Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6692390, 11 pages
https://doi.org/10.1155/2021/6692390

Research Article

A Robust Data-Driven Method for Multiseasonality and Heteroscedasticity in Time Series Preprocessing

Bin Sun,[1] **Liyao Ma**,[1] **Tao Shen**,[1] **Renkang Geng**,[1] **Yuan Zhou**,[2] **and Ye Tian**[3]

[1]*School of Electrical Engineering, University of Jinan, Jinan 250022, China*
[2]*Blekinge Institute of Technology, Karlskrona 37179, Sweden*
[3]*China Information Communication Technologies Group Corporation (CICT), Wuhan 430074, China*

Correspondence should be addressed to Liyao Ma; cse_maly@ujn.edu.cn and Tao Shen; cse_st@ujn.edu.cn

Received 27 November 2020; Accepted 28 July 2021; Published 16 August 2021

Academic Editor: Shaohua Wan

Internet of Things (IoT) is emerging, and 5G enables much more data transport from mobile and wireless sources. The data to be transmitted is too much compared to link capacity. Labelling data and transmit only useful part of the collected data or their features is a promising solution for this challenge. Abnormal data are valuable due to the need to train models and to detect anomalies when being compared to already overflowing normal data. Labelling can be done in data sources or edges to balance the load and computing between sources, edges, and centres. However, unsupervised labelling method is still a challenge preventing to implement the above solutions. Two main problems in unsupervised labelling are long-term dynamic multiseasonality and heteroscedasticity. This paper proposes a data-driven method to handle modelling and heteroscedasticity problems. The method contains the following main steps. First, raw data are preprocessed and grouped. Second, main models are built for each group. Third, models are adapted back to the original measured data to get raw residuals. Fourth, raw residuals go through deheteroscedasticity and become normalized residuals. Finally, normalized residuals are used to conduct anomaly detection. The experimental results with real-world data show that our method successfully increases receiver-operating characteristic (AUC) by about 30%.

1. Introduction

Together with rapid development of 5G, the connection requirement of wireless devices is also developing due to the eased connectivity and much shorter (in milliseconds) delay. A result is that Internet of Things (IoT) technologies are now used by more than a quarter of mainstream business compared to 13% six years ago. A great number of industry companies started to put attention on their IoT time series data, including but not limited to health care [1] and transportation [2]. While lots of mobile vehicles are connected to the IoT network as data sources [3], much more data is produced. On one aspect, it is an opportunity for machine learning-based data processing methods. On the other aspect, data transmission is now more challenging.

Moving and remote data source create a challenge that it is hard to send data, especially using wireless ways, as it is still expensive to use limited wireless resource to transfer data even for 5G service providers. In some situations, if real-time moving vehicle information is needed while radio signal is limited, then wireless and wired connection may be both needed to provide support together [4, 5]. This situation is shown in Figure 1.

For this situation, one way to solve it is to label data near to sources. It not only reduces the amount of data to transfer but also balances the computing load between edges and centres [6]. One more benefit is that labelling different types of data is good for later prediction [7]. However, most solutions require labelled data to train labelling models or human expert rich experience to configurate parameters.

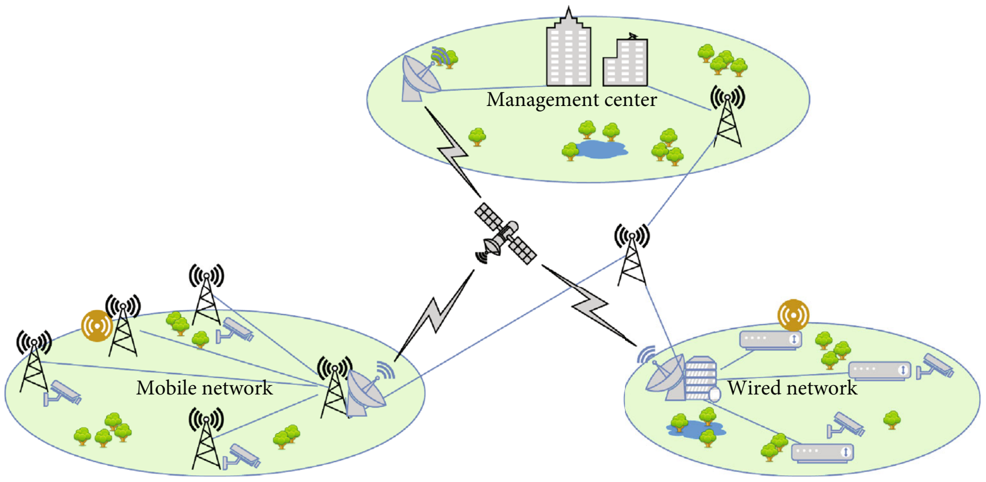

Figure 1: Data transmission across multiple 5G wireless and wired networks among sources, edges, and centres.

In this work, we try to solve this problem by enhancing data preprocessing. Our previous initial feasibility experiments show promising results [8] and we complete the design here. The main contributions of this work include detailed steps of the data-driven method to handle heteroscedasticity of Internet of Things (IoT) data and comparison of possible unsupervised labelling methods as well as analysis of the reasons.

The remaining content is organized as follows. First, the section introduces the problem and related definitions, together with previous research that tried to tackle this problem. Second, the proposed method is thoroughly documented in the section including steps of data preprocessing, model building, model adaptation, residual matrix construction, and anomaly detection. Third, the section describes a series of experiments using real-world data that are carried out in order to evaluate and compare the performance of the proposed method in terms of different metrics. Finally, experimental results are shown and analysed in the section, and conclusions are made in the section.

2. Background and Related Work

Here, we consider a system with a centre node. IoT data processing happens across the entire system [9]. It starts as early as the source application data part as shown in the updated TCP/IP architecture in Figure 2. Example source application data include camera images, video streaming, temperature, and other environmental sensed values [10]. The sensed data are then sent via possible networking routing which could be fully used for distributed processing [11], especially together with the application layer [12–14]. Physical layer choice matters as the emergency level and importance level differ among transported data which should be optimized carefully [15, 16]. When the data finally arrive at the centre, data mining algorithms could be applied [17] to analyse and conduct prediction in most cases.

Regarding labelling and detection of anomalies in time series, much work has been done. Previous work can be categorized in different ways from different aspects [18]. A typical categorization includes the following categories. Probability-based methods calculate a density distribution and use some kind of thresholds to the distribution centre to label anomalies [19]. Distance-based methods set thresholds regarding how far an instance deviates from its neighbours. The measurement can be defined distances, such as in k-nearest neighbours [20], or some kind of cost of separation such as decision tree-based methods [21]. Reconstruction-based methods catch patterns and calculate the expected values of instances to get the difference, i.e., residuals, and then use residuals to conduct labelling [22, 23]. Boundary-based methods, such as support vector machine [24], provide a boundary or hyperplane to separate abnormal instances from normal ones. In addition, ensemble methods can be used to improve the accuracy and robustness of above methods [25]. For the above-mentioned methods, reconstruction-based methods give not only residuals but also comprehensive patterns and models. Thus, this work focuses on providing a preprocessing procedure to calculate and standardize residuals as the first step of reconstruction-based methods.

