91620e-4	0.00153	−0.00247
Error term of ARIMA	3,530	8,942.98	129,523.973	36.87	0

Model parameters				
	ϕ_1	ϕ_2	θ_1	θ_2
Parameters of ARIMA model	0.344	−0.0174	0.0464	0. 0517

3.3 Profiling and Outlier Detection Module

To get sufficient information for explaining the subsequent outlier detection, the uniform profiles should contain several contexts of the input data, such as (1) amount, (2) sum, (3) quadratic sum, (4) maximum value, (5) minimum value, (6) parameters in ARIMA model, (7) other information that user-defined outlier detection algorithm needs. With the help of uniform profiles, which are updated in processing of each window, we can easily outline the distribution of input data. For example, the *mean* and *variance*, which can be calculated by amount, sum, and quadratic sum in uniform profiles, are often used to estimate the parameters of a normal distribution.

ARIMA model (Autoregressive Integrated Moving Average model) is a time series model, and the methodology of ARIMA was proposed in Reference [2] to study time series. ARIMA is used for forecasting analysis of time series, e.g., electricity price forecasting [7], therefore it can be used for anomaly detection [23, 31]. As the name implies, the ARIMA model integrates autoregressive and moving average. And the ARIMA model is based on a stationary series. In an $ARIMA(p, d, q)$ model, p represents the number of autoregressive terms, q is the number of moving average terms, and d is the number of differences made to make the time series become stationary series. $ARIMA(p, d, q)$ model is expressed as Equation (1), where L is the lag operator, X is the time series value, and ϵ_t is white noise sequence with zero mean. ARIMA model is used in time series prediction, which can be used as outlier detection as well, by comparing the original value and the predicted value.

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right)(1 - L)^d X_t = \left(1 + \sum_{i=1}^{q} \theta_i L^i\right)\epsilon_t \tag{1}$$

Table 1 gives an example of uniform profiles over an attribute with ARIMA(2, 1, 2) model. Most profiles can be easily computed incrementally and the parameters of $ARIMA(p, d, q)$ can be calculated incrementally according to Reference [31].

The uniform profiles contain not only the distribution statistics of original value. Interval Δt indicates the data collecting frequency that is mostly a fixed value. Too large or small intervals are often considered as outliers, which possibly means missing or redundant data.

$$\Delta t_i = t_i - t_{i-1}$$

The speed v and acceleration u defined in References [29, 30] constrain the change speed of IoT data with timestamps and can intuitively describe why the outlier appears with its neighbor data

points in real-world situation.

$$v_i = \frac{x_i - x_{i-1}}{t_i - t_{i-1}} \tag{2}$$

$$u_i = \frac{x_{i+1} - x_i}{t_{i+1} - t_i} - \frac{x_i - x_{i-1}}{t_i - t_{i-1}} \tag{3}$$

Outlier detection algorithms are implemented as window-based operation. Since the detection process often needs contextual information, *step size* is set to (*window size−2*). The first and last block of the window are only used as extra information and are not considered as outlier candidates in the detection algorithm.

TsClean integrates with various outlier detection models, ranging from simple speed, acceleration models (online learned), to traditional statistical models ARIMA. Deep learning models like LSTM (offline learned) are also in the ongoing work.

3.4 Outlier Repairs Module

Dirty values are very common in IoT data. Song et al. [29] propose SCREEN, a constraints-based stream data cleaning method. SCREEN detects and repairs outliers based on maximum speed and minimum speed. Speed represents the rate of data changes and its definition formula is Equation (2). Maximum speed and minimum speed are meaningful in industrial production and real life. For example, some accumulated data indicators such as the mileage of the car should not decrease, which means the minimum speed of data changes should be greater than 0. And there are many other real cases in Reference [29] used to describe the maximum and minimum speeds.

The goal of SCREEN's repair is to make the repaired stream data meet the given speed constraints while minimizing the repair distance. Song et al. [29] propose speed-based global optimum and local optimum. Global optimum is more time-consuming than local optimum. Therefore, we apply speed-based local optimal repair to our tool TsClean for outlier repairs. The local optimal repair problem means that a data point x_k locally satisfies the speed constraint and it is defined as

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} |x_i - x_i'| \\ \text{s.t.} \quad & \frac{x_k' - x_i'}{t_k - t_i} \le s_{max}, \qquad t_k < t_i \le t_k + w, 1 \le i \le n \\ & \frac{x_k' - x_i'}{t_k - t_i} \ge s_{min}, \qquad t_k < t_i \le t_k + w, 1 \le i \le n, \end{aligned} \tag{4}$$

where x_i' is the value of data point x_i after repair, t_i is the timestamp of data point x_i, S_{max} is the maximum speed, S_{min} is the minimum speed, w is the window size.

We use SCREEN as a data repair method in TsClean. In fact, many industrial data in IoT, e.g., GPS data, vehicle speed, water temperature, and some accumulated data, have maximum and minimum speed requirements. Therefore, the speed constraint-based temporal data repair method SCREEN in TsClean can help to repair abnormal values in IoT data.

