Systems & Control Transactions

Research Article - Peer Reviewed Conference Proceeding
ESCAPE 35 - European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale,
Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Diagnosing Faults in Wastewater Systems: A Data-Driven Approach to Handle Imbalanced Big Data

M. Zadkarami[a], K.V. Gernaey[b*], A.A. Safavi[a], and P. Ramin[b]

[a] School of Electrical and Computer Engineering, Shiraz University, Iran
[b] Process and Systems Engineering Center (PROSYS), Department of Chemical and Biochemical Engineering, Technical University of Denmark (DTU), Denmark
* Corresponding Author: kvg@kt.dtu.dk

ABSTRACT

Process monitoring is essential in industrial settings to ensure system functionality, necessitating the identification and understanding of fault causes. While a substantial body of research focuses on fault detection, fault diagnosis has received significantly less attention. Typically, faults originate either from abnormal instrument behavior, indicating the need for calibration or replacement, or from process faults, signaling a malfunction within the system. A primary objective of this study is to apply the proposed fault diagnosis methodology to a benchmark that closely mirrors real-world conditions. Specifically, we introduce a fault diagnosis framework for a wastewater treatment plant (WWTP) that effectively addresses the challenges posed by imbalanced big data commonly encountered in large-scale systems. In our study, four distinct fault scenarios were investigated: fault-free conditions, process faults only, sensor faults only, and simultaneous sensor and process faults. To enhance our research, we explored various techniques and approaches to optimally address the fault diagnosis problem. Implementing an undersampling technique based on clustering and determining appropriate hyperparameters proved challenging, alongside deciding the most effective approach for undersampling. Additionally, advanced time-series models, including Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM), were employed, with particular emphasis on identifying the correct regularization factor and number of epochs being critical. When optimal model parameters are selected, each fault scenario achieves an accuracy of at least 90%, and the maximum simulation time does not exceed 15 minutes. Furthermore, we developed a Python-based GUI for this study to enable users to fully explore the capabilities of the proposed fault diagnosis algorithm.

Keywords: Artificial Intelligence, Big Data, Wastewater, Industry 4.0, Process Monitoring

INTRODUCTION

Today, process monitoring has become an increasingly critical task in industries, driven by cutting-edge technologies capable of handling large volumes of data. This advancement has introduced new challenges, particularly related to big data aspects. In scientific literature, "big data" typically refers to datasets that exhibit at least two of the following characteristics, collectively known as the 5V: Volume, indicating a substantial quantity of data samples; Velocity, reflecting a high sampling rate of incoming data; Veracity, which denotes significant uncertainty and noise within the data; Variety, meaning the dataset is composed of smaller datasets of different types and formats; and Variability, referring to the degree of complexity in the data, such as noticeable fluctuations and peaks in data patterns. It is important to note that whether a dataset is classified as big data also depends on the specific application [1].

Deep learning architectures have emerged as an effective approach for handling big data in process monitoring. Their primary advantage lies in their ability to perform feature analysis systematically, enabling the efficient extraction of key information from large datasets without the need for designing and implementing separate feature extraction mechanisms. However, utilizing deep learning architectures presents certain challenges. Tuning the hyperparameters of these deep structures is

often time-consuming and complex. Additionally, due to their extensive layers and intricate structures, these models are highly susceptible to the vanishing gradient problem, which can lead to inaccurate and erroneous results [2].

In real-world applications, normal operating conditions occur more frequently than faulty ones. From an engineering perspective, process monitoring in such settings constitutes an imbalanced classification problem, where the majority class represents normal conditions. Most statistical and neural network models are typically designed for balanced or nearly balanced classification scenarios, making them ineffective for highly imbalanced tasks. This is because, in imbalanced datasets, the majority class tends to dominate the decision-making process of these models.

One common approach to addressing the imbalanced classification problem is undersampling the majority class, which involves removing samples from the majority class [3]. A straightforward method is to randomly eliminate samples from the majority class. However, this simplistic approach is often naive, as it can result in the loss of important information pertaining to the majority class.

A recent study on fault detection in wastewater treatment plants (WWTPs) utilized an undersampling technique based on Gaussian Mixture Models (GMMs) [3]. The core idea of their research was to cluster the majority class and remove samples associated with the smallest clusters. This method achieved promising fault detection results by effectively balancing the removal of samples while preserving key information and the inherent behavior of the majority class.

