12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

Reservoir computing-based slow feature analysis: Application in fault classification

Alireza Memarian [*,**] , **Amirreza Memarian** [*]
Seshu Kumar Damarla [*] , **Rahul Raveendran** [**]
, **Biao Huang** [*,1]

[*] *Department of Chemical and Materials Engineering, University of Alberta, Edmonton, Canada, T6G 1H9.*
[**] *Spartan Controls, Edmonton, Alberta, Canada, T6E 5L9.*

Abstract: Differentiating between various types of faults and classifying them based on their importance is essential for process fault detection and diagnosis. This classification helps operators to prioritize their actions based on the severity of the faults. This paper proposes a reservoir computing-based slow feature analysis (RCSFA) to model complex and nonlinear industrial processes and study its application in fault classification while integrated with a graph neural network (GNN) and majority voting ensemble causality detection. To make the algorithm robust to unseen faults, real-time operator feedback is included by utilizing operator eye tracking. The practical applicability of the proposed method and its application in fault classification is studied through an industrial application.

Keywords: Fault detection, Operator feedback, Slow feature analysis, Graph neural network, Majority voting, Reservoir computing neural network.

1. INTRODUCTION

The safety and reliability of industrial processes are of paramount importance, especially within the oil industry (Baker et al., 2020; El Kouche et al., 2012). To uphold these objectives, an effective approach is through fault classification, which plays a crucial role in maintaining the integrity of process operations (Wang et al., 2020). By implementing process monitoring and fault detection algorithms (Yin et al., 2022; Divya et al., 2023; Memarian et al., 2023), we can identify faults within the system. However, it is not enough to simply detect faults; it is equally critical to differentiate and classify them accurately. This classification process empowers operators to make informed decisions and take appropriate actions based on the specific type of fault, enabling effective planning and execution of maintenance activities.

In the context of chemical processes, faults refer to undesirable deviations from normal operating conditions, often characterized by gradual growth or incipient behavior. Traditional fault detection and classification (FDC) algorithms have been developed to address these faults, providing operators with timely notifications based on comprehensive information. Nevertheless, conventional classification methods sometimes suffer from the inclusion of unnecessary information, resulting in increased computational costs and false diagnosis. Therefore, the development of an efficient FDC algorithm that optimizes the utilization of relevant information and incorporates a robust model capable of accurately representing underlying faults is crucial. Causal analysis emerges as a diagnostic approach, leveraging pertinent information to enhance fault detection and classification accuracy.

Causal analysis is commonly employed in fault detection and classification to improve the accuracy of diagnostic procedures. Various approaches exist for causal analysis, including transfer entropy, Granger causality, autoencoder, and Bayesian inference. However, each method has its own strengths and limitations in detecting causal relationships. To mitigate the potential for erroneous causality analysis, majority voting ensemble causality detection algorithm integrates information from multiple algorithms simultaneously. These algorithms make decisions based on the most frequently occurring results from various causality detection methods, thereby reducing false causality detections (Pastorino et al., 2021). In this paper, a majority voting ensemble algorithm utilizing random forest, support vector machine (SVM), and autoencoder models for causal detection is proposed. Section 2.2 provides a concise overview of majority voting ensemble algorithms.

As mentioned earlier, the fault model should be compatible with the gradual development commonly observed in process faults. While slow feature analysis (SFA) offers a linear feature-based model capable of extracting features based on the slowness principle, it does not provide an accurate representation for processes with multiple modes or nonlinearity. Therefore, in this study, a reservoir computing-based slow feature analysis is proposed to extract slowly varying features from nonlinear processes. Section 2.3 briefly discusses SFA in this context.

To the best of the author's knowledge, this study represents the first integration of the statistical feature model

[1] This work was supported by Natural Sciences and Engineering Research Council of Canada and Mitacs Accelerate Program. Corresponding author: Biao Huang (biao.huang@ualberta.ca)

Copyright © 2024 the authors. Accepted by IFAC for publication under a Creative Commons License CC-BY-NC-ND.

within a reservoir computing neural network, facilitating seamless integration with other machine learning algorithms. This study provides a novel fault detection and classification algorithm based on reservoir computing, leveraging the power of graph neural networks for fault prediction and classification. The remaining sections of this paper are organized as follows. Section 2 provides a concise overview of key concepts, including the reservoir computing neural network, majority voting ensemble algorithm, slow feature analysis, and graph neural network. This background information enhances the reader's understanding of the proposed fault detection and classification algorithm. In Section 3, the proposed algorithm is elaborated and its structure is presented in detail. To validate the effectiveness of the approach, an industrial case study is presented in Section 4, showcasing the practical application of the proposed fault detection and classification algorithm. Finally, in Section 5, conclusion summarizing the contributions and implications of the proposed algorithm is provided.