However, SCREEN needs to understand the business background and prior knowledge. For IoT data with insufficient prior knowledge, inspired by Median Filter [3], we further devise a median repair method based on the original value, speed, and acceleration and called it Median Repair (MR). When there is no prior knowledge, an intuitive idea is to use the average of neighbors to repair the outlier, e.g., Exponentially Weighted Moving Average [15]. However, because outliers can occur in succession, using the average of nearest neighbor points to repair them may not be

Table 2. The Datasets for Case Study

Dataset	Data size	Rows	Columns	Period	Equal interval
Water temperature data	35 KB	584	4	2019/9/23–2019/9/24	No
Subway operation data	81.7 MB	1,259,003	4	2018/12/01–2018/12/31	No
Vibration angle data	2,531 KB	67,389	2	2019/11/24–2019/11/25	No
Tank gauge data	350 KB	9,599	4	2019/5/6–2019/5/24	No

accurate. Moreover, experiments show that the average is more susceptible to outliers than the median [4]. Therefore, we decide to use the median of neighbors to repair outliers.

Different from Median Filter, MR repairs outliers based on the original value, speed, and acceleration. We use x_t to represent data point in time series, where t is the time of data point. Assuming that x_t is detected as an outlier, and x_t can be repaired using MR with the following Equation (5), where k represents k nearest neighbors before and after time t. It should be noted that in the original-value-based repair, x_t is the original data point. In the speed-based repair, x_t is the calculated speed data with timestamp, and in the acceleration-based repair, x_t is the calculated acceleration data with timestamp.

$$x_t = median(x_{t-k}, \ldots, x_{t-2}, x_{t-1}, x_{t+1}, x_{t+2}, \ldots, x_{t+k}) \tag{5}$$

For the MR based on the original value, the repair value can be directly obtained by Equation (5). Next, we will describe how MR based on speed and acceleration gets the repair value. For speed-based MR, we first obtain the median speed v_i by Equation (6), where v_i represents the calculated speed data with timestamp. And the repair value x_i can be calculated by Equation (7). It is easy to find that Equation (7) is inferred from Equation (2). When the outlier is detected by speed-based 3-sigma, if there is no prior knowledge, a reasonable repair method is using speed-based MR to repair.

$$v_i = median(v_{i-k}, \ldots, v_{i-2}, v_{i-1}, v_{i+1}, v_{i+2}, \ldots, v_{i+k}) \tag{6}$$

$$x_i = x_{i-1} + (t_i - t_{i-1})v_i \tag{7}$$

For acceleration-based MR, we first obtain the median acceleration u_i by Equation (8). And the repair value x_i can be calculated by Equation (9). Similarly, Equation (9) is transformed from Equation (3).

$$u_i = median(u_{i-k}, \ldots, u_{i-2}, u_{i-1}, u_{i+1}, u_{i+2}, \ldots, u_{i+k}) \tag{8}$$

$$x_i = \frac{(t_{i+1} - t_i)x_{i-1}}{(t_{i+1} - t_{i-1})} + \frac{(t_i - t_{i-1})x_{i+1}}{(t_{i+1} - t_{i-1})} - \frac{(t_i - t_{i-1})(t_{i+1} - t_i)u_i}{(t_{i+1} - t_{i-1})} \tag{9}$$

4 CASE STUDY

In this part, we present four case studies in real-world industry scenario. The data of these four cases are real sensor data of some companies, and TsClean is used to help detect, explain outliers, and repair them. In fact, the explanation of TsClean's outlier repair is to repair outliers that fall outside of 3-sigma into the range of 3-sigma. Table 2 shows the basic information of the datasets for our case study. Statistical explanations and repairing value are given for the detected outliers. Table 3 also shows the abnormal rates of different types of profiles over these industrial datasets. In general, the abnormal rate of tank gauge data is the lowest, while the abnormal rate of subway operation data is the highest. This is because the tank gauge data is measured with a high-precision liquid level gauge, thus the data is less likely to produce errors. The subway operation data has a longer time span (the entire month of December). During this period, the subway will be in a state

Table 3. Abnormal Rates of Different Types of Profiles over Various Real-world Industrial Datasets

Dataset	Original value	Interval	Speed	Acceleration	ARIMA model
Water Temperature Data	0.3460%	0.3460%	0.3460%	0.3472%	0.2873%
Subway Operation Data	0%	1.421%	2.358%	3.127%	1.811%
Vibration Angle Data	0.3622%	0.2954%	0.7867%	0.8609%	0.3503%
Tank Gauge Data	0%	0.3021%	0.02084%	0.04168%	0.04856%

of working and sometimes rest, resulting in the status data obtained may be very different, thus the overall abnormal rate is relatively large.

4.1 Excavator Water Temperature Data

Excavator water temperature data is real industrial data. Excavator water temperature data can be used to detect whether the working environment of excavators is normal. When the water temperature data of the excavator is too high, it indicates that the working environment of the excavator is abnormal, which may cause excavator damage, and so on. However, due to sensor measurement errors and data errors during data transmission, there are some erroneous data in excavator water temperature data. These erroneous data greatly affect the monitoring of the working environment of the excavator. Therefore, it is very important to detect and repair outliers of excavator water temperature data.