In this study, four distinct scenarios for fault diagnosis were generated for a typical WWTP using Benchmark Simulation Model No.2 (BSM2). The research addresses a complex multi-class imbalanced big data problem, which poses significant challenges in accurately identifying and diagnosing faults within the system. To tackle this issue, we expanded our investigation to explore various undersampling techniques, advanced time-series models, and hyperparameter tuning strategies. These approaches aim to enhance the performance of fault diagnosis by effectively managing the imbalance in the dataset and improving the robustness of the models. Furthermore, we developed a Python-based Graphical User

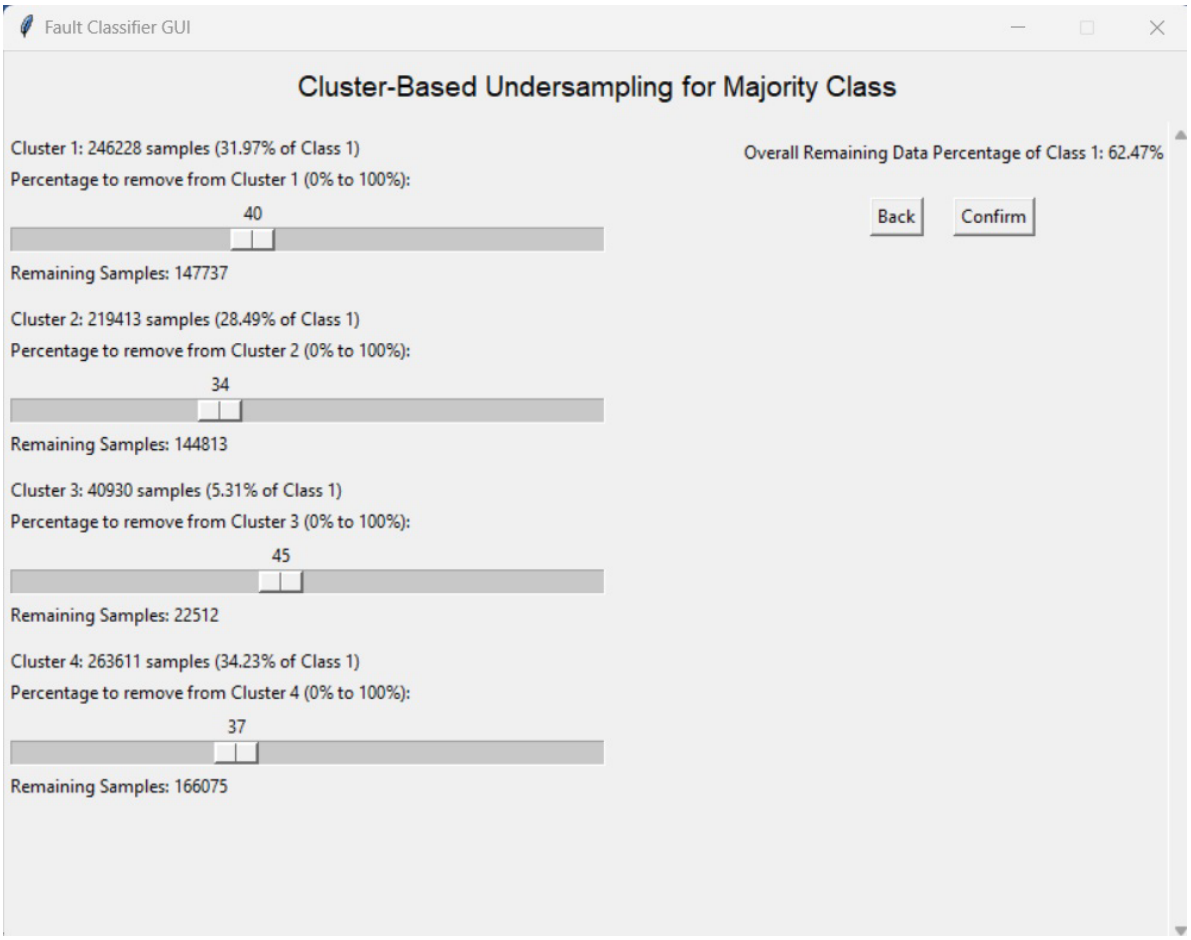

Figure 1: The second GUI window

Interface (GUI) to facilitate user interaction with the proposed models. The GUI allows users to deliberately select different models and their corresponding parameters, providing flexibility and ease of use. Additionally, it offers the option to save each modification made during the analysis process, ensuring that users can efficiently track and manage their experiments. The complete code for this study is publicly available on GitHub.