2. PRELIMINARIES

In this section, the required preliminaries about reservoir computing neural networks (RCNN), majority voting ensemble algorithm, slow feature analysis (SFA), and graph neural networks (GNN) are provided for a better understanding of the proposed algorithm.

2.1 RESERVOIR COMPUTING NEURAL NETWORK

The rapid advancements in artificial intelligence (AI) and its wide-ranging applications in the oil sands industry have spurred numerous efforts to address prevailing industrial challenges (Gupta and Shah, 2022). Reservoir computing neural networks have gained significant popularity as a class of powerful machine learning tools. However, their application in the process industry remains limited.

Reservoir computing is a type of recurrent neural network (RNN) that employs fixed, random internal dynamics and hidden layers to process inputs and generate outputs that follow a specific distribution, like Glorot uniform, normal, and truncated normal distributions. Notably, the parameters of the internal layers are not updated during training, resulting in a faster and simpler training process. Unlike traditional RNNs, reservoir computing neural networks do not encounter the issue of vanishing gradients due to their fixed internal layers. Consequently, the modeling process achieves improved convergence. The network generates outputs by training the readout layer, typically the last hidden layer, to predict the desired outputs based on the internal states. Reservoir computing neural networks have been successfully applied in various domains, including time-series prediction (Kutvonen et al., 2020), control systems design (Zhu et al., 2019), and in combination with other machine learning techniques (Mlika et al., 2023).

In RCNN, the outputs (Y_t) are generated using the input (X_t) through the reservoir layers and their neurons (R_t) as depicted in Fig. 1. The generative equations for RCNN are provided in Eqs. (1) and (2).

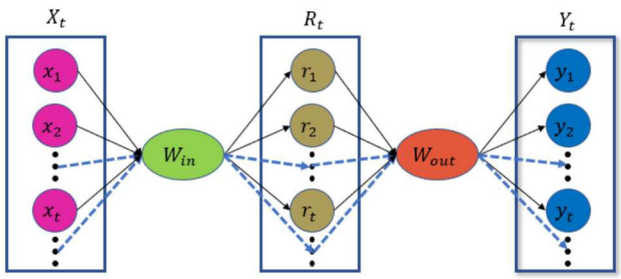


Fig. 1. Representation of reservoir computing neural networks

$$r_t = (1-\lambda)r_{t-1} + \lambda\ tanh(A.r_{t-1}) + W_{in}x_t + b \quad (1)$$

$$\hat{y}_t = W_{out} \times r_t + c \quad (2)$$

where r_t is the reservoir state correlating the current state to its previous time $(t-1)$. x_t and y_t are the input and output, respectively. b is a bias that is usually a constant vector. A is the sparse weighted adjacency matrix for the reservoir layer. W_{in} and W_{out} are linear input and output weight matrices, respectively. $\lambda(0 < \lambda \leq 1)$ is the leaking factor that defines the reservoir's dynamics.

The previous studies (Jaeger, 2007) have demonstrated that choosing λ close to 1 maximizes the memory capacity of the model. The rationale for setting the leaking factor close to 1 is based on the desire to preserve information from the recent past without letting the reservoir states change too rapidly, which could cause the network to forget important temporal information. A leaking factor close to 1 means that the reservoir's state is updated in a manner that closely integrates its current state with incoming inputs, allowing the network to maintain a rich dynamical memory of past inputs. This is essential for tasks that require the network to leverage long-term dependencies and temporal patterns in the data. On the other hand, the nonlinearity of the model is defined using the spectral radial (ρ) of A representing the forgetting factor, which is defined in Eq. (3).

$$\rho = \max|\lambda_i| \quad (3)$$

where λ_i's are the eigenvalues of matrix A. Selecting appropriate values for λ and ρ involves striking a balance between memory capacity and nonlinearity in the model.