Figure 2 shows the results of outlier detection and repair based on SCREEN in excavator water temperature data. The upper subfigure of Figure 2 illustrates the distribution of attribute water temperature and the dotted line stands for the 3-sigma range border. Any point having a value out of such range is marked as an outlier. A line chart showing how the original water temperature value changes over time is plotted in the lower subfigure of Figure 2. By comparing to its neighbors, the highlighted outlier point exhibits a sudden spike, which is obviously abnormal in the process of temperature monitoring. As a matter of fact, it is impossible for the excavator's water temperature value to rise from less than 100 to more than 500 in just a few seconds.

And this outlier is detected by our anomaly detection algorithm based on the original value, speed, acceleration. We use Figure 2 to explain why this point is considered an outlier. The upper subfigure of Figure 2 shows the distribution of water temperature data, and any point that falls outside the dotted line (3-sigma line) in the figure is considered a statistically potential abnormal point. From the histogram, we can see that this point (the value is 511.992, and its position is marked with a red arrow) is far from the distribution area of most data. Therefore, the point is likely to be an outlier. Further, we use a line chart to depict the changes of this point and its neighbors, i.e., the lower subfigure of Figure 2. It can be seen that the water temperature has increased by more than 400 within a few seconds compared to the data point at the previous moment. And the specific performance of this sudden increase is that its speed will be large calculated by Equation (2) as well as its acceleration will be very small calculated by Equation (3). And too large speed and too small acceleration can be found by our anomaly detection algorithm based on speed and acceleration.

In summary, we use a histogram to show the distribution of data and a line chart to show the change of data over time. Combined with some common statistical results, based on visual information, we can explain why the point is regarded as an outlier and speculate on the cause of the abnormality. Water temperature data is the real sensor data of a company's equipment. Through detection and abnormal point display of TsClean, business personnel recognize that it is impossible to have such big change in water temperature. Therefore, we infer that the data is most likely to be a sensor measurement error or an error generated during data transmission and storage. Because

there is no such large temperature change at this point, to minimize the repair distance, we choose SCREEN to repair the outlier. We set the maximum and minimum rise and fall speed of water temperature to repair outliers. From the histogram in Figure 2, we can also see that the original outlier falls outside the 3-sigma range border, and after repairing, the repaired value (marked with a blue arrow) falls within the 3-sigma range border. We use front-end visualization to explain the outlier repair, i.e., the outliers that fall outside of 3-sigma are repaired into the range of 3-sigma.

The water temperature data is used to monitor whether the equipment is working normally, and some outliers such as excessive water temperature will cause an alarm. These false alarms caused by these outliers make domain experts have to verify whether the equipment is working properly, which brings them a lot of unnecessary workload. After detecting and repairing these outliers through TsClean, outliers are repaired to within the range of 3-sigma. These false alarms caused by outliers are automatically filtered, which greatly reduces the workload of domain experts.

4.2 Subway Operation Data

Operation data collected from a subway is analyzed in this use case. There are 53 attributes indicating the operation states of the subway, such as ground speed, load of vehicle frame, fuel consumption of traction, ID of the current station, along with the timestamp. Outliers often trigger the alarms. Subway staff should distinguish the true alarm, which potentially reflects the abnormal operation states. A good explanation of outliers will assist them to make a wiser decision.

We obtained data for multiple operation states of the subway in December. Table 2 only shows one operation state of the data. We use TsClean to analyze the data and find that there is no data conflict in the operation data, i.e., there are not two different tuples at the same time. Certainly, we also find some abnormal value changes, e.g., the temperature of the measuring point suddenly dropped from 20 °C to 0 °C, the opening time of a vehicle door suddenly dropped from 30,000 milliseconds to 0 milliseconds. We combine the meaning of the operation data and the visualization of the abnormal points; we can find that when the subway door appears in the state of 3,000 milliseconds opening, it is actually getting on and off, and when it is in the open state of 0 milliseconds, it is in the driving process. For such outliers, we do not need to repair. This kind of outlier can be detected by anomaly detection algorithms based on speed and acceleration, and the explanation of these outliers is similar to the previous case.

From Table 3, we can see that the subway data has the highest anomaly rate in the interval among the four cases. And in the process of analyzing subway data with TsClean, it was found that the average collection interval of subway data is about 2 seconds, while the maximum interval can reach 64,244 seconds (about 18 hours). Through displaying these outliers in the form of a line chart, we can explain and infer the cause of the anomaly. For example, for an outlier with an interval of up to 18 hours, we can infer that the cause may be that the device is turned off or the sensor may be broken. If the staff determines that there is no problem with the sensor, then this abnormality may be that the device is shut down normally, hence no action is required at that time. While for those outliers with interval of several minutes, we can infer that the cause may be data loss in the process of transmission. Users can choose whether to fill missing data or not. Through anomaly discovery and interpretation, staff can understand and discover the cause of anomalies. Our tool allows the staff to take further actions, e.g., setting the maximum and minimum speed to repair relatively large spike errors. More importantly, through the explanation of outliers, the staff can spot faults in subway equipment and take further steps.