The structure of this paper is organized as follows. Section 2 presents the case study along with the generated data. Section 3 details the proposed fault diagnosis approach and its GUI. In Section 4, the process monitoring results are analyzed. Finally, Section 5 provides the conclusions of the study.

CASE STUDY

The design of fault scenarios in this study is the same as the methodology outlined in [3, 4]. The faults are carried out on BSM2, closely simulating a real-world WWTP, containing process units listed as: a primary clarifier, activated sludge reactors, a secondary clarifier, a thickener, and a dewatering unit [3, 4]. Data acquisition is performed using 31 easily measurable variables, such as flow rate, Total Suspended Solids (TSS), Volatile Fatty Acids (VFA), among others, collected from various process units. The simulation duration of this plant-wide is 609 days with a sampling time of 1 minute, resulting in 876960 sample points. Therefore, for each fault scenario, a big dataset including 876960 sample points over 31 different measurements is investigated. To ensure the case study aligns more closely with real-world scenarios, the simulation period for normal operating conditions is set to be significantly longer than that for faulty conditions, resulting in an imbalanced big data distribution. The four fault scenarios and their contribution percentage is listed in Table 1.

As illustrated in Table 1, the majority class in this study, which is the normal operation condition, consists of 770182 samples and is designated as Class 1. Independent sensor faults and process faults are labeled as Class 2 and Class 3, respectively, each accounting for approximately 6% of the entire dataset. In this study, the process faults refer to the sludge bulking phenomenon and sensor faults represent an abnormality in the aerated reactors related the airflow. Lastly, only 330 samples (i.e. less than 0.1% of the dataset) indicate the simultaneous occurrence of both sensor and process faults, which are categorized as Class 4.

The aim of this study is to develop a fault diagnosis approach that by analyzing the 31-measurements anticipates which fault scenario is occurring. The primary challenge of this research work is to handle imbalance big datasets while striking a meaningful balance between the accuracy and the computational burden. Although the case study utilized in this research is identical to those presented in [3] and [4], the focus is on fault diagnosis, a multi-class classification problem, rather than fault detection, which is addressed as a binary classification problem. Moreover, aspects of big data must be considered when selecting appropriate parameters to ensure the effectiveness of the proposed approach. Finally, a GUI was developed for the proposed fault diagnosis algorithm, enabling users to easily select the desired model and adjust its parameters. Additionally, the GUI provides the capability to save the results of each stage of the algorithm independently.

Table 1: Fault Scenarios

Stream	Description	Percentage
Class 1	Normal	87.82
Class 2	Only Sensor Fault	6
Class 3	Only Process Fault	6.14
Class 4	Both Sensor and Process Fault	0.04

FAULT DIAGNOSIS APPROACH

The proposed fault diagnosis workflow initiates by loading and preprocessing the dataset, generated by the update version of BSM2 [5], followed by identifying the majority and minority classes. The user is then prompted to input the desired number of clusters, after which the majority class data is segmented into the specified clusters. For each cluster, relevant information is displayed to the user, who determines the percentage of samples to remove. Based on this input, the appropriate number of samples is calculated and removed from each cluster. Subsequently, the undersampled majority class is combined with the minority classes, and the updated class distribution is presented. The user has the option to save the newly balanced dataset before proceeding to the modeling phase, thereby completing the preprocessing sequence.

The subsequent subsections present each window of the proposed fault diagnosis GUI designed to manage multi-class imbalanced big data distributions.

The First Window: Data information

In the initial window, detailed information is presented, including the sample size of each class and the various imbalance ratios relative to the majority class (Class 1) for the generated fault scenarios of BSM2. In other words, the first window of the GUI displays the number of samples corresponding to each fault scenario: Class 1 contains 770,182 samples, Class 2 contains 52,583 samples, Class 3 includes 53,865 samples, and Class 4 has only 330 samples. Furthermore, the imbalance ratios (IR) are calculated to quantify the degree of class imbalance, yielding IR_1 = Class 1 / Class 2 = 14.65,

IR_2 = Class 1 / Class 3 = 14.30, and IR_3 = Class 1 / Class 4 = 2,333.88. More importantly, users have the option to select the number of clusters into which the majority class should be segmented. In this study, the diagnosis performance were studies for 2 to 6 clusters.