For reconstructed residuals, as the original saved data is huge and long-term, one common problem is the variance of residuals are time-dependent, i.e., heteroscedasticity [26]. Using traffic flow as an example, the variance is high during noon time when the flow itself is high as shown in Figure 3. Vice versa, the flow and its variance are both low after midnight. This causes problems for labelling algorithms as many of them cannot distinguish high variances with anomalies.

During literature review, we found two methods that try to solve the above two problems at the same time. One method is SARIMA-GARCH (Seasonal Auto-Regressive Integrated Moving Average-Generalized Auto-Regressive Conditional Heteroscedasticity) [27]. Another one is TBATS (Trigonometric Box-cox transform, ARMA errors, Trend and Seasonal component) [26]. Thus, those two methods are also tested in this work. For the final detection part, SHESD (Seasonal Hybrid Extreme Studentized Deviate test) [28] shows promising results in experiments [29–31] and is

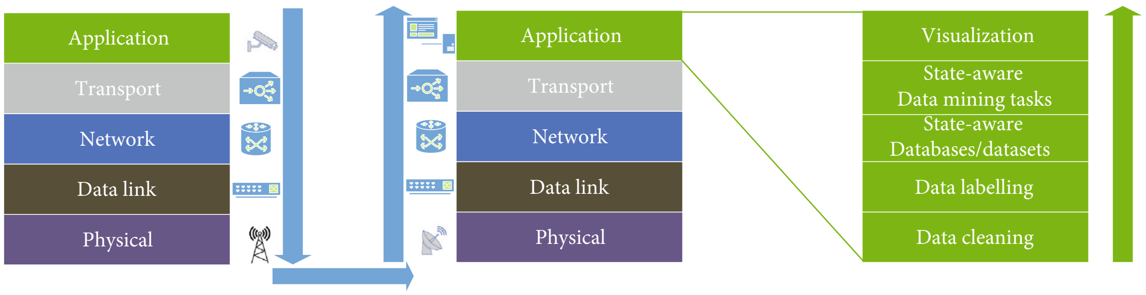

Figure 2: Data flow and process architecture.

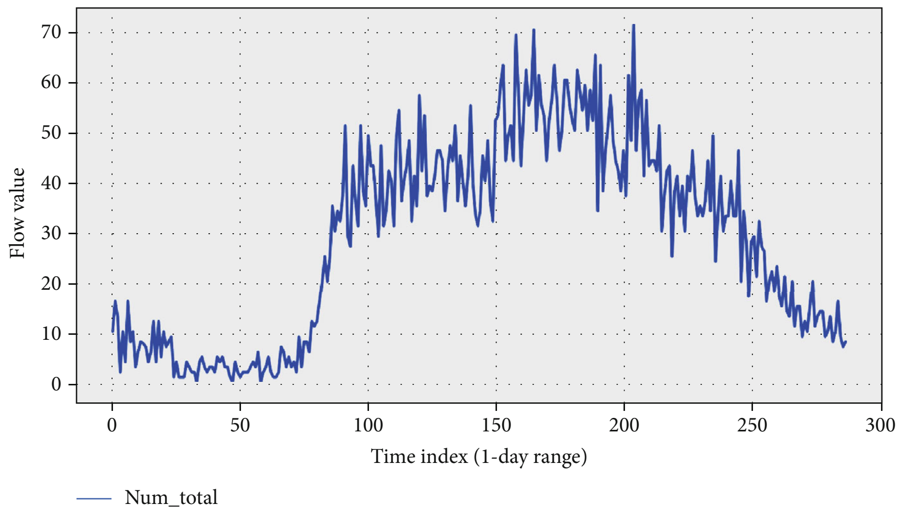

Figure 3: Overview of typical time series with heteroscedasticity.

used here. It is worth mentioning that there are plenty of alternative methods while this work focuses on preprocessing.

3. Methodology

The proposed method includes three main steps which are preprocessing, building day-of-week (DOW) models, and solving flow-level-heteroscedasticity problem. This part describes the method in detail. The entire procedure is summarized in Figure 4.

3.1. Preprocess Data. In this part, data are loaded and then divided into seven groups according to day of week.

For consecutive zeros (continuous three or more zeros) which means controlled access or device malfunction, set flags and replace the instances with *null*:

$$v_i^{\text{flag}} = \begin{cases} 1, & \text{if } r_i \text{ is in consecutive zeros,} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where r_i is the ith measured flow rate value.

Instead of using original natural daily periods, we use a new starting point. The purpose is to find a base where the starting flow rates of seasons are low and similar so that the robust fitting could work better in latter steps. It is worth mentioning that (daily) seasons may start from other time than midnight. Actually, the starting point is calculated to be around 3 am in the experiments.

$$N^{\text{seasons}} = \left\lfloor \frac{N^{\text{origin}}}{N^{\text{periods}}} \right\rfloor, \quad (2)$$

where N^{seasons} is the number of complete seasons, N^{origin} is the original number of instances (about 288×406 days), and N^{periods} is the number of periods (i.e., instances) per day (e.g., 288 per day for 5-minute interval data).

All complete seasons are put together to construct a matrix:

$$R = \left[s_1 \; s_2 \; \cdots \; s_{i_s} \; \cdots \; s_{N^{\text{seasons}}} \right], \quad (3)$$

$$= \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,i_s} & \cdots & r_{1,N^{\text{seasons}}} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,i_s} & \cdots & r_{2,N^{\text{seasons}}} \\ \vdots & \vdots & \ddots & & & \vdots \\ r_{i_p,1} & r_{i_p,2} & & r_{i_p,i_s} & & r_{i_p,N^{\text{seasons}}} \\ \vdots & \vdots & & & \ddots & \vdots \\ r_{N^{\text{periods}},1} & r_{N^{\text{periods}},2} & \cdots & r_{N^{\text{periods}},i_s} & \cdots & r_{N^{\text{periods}},N^{\text{seasons}}} \end{bmatrix}, \quad (4)$$