Subway operation data is used to monitor the operating status of the subway. TsClean conducts outlier detection on subway operation data, visualizes the outliers, and analyzes whether there is a malfunction in the subway. The subway monitoring personnel use the statistical analysis function of TsClean to have a general understanding of the data and analyze whether there

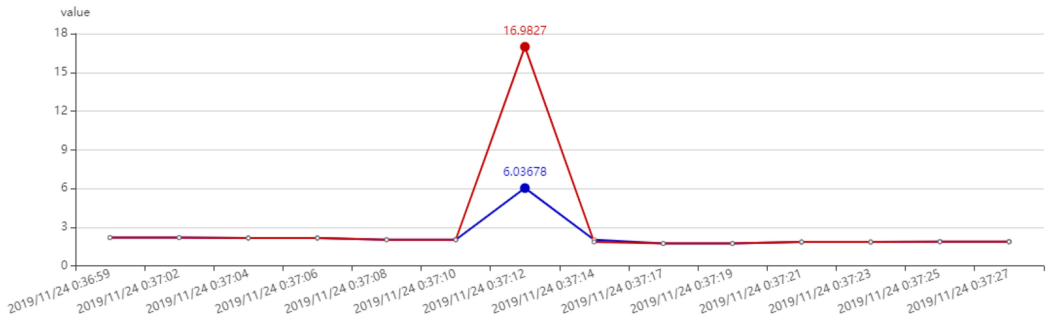

Fig. 5. Outlier detected by acceleration.

is a potential malfunction in the subway through anomaly detection. With the help of TsClean, they can directly obtain the summary information of the data and better realize the monitoring of the subway operating status.

4.3 Vibration Angle Data

The vibration angle data comes from many excavators, and it is calculated from the angular acceleration of the excavator in the *X*, *Y*, and *Z* axes. We can analyze whether the excavator is in danger of overturning by analyzing the change of the vibration angle data and further monitor the working status of the excavator.

Similarly, we detect outliers in the data based on the 3-sigma principle of multiple algorithms. For example, in the case of outliers of vibration data found through 3-sigma detection of acceleration, we found that some abnormal points reflected a very large angle of the vibration, e.g., the outlier in Figure 5. Anomaly detection algorithm based on acceleration is usually used to find some jumping data points. Jumping data points usually refer to a large change between the current point and the two sequential data points before and after. In the vibration angle data, the larger the vibration angle, the more the excavator leans to one side. The excavator vibration angle shown in Figure 5 suddenly increases from about 3 to about 17 and then returns to about 3. Such outliers are the jumping data points mentioned above. In fact, for the excavator, when the vibration angle reaches about 17, it means that it will be dumped, and it is impossible to recover its vibration angle to about 3 again. Therefore, through our anomaly detection and visual analysis of outliers, we can combine business knowledge to explain why the point is an outlier.

Hence, in combination with the actual operating environment of the excavator, the vibration angle cannot be increased so much, so we set the maximum change rate through SCREEN to repair the outlier. In addition, TsClean analyzes that the acquisition interval is abnormal in the vibration angle data, i.e., the acquisition interval between the two data points is too long or too short. In the visualization of the acquisition interval outliers, it is found that there are multiple acquisition intervals of zero, i.e., there are two different values at the same timestamp, which means data conflict. This phenomenon may be caused by a defective sensor or a data error during data transmission and storage.

Vibration angle data is used to monitor the working status of the excavator. Monitoring personnel analyze the vibration data to determine whether the excavator is in danger of overturning. Similarly, due to sensor measurement errors and errors caused during transmission and storage, some erroneous data triggers false alarms. These false alarms bring troubles and additional workload to the monitoring personnel. The monitoring personnel use TsClean's SCREEN repair method to repair the outliers and realize the filtering of false alarms.