2.2 Majority Voting Ensemble Algorithm

The majority voting ensemble is a technique that leverages the collective wisdom of multiple models to make predictions on a given dataset. In this approach, several models are trained independently, each providing their own predictions. These individual predictions are then combined by means of a majority vote to determine the final prediction.

The strength of majority voting ensembles lies in their ability to mitigate inaccuracies present in individual models. By aggregating the predictions from multiple models, the ensemble can produce a more robust and accurate prediction compared to relying on a single model alone.

The application of majority voting ensembles spans various domains, including health care (Qin et al., 2017),

economics (An et al., 2019), and process control (Meng and Kwok, 2013).

2.3 SLOW FEATURE ANALYSIS

Slow feature analysis (SFA) was first introduced by Wiskott and Sejnowski (Wiskott and Sejnowski, 2002). Slow feature analysis maps input data (X) into lower dimensional feature space to generate slow features. These features are trying to capture the slowly varying information from the data. The representation of the SFA model is provided in Eqs. (4)-(7).

$$\min \Delta(s_j) \tag{4}$$
$$s.t. < s_{j_t} >_t = 0 \tag{5}$$
$$< s_{j_t}^2 >_t = 1 \tag{6}$$
$$\forall i \neq j, < s_{i_t} s_{j_t} >_t = 0 \tag{7}$$

The SFA model tries to minimize the variation of slow features by constraining them to follow zero mean and unit variance. In addition, slow features are mutually independent. The outcome of the slow feature analysis is a set of features where s_{1_t} is the slowest and s_{2_t} is the second slowest, etc.

2.4 GRAPH NEURAL NETWORK

The Graph Neural Network (GNN) is a deep learning algorithm designed for structured data analysis. Recently, several types of GNNs have been proposed to address the challenges associated with structured data, such as the Graph Convolutional Network (GCN), Graph Attention Network (GAT), and Graph Isomorphism Network (GIN) (Park et al., 2022; Yuan et al., 2020). In GNN, the inputs consist of an adjacency matrix and features. The features represent nodes, and the adjacency matrix helps create connections between nodes, forming edges (Zhou et al., 2020). One important application of graph neural networks is in fault detection and diagnosis (Li et al., 2020).

To update the forward-pass propagation in GNN, Eq. (8) is employed for each node.

$$s_i^{(l+1)} = \sigma \left(s_i^{(l)} \times W_1^{(l)} + \sum_{j \in N(i)} s_j^{(l)} \times W_2^{(l)} \right) \tag{8}$$

where l and s_i represent the hidden layer and hidden state of the node, respectively. W_1 and W_2 are the parameters of the neural network. $N(.)$ denotes the neighbors of the node. This equation implies that each node's state depends solely on the information received from its neighbors, thereby capturing the causal relationship among variables. This approach effectively avoids incorporating redundant information from high-dimensional input variables.

3. PROBLEM FORMULATION

In this section, a fault detection and classification algorithm that is based on the reservoir computing slow feature analysis and the majority voting ensemble is presented. The algorithm consists of three steps, feature extraction, causality analysis, and fault classification. In the rest of this section, these steps are elaborated.

3.1 FEATURE EXTRACTION

In this step, the slow features are extracted from the input data using a reservoir computing-based SFA model. Fig. 2 illustrates the proposed model, which employs reservoir computing for slow feature extraction. The input data is fed into the Reservoir Computing Neural Network (RCNN) to extract the slow features in the encoding section. Subsequently, these slow features are used to reconstruct the input data in the decoding section. In order to maintain model accuracy, it is crucial to minimize the error between the input data ($X \in \mathcal{R}^{m \times T}$) and the reconstructed input ($\hat{X}$). Additionally, to ensure that the extracted features are indeed slow features, the optimization problem described in section 2.3 must be solved. Thus, a multi-objective optimization problem needs to be addressed to obtain the slow features. The formulation of the optimization problem is presented in Eq. 9.

$$\begin{aligned} \text{Objective function 1} &: \min_{\theta} (x - \hat{x}) \\ \text{Objective function 2} &: \min_{\theta} \Delta(s_j) \\ \text{Constraints} &: < s_{j_t} >_t = 0 \\ &< s_{j_t}^2 >_t = 1 \\ &\forall i \neq j, < s_{i_t} s_{j_t} >_t = 0 \end{aligned} \tag{9}$$

where θ is the set of parameters for linear regression 1 and linear regression 2. $S \in \mathcal{R}^{q \times T}$ is the slow feature where $q \leq m$. T is the total number of measurements. The optimization problem defined in Eq. (9) can be further simplified to a constraint-free objective function as represented in Eq. (10).

