Unsupervised Outlier Detection in Continuous Nonlinear Systems: Hybrid Approaches with Autoencoders and One-Class SVMs

Roland Bolboacă and Bela Genge

George Emil Palade University of Medicine, Pharmacy, Science and Technology of Târgu Mureş, 540142, Târgu Mureş, Romania.
{roland.bolboaca,bela.genge}@umfst.ro

Abstract. Outlier detection in continuous nonlinear systems is essential as the presence of outliers might be indicators of faults, diseases, cyberattacks, or system malfunctions. However, the complex nature of such systems significantly increases the difficulty of developing outlier detection techniques. Due to the high complexity of such systems, accurate model based approaches are often difficult to design. While supervised outlier detection techniques yield great performance, the scarcity of labeled data motivates the necessity of unsupervised approaches. This paper introduces a data-driven approach for unsupervised outlier detection, which utilizes a hybrid combination of Autoencoders and One-Class Support Vector Machines. Experimental assessment was performed on the Tennessee Eastman Process dataset, and the performance of the proposed solution was measured using nine independent metrics, including detection delay and, true and false positive rates. Furthermore, a comparison with other recent techniques was performed, with notable results in terms of false alert rates and detection delay.

Keywords: Continuous nonlinear systems, outlier detection, fault detection, autoencoders, MLP, RNN, LSTM, one-class support vector machines (OCSVM), Tennessee Eastman process

1 Introduction

Continuous nonlinear systems are prevalent and extensively utilized across a diverse range of domains, including Physics [1], Engineering [2], Biology [3], Medicine [4], Astronomy [5] and various other ones as well [6–8]. Monitoring and identifying outliers in such systems is essential since the presence of outliers can serve as crucial indicators of various conditions, such as diseases [9], system faults and malfunctions [10], cyberattacks, erroneous readings, and even potential discoveries in new physics [11], depending on the specific context.

As illustrated by Aggarwal et al. [12] three primary methodologies are utilized for detecting outliers: supervised, unsupervised, and semi-supervised. The choice between these approaches depends on the availability of labeled data and the selected detection models. In the supervised approach, classifiers are trained using both normal and anomalous data. In contrast, unsupervised techniques do not rely on labeled anomalous data, presenting a more formidable challenge in anomaly detection. Regarding semi-supervised approaches, they represent a middle ground between supervised and unsupervised methods. In semi-supervised outlier detection, the algorithm leverages a combination of labeled normal data and unlabeled data, which may include both normal and anomalous instances [13]. Without loss of generality, the terms *outlier* and *anomalous* point, or instance, will be used interchangeably. Similarly, the terms *normal*, *clean*, *outlier-free* and *inlier* will denote an observation that falls within the expected or typical range of values.

Detecting outliers in continuous nonlinear systems poses significant challenges due to the high complexity and the presence of nonlinear patterns and relationships. The increased complexity makes the creation of accurate system models challenging [14], particularly when employing a data-driven approach for unsupervised outlier detection [15]. In such scenarios, linear models may not be sufficient to capture the intricate relationships present in the data. Nonlinear patterns may lead to deviations that are harder to identify using traditional linear techniques [16]. As a result, specialized algorithms and techniques are required to handle the complexities of nonlinear outlier detection effectively [15].

This paper introduces a data-driven approach for unsupervised outlier detection for continuous nonlinear systems. The proposed solution incorporates an Autoencoder (AE) and a One-Class Support Vector Machine (OCSVM). The AE is utilized for modeling the normal behavior of the system from data obtained during normal operating conditions. Subsequently, the OCSVM is used for outlier detection in the reconstruction residuals provided by the AE. Furthermore, this study presents and evaluates three different AE architectures based on Multilayer Perceptrons (MLP), Recurrent Neural Network (RNN) [17], and lastly, on Long-Short Term Memory (LSTM) models [18]. The proposed outlier detection techniques are easily applicable with low overhead brought on by feature selection methodologies. Additionally, these techniques function in multi-variate settings as well.

The experimental assessment was performed on the Tennessee Eastman Process (TEP) dataset [19]. The TEP is popular for being a reference complex nonlinear system utilized in diverse challenges, including modeling, fault detection, fault analysis, and the design of control techniques. Outlier detection on such systems is valuable for identifying system faults and malfunctions.