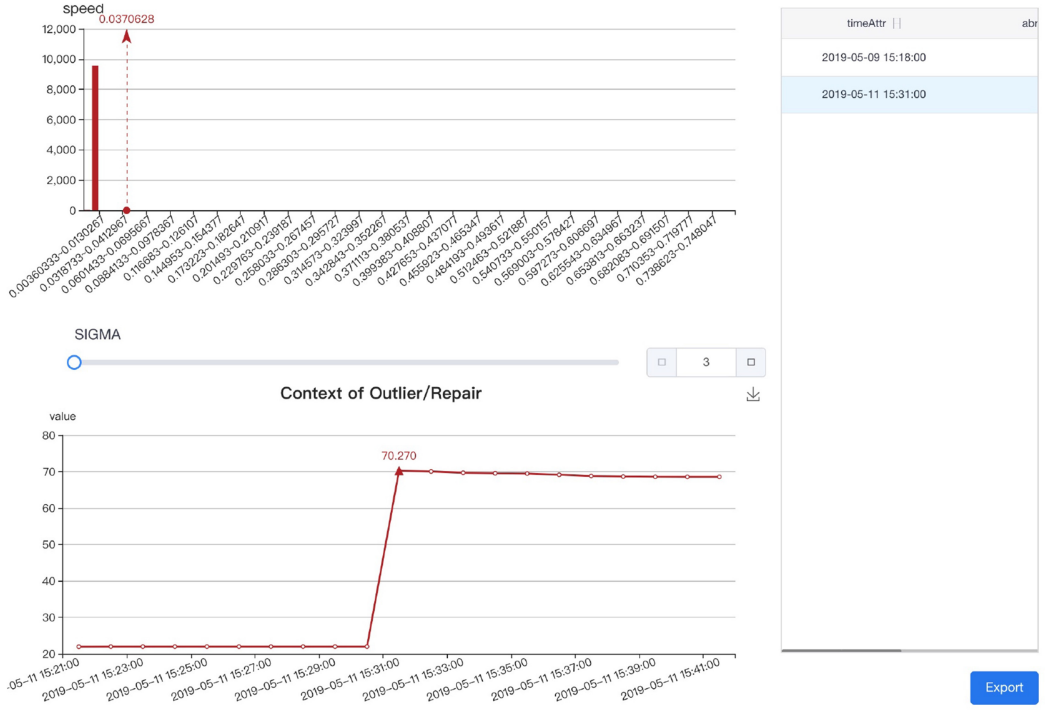

Fig. 6. Outlier detected by speed.

4.4 Tank Gauge Data

The final use case gives an analysis about outlier detection in tank gauge data of a car. Under normal circumstances, the liquid level should slowly drop with the car and stop changing when the car is turned off; only when the car is refueling will the liquid level rise. Incorrect data will affect the company's judgment on the consumption of the vehicle during the driving process.

Once the level data is abnormal, such as falling rapidly in a short time, we can detect these abnormalities through our 3-sigma principle of several algorithms, especially by 3-sigma based on speed algorithm to judge vehicle fueling and fuel consumption abnormality. For example, Figure 6 shows the outlier found based on speed algorithm. Figure 6 shows that the amount of fuel in the fuel tank increases rapidly. The right sub-picture of Figure 6 shows the list of abnormal points detected by the speed-based abnormal detection algorithm. It can be seen that the speed-based anomaly detection algorithm detects two outliers. Below, we use Figure 6 to explain the cause of the anomaly. The upper subgraph of Figure 6 shows that the normal speed calculated by 3-sigma should be less than 0.0370628 (the dotted line with arrow). The following sub-graph shows that the amount of oil in the fuel tank increases from about 20 (actually 21.97) to about 70 (actually 70.270) in 1 minute. The speed is 0.805 calculated by Equation (2), which is much greater than 0.0370628; therefore, this point is detected as an outlier. And through this line chart, we can see that the amount of fuel in the fuel tank has been stable at about 20 before this time. This sudden increase can be inferred that the driver refuels the car. And this detected outlier does not need to be repaired.

As can be seen from the above example, through TsClean's outlier detection and visualization, we can find out what happened in reality represented by time series, e.g., the driver refuels the car. After explaining and inferring the cause of the anomaly, we can decide whether to fix the

Table 4. Evaluation Dataset

Dataset	Number of classes	Size of training set	Size of testing set	Time series length
Synthetic Control	6	300	300	60
Two Patterns	4	1,000	4,000	128

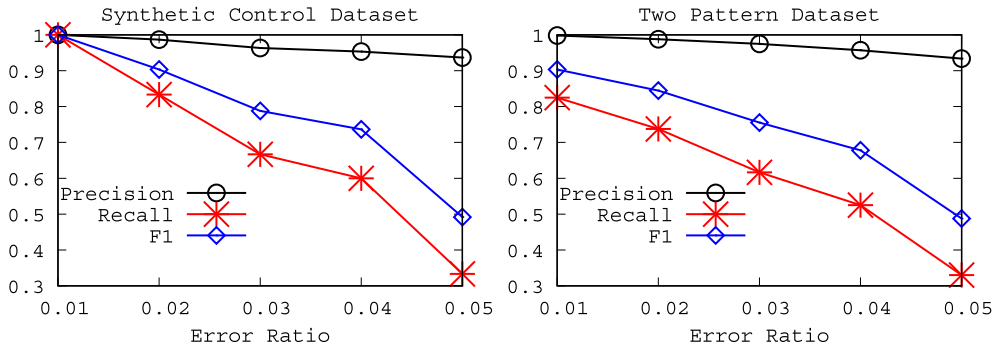

Fig. 7. Precision, recall, and F-value on various error ratios.

anomaly or take no action. Tank gauge data of a car is used to monitor whether the fuel volume changes normally during the driving of the car. TsClean can detect these sudden increases and sudden drops in oil, and these events correspond to refueling and oil spills. The driver refueling incident given in the above case has received feedback from the business personnel, hence the business personnel use TsClean to detect vehicle refueling events automatically.

5 PERFORMANCE EVALUATION

In this section, we will show the evaluation for outlier detection and repairs. Evaluation in outlier detection is focused on the precision, recall, and F-value under different settings of error ratio. For evaluation in outlier repairs, we take root-mean-square error (RMSE) and accuracy of classification as the evaluation index.