The Second Window: Undersampling

In the second GUI window, the majority class is segmented based on the number of clusters selected in the preceding window. This window presents detailed information for each cluster, including the cluster size and its contribution percentage. The undersampling technique employed in this study involves initially clustering the majority class and subsequently removing samples randomly from each cluster. The removal percentage for each cluster can be determined through various methods: applying a uniform elimination percentage across all clusters, removing all samples from the smallest clusters sequentially until a predetermined overall removal percentage is achieved, or allowing the user to specify the removal percentage for each cluster individually. Figure 1 illustrates the second window of the developed fault diagnosis GUI.

The Third Window: Confirmation

In this window, users can view the updated sample size of the majority class along with the revised imbalance ratios. To illustrate, the undersampling percentage applied to each cluster within the majority class (i.e., Class 1) is based on the values presented in Figure 1. As a result, the modified Class 1 is reduced to 481,137 sample points, leading to updated imbalance ratios of IR_1 = 9.15, IR_2 = 8.93, and IR_3 = 1,457.99. Although the dataset does not achieve a fully balanced distribution among the classes, removing the less informative samples from the majority class effectively reduces the degree of imbalance. Additionally, users have the option to save comprehensive information, including input measurements and output labels, for both the modified dataset and the removed samples. If the user is unsatisfied with the current elimination strategy, they can navigate back to the previous window to make adjustments. Otherwise, they can confirm and proceed to the next window.

The Fourth Window: Time-Series Modelling

In this window, users can select their deep learning time-series model from Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM). Upon selecting a model, users can adjust various model parameters, including learning rate, batch size, number of epochs, regularization factor, number of LSTM units, and dropout rate. Selecting appropriate values for epochs and the regularization factor is particularly challenging, as it requires balancing accuracy with computational efficiency. It is important to note that selecting lower values for the L2 regularization parameter can significantly increase the computational effort, improving the accuracy. On the other hand, increasing the number of epochs will extend the simulation time without necessarily guaranteeing an improvement in the classification accuracy for each class. Figure 2 illustrates the fourth window of the developed fault diagnosis GUI.

The Fifth Window: Training Phase

In the fifth GUI window, the model begins training based on the previously provided information, while simultaneously displaying the ongoing simulation time. Monitoring the training time allows the user to terminate the process if the simulation duration exceeds expected limits.

The Sixth Window: Results

In the final window, once the training phase is completed, the results are displayed, including simulation time, overall accuracy, confusion matrices, and class-specific accuracies for both training and testing datasets. Additionally, users have the option to save these results. It is important to note that overall accuracy may not be a reliable indicator for imbalanced data distributions; instead, ensuring high accuracy for each individual class is crucial. Figure 3 illustrates the last window of the developed fault diagnosis GUI.