To achieve the optimal selection of hyperparameters for both the AE and the OCSVMM, we employed a combination of manual and automatic hyperparameter optimization techniques on the validation loss objective function. Additionally, to measure the performance of the proposed outlier detection methodology, nine distinct metrics are employed, including detection delay, false positive rate, true positive rate and balanced accuracy score. Furthermore, the proposed detection technique is compared to recent supervised [20] and unsupervised [21] techniques, with promising results in terms of false alarm rates and detection delays on both clean and anomalous datasets.

The remainder of the paper is structured as follows. Section 2 provides a discussion on relevant related studies. This is followed by Section 3, which presents the proposed outlier detection methodology. In Section 4, the experimental assessment is described. The experimental results are showcased in the fifth section, and finally, the paper concludes in Section 6.

2 Related Work

This section explores recent studies that are relevant to the current paper. It is structured into three subsections, each addressing distinct aspects. The initial subsection investigates recent and relevant studies that utilize AE and SVM for outlier detection in various fields. The second subsection concentrates on studies proposing detection strategies for nonlinear systems. Lastly, the third subsection analyzes outlier detection techniques applied to the TEP dataset.

2.1 Autoencoder and One-Class SVM Outlier Detection

In a recent paper, Wei et al. [22] proposed an LSTM-Autoencoder based solution for outlier detection for indoor air quality monitoring systems. Their solution utilizes the AE to reconstruct a univariate time-series data containing CO_2 readings measured in several schools. The Encoder takes as input a sequence of t time-steps, where each time-step is treated as an indivi dual input, while the decoder reconstructs the same sequence. This solution utilizes a threshold based detection methodology, where the threshold is computed as the maximum value of the reconstructed residuals on the training set. However, such an approach is sensitive to outliers and selecting the maximum value might yield large threshold values, making the solution inefficient to non-obvious outliers encountered during inference. The authors addressed this issue by removing certain observations from the training set, which is not considered a standard practice. Additionally, this approach is not designed to function in multivariate settings.

Riberolles et al. [23] proposed a similar LSTM-Autoencoder based outlier detection solution for multivariate time-series data originating from Industrial Control Systems. Their proposed solution also utilizes a threshold based detection approach, where the threshold is set to a fixed value so that the majority of the training residuals are below it. This thresholding method, as also identified by the authors, is context specific. Selecting a higher value for the threshold might yield poor detection results, while a lower value might increase the false alert rate. Additionally, such an approach is not scalable, as each feature from the multi-variate space is monitored individually, increasing the number of features will increase the complexity. On the other hand, with a smaller set of inputs, this method offers the benefit of pinpointing the cause of the anomaly, such as the specific anomalous feature.

In a similar direction to our proposed solution, Said et al. [24] developed an AE and OCSVM based solution, which was tested on a recent SDN Intrusion Detection System dataset. Here, the authors experimented with two approaches.

First, by setting a fixed threshold on the reconstruction residuals, as in the previous paper. Second, training an additional OCSVM model using the latent (e.g., compressed) data given by the encoder. Their results showed that the AE with OCSVM yielded superior results compared to the thresholding method, and even compared to utilizing only the OCSVM without the AE. An overall of 18% and 22% increase in Precision was observed when utilizing the AE with OCSVM approach compared to thresholding and OCSVM, respectively. Similar differences in performance were also observed while using other metrics, such as Recall, F1-score and Accuracy. While this approach further proves that such methods are efficient for outlier detection tasks, the authors only experimented with LSTM based AE. Additionally, the OCSVM model was trained and tested with the latent representation of the data. In this compressed form, valuable outlier information might be lost. Moreover, to enhance this solution with extra capabilities, like identifying the feature responsible for the outlier, the inclusion of a decoder would be necessary.

2.2 Nonlinear Systems Detection Approaches

Outlier detection in nonlinear systems has been addressed by several researchers' in various domains [25–28].