For evaluation datasets, we take UCR Time Series [6] dataset to perform our evaluation experiment. And the basic information of evaluation datasets is shown in Table 4. These two datasets are divided into training dataset and test dataset, and their size indicates the number of tuples, and the length of the time series indicates the number of columns in each data point. In the experiment, we use the training dataset to train the classifier and use the test dataset to evaluate outlier detection and outlier repairs. In addition, we manually insert errors into any position of each data point based on the error ratios and we set the error ratio to range from 0.01 to 0.05, increasing the error ratio by 0.01 each time.

As Figure 7 illustrates, precision, recall, and F-value all decrease with the increasing of error ratio. It is reasonable that the variance increases gradually as more error points are introduced in the series. As a result, more points lie within 3-sigma section and outliers cannot be distinguished easily anymore. In real-world scenario, outliers occur with a relatively small probability generally (as shown in Table 3). Therefore, most outliers can be detected and possibly explained.

For the evaluation of anomaly repair, we compare the RMSE and classification accuracy of the dirty data and the repaired data after the use of SCREEN or MR. We can find from Figure 8 that after outlier repairs of TsClean, the RMSE of the repaired data is lower than the dirty data on both datasets. In addition, we can find that the RMSE of the data repaired using MR is lower than the data

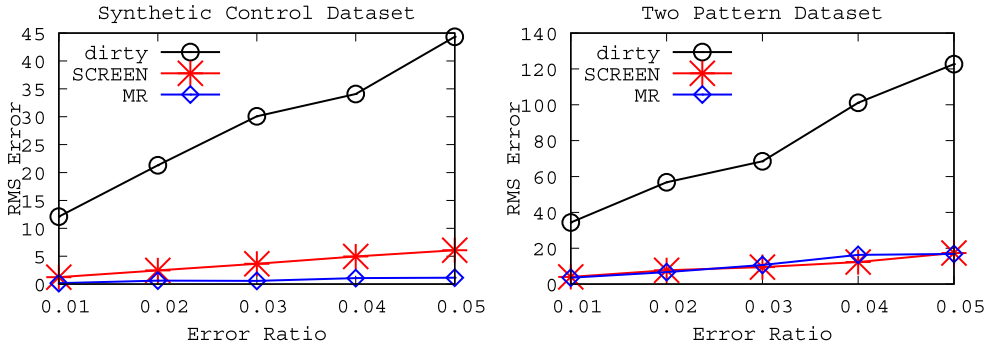

Fig. 8. RMSE of data repaired by SCREEN and MR on various error ratios.

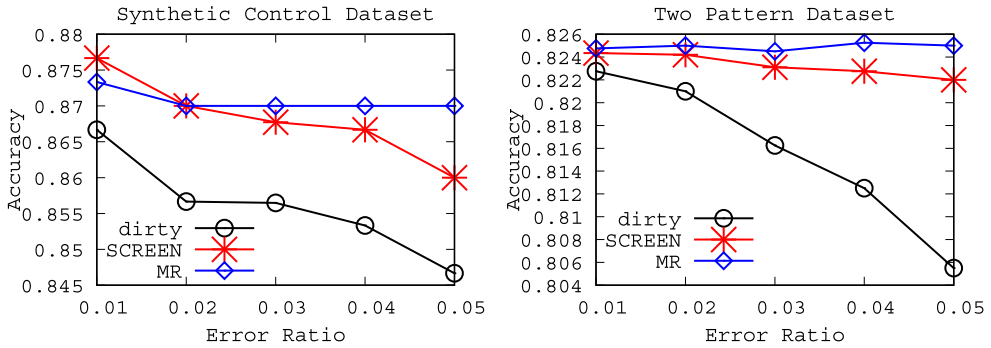

Fig. 9. Classification accuracy of data repaired by SCREEN and MR on various error ratios.

repaired by SCREEN on the Synthetic Control dataset and is close to the data repaired by SCREEN on the Two Patterns dataset. Furthermore, it can be seen from Figure 9 that after data repair by TsClean, the classification accuracy on the repaired data is higher than the classification accuracy on dirty data. On the two datasets, the classification accuracy of data based on MR repair is close to that of data based on SCREEN repair, and sometimes even higher. This is because SCREEN is fixed based on the minimum repair distance, which may still be far from the real data. The MR is repaired based on the median of the neighboring points, since the data will not have a sudden change in most cases, and more often it is close to the value of the neighboring points. Therefore, MR method is reasonable and the experiments have proved that MR can reduce RMSE and improve classification accuracy of dirty data.

6 CONCLUSIONS AND FUTURE WORK

Outliers in IoT data are very common, but few systems support outlier detection, repair, and explanation at the same time. In this article, we present TsClean, a system for detecting, repairing, and explaining outliers in IoT data. The major highlights of TsClean include: (1) using a variety of detection algorithms based on the 3-sigma rule to detect outliers; (2) implementing SCREEN and MR to help repair the outliers and providing front-end visualization to explain outlier repair, i.e., repair the outliers into the range of 3-sigma; (3) supporting both batch processing and real-time stream processing. Case studies of applying this system in real-world industry scenario demonstrate promising outlier detection, reasonable explanation results, and reliable outliers repair. It is also worth mentioning that TsClean has been deployed on the servers of some partner companies

for IoT data outlier detection and repair for a long time, and its usability and effect have been verified. Moreover, TsClean implements statistical analysis of data, uses histograms to indicate the distribution of data, and calculates the abnormal rate of IoT data. Therefore, TsClean can also help data scientists have a general understanding of data and help them conduct scientific research.