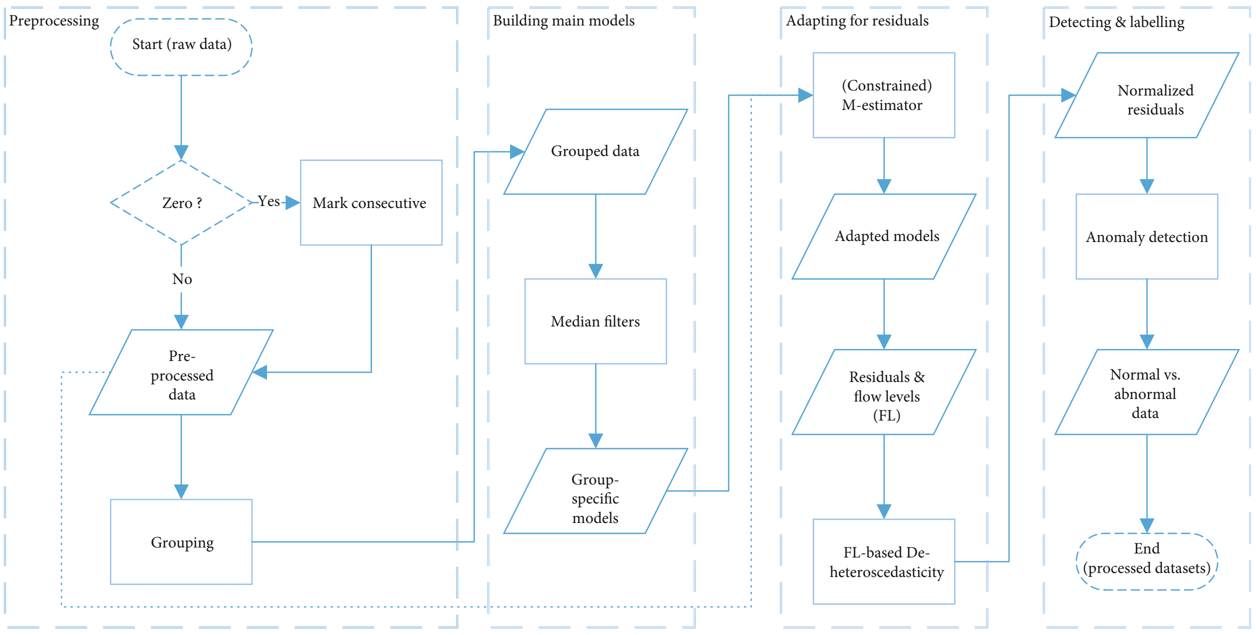

FIGURE 4: Summary of the proposed procedure.

with each season constructing a column, e.g., $s_1 = [r_1, r_2, \cdots, r_{N^{\text{periods}}}]^T$.

Then, separate the seasons/columns into groups; here, we use the day of week of the season starting point as the criteria; thus, there are 7 groups ($G_1, \cdots G_7$) with similar number of instances for each group.

$$G_i = \{s_{7n+i} \mid n = 0, 1, 2, \cdots \text{and } 7n + i \le N^{\text{seasons}}\}. \qquad (5)$$

3.2. Build the Main Models. Now, seven day-of-week (DOW) models are built with the key concept of median. The building algorithm is designed in the way that it can set up several workers in parallel to improve building performance.

To get a specific model $\mathbf{M}_i$, a matrix $\check{\mathbf{R}}_{i_m}$ is constructed by using all seasons (all columns) of G_i:

$$\check{\mathbf{R}}_{i_m} = \begin{bmatrix} \check{r}_{1,1} & \check{r}_{1,2} & \cdots & \check{r}_{1,i_r} & \cdots & \check{r}_{1,N^{M_i}} \\ \check{r}_{2,1} & \check{r}_{2,2} & \cdots & \check{r}_{2,i_r} & \cdots & \check{r}_{2,N^{M_i}} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \check{r}_{i_p,1} & \check{r}_{i_p,2} & & \check{r}_{i_p,i_r} & & \check{r}_{i_p,N^{M_i}} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \check{r}_{N^{\text{periods}},1} & \check{r}_{N^{\text{periods}},2} & \cdots & \check{r}_{N^{\text{periods}},i_r} & \cdots & \check{r}_{N^{\text{periods}},N^{M_i}} \end{bmatrix}. \qquad (6)$$

N^{M_i} is the number of complete seasons for a specific model m_i.

Seven DOW models M_i^{dow} ($i = 1, \cdots, N^{\text{models}}$ where N^{models} is 7 in this paper) are built by applying median filters to $\check{\mathbf{R}}_{i_m}$.

We can present all models as columns of a matrix:

$$M = \left[m_1 \quad m_2 \quad \cdots \quad m_{i_m} \quad \cdots \quad m_{N^{\text{models}}} \right], \qquad (7)$$

$$= \begin{bmatrix} \tilde{r}_{1,1} & \tilde{r}_{1,2} & \cdots & \tilde{r}_{1,i_m} & \cdots & \tilde{r}_{1,N^{\text{models}}} \\ \tilde{r}_{2,1} & \tilde{r}_{2,2} & \cdots & \tilde{r}_{2,i_m} & \cdots & \tilde{r}_{2,N^{\text{models}}} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \tilde{r}_{i_p,1} & \tilde{r}_{i_p,2} & & \tilde{r}_{i_p,i_m} & & \tilde{r}_{i_p,N^{\text{models}}} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \tilde{r}_{N^{\text{periods}},1} & \tilde{r}_{N^{\text{periods}},2} & \cdots & \tilde{r}_{N^{\text{periods}},i_m} & \cdots & \tilde{r}_{N^{\text{periods}},N^{\text{models}}} \end{bmatrix}, \qquad (8)$$

where the $i_m = 1, 2, \cdots, N^{\text{models}}$ indicates model index and the $i_p = 1, 2, \cdots, N^{\text{periods}}$ indicates time point (period) index of day. Thus,

$$\tilde{r}_{i_p,i_m} = \text{med}\left(\text{row}_{i_p} \check{R}_{i_m}\right), \qquad (9)$$

where $\check{R}_{i_p,i_m} = \{\check{r}_i \in R\}$, i.e., $\check{R}_{i_p,i_m} \subset R$ and contains all r's with the time point index of day i_p which belongs to model M_{i_m}.

3.3. Adapt like Regressors. This part calculates fitted models using M-estimation considering the above model matrix and each individual season.

An M-estimator is then computed iteratively with reweighted least squares (IRLS):

$$\beta^{(t+1)} = \arg\min_{\beta} \sum_{i_p=1}^{N^{\text{periods}}} w_{i_p}\left(\beta^{(t)}\right) \left|\varepsilon_{i_p}(\beta)\right|^2, \qquad (10)$$

where the scaling and addition parameters $\beta = [k, b]$, and residuals from the previous fit (using season i_s belongs to model i_m as an example):

$$\varepsilon(\beta) = s_{i_s} - f_a\left(m_{i_m}, \beta_{i_s,i_m}\right) = col_{i_s} R - \left(k_{i_s,i_m}\, col_{i_m} M + b_{i_s,i_m}\right). \tag{11}$$

Thus, the residual matrix:

$$E = \begin{bmatrix} \varepsilon_{1,1} & \varepsilon_{1,2} & \cdots & \varepsilon_{1,i_s} & \cdots & \varepsilon_{1,N^{\text{seasons}}} \\ \varepsilon_{2,1} & \varepsilon_{2,2} & \cdots & \varepsilon_{2,i_s} & \cdots & \varepsilon_{2,N^{\text{seasons}}} \\ \vdots & \vdots & \ddots & & & \vdots \\ \varepsilon_{i_p,1} & \varepsilon_{i_p,2} & & \varepsilon_{i_p,i_s} & & \varepsilon_{i_p,N^{\text{seasons}}} \\ \vdots & \vdots & & & \ddots & \vdots \\ \varepsilon_{N^{\text{periods}},1} & \varepsilon_{N^{\text{periods}},2} & \cdots & \varepsilon_{N^{\text{periods}},i_s} & \cdots & \varepsilon_{N^{\text{periods}},N^{\text{seasons}}} \end{bmatrix}. \tag{12}$$