Recently, Wanli et al. [27] proposed a tree-based ensemble methodology combined with multivariate control charts for fault detection in centrifugal chillers, namely on the ASHRAE-1043 dataset. Their extensive study analyses the detection performance of various ensemble prediction methods, such as Random Forest (RF), extreme gradient boosting (XGBoost) and light gradient boosting machine (LightGBM) coupled with three multivariate control charts, namely Hotelling's T^2, multivariate cumulative sum (MCUSUM) and multivariate exponentially weighted moving average (MEWMA). Additionally, the authors compare the prediction performance of the three models with Support Vector Regression (SVR) models. This approach utilizes the tree-based ensembles to create a model of a centrifugal chiller during normal operating conditions, and subsequently apply control charts as a means of monitoring the prediction residuals of the system. The results of their study indicate that, on this specific dataset, the tree-based ensembles can outperform SVR models. Moreover, in terms of detection, their results highlight that MEWMA yields the best results, with a 1.2% false alert rate and an average detection rate of 88.71%. However, while the tree-based ensembles exhibit a great performance, the authors don't address the shortcoming of the multivariate control charts. All the multivariate control charts utilize the inverse covariance matrix in the computation process. As identified by others, the covariance matrix is not always invertible [29,30]. Mathematically, the existence of the inverse covariance matrix is limited by multiple conditions, such as: the matrix must be positive definite [31], and non-singular [32]. Assuring the existence of the inverse covariance matrix thus requires overhead brought on by carefully selecting the monitored variables, even so, such methods might not always be applicable.

Tan et al. in [26] proposed an LSTM based anomaly detection system for nonlinear dynamic systems. Their solution involves modeling the nonlinear system using an LSTM, and detecting the changes in the mean, standard deviation and slope values of the predicted output values given by the model, using a sliding window technique. In terms of accuracy scores, the solution yields scores of up to 99.5%. However, the proposed outlier detection solution is applied to a univariate self generated dataset while no architectural information about the model are supplied, nor is the dataset available.

2.3 TEP Outlier Detection Solutions

The TEP serves as a prominent reference environment for testing and evaluating various approaches in domains such as outlier detection [33], fault diagnosis [34], and performance optimization [35]. In this subsection, recent outlier and fault detection approaches are presented. The outlier detection methods applied on this dataset are specifically addressing fault detection. Nonetheless, fault detection is a specific application of outlier detection.

In the work of Hu et al. [36] the authors introduce a novel supervised method called the "kernel limit learning machine". Initially they employ eXtreme Gradient Boosting to compress the features, and subsequently, they utilize a failure classifier in conjunction with adjusted network hyperparameters. As reported by the authors, their proposed approach demonstrates an impressive average detection rate of 91%.

Avinash and Ajaya [20] proposed Gaussian Process Regression (GPR) based detection approach for the TEP. Additionally, this paper also investigates the effect of GPR hyperparameters, such as the covariance and mean functions, influence the detection performance. The tested covariance functions include squared exponential and the matern function, while the tested mean function include the zero, constant, sum and polynomial functions. While the authors obtained notable results in terms of fault detection rates and detection delays, the false alarm rate were high, averaging 20.20%. However, the authors attributed the high false alert rate to the low detection threshold value.

Moving on to SVM based detection approaches, we find the work of Onel et al. [21]. In this paper, the authors proposed a two-class SVM method for outlier detection, together with a feature selection methodology based on nonlinear Kernel-dependent SVM feature rank criteria, This feature selection methodology is derived from the sensitivity analysis of the dual C-SVM objective functions. The authors utilized five performance metrics, including the detection delay. Their results indicate that the SVM classifier is capable of detecting numerous faults, with great accuracy. Nonetheless, in certain scenarios their proposed approach yields a high false alert rate, upwards of 50%. Their detection delay is also high, on the hard to detect faults 3 and 15, with measured delays reaching 500 and 800 samples.

Deep learning methods have also been utilized towards fault detection on the TEP. Lomov et al. [37] investigated the fault detection capabilities of a wide range of deep Recurrent and Convolutional Neural Network classifiers. The studied classifiers include Gated Recurrent Units (GRUs), LSTMs and Transformers, with and without attention mechanisms, for all the tested models. Additionally, a Generative Adversarial Network (GAN) is introduced to enrich and extend the existing training dataset. The experimental assessment is performed using 60 and 800 sequence lengths, and only on the faulty 800 data points for each faulty scenario. Their results show outstanding performance with a sequence length of 800, with detection rates up to 100% on almost all the faulty datasets, and by almost all the tested models. Using a sequence length of 60 the detection rates are lower, ranging from 47% up to 100%. While the results are promising, realistically, considering that the measurement frequency is 3 minutes, the detection time would be once every 40 hours for sequences of 800 and once every 3 hours for sequence lengths of 60. Moreover, as also stated by the authors, in measuring the detection delay, only the simulation runs that were correctly predicted were considered. Any simulation runs that were not detected with the correct class were not included in the calculation of the detection delay. This often resulting in conflicting results, with large detection delays but with 100% detection rate.