Finally, we discuss the generalizability of the proposed system. In this article, we mainly focus on IoT data. For multi-dimensional time series, TsClean can also be used for outlier detection and repair, where TsClean performs outlier detection and repair for each field separately. The current anomaly detection implemented by TsClean is all aimed at IoT data, e.g., establishing a distribution model for a single attribute. For multi-dimensional time series, if there is no correlation between each attribute, then statistical analysis, anomaly detection, and anomaly repair are performed on each attribute separately, which is exactly the same as IoT data. The difference is that if there are association rules in a multi-dimensional time series, it is more reasonable to use association rules for anomaly detection and anomaly repair than the current algorithm. Therefore, TsClean cannot yet combine information such as differential dependencies [27] to achieve global outlier detection. The realization of anomaly detection and repair of multi-dimensional time series based on association rules and differential dependencies will be the next direction for TsClean to expand.

In the future, we will allow users to customize the rules for anomaly detection, i.e., allowing users to combine their domain knowledge and experience to find outliers in temporal data. In addition, we will add more temporal data repair methods to the tool. Combined with the outlier interpretation function of our tool, it allows users to choose the appropriate temporal data repair method to repair the outliers. More importantly, we will extend the system from IoT to multi-dimensional time series and correspondingly increase the method of anomaly detection and repair for multi-dimensional time series.

REFERENCES

[1] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. 2017. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing* 262 (2017), 134–147. DOI: https://doi.org/10.1016/j.neucom.2017.04.070

[2] G. E. P. Box and G. M. Jenkins. 2010. Time series analysis : forecasting and control. *Journal of Time* 31, 3 (2010).

[3] D. R. K. Brownrigg. 1984. The weighted median filter. *Commun. ACM* 27, 8 (1984), 807–818. DOI: https://doi.org/10.1145/358198.358222

[4] Wei Cao, Yusong Gao, Bingchen Lin, Xiaojie Feng, Yu Xie, Xiao Lou, and Peng Wang. 2018. TcpRT: Instrument and diagnostic analysis system for service quality of cloud databases at massive scale in real-time. In *Proceedings of the International Conference on Management of Data (SIGMOD'18)*. 615–627. DOI: https://doi.org/10.1145/3183713.3190659

[5] Liang-Chieh Chen, Tsung-Ting Kuo, Wei-Chi Lai, Shou-De Lin, and Chi-Hung Tsai. 2012. Prediction-based outlier detection with explanations. In *Proceedings of the IEEE International Conference on Granular Computing (GrC'12)*. 44–49. DOI: https://doi.org/10.1109/GrC.2012.6468672

[6] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. 2015. The UCR Time Series Classification Archive. Retrieved from www.cs.ucr.edu/~eamonn/time_series_data/.

[7] Javier Contreras, Rosario Espinola, Francisco J. Nogales, and Antonio J. Conejo. 2003. ARIMA models to predict next-day electricity prices. *IEEE Trans. Power Syst.* 18, 3 (2003), 1014–1020.

[8] Xuan-Hong Dang, Barbora Micenková, Ira Assent, and Raymond T. Ng. 2013. Local outlier detection with interpretation. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD'13)*. 304–320. DOI: https://doi.org/10.1007/978-3-642-40994-3_20

[9] Yijun Duan, Adam Jatowt, and Katsumi Tanaka. 2019. Discovering latent threads in entity histories. *Data Sci. Eng.* 4, 4 (2019), 336–351. DOI: https://doi.org/10.1007/s41019-019-00108-x

[10] Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. 2014. *Outlier Detection for Temporal Data*. Morgan & Claypool Publishers. DOI: https://doi.org/10.2200/S00573ED1V01Y201403DMK008

[11] Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. 2014. Outlier detection for temporal data: A survey. *IEEE Trans. Knowl. Data Eng.* 26, 9 (2014), 2250–2267. DOI: https://doi.org/10.1109/TKDE.2013.184

[12] Nikhil Gupta, Dhivya Eswaran, Neil Shah, Leman Akoglu, and Christos Faloutsos. 2018. Beyond outlier detection: LookOut for pictorial explanation. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD'18)*. 122–138. DOI: https://doi.org/10.1007/978-3-030-10925-7_8

[13] Riyaz Ahamed Ariyaluran Habeeb, Fariza Nasaruddin, Abdullah Gani, Ibrahim Abaker Targio Hashem, Ejaz Ahmed, and Muhammad Imran. 2019. Real-time big data processing for anomaly detection: A survey. *Int. J. Inf. Manag.* 45 (2019), 289–307. DOI: https://doi.org/10.1016/j.ijinfomgt.2018.08.006

[14] Ruihong Huang, Zhiwei Chen, Zhicheng Liu, Shaoxu Song, and Jianmin Wang. 2019. TsOutlier: Explaining outliers with uniform profiles over IoT data. In *Proceedings of the IEEE International Conference on Big Data (Big Data'19)*. 2024–2027. DOI: https://doi.org/10.1109/BigData47090.2019.9006232

[15] J. Stuart Hunter. 1986. The exponentially weighted moving average. *J. Qual. Technol.* 18, 4 (1986), 203–210.