$$\begin{aligned} \min_{\theta} \frac{1}{T} \Bigg(& \sum_{t=1}^{T} ||X_t - \hat{X}_t||_2^2 + \frac{T}{T-1} \sum_{t=2}^{T} ||S_t - S_{t-1}||_2^2 \\ & + \sum_{i=1}^{q} (\sum_{t=1}^{T} s_{i,t})^2 + \sum_{i=1}^{q} \left(\sum_{t=1}^{T} (s_{i,t} - \bar{s_i})^2 - 1 \right)^2 + \\ & + \sum_{i=1}^{q} \sum_{j=1, j \neq i}^{q} \left(\sum_{t=1}^{T} (s_{i,t} - \bar{s_i})(s_{j,t} - \bar{s_j}) \right)^2 \Bigg) \end{aligned} \tag{10}$$

In the given equation, X_t represents the measurements at time t, while S_t represents the slow features at time t. $s_{i,t}$ denotes the value of feature i at time t, and $\bar{s_i}$ represents the mean value of i^{th} slow feature.The first term in the equation represents the reconstruction error, while the last three terms are the constraints of Eq. (9). The second term defines the slowness principle. This optimization problem can be effectively solved using numerical solvers.

3.2 CAUSALITY ANALYSIS

After extracting the slow features in section 3.1, the proposed majority voting ensemble algorithm is utilized to identify causes and effects between the features. To identify the causes of a specific slow feature (s_i), the surrogate datasets are generated as $[s_i, s_{j \neq i}]$ in addition to the main dataset $[s_i]$ where $s_{j \neq i}$ is a lagged variable to study the causalities. These datasets are partitioned into two sections using partitioning size (L). The first section

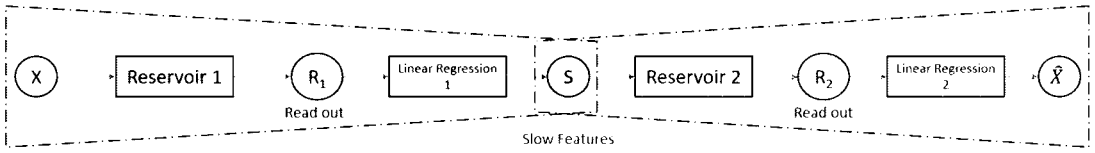


Fig. 2. Reservoir computing-based slow feature analysis

is used for creating the models while the second section is used for calculating the MSE of the predictions using the models. These datasets are given to three different models, random forest (RF), support vector machine (SVM), and autoencoder (AE). In each model, the MSE is calculated for each dataset. If any of the surrogate datasets' MSE is smaller than the MSE for the actual dataset, the presence of the other slow feature helps improve the prediction accuracy. Thus, the augmented slow feature (s_j) is considered as a "cause" for the specific slow feature (s_i). The final decision on identifying causes is based on the majority voting among those three models, i.e. the slow features are the cause for the specific slow feature (s_i) if at least two models detect their causality. The schematic for the proposed majority voting ensemble causality detection algorithm is provided in Fig. 3.

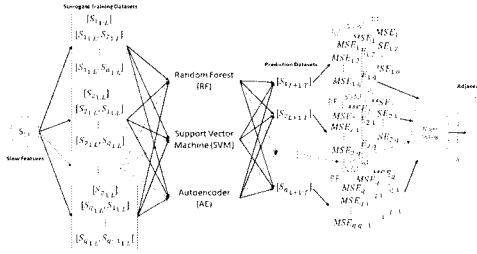


Fig. 3. Majority voting ensemble causality detection algorithm

3.3 FAULT CLASSIFICATION

After extracting the slow features (nodes) in step 3.1 and creating their corresponding adjacency matrix (edges) in step 3.2, the graph representation can be constructed. This graph is then fed into a GNN as input, which establishes connections based on the adjacency matrix to learn the model. The GNN processes the graph and produces output that classifies faults based on prior fault information. Fig. 4 illustrates the schematic of the GNN and how the extracted slow features and the identified adjacency matrix are employed for fault classification.