Similarly, Heo and Lee [38] proposed a deep MLP classifier for fault detection in the TEP. In their study, the authors did not include Faults 3,9 and 15, which are considered difficult to detect due to the absence of observable change in the mean and variance of the signals. Their proposed solution yielded notable results, with detection rates ranging from 93% to 100% on 17 faults. While their proposed classifiers, as well as the ones from the previous studies, showed remarkable results, we have to consider the fact that all the models were trained using both clean and faulty data. Specifically, these classifiers are compelled to detect only learned faults. Moreover, in real-life scenarios it is often difficult to generate faulty data, covering all the possible faults or scenarios, especially considering, for this specific scenario, that this would imply shutting down or altering the functionality of a real chemical plant.

3 Proposed Outlier Detection Solution

The following is a broad overview of the components included in the proposed outlier detection methodology. The methodology comprises several components that work together to detect outliers. These components are as follows:

Autoencoder: An AE is used to reconstruct input signals. It takes the input data and generates reconstructed signals at each time step. The AE learns to compress and then reconstruct the input, capturing the underlying patterns and features of the data.

Reconstruction Residuals: From the reconstructed signals, the methodology computes the reconstruction residuals. These residuals represent the differences between the original input signals and their corresponding reconstructions. They capture the errors in the reconstruction process.

One-Class SVM: The OCSVM is a machine learning algorithm used for outlier detection. It treats the data points as high-dimensional points, where each dimension corresponds to a reconstruction residual for an input signal. The OCSVM is trained on normal data points and learns to distinguish between normal and outlier instances based on their representation in the high-dimensional space.

Outlier Decision: When a new data point (e.g., reconstruction residual) is provided to the OCSVM, it evaluates the binary decision whether it is classified as a normal point or as an outlier. The decision is based on the learned representation of normal data points and the distance of the new point in the high-dimensional space.

Fig. 1 illustrates an overview of the proposed method, including the three AE variants, and the OCSVM. While this detection methodology is categorized as unsupervised outlier detection, there is a training process, where both the AE and the OCSVM model are trained using only clean, outlier free, data. The AE is trained using the selected training dataset, containing real measured values, while the OCSVM is trained on the reconstruction residuals over the same training dataset.

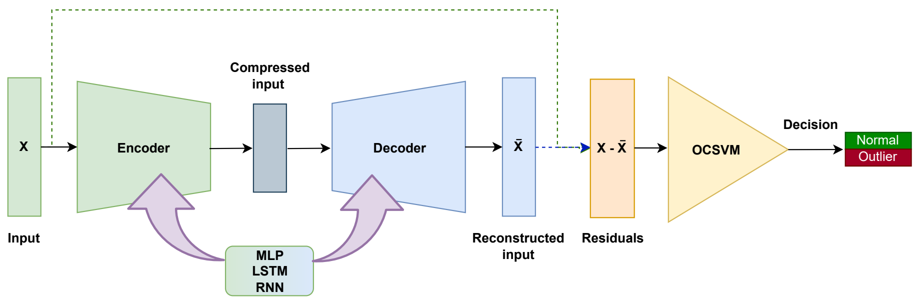

Fig. 1. Illustration of the proposed unsupervised outlier detection method which includes the three AE variants and the OCSVM.

The following subsections provide a broad overview of the major components included in the detection methodology.

3.1 Autoencoders

An AE is described as an artificial neural network that has the ability to reproduce an m-dimensional input vector X^m by generating an m-dimensional output vector $\bar{X}^m$. The network is designed to learn an efficient representation of the input data by compressing it into a lower-dimensional latent space and then reconstructing it back to its original dimensionality [17].