During the estimation, the weights are calculated as:

$$w = \left[w_1, \cdots, w_{i_p}, \cdots, w_{N^{\text{periods}}}\right] = \frac{\psi_\gamma(\varepsilon/c)}{\varepsilon}, \tag{13}$$

where c is a scaling factor:

$$c = \frac{\text{med}(abs(\varepsilon))}{\eta}, \tag{14}$$

and ψ is in Huber family:

$$\psi_\gamma(x) = \begin{cases} x, & \text{if} |x| \le \gamma, \\ \gamma \ \text{sign}\ (x), & \text{if} |x| > \gamma, \end{cases} \tag{15}$$

while η is a constant 0.675 and γ is 1.345 which correspond to regression estimator 95% efficiency. If M-estimation fails (rarely), then constrained M-estimation (CM) [32] is used (which is always working for our data). CM is proposed by Mendes and Tyler for regression and is more robust while keeping the same breakdown point (i.e., 1/2) though slower.

3.4. Construct the Residual Matrix. While having the adapted models, the raw residuals can be calculated directly. However, the raw residuals contain different variations on different flow levels. Thus, this part also removes flow-level-related heteroscedasticity.

For adapted models, i.e., $f_a(m, \beta)$, let us take values of adapted models and round them to integers then we get flow levels as integers l of each time point.

$$L = \lfloor f_a(m, \beta) \rceil, \tag{16}$$

$$= \left[\hat{s}_1 \quad \hat{s}_2 \quad \cdots \quad \hat{s}_{i_s} \quad \cdots \quad \hat{s}_{N^{\text{seasons}}}\right], \tag{17}$$

$$= \begin{bmatrix} l_{1,1} & l_{1,2} & \cdots & l_{1,i_s} & \cdots & l_{1,N^{\text{seasons}}} \\ l_{2,1} & l_{2,2} & \cdots & l_{2,i_s} & \cdots & l_{2,N^{\text{seasons}}} \\ \vdots & \vdots & \ddots & & & \vdots \\ l_{i_p,1} & l_{i_p,2} & & l_{i_p,i_s} & & l_{i_p,N^{\text{seasons}}} \\ \vdots & \vdots & & & \ddots & \vdots \\ l_{N^{\text{periods}},1} & l_{N^{\text{periods}},2} & \cdots & l_{N^{\text{periods}},i_s} & \cdots & l_{N^{\text{periods}},N^{\text{seasons}}} \end{bmatrix}. \tag{18}$$

Be aware that the flow levels are rounded from adapted model values instead of measured. For example, suppose 9 am traffic is 85 in the DOW model, 90.3 in the adapted model, but only 10 in the measured traffic (due to an incident or so); then, the traffic flow level is 90, i.e., flow level is a adapted and generalized description which represents what the traffic should be during a similar day.

Suppose the minimum and maximum integers (levels) in L are:

$$l_{\min} = \min\left(l_{i_p,i_s}\right), l_{\max} = \max\left(l_{i_p,i_s}\right), \quad l_{i_p,i_s} \in L, \tag{19}$$

then we can generate a level vector

$$\check{L} = \left[l_{\min} \quad l_{\min}+1 \quad \cdots \quad l_{i_l} \quad \cdots \quad l_{\max}\right], \tag{20}$$

$$= \left[\check{l}_1 \quad \check{l}_2 \quad \cdots \quad \check{l}_{i_l} \quad \cdots \quad \check{l}_{N^{\text{levels}}}\right], \tag{21}$$

which contains all integers from $l_{\min}$ to $l_{\max}$ and $N^{\text{levels}} = l_{\max} - l_{\min} + 1$ denotes the number of total flow levels.

For all level items/values in adapted models L, adapted models do element-wise XNOR logic and we get a mask matrix A with ones indicating the time points/instances with flow levels of l_{i_l}.

$$A_{i_l} = l_{i_l} \,\bar{\oplus}\, L = \left\{a_{i_a,j_a} \mid i_a = 1, 2, \cdots, N^{\text{periods}}; j_a = 1, 2, \cdots, N^{\text{seasons}}\right\}, \tag{22}$$

$$a_{i_a,j_a} = \begin{cases} 1, & \text{if } l_{i_p,i_s} = l_{i_l}, \\ \text{null}, & \text{otherwise}. \end{cases} \tag{23}$$

Let us apply this mask A_{i_l} to E and take all the matched values then calculate the variance (standard deviation) for an arbitrary level *null* items and related calculation are ignored during this process.

$$\check{E}_{i_l} = A \cdot E = \left\{\varepsilon_{i_p,i_s} \mid a_{i_p,i_s} = 1\right\}, \tag{24}$$

$$v_{i_l} = std\left(\check{E}_{i_l}\right). \tag{25}$$

The variances for different levels vary, thus heteroscedasticity. When putting all variances for all levels to get a variance/-heteroscedasticity vector, note that residuals from neighbour levels are used when the amount of residuals is insufficient.

```
1: procedure DOW-FLH (Original Time Series)
2:     set flags for consecutive zeros ▹Handel Dirty Data
3:     for each day do
4:         find the time point index (TPI) of the lowest flow
5:     end for
6:     find TPIs' median number as starts of daily seasons, e.g., 3 am
7:     for model $m_{i_m}$ in all DOW models do▹Build DOW Models
8:         take all seasons related to $m_{i_m}$ to a group
9:         remove flagged consecutive zeros
10:        calculate median of grouped seasons as the model $m_{i_m}$
11:    end for
12:    for model $m_{i_m}$ in all DOW models do ▹Fit/Adapt to Get Scalings $k$ and Additions $b$
13:        for each realted season do
14:            remove flagged consecutive zeros
15:            estimate $k$, $b$ by robustly fitting $m_{i_m}$ to the season
16:            rounding all values of the fitted model to integers as the season's flow levels
17:            get residuals as the difference between the fitted and the season
18:        end for
19:    end for
20:    for each flow level Standardize Residuals (FLH) do▹Standardize Residuals (FLH)
21:        take all residuals for this flow level (or with neighbours if not enough)
22:        calculate standard deviations (STD)
23:    end for
24:    consider all STDs with all flow levels as the flow level heteroscedasticity (FLH)
25:    divide each residual with timely corresponding STD to standardize
26:    for each detection algorithm do ▹Detection
27:        feed the entire standardized residual time series to the algorithm
28:        get algorithm-specific anomalies or anomaly scores
29:    end for
30:    return the list of anomalies or anomaly scores
31: end procedure
```

Algorithm 1: DOW-FLH Modelling for Data Preprocessing

$$V = \begin{bmatrix} v_1 & v_2 & \cdots & v_{i_l} & \cdots & v_{N^{\text{levels}}} \end{bmatrix}. \quad (26)$$