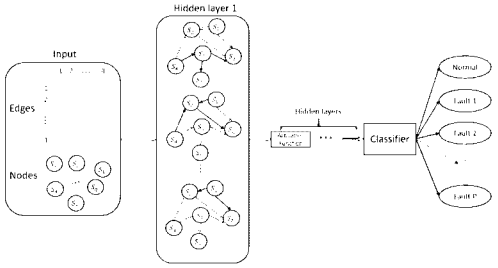


Fig. 4. Fault classification using graph neural network

The graph neural network classifies the data into $P + 1$ different classes including normal condition and P different faulty conditions. The trained Graph Neural Network (GNN) currently lacks the capability to detect and classify faults that have not been encountered before. To enhance the algorithm's resilience in handling such unforeseen faults, eye-tracking information from operators is integrated into the methodology. Operators continuously monitor various process variables, leveraging their expertise to potentially identify faults. When operators spot a potential fault, it is flagged for further investigation once their gaze becomes stationary or exhibits a specific eye gesture. Consequently, a new fault category is introduced. The GNN model is subsequently retrained, incorporating this newly identified fault alongside the existing ones in the dataset (P+2 classes). The schematic of operator feedback is demonstrated in Fig. 5.

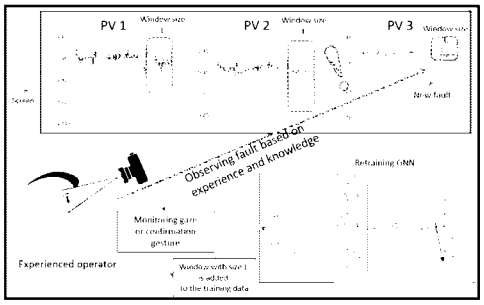


Fig. 5. Schematic of operator feedback using eye tracking

4. CASE STUDY

In this section, a real-life experiment was conducted to test the applicability of an algorithm for health monitoring of pumps using ultrasonic flow meter measurements. Ultrasonic flow meters are devices that use high-frequency sound waves to measure the flow rate of fluids in pipes. They emit sound waves into the fluid and measure the time it takes for the waves to travel through the fluid and return to the sensor. These sensors are commonly used in the oil sand industry (Drenthen et al., 2009).

The experiment focused on monitoring the pumps used to increase the pressure of oil passing through pipelines in the oil sand industry. The mechanical characteristics of these pumps make them prone to failure. The schematic in Fig. 6 represents the system studied, but specific details such as tag names, devices, and process information have been removed and the data has been normalized for proprietary reasons. The system consists of four pumps: $U1$, $U2$, $U3$, and $U4$. Pumps $U1$ and $U2$ work with pumps $U3$ and $U4$, respectively, using bypass lines. Measurements from ultrasonic flow meters were collected for each pump

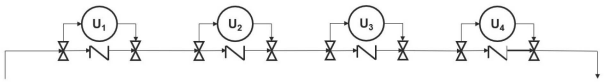


Fig. 6. Series-operation pumps

Table 1. Computational time comparison between the proposed algorithm and RNN-based SFA

Algorithm	Training Time (s)
Proposed Algorithm	1045
Recurrent Neural Network-based SFA	1384

over a period of three months, resulting in 30 different measurements per pump. The data was divided, with 65% used for training the algorithm and the remaining portion for validation.

In addition to the flow meter measurements, an alarm log was also available, indicating the condition of the process (e.g., normal condition, low-low alarm, low alarm, high alarm, high-high alarm). The algorithm used the reservoir computing-based Slow Feature Analysis (SFA) model to extract slow features from the training data. The "BoTorch" library, a Bayesian optimization framework in "PyTorch," was employed for this purpose. Only the first six slow features, which explain 95% of the actual variability, were retained and shown in Fig. 7.

The extracted slow features were then used to generate an adjacency matrix using the majority voting ensemble algorithm. This matrix represents the causal relationships between the pumps and is depicted as a graph in Fig. 8. Subsequently, a graph neural network was trained using the extracted slow features and detected causalities.

The algorithm was validated using test data, and the results were evaluated using a confusion matrix shown in Fig. 9. The confusion matrix demonstrates the algorithm's performance in detecting faults and classifying them into the four available fault types. Table 1 provides a comparison based on the training time of the proposed algorithm and the RNN-based SFA which demonstrates a reduction in training time.