The AE consists of two main parts: an Encoder and a Decoder. The Encoder takes the input vector X^m and maps it to a lower-dimensional latent space representation L^d, with $d < m$. This latent representation captures the essential features and patterns of the input data. The Decoder, on the other hand, takes the latent representation L^d and maps it back to the original input dimensionality, generating the reconstructed output vector $\bar{X}^m$. The goal of the AE is to minimize the reconstruction error between the original input vector X and its reconstructed output $\bar{X}^m$ during the training process.

The two components of the AE, namely the Encoder and the Decoder, can be formally defined as the following two functions, $f : \mathbb{R}^m \rightarrow \mathbb{R}^d$ and $g : \mathbb{R}^d \rightarrow \mathbb{R}^m$. A general form of the objective function of the AE can be defined as follows:

$$argmin_{f,g}||X - g(f(X))||_2^2. \tag{1}$$

The scope here is to approximate the function that minimizes the error between the input X^m and the output $\bar{X}^m$, where $\bar{X}^m = g(f(X^m))$. The general form of the mapping relationship, during encoding and decoding, is defined as $L^d = h(WX^m + B_1)$ and $\bar{X}^m = h(UL^d + B_2)$. Here, $W, U \in \mathbb{R}^{mxd}$ are weight matrices, B_1 and B_2 are the biases, and h represents the activation function.

3.2 Autoencoder Architecture

As previously mentioned, the proposed AE architecture consists of three different variants: MLP, RNN and LSTM, each utilizing different basic units.

The MLP architecture is a feed-forward neural network comprised of multiple layers. It takes the input data and processes it through a series of fully connected layers. Each layer consists of neurons that compute weighted sums of their inputs, apply an activation function, and pass the output to the next layer. This architecture is suitable for capturing nonlinear relationships in the data. Additional information on the MLP architecture can be obtained in [17].

The vanilla RNN architectures also operate on sequential data having a single recurrent connection that allows information to be passed from one time-step to the next. They can maintain an internal state or memory, which enables them to process sequences of variable lengths. However, traditional RNNs may suffer from vanishing/exploding gradient issues when processing long sequences. Additional information about RNN architectures are presented in [17].

The LSTM architecture is a type of RNN that incorporates memory cells and gates to better handle sequential data. It can retain information over longer sequences and alleviate the vanishing/exploding gradient problem. LSTM units contain memory cells that can store and retrieve information, as well as gates that control the flow of information. This architecture is effective for capturing dependencies in time-series or sequential data. Additional information on the LSTM units, architecture, and the gating mechanism can be obtained in [17,18].

3.3 One-Class Support Vector Machines

In the literature, two main approaches proposing OCSVM are identified, namely the work of Scholkopf et al. [39] and the work of Tax et al. [40]. This paper will utilize the implementation of the former. The OCSVM algorithm, as proposed by Scholkopf et al. in [39] consists of an adaptation of the two class Support

Vector Machine (SVM) [41] to a one class problem. Specifically, for the OCSVM the training data is enclosed in a single class and the algorithm can either classify new data as inliers (e.g., in the same class as the training data) or as outliers (e.g., outside the learned class). In short, during training, the OCSVM maps the input data to higher dimensional feature space by the use of kernel functions and approximates a decision boundary around the first (and only) class, separating the data points from the origin. During detection, a given data point is considered an outlier if it falls below the hyper-plane and closer to the origin.

Specifically, let $e_i \in \mathbb{R}^m$ represent the reconstruction residuals, computed as the difference $X^m - \bar{X}^m$, i takes values from 1 to n, where n represents the size of the training set, and m represents the dimensionality of the input space, namely the number of features. Let $\Phi(e_i)$ denote a mapping function that transforms e_i to a high dimensional feature space F generated by the kernel $k(e_i, e_j)$. The OCSVM algorithm identifies a hyperplane in the kernel space, separating the data points from the origin with a maximum margin. To handle the case where such a hyperplane doesn't exist, a set of slack variables ξ_i are introduced, which allows some points to fall within the margin (e.g., outliers). Additionally, a parameter $\nu \in (0, 1]$ is introduced, as an upper bound on the fraction of outliers in the training data.