Later, all residuals **E** are divided by the time point's level's variance to get "normalized residuals." First, for levels of each time point, i.e., l_{i_p,i_s}, find its corresponding variance:

$$\hat{i}_l(i_p, i_s) = \underset{i_l}{\arg} \text{ where } l_{i_p,i_s} = \check{l}_{i_l}, \quad (27)$$

$$\hat{v}_{i_p,i_s} = v_{\hat{i}_l(i_p,i_s)}. \quad (28)$$

Generate a matrix of all residual's corresponding variance:

$$\hat{V} = \begin{bmatrix} \hat{v}_{1,1} & \hat{v}_{1,2} & \cdots & \hat{v}_{1,i_s} & \cdots & \hat{v}_{1,N^{\text{seasons}}} \\ \hat{v}_{2,1} & \hat{v}_{2,2} & \cdots & \hat{v}_{2,i_s} & \cdots & \hat{v}_{2,N^{\text{seasons}}} \\ \vdots & \vdots & \ddots & & & \vdots \\ \hat{v}_{i_p,1} & \hat{v}_{i_p,2} & & \hat{v}_{i_p,i_s} & & \hat{v}_{i_p,N^{\text{seasons}}} \\ \vdots & \vdots & & & \ddots & \vdots \\ \hat{v}_{N^{\text{periods}},1} & \hat{v}_{N^{\text{periods}},2} & \cdots & \hat{v}_{N^{\text{periods}},i_s} & \cdots & \hat{v}_{N^{\text{periods}},N^{\text{seasons}}} \end{bmatrix}. \quad (29)$$

Normalized residuals are:

$$R' = \left\{ r'_{i_p,i_s} \mid i_p = 1, 2, \cdots, N^{\text{periods}}; i_s = 1, 2, \cdots, N^{\text{seasons}} \right\}, \quad (30)$$

where

$$r'_{i_p,i_s} = \frac{r_{i_p,i_s}}{\hat{v}_{i_p,i_s}}. \quad (31)$$

3.5. Detect Using Normalized Residuals. Finally, normalized residuals are sent to detection algorithms. The entire procedure is also presented in pseudocode (Algorithm 1).

4. Experiments

This section describes data, practical procedure, and the way we conduct experiments.

4.1. Data Specification. The one-year long real-world data are collected from a highway. Ground truth anomaly (incidents) labels are generated by using the extended system mentioned in [33]. The data are imputed using the method from [34] before any processing. One device sends a monitored flow

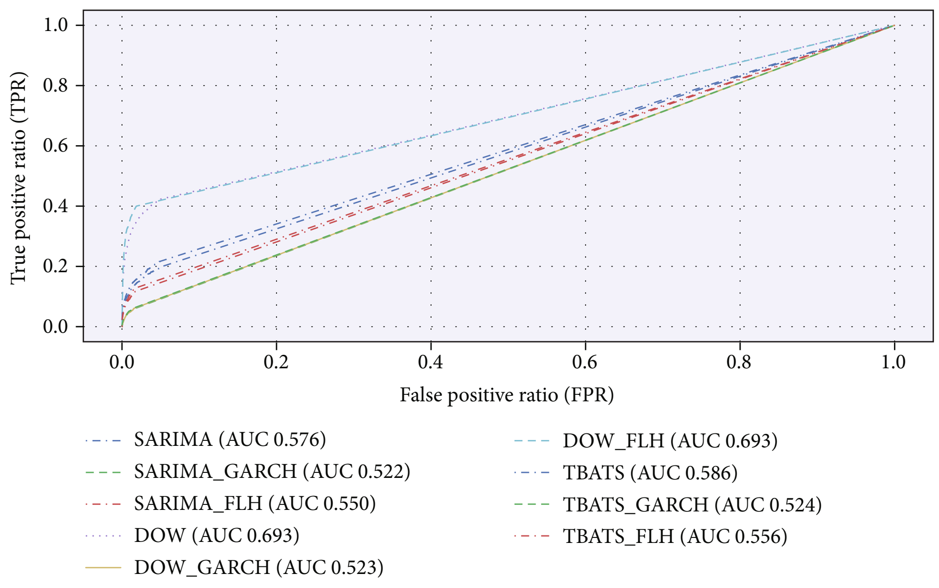

FIGURE 5: The proposed DOW and DOW-FLH gives similar AUC and around 27% bigger coverage than the others on average. Besides, DOW-FLH produces only half false positives compared to DOW without FLH.

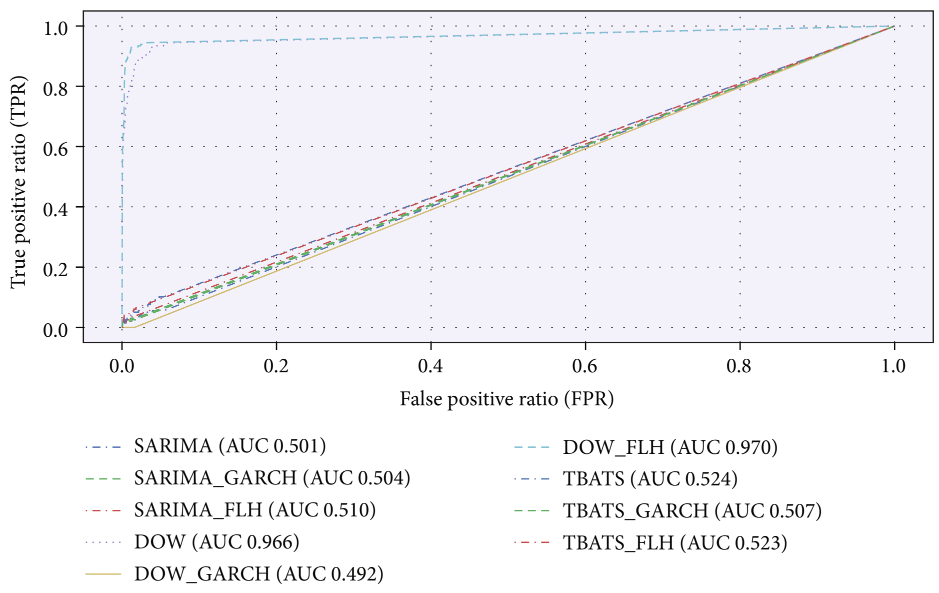

FIGURE 6: Only DOW-based algorithms can detect device malfunction efficiently while others are close to no discrimination line.

record at five-minute intervals. Each record contains some traffic statistics such as flow rate and average speed. This road carries undersaturated flow except in holidays' noons, where is 15 min average?

4.2. Experimental Setup. The experiments are done in a desktop computer with AMD Ryzen 5-3600 (6 Cores, 3600 MHz) and 16 GB DDR4 memory. To be fair, we only implement our method; other algorithms are taken from public domain such as GitHub.

Our implementation is done in the R programming environment version 3.4.3 with RStudio 1.3.1056, AnomalyDetection 1.0, forecast 8.2, feather 0.3.3 as well as the Python programming environment version 3.6.7/3.6.9 with library arch 4.8.1, statsmodels 0.9.0, feather-format 0.4.0/0.4.1, numpy 1.16.0/1.19.4, pandas 0.23.4/1.1.4, scikit-learn 0.19.2/0.23.2, scipy 1.2.2/1.5.4, ipykernel 5.3.4, and ipython 7.16.1.