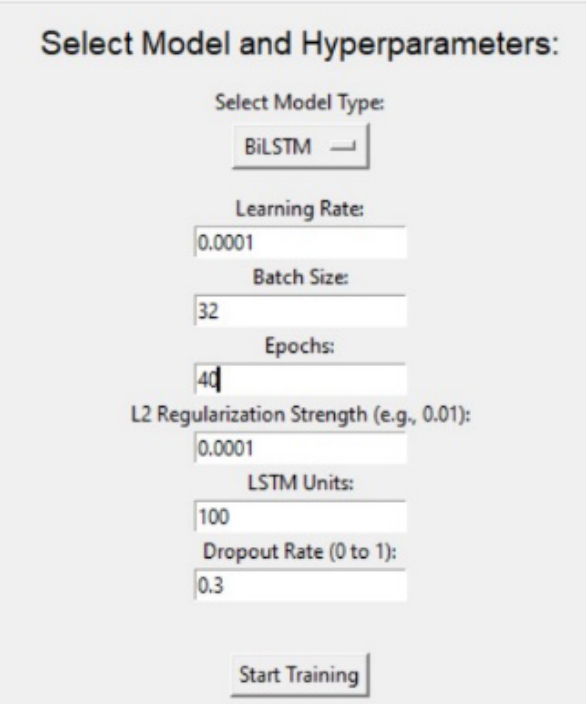

Figure 2. The fourth GUI window

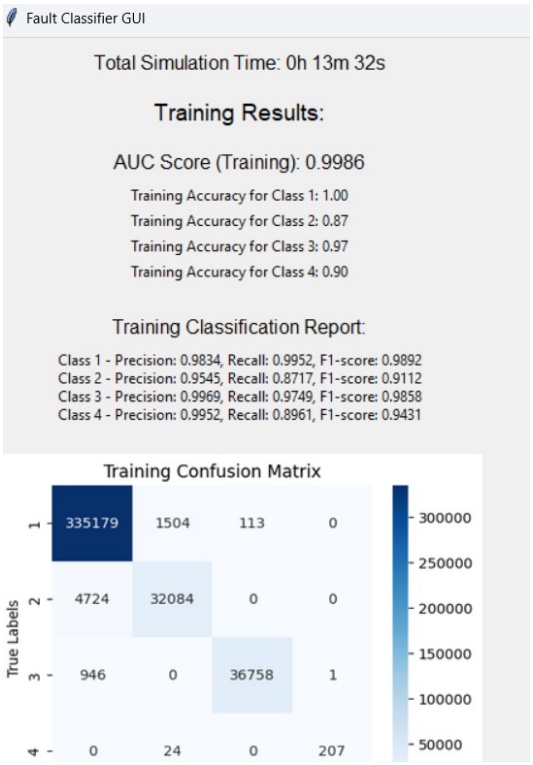

Figure 3. The sixth GUI window

RESULTS AND DISCUSSION

As previously discussed, the regularization factor is crucial for obtaining optimal results. A regularization factor closer (L2) to zero typically enhances accuracy but also increases simulation time. Initially, we examine a fault detection algorithm that does not incorporate any undersampling techniques (Table 2), thereby directly inputting the imbalanced data into either an LSTM or BiLSTM model. In this setup, the parameters are configured with 40 epochs and a dropout rate of 30%.

Table 2: No undersampling approach (C denotes here as Class Accuracy)

L2	Model	C1	C2	C3	C4	Time(min)
0.001	BiLSTM	99	43	76	0	17
0.001	LSTM	99	44	77	0	14
0.0005	BILSTM	99	60	84	0	20
0.0005	LSTM	99	58	84	0	16
0.0001	BiLSTM	99	80	95	94	20
0.0001	LSTM	99	80	94	91	18

Table 3: Comparing different epoch values (C denotes here as Class Accuracy)

Epoch	C1	C2	C3	C4	Time (min.)
100	100	91	98	96	34
40	99	89	97	91	14

According to Table 2, a regularization factor of 0.0001 is considered the highest acceptable value (L2 ≤ 0.0001), as exceeding this threshold leads to the accuracy of Class 4 dropping to zero or near zero.

With an established regularization factor of 0.0001, we proceed to identify the optimal number of epochs that achieves promising performance while minimizing computational time. The parameters fixed for this analysis are as follows: an undersampling rate of 40% (preserving 60% of the majority class) with 3 clusters for the majority class, a BiLSTM model configured with 100 units and a 30% dropout rate. Table 3 presents the classification accuracy for each class at epoch counts of 40 and 100, along with the corresponding simulation times.

Upon analyzing Table 3, it is evident that while the class accuracy of the model trained for 100 epochs is slightly higher than that of the model trained for 40 epochs, the computational cost is significantly greater for the 100-epoch model. Specifically, the fault diagnosis model with 100 epochs requires an additional 20 minutes of simulation time compared to the 40-epoch model. Therefore, it is reasonable to conclude that the 40-epoch configuration is more suitable in terms of both accuracy and computational efficiency.

With the L2 regularization value and the number of epochs established, we can now focus on investigating the number of clusters, determining the undersampling technique, and selecting a time-series model (LSTM or BiLSTM). Table 4 illustrates an undersampling rate of 40% applied uniformly across all clusters, whereas Table 5 demonstrates an undersampling rate of 40% achieved by eliminating the smallest clusters. Please note that in the simulation of Table 4 and 5, the L2 regularization value is 0.0001, drop out is 30%, batch size is 32, number of epoch is 40, the learning rate is 0.0001, and the employed LSTM units is 100.