To test the robustness of the algorithm to unseen faults, a class of new faults is manually introduced in test data with the label "Warning" operating around the normal condition and 43 alarms are generated relating to this label. The experimental operator's eye-tracking feedback is utilized to update the fault classification model. Once the experimental operator detects a deviation in process variables and captures the mismatch between their process knowledge with the model output, a "30-second" gaze pattern is utilized to retrain the GNN model using the new label and deviated portion of the process variable as depicted in Fig 5. As can be seen from Fig. 10(a), the updated algorithm performs well in classifying the new class compared to Fig. 10(b) where all introduced faults are misclassified.

Overall, the algorithm showed promising results in fault detection and classification, indicating its effectiveness in monitoring pump health using ultrasonic flow meter measurements.

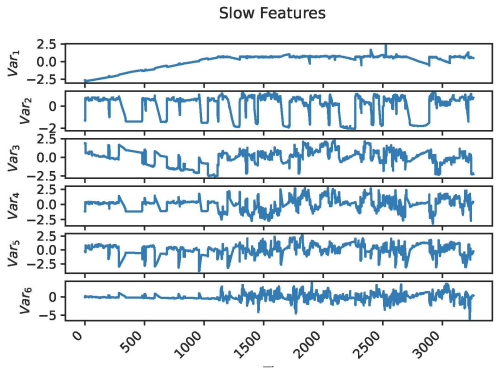


Fig. 7. Extracted slow features using RCSFA

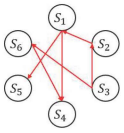


Fig. 8. Estimated graph using RCSFA and ensemble causality detection

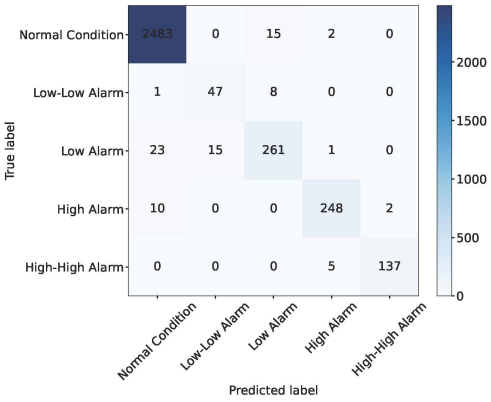
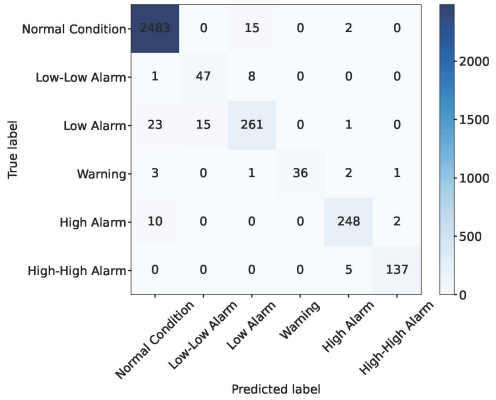


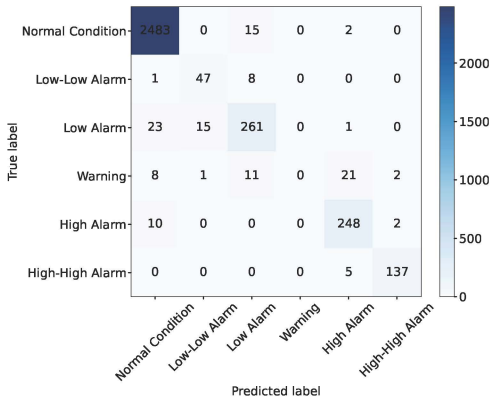
Fig. 9. Fault detection and classification algorithm performance

5. CONCLUSIONS

In this paper, a reservoir computing-based slow feature analysis model is proposed for extracting the slowly varying features from nonlinear data. The proposed model is integrated with the majority voting ensemble and graph neural network to detect the faults and classify them. The reservoir computing-based slow feature analysis and majority voting ensemble are able to extract the slow features and obtain the nonlinear causal relationship between input variables. In addition, operator feedback is included by means of the operator eye tracking for real-time algorithm updates to handle the problem of unseen faults. The proposed algorithm is tested on an industrial data set to demonstrate real-life applicability. The results presented through the industrial case study on pump health monitoring using ultrasonic flow meters verify the effectiveness of the nonlinear SFA model and the application of the proposed algorithm in detecting and classifying faults.