SHESD was originally implemented to give only binary results so we modified it by adding test$_r$esult – critical$_v$alue to get anomaly/outlier scores. Also, as the max allowed anomaly (outliers) ratio is 50%, we mark all nontested ones the same score as the lowest score.

4.3. Evaluation Measurement and Metrics. Receiver-operating characteristic (ROC) is used as the main evaluation metric as it provides an accurate and visualized way to present detecting results. One important value from ROC is area

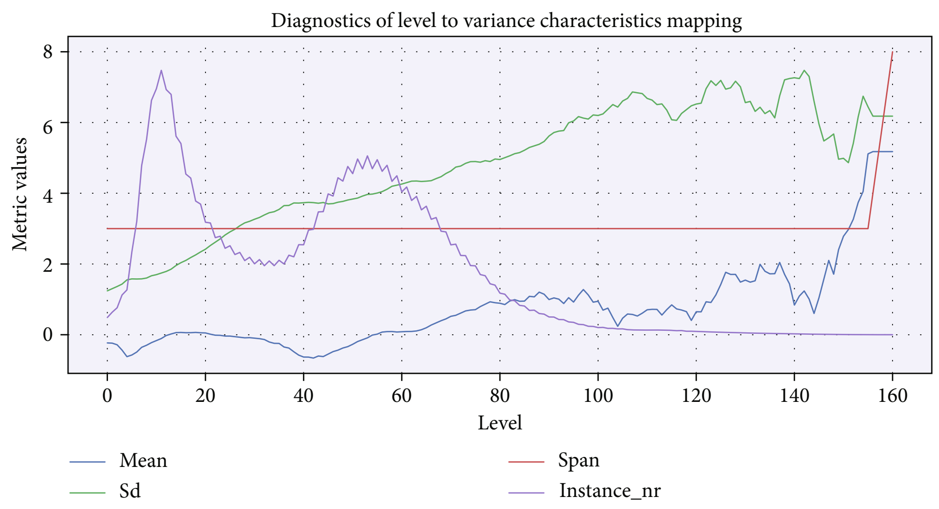

Figure 7: Flow level with corresponding instance numbers and mean, STD, and span of residuals (instance number per level is scaled down to be shown in this figure).

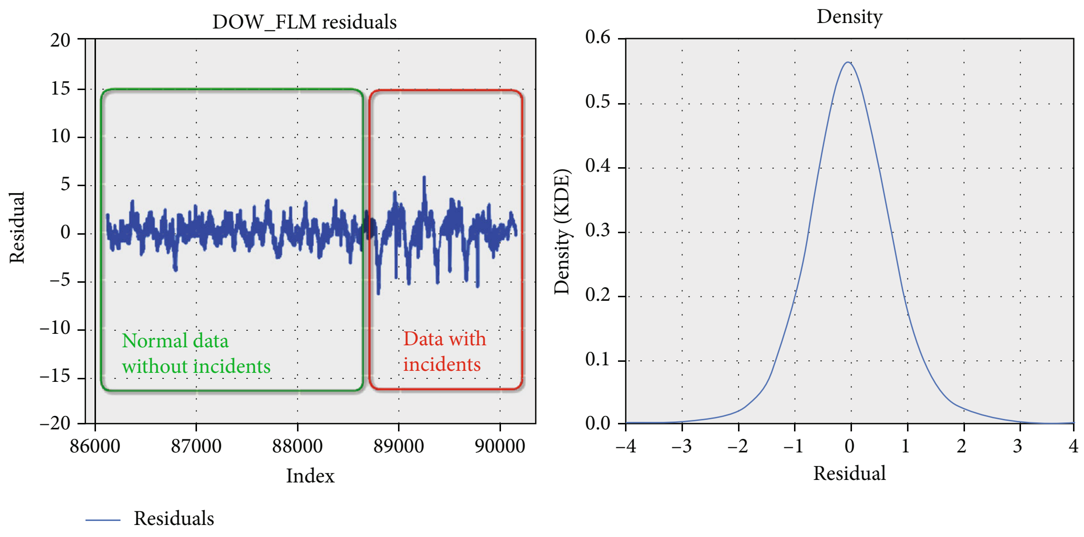

Figure 8: The left one shows DOW-FLH residuals of time series data without and with abnormal data and the right one shows overall residual density for the entire dataset. DOW-FLH successfully suppressed heteroscedasticity for normal data.

under curve (AUC) which is also known as A$'$ ("a-prime"), or concordance-statistic (c-statistic). It is a measure of goodness of fit that is often used for binary classification modelling results evaluation; therefore, we use it here.

5. Results and Analysis

As shown in Figure 5, our DOW and DOW-FLH methods are superior with regard to AUC. DOW with and without FLH performs similar considering AUC of 0.693 from both algorithms which are 26.9% better coverage than other algorithms on average (AUC 0.546). What is more, DOW-FLH is preferred for less false positives on the optimal cut-off point compared to DOW without FLH due to the data sensitivity to false positive. May move below to analysis? For unbalanced datasets such as traffic flows, this behaviour gives positive influence. The reason is that some false-negative instances introduce only minor issues for true-negative ones as negative instances are majority while the same amount false-positive instances impact true anomalies (incidents) much more.

We analysed the detection ratio and AUCs for different situations and found some interesting results. For device malfunction incidents, most algorithms cannot notice it as shown in Figure 6. The possible reason is that other algorithms are tracking no-flow situation without considering normal situation. Note that good seasonal modelling (DOW) should work with suitable variance handling methods, as inappropriate variation handling (i.e., GARCH) may otherwise reduce the effectiveness.

Figure 7 shows level to residual characteristics diagnostics. The mean of residuals (blue line) is mostly under 2 but

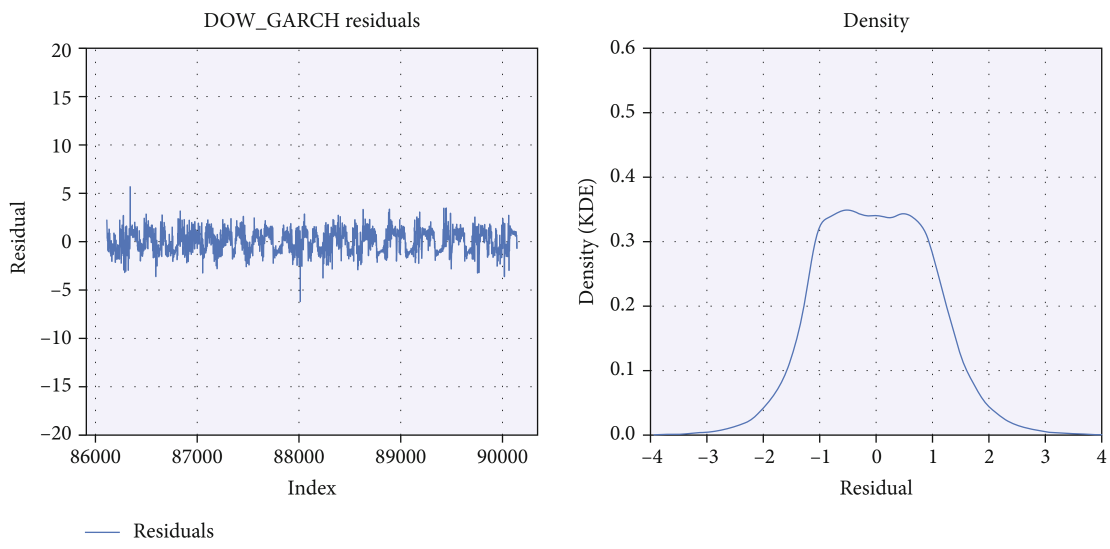

FIGURE 9: DOW-GARCH suppresses not only extreme values and heteroscedasticity for data with all normal instances but also for data with abnormal instances, when being compared to DOW-FLH (Figure 9) (same data range).