[16] Shawn R. Jeffery, Gustavo Alonso, Michael J. Franklin, Wei Hong, and Jennifer Widom. 2006. Declarative support for sensor data cleaning. In *Proceedings of the 4th International Conference on Pervasive Computing (PERVASIVE'06)*. 83–100. DOI: https://doi.org/10.1007/11748625_6

[17] Shawn R. Jeffery, Minos N. Garofalakis, and Michael J. Franklin. 2006. Adaptive cleaning for RFID data streams. In *Proceedings of the 32nd International Conference on Very Large Data Bases*. 163–174. Retrieved from http://dl.acm.org/citation.cfm?id=1164143.

[18] Aimad Karkouch, Hajar Mousannif, Hassan Al Moatassime, and Thomas Noël. 2016. Data quality in internet of things: A state-of-the-art survey. *J. Netw. Comput. Applic.* 73 (2016), 57–81. DOI: https://doi.org/10.1016/j.jnca.2016.08.002

[19] Eamonn J. Keogh, Selina Chu, David M. Hart, and Michael J. Pazzani. 2001. An online algorithm for segmenting time series. In *Proceedings of the IEEE International Conference on Data Mining*. 289–296. DOI: https://doi.org/10.1109/ICDM.2001.989531

[20] Shou-De Lin and Hans Chalupsky. 2008. Discovering and explaining abnormal nodes in semantic graphs. *IEEE Trans. Knowl. Data Eng.* 20, 8 (2008), 1039–1052. DOI: https://doi.org/10.1109/TKDE.2007.190691

[21] Manuel Mejía-Lavalle and Atlántida Sánchez Vivar. 2009. Outlier detection with explanation facility. In *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM'09)*. 454–464. DOI: https://doi.org/10.1007/978-3-642-03070-3_34

[22] Barbora Micenková, Raymond T. Ng, Xuan-Hong Dang, and Ira Assent. 2013. Explaining outliers by subspace separability. In *Proceedings of the IEEE 13th International Conference on Data Mining*. 518–527. DOI: https://doi.org/10.1109/ICDM.2013.132

[23] Eduardo H. M. Pena, Marcos V. O. de Assis, and Mario Lemes Proença Jr. 2013. Anomaly detection using forecasting methods ARIMA and HWDS. In *Proceedings of the 32nd International Conference of the Chilean Computer Science Society (SCCC'13)*. 63–66. DOI: https://doi.org/10.1109/SCCC.2013.18

[24] Umaa Rebbapragada, Pavlos Protopapas, Carla E. Brodley, and Charles R. Alcock. 2009. Finding anomalous periodic time series. *Mach. Learn.* 74, 3 (2009), 281–313. DOI: https://doi.org/10.1007/s10994-008-5093-3

[25] Siwoon Son, Myeong-Seon Gil, and Yang-Sae Moon. 2017. Anomaly detection for big log data using a Hadoop ecosystem. In *Proceedings of the IEEE International Conference on Big Data and Smart Computing (BigComp'17)*. 377–380. DOI: https://doi.org/10.1109/BIGCOMP.2017.7881697

[26] Fei Song, Yanlei Diao, Jesse Read, Arnaud Stiegler, and Albert Bifet. 2018. EXAD: A system for explainable anomaly detection on big data traces. In *Proceedings of the IEEE International Conference on Data Mining Workshops (ICDM Workshops)*. 1435–1440. DOI: https://doi.org/10.1109/ICDMW.2018.00204

[27] Shaoxu Song, Lei Chen, and Hong Cheng. 2014. Efficient determination of distance thresholds for differential dependencies. *IEEE Trans. Knowl. Data Eng.* 26, 9 (2014), 2179–2192. DOI: https://doi.org/10.1109/TKDE.2013.84

[28] Shaoxu Song, Chunping Li, and Xiaoquan Zhang. 2015. Turn waste into wealth: On simultaneous clustering and cleaning over dirty data. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams (Eds.). ACM, 1115–1124. DOI: https://doi.org/10.1145/2783258.2783317

[29] Shaoxu Song, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2015. SCREEN: Stream data cleaning under speed constraints. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 827–841. DOI: https://doi.org/10.1145/2723372.2723730

[30] Aoqian Zhang, Shaoxu Song, and Jianmin Wang. 2016. Sequential data cleaning: A statistical approach. In *Proceedings of the International Conference on Management of Data*. 909–924. DOI: https://doi.org/10.1145/2882903.2915233

[31] Aoqian Zhang, Shaoxu Song, Jianmin Wang, and Philip S. Yu. 2017. Time series data cleaning: From anomaly detection to anomaly repairing. *PVLDB* 10, 10 (2017), 1046–1057. DOI: https://doi.org/10.14778/3115404.3115410

Received February 2020; revised October 2020; accepted November 2020