(a) Classification algorithm performance in the presence of unseen fault with operator feedback



(b) Classification algorithm performance in the presence of unseen fault without operator feedback

Fig. 10. Abnormalities observed by using the proposed algorithm in

REFERENCES

An, J., Mikhaylov, A., and Sokolinskaya, N. (2019). Machine learning in economic planning: Ensembles of algorithms. In *Journal of Physics: Conference Series*, volume 1353, 012126. IOP Publishing.

Baker, K.E., Macciotta Pulisci, R., Hendry, M.T., and Lefsrud, L.M. (2020). Combining safety approaches to bring hazards into focus: An oil sands tailings case study. *The Canadian Journal of Chemical Engineering*, 98(11), 2330–2341.

Divya, D., Marath, B., and Santosh Kumar, M. (2023). Review of fault detection techniques for predictive maintenance. *Journal of Quality in Maintenance Engineering*, 29(2), 420–441.

Drenthen, J.G., Kurth, M., and Vermeulen, M. (2009). Verification of ultrasonic gas flow meters. In *CEESI ultrasonic seminar*.

El Kouche, A., Hassanein, H., and Obaia, K. (2012). Monitoring the reliability of industrial equipment using wireless sensor networks. In *2012 8th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 88–93. IEEE.

Gupta, D. and Shah, M. (2022). A comprehensive study on artificial intelligence in oil and gas sector. *Environmental Science and Pollution Research*, 29(34), 50984–50997.

Jaeger, H. (2007). Echo state network. *scholarpedia*, 2(9), 2330.

Kutvonen, A., Fujii, K., and Sagawa, T. (2020). Optimizing a quantum reservoir computer for time series prediction. *Scientific reports*, 10(1), 14687.

Li, C., Mo, L., and Yan, R. (2020). Rolling bearing fault diagnosis based on horizontal visibility graph and graph neural networks. In *2020 international conference on sensing, measurement & data analytics in the era of artificial intelligence (ICSMD)*, 275–279. IEEE.

Memarian, A., Raveendran, R., and Huang, B. (2023). Robust multi-mode probabilistic slow feature analysis with application to fault detection. *Journal of Process Control*, 132, 103130.

Meng, Y. and Kwok, L.F. (2013). Enhancing false alarm reduction using voted ensemble selection in intrusion detection. *International Journal of Computational Intelligence Systems*, 6(4), 626–638.

Mlika, Z., Cherkaoui, S., Laprade, J.F., and Corbeil-Letourneau, S. (2023). User trajectory prediction in mobile wireless networks using quantum reservoir computing. *IET Quantum Communication*.

Park, J., Sung, G., Lee, S., Kang, S., and Park, C. (2022). Acgcn: graph convolutional networks for activity cliff prediction between matched molecular pairs. *Journal of Chemical Information and Modeling*, 62(10), 2341–2351.

Pastorino, M., Montaldo, A., Fronda, L., Hedhli, I., Moser, G., Serpico, S.B., and Zerubia, J. (2021). Multisensor and multiresolution remote sensing image classification through a causal hierarchical markov framework and decision tree ensembles. *Remote Sensing*, 13(5), 849.

Qin, C.J., Guan, Q., and Wang, X.P. (2017). Application of ensemble algorithm integrating multiple criteria feature selection in coronary heart disease detection. *Biomedical Engineering: Applications, Basis and Communications*, 29(06), 1750043.

Wang, Y., Yang, H., Yuan, X., Shardt, Y.A., Yang, C., and Gui, W. (2020). Deep learning for fault-relevant feature extraction and fault classification with stacked supervised auto-encoder. *Journal of Process Control*, 92, 79–89.

Wiskott, L. and Sejnowski, T.J. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4), 715–770.

Yin, X., Qin, Y., Chen, H., Du, W., Liu, J., and Huang, B. (2022). Process decomposition and distributed fault detection of large-scale industrial processes. In *2022 IEEE International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, 176–181. IEEE.

Yuan, H., Tang, J., Hu, X., and Ji, S. (2020). Xgnn: Towards model-level explanations of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 430–438.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI open*, 1, 57–81.

Zhu, Q., Ma, H., and Lin, W. (2019). Detecting unstable periodic orbits based only on time series: When adaptive delayed feedback control meets reservoir computing. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(9).