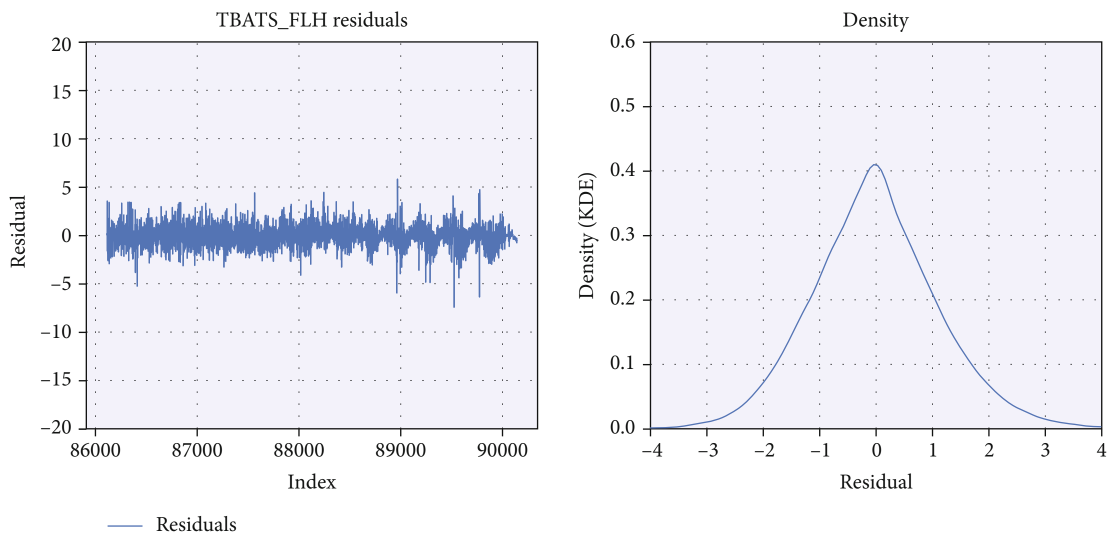

FIGURE 10: TBATS-based algorithms produce residuals with much noise when being compared to DOW-FLH (Figure 9) (same x-axis time range). (S)ARIMA-based algorithms give similar results.

increases rapidly to be about 5 when the flow level is greater than 150. This is due to the fact that extreme levels (greater than 150) occur only during few big holidays, so this scenario is hard to be caught by models. The standard deviations (green line) is mainly increasing which represents one key problem, i.e., heteroscedasticity. The purple line represents the number of instances per level, and it becomes very small for extreme scenarios in both directions of x-axis. The number of span is used to include neighbour levels when one level's corresponding instances are too few to calculate reasonable statistics. In summary, it can be seen that the relation mapping from levels to residual characteristics are nonlinear. This explains why the proposed data-driven algorithms perform better.

DOW successfully modelled patterns and FLH successfully suppressed heteroscedasticity for normal data compared to others as the residuals are shown in Figure 8. Other algorithms, when being compared to DOW-FLH, cannot distinguish data with vs. without abnormalities, such as shown in Figure 9. This could be an advantage for GARCH-based methods when tracing rapid change in (nonseasonal) time series with heteroscedasticity, but it becomes an disadvantage and hides possible abnormal data instances here. The problem with TBATS and SARIMA is that they could not successfully model the patterns and produces residuals with much noise which leads to low signal-noise ratio as shown in Figure 10.

Previous work has shown that ARIMA and GARCH cannot be adapted to seasonality with many periods such as here

288 periods per season. Instead, they will adapt to local trend or rapid change add plots; therefore, they are not suitable to detect anomalies lasting beyond their detection abilities. This characteristic could be an advantage when quick predicting traffic for short-term time is needed.

6. Conclusion and Future Work

The experiment results show that the proposed DOW algorithm is good at matching multiseasonality time series patterns, and FLH can solve heteroscedasticity problem. DOW-FLH-modelled residuals can be used for labelling anomalies; then, the chosen data can be sent to either edges or centres for further process.

As discussed above, the proposed DOW-FLH in this work is good at modelling and labelling multiseasonal IoT time series for the edge-centre structure. However, other compared algorithms, including SARIMA- and TBATS-based ones, are more mature and may be good at local trend prediction. Also, edge computing can engage crowdsourcing and related active learning [35] to make full use of advantages provided by edge-centre structure.

This point can be further tested in later research.

Labelling can be treated as a classification question, and many new algorithms can work on this task. Especially, recent development regarding classification using belief theory is showing promising results [36], and it is good for multisource scenarios in edge-centre computing. Thus, this might be a good enhancement for our current work, and we look forward to investigate more about it in the future work.

In summary, the proposed DOW-FLH method performs well during experiments using multiseasonal IoT time series and should be considered to use when labelling is needed in edge-centre computing structure.

Data Availability

Access to data is restricted in general; please contact the authors for access when necessary.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We would like to thank the reviewers for their detailed suggestions that have indeed made great contribution to this work. This work is supported by the Shandong Natural Science Foundation under ZR2020MF067 and Shandong Key Research and Development Program under 2019JZZY021005.

References

[1] M. Sun, C. Smith, D. Howard et al., "FES-UPP: a flexible functional electrical stimulation system to support upper limb functional activity practice," *Frontiers in Neuroscience*, vol. 12, 2018.

[2] B. Mandler, J. Barja, M. E. M. Campista et al., *Internet of Things-IoT Infrastructures, Volume 169*, Springer, 2016.

[3] Q. Liu, K. M. Kamoto, X. Liu et al., "A sensory similarities approach to load disaggregation of charging stations in Internet of electric vehicles," *IEEE Sensors Journal*, vol. 9, 2020.

[4] Y. Xu, C. K. Ahn, Y. S. Shmaliy, X. Chen, and Y. Li, "Adaptive robust INS/UWB-integrated human tracking using UFIR filter bank," *Measurement*, vol. 123, pp. 1–7, 2018.

[5] Y. Xu, Y. S. Shmaliy, C. K. Ahn, T. Shen, and Y. Zhuang, "Tightly-coupled integration of INS and UWB using fixed-lag extended UFIR smoothing for quadrotor localization," *IEEE Internet of Things Journal*, vol. 8, no. 3, p. 1, 2020.