Table 4: Uniformly removing samples from the clusters of the majority class (C denotes here as Class Accuracy)

Cluster	Model	C1	C2	C3	C4	Time(min)
2	BiLSTM	99	88	97	91	14
2	LSTM	99	87	96	86	11
3	BILSTM	99	89	97	91	14
3	LSTM	99	87	97	90	11
4	BiLSTM	99	87	97	89	13
4	LSTM	99	87	97	86	11
5	BiLSTM	99	87	97	89	13
5	LSTM	99	87	97	87	10.5
6	BiLSTM	99	88	97	94	11
6	LSTM	99	87	97	83	10.5

According to Tables 4 and 5, time-series models based on LSTM generally exhibit shorter computational times, averaging approximately three minutes. In contrast, fault diagnosis models utilizing BiLSTM achieve

higher accuracy values. Regarding the undersampling technique, uniformly removing samples from each cluster results in better accuracy compared to the approach of eliminating the smallest clusters. Additionally, the number of clusters has minimal impact on the performance of the proposed fault diagnosis algorithm.

Table 5: Removing samples from small clusters of the majority class (C denotes here as Class Accuracy)

Cluster	Model	C1	C2	C3	C4	Time(min)
2	BiLSTM	99	88	97	96	14.5
2	LSTM	99	88	98	83	11.5
3	BILSTM	99	87	96	91	13
3	LSTM	99	87	97	88	11
4	BiLSTM	99	87	96	96	14
4	LSTM	99	87	97	92	10.5
5	BiLSTM	99	89	97	85	13
5	LSTM	99	87	97	84	10.5
6	BiLSTM	99	87	98	99	13
6	LSTM	99	87	97	96	11

Finally, some implementation challenges related to parameter selection for the proposed fault diagnosis approach is highlighted. The total undersampling ratio is critical, as excessive removal of majority class samples may cause information loss. A 40% undersampling rate is recommended, retaining 60% of the majority class (i.e. class 1) samples [3]. For clustering, although the number of clusters has minimal effect, setting three or four clusters and uniformly removing 40% of samples from each cluster prevents losing essential data. Furthermore, both LSTM and BiLSTM models perform similarly, with suggested hyperparameters: learning rate of 0.0001, dropout rate of 0.3, batch size of 32, and 100 LSTM units. These parameters have limited effect on accuracy and simulation time. However, the L2 regularization parameter has a highly significant impact on model performance. Specifically, values greater than 0.0001 prevent the model from correctly identifying fault scenarios associated with Class 4, while values smaller than 0.0001 introduce a substantial computational burden. Additionally, the recommended number of training epochs ranges between 40 and 60. Increasing the number of epochs beyond this range does not necessarily improve classification accuracy but will considerably increase computational cost.

CONCLUSION

This study presents a fault diagnosis framework specifically for a wastewater treatment plant, addressing the challenges of imbalanced big data typical in large-scale systems. It investigates four fault scenarios: fault-free conditions, process faults, sensor faults, and simultaneous sensor and process faults. The research utilizes clustering-based undersampling and optimizes hyperparameters, employs advanced time-series models such as LSTM and BiLSTM with careful regularization and epoch selection, and develops a Python-based GUI to allow users to effectively explore the fault diagnosis algorithm. The objective is to apply this methodology to benchmarks that closely reflect real-world conditions, enhancing the reliability and efficiency of industrial process monitoring. For future research, the development of an advanced optimization algorithm could be pursued to systematically and more effectively tune hyperparameters.

DIGITAL SUPPLEMENTARY MATERIAL

The Python code for the proposed fault diagnosis GUI developed in this study is publicly available at https://github.com/Zadkarami/ESCAPE-2025.

REFERENCES

1. Gandomi A, Haider M. Beyond the hype: Big data concepts, methods, and analytics. International journal of information management. 2015 Apr 1;35(2):137-44.
2. Zadkarami M, Safavi AA, Taheri M, Salimi FF. Data driven leakage diagnosis for oil pipelines: An integrated approach of factor analysis and deep neural network classifier. Transactions of the Institute of Measurement and Control. 2020 Oct;42(14):2708-18.
3. Zadkarami M, Safavi AA, Gernaey KV, Ramin P. A Process Monitoring Framework for Imbalanced Big Data: A Wastewater Treatment Plant Case Study. IEEE Access. 2024 Sep
4. Zadkarami M, Gernaey KV, Safavi AA, Ramin P. Big Data Analytics for Advanced Fault Detection in Wastewater Treatment Plants. InComputer Aided Chemical Engineering 2024 Jan 1 (Vol. 53, pp. 1831-1836). Elsevier.
5. Ramin P, Flores-Alsina X, Topalian SO, Jeppsson U, Gernaey K. Fault detection in a benchmark simulation model for wastewater treatment plants. InComputer Aided Chemical Engineering 2022 Jan 1 (Vol. 49, pp. 1363-1368). Elsevier.