To separate the data points from the origin, the following quadratic program is solved:

$$\begin{aligned} &\min_{w,\xi,\rho} \frac{1}{2}||w||^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho, \\ &\text{s.t. } \langle w, \Phi(e_i)\rangle \geq \rho - \xi_i, \xi_i \geq 0, \end{aligned} \tag{2}$$

here, w denotes the norm perpendicular to the hyperplane and ρ denotes the margin. The decision function, further denoted as $\gamma(e_n)$, which determines if a new datapoint e_n belongs to the estimated set, is defined as follows:

$$\gamma(e_n) = sign(\langle w, \Phi(e_n)\rangle - \rho). \tag{3}$$

If the function γ yields a negative value for a newly introduced point e_n, it is classified as an outlier; otherwise, it is categorized as normal.

As stated by numerous authors [42, 43], in practice the Gaussian kernel is usually chosen. The Gaussian kernel, with a width parameter σ, is defined as:

$$k(e_i, e_j) = exp\left(-\frac{||e_i - e_j||^2}{2\sigma^2}\right). \tag{4}$$

Additional information about SVMs and OCSVMs can be found in [39–41].

4 Experimental Assessment

The proposed approaches were implemented in Python, utilizing several libraries such as Keras [44] with the TensorFlow back-end [45], as well as Scikit-learn [46].

The implementation was carried out on a Lenovo Legion laptop featuring an AMD Ryzen 5 5600H CPU, 32 GB DDR4 RAM, running the Windows 10 PRO operating system. Three repetitions of each experiment were conducted, and the results showcased in the subsequent section represent the average outcome obtained from these three executions.

In what follows, the dataset used for the experimental assessment is presented, as well as the performance metrics, architecture, and hyperparameter values selected for each model.

4.1 Dataset Description

First introduced in 1992 by Downs and Vogel [19], the TEP serves as a representation of an industrial chemical process. Its primary objective is to facilitate the exploration, design, and evaluation of process control technologies. Comprised of key units including the product condenser, separator, compressor, and product stripper, the TEP produces two liquid products and two byproducts from a combination of four reactants. Within this process, the chemical reactions are irreversible and exothermic, with their rates being influenced by temperature. The initial paper provides details on a model featuring a total of 52 measurements, with 41 measurements pertaining to process variables and the remaining 11 to manipulated variables.

While the TEP is an industrial process, it's also a complex continuous nonlinear system describable by differential equations. As identified by Jockenhövel et al. [35] and by Ricker and Lee [47], this complex nonlinear system is described by 30 differential equations, 149 algebraic equations, 160 algebraic variables, 11 control variables and 26 states.

The TEP dataset has found extensive application in various research studies, covering areas such as plant-wide control strategy design, multivariate control analysis, educational purposes, anomaly detection, and fault diagnosis. Furthermore, the dataset is publicly accessible [48]. This dataset comprises both clean and anomalous subsets. The anomalous datasets include 20 system faults on several components. By introducing faults into the TEP dataset, abnormal operating conditions or equipment malfunctions can be simulated. Malfunctions that may occur in real-world systems. These faults can include deviations in process variables, manipulated variables, or abnormal behavior in the chemical reactions within the TEP.

The training subset consists of 500 simulations, where each simulation encompasses 500 observations, leading to a total of 250,000 observations. In each simulation, variables were sampled at intervals of 3 minutes, and the simulations ran for a duration of 25 hours in the case of the training subset. Conversely, for the testing clean subsets the simulation ran for 48 hours, comprising 500 simulations with 960 observations per simulation, resulting in a total of 480,000 data points. Each subset contains 55 columns, encompassing 52 variables, the simulation number, the sample number, and an additional column for supplementary information.

The anomalous datasets consist of also 500 simulations, where each simulation comprises 960 observations. Each fault is introduced after the 160th observation and is active for the next 800 observations. In what follows, the 20 anomalous datasets, which encompass the 20 faults, will be denoted as F1 to F20. A recent study [49] discusses the nature of the faults in the TEP dataset and the effects they have on the product quality control within the plant. The authors identified 9 faults (i.e., 1, 2, 5–8, 10, 12 and 13) as quality-related faults, and 5 faults (i.e., 3, 4, 9, 11 and 15) as quality-unrelated faults. This paper considers all the available faults.