[6] M. Sun, X. Zhu, P. Zhang et al., "Xxx an EEG signal-based music treatment system for autistic children," *Security and Communication Networks*, vol. 2020, Article ID 8868311, 2021.

[7] B. Sun, W. Cheng, P. Goswami, and G. Bai, "Flow-aware WPT k-nearest neighbours regression for short-term traffic prediction," in *22nd IEEE symposium on computers and communication (ISCC)*, pp. 48–53, Heraklion, Greece, July 2017.

[8] B. Sun, W. Cheng, L. Ma, and G. Prashant, "Anomaly-aware traffic prediction based on automated conditional information fusion," in *International conference on information FUSION (FUSION)*, pp. 2283–2289, Cambridge, United Kingdom, July 2018.

[9] K. Jia, Z. Wang, S. Fan, S. Zhai, and G. He, "Data-centric approach: a novel systematic approach for cyber physical system heterogeneity in smart grid," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 14, no. 5, pp. 748–759, 2019.

[10] P. N. Borza, M. Machedon-Pisu, and F. Hamza-Lup, "Design of wireless sensors for IoT with energy storage and communication channel heterogeneity," *Sensors*, vol. 19, no. 15, article 3364, 2019.

[11] C. Chen, Y. Zhang, and W. Zheng, "Distributed computation offloading method based on deep reinforcement learning in ICV," *Applied Soft Computing*, vol. 103, article 107108, 2021.

[12] M. Diyan, B. N. Silva, J. Han, Z. B. Cao, and K. Han, "Intelligent Internet of Things gateway supporting heterogeneous energy data management and processing," *Transactions on Emerging Telecommunications Technologies*, vol. 3, article 3919, 2020.

[13] K. Sood, K. K. Karmakar, S. Yu, V. Varadharajan, S. R. Pokhrel, and Y. Xiang, "Alleviating heterogeneity in SDN-IoT networks to maintain QoS and enhance security," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5964–5975, 2020.

[14] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: a self-organized federated learning framework for IoT," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3088–3098, 2021.

[15] S. K. Goudos, P. Sarigiannidis, P. I. Dallas, and S. Kyriazakos, "Communication protocols for the IoT-based smart grid," in *IoT for Smart Grids: Design Challenges and Paradigms, Power Systems*, K. Siozios, D. Anagnostos, D. Soudris, and E. Kosmatopoulos, Eds., pp. 55–83, Springer International Publishing, Cham, 2019.

[16] C. Chen, J. Li, V. Balasubramaniam, Y. Wu, Y. Zhang, and S. Wan, "Contention resolution in Wi-Fi 6-enabled Internet of Things based on deep learning," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5309–5320, 2021.

[17] L. Chhaya, P. Sharma, A. Kumar, and G. Bhagwatikar, "Application of data mining in smart grid technology," in

Encyclopedia of Information Science and Technology, pp. 815–827, IGI Global, 2021.

[18] B. Sun and L. Ma, "An overview of outliers and detection methods in general for time series from IoT devices," in *The 10th international conference on computer engineering and networks*, vol. 135, pp. 1180–1186, Xi-An, China, October 2020.

[19] Z. Zheng, H.-Y. Jeong, T. Huang, and J. Shu, "KDE based outlier detection on distributed data streams in multimedia network," *Multimedia Tools and Applications*, vol. 76, no. 17, pp. 18027–18045, 2017.

[20] S.-E. Benkabou, K. Benabdeslem, and B. Canitia, "Unsupervised outlier detection for time series by entropy and dynamic time warping," *Knowledge and Information Systems*, vol. 54, no. 2, pp. 463–486, 2018.

[21] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, p. 3, 2012.

[22] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[23] B. Sun, W. Cheng, P. Goswami, and G. Bai, "Short-term traffic forecasting using self-adjusting k-nearest neighbours," *IET Intelligent Transport Systems*, vol. 12, no. 1, pp. 41–48, 2018.

[24] M. Sun, J. Amor, C. J. James et al., "Methods to characterize the real-world use of rollators using inertial sensors–a feasibility study," *IEEE Access*, vol. 7, pp. 71387–71397, 2019.

[25] L. Ma, B. Sun, and C. Han, "Learning decision forest from evidential data: the random training set sampling approach," in *4th International Conference on Systems and Informatics (ICSAI)*, Hangzhou, China, November 2017.

[26] H. Shi, K. Worden, and E. J. Cross, "A cointegration approach for heteroscedastic data based on a time series decomposition: an application to structural health monitoring," *Mechanical Systems and Signal Processing*, vol. 120, pp. 16–31, 2019.

[27] T. Andrysiak, L. Saganowski, M. Maszewski, and A. Marchewka, "Detection of network attacks using hybrid ARIMA-GARCH model," in *Advances in Dependability Engineering of Complex Systems*, W. Zamojski, J. Mazurkiewicz, J. Sugier, T. Walkowiak, and J. Kacprzyk, Eds., vol. 582, pp. 1–12, Springer International Publishing Ag, Cham, 2018.

[28] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," 2017, https://arxiv.org/abs/1704.07706.

[29] S. Kelly and K. Ahmad, "Propagating disaster warnings on social and digital media," in *Intelligent Data Engineering and Automated Learning–IDEAL 2015*, pp. 475–484, Springer, Cham, 2015.

[30] L. Bodrog, M. Kajo, S. Kocsis, and B. Schultz, "A robust algorithm for anomaly detection in mobile networks," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6, Valencia, Spain, September 2016.

[31] K. Jensen, T. V. Do, H. T. Nguyen, and A. Arnes, "Better protection of SS7 networks with machine learning," in *2016 6th International Conference on IT Convergence and Security (ICITCS)*, pp. 1–7, Prague, Czech Republic, September 2016.

[32] B. Mendes and D. E. Tyler, "Constrained M-estimation for regression," in *Robust Statistics, Data Analysis, and Computer Intensive Methods: In Honor of Peter Huber's 60th Birthday, Lecture Notes in Statistics*, H. Rieder, Ed., pp. 299–320, Springer New York, New York, NY, 1996.

[33] B. Sun, W. Cheng, G. Bai, and P. Goswami, "Correcting and complementing freeway traffic accident data using mahalanobis distance based outlier detection," *Tehnicki vjesnik - Technical Gazette*, vol. 24, no. 5, pp. 1597–1607, 2017.

[34] B. Sun, L. Ma, W. Cheng, W. Wen, P. Goswami, and G. Bai, "An improved k-nearest neighbours method for traffic time series imputation," in *Chinese automation congress (CAC)*, Jinan, China, October 2017.

[35] L. Ma, S. Destercke, and Y. Wang, "Online active learning of decision trees with evidential data," *Pattern Recognition*, vol. 52, Supplement C, pp. 33–45, 2016.

[36] X. Ke, L. Ma, and Y. Wang, "A dissimilarity measure based on singular value and its application in incremental discounting," in *Proceedings of the 16th International Conference on Information Fusion*, pp. 1391–1397, Istanbul, Turkey, July 2013.