For the experimental assessment, namely for training the models, 20 clean simulations were selected from the training set, containing 10,000 observations in total. For validation purposes 1,000 data points were selected, representing 2 simulations. For outlier detection purposes, the entire testing datasets were used, namely, 480,000 data points for the clean testing set and for each subsequent anomalous dataset. As the proposed outlier detection techniques are unsupervised approaches, only clean data was used for training the models. For a comprehensive list of variables, faults and additional information, please refer to the original TEP paper [19].

The datasets used in the experimental assessment were normalized using the feature scaling method, where all the features were scaled in the [0,1] range, using the maximum and minimum value from the training dataset [50].

4.2 Architectures

To achieve the optimal selection of hyperparameters for both the AE and the OCSVMM, we employed a combination of manual and automatic hyperparameter optimization techniques on the validation loss objective function

To ovoid overfitting the models to the training data, an early stopping [17] approach was utilized, with the patience parameter set to five epochs. For the LSTM and RNN architectures, a fully connected output layer was introduced, with the same number of units as the input layer. The complete list of the selected hyperparameters used for the experimental assessment are summarized in Table 1.

4.3 Performance Metrics

For a thorough investigation of the detection performance, nine metrics were utilized, namely True Negative Rate (TNR), False Positive Rate (FPR), True Positive Rate (TPR), False Negative Rate (FNR), Accuracy (ACC), Precision (PRC), F1 score (F1), Balanced Accuracy (BACC), and Detection Delay (DD).

– **TNR**. The proportion of correctly classified negative observations out of all the true negative observations.

$$TNR = \frac{TN}{TN + FP}. \tag{5}$$

	Autoencoder			OCSVM
	MLP	LSTM	RNN	
Activation	Relu	Sigm/Tanh	Tanh	-
Batch Size		32		-
Bias Initializers		Zeroes		-
Early Stopping Patience		5		-
Hidden Size		[32, 16, 16, 32]		-
Input/Output Size		52		-
Kernel	-	-	-	Gaussian
Learning Rate	0.005	0.013	0.002	-
Maximum Training Epochs		100		no limit
ν	-	-	-	0.00001
Optimizer		Adam [51]		-
Training Sequence	-	40	40	-
Weight Initializers		Glorot (Xavier) [52]		-

Table 1. The selected AE and OCSVM hyperparameter values utilized in the experimental assessment. In this table, Sigm denotes the Sigmoid activation [53], Tanh denotes the Hyperbolic tangent activation [54], and Relu represents the Rectified linear unit activation [55].

- **TPR**. The proportion of correctly identified positive observations out of all the true positive observations. In fault detection approaches, this is often called Fault Detection Rate (FDR).

$$TPR = \frac{TP}{TP + FN}. \tag{6}$$

- **FPR**. The number of negative observations misclassified as positive observations. In fault detection approaches, this is often called False Alarm Rate (FAR).

$$FPR = \frac{FP}{FP + TN} = 1 - TNR. \tag{7}$$

- **FNR**. The number of positive observations misclassified as negatives observations.

$$FNR = \frac{FN}{FN + TP}. \tag{8}$$

- **ACC**. The ratio between the correctly identified observations to the total observations.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}. \tag{9}$$

– **PRC**. The proportion of correctly classified positive observations out of all the predicted positive observations.

$$PRC = \frac{TP}{TP + FP}. \tag{10}$$

– **F1**. The harmonic mean of PRC and TPR. It serves as a statistical measure of the accuracy of a test or a model, taking into account both PRC and TPR simultaneously.

$$F1 = \frac{2 * PRC * TPR}{PRC + TPR}. \tag{11}$$

– **BACC**. Balanced accuracy is expressed as the average of the TPR and TNR. This metric is particularly useful for imbalanced classes, as it is the case for the anomalous datasets from the experimental assessment.

$$BACC = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right). \tag{12}$$

– **DD**. The median delay (e.g., number of elapsed samples) between the appearance of outliers and the first generated alert by the detection solution, specifically, the detection speed. In the case of the clean testing dataset, DD measures the delay until the first false alert is generated. A DD value of one indicates instantaneous detection.

The above metrics are dependent on the following: True Positives (TP), which represent the accurate detection of positive observations. True Negatives (TN) denote the correct identification of negative values. On the other hand, False Negatives (FN) indicate the erroneous classification of negative values as positive, False Positives (FP) refer to the misclassification of negative values as positive. In this study, the positive class signifies the anomalous observations, while the negative class represents the clean, outlier-free, observations.

5 Experimental Results

The results of the experimental assessment for the three tested architectures are illustrated in Tables 2, 4 and 3. It is important to mention that the results were not rounded upwards or downwards. The precision and accuracy of the reported results were maintained without any manipulation to ensure the integrity of the findings.

As shown in the three tables, the MLP based methodology obtained the overall best results, followed by the LSTM based architecture, while the worst results were obtained by the RNN based solution.

On the clean testing dataset,in terms of FPR, the LSTM based solution yielded the best results with an overall score of 0.0049 with the first false alert generated after 171 samples, followed by the RNN approach with 0.0051 with the first false alert after 164 samples, and the MLP based solution with an FPR of 0.0090 with the first false alert generated after the 74th sample.

	Detection Results MLP Autoencoder								
	TNR	FPR	TPR	FNR	ACC	PRC	F1	BACC	DD
Clean	0.9909	0.0090	-	-	-	-	-	-	74
F1	0.9899	0.0100	0.9962	0.0037	0.9951	0.9979	0.9970	0.9930	2
F2	0.9899	0.0100	0.9874	0.0125	0.9878	0.9979	0.9926	0.9886	10
F3	0.9899	0.0100	0.0145	0.9854	0.1770	0.8779	0.0285	0.5022	70
F4	0.9899	0.0100	0.7667	0.2332	0.8039	0.9973	0.8669	0.8783	1
F5	0.9899	0.0100	0.2339	0.7660	0.3599	0.9914	0.3786	0.6119	1
F6	0.9899	0.0100	1.0	0.0	0.9983	0.9979	0.9989	0.9949	1
F7	0.9899	0.0100	1.0	0.0	0.9983	0.9979	0.9989	0.9949	1
F8	0.9899	0.0100	0.9725	0.0274	0.9754	0.9979	0.9850	0.9812	17
F9	0.9899	0.0100	0.0149	0.9850	0.1774	0.8813	0.0294	0.5024	71
F10	0.9899	0.0100	0.2156	0.7843	0.3447	0.9907	0.3542	0.6027	42
F11	0.9899	0.0100	0.6089	0.3910	0.6724	0.9966	0.7560	0.7994	10
F12	0.9899	0.0100	0.9817	0.0182	0.9830	0.9979	0.9897	0.9858	6
F13	0.9899	0.0100	0.9439	0.0560	0.9515	0.9978	0.9701	0.9669	34
F14	0.9899	0.0100	0.9992	0.0007	0.9976	0.9979	0.9986	0.9945	1
F15	0.9899	0.0100	0.0166	0.9833	0.1788	0.8918	0.0326	0.5032	65
F16	0.9899	0.0100	0.1092	0.8907	0.2560	0.9818	0.1966	0.5495	37
F17	0.9899	0.0100	0.8554	0.1445	0.8778	0.9976	0.9210	0.9226	25
F18	0.9899	0.0100	0.9330	0.0669	0.9425	0.9978	0.9643	0.9614	38
F19	0.9899	0.0100	0.1182	0.8817	0.2635	0.9832	0.2110	0.5540	8
F20	0.9899	0.0100	0.4163	0.5836	0.5119	0.9951	0.5871	0.7031	39

Table 2. Illustration of the detection results on the clean and the 20 anomalous datasets using the MLP based autoencoder. The clean dataset doesn't contain anomalous points, thus only the TNR and FPR are shown. Additionally, the anomalous datasets contain both clean and anomalous data points.

All the tested solutions obtained the best scores on the F1, F2, F4, F8, F11, F12, F13, F14, F17, F18 and F20 anomalous datasets, and encountered difficulties on the F3, F9, F15 and F19 datasets. However, out of the three architectures, the MLP AE obtained the best scores for these four datasets. As highlighted in [38], the detection of F3, F9 and F15 is particularly challenging because there is no observable change in the mean, variance, and higher-order variances of the signals. Moreover, a recent study [49] discusses the nature of the faults in the TEP dataset and the effects they have on the product quality

	Detection Results RNN Autoencoder									
	TNR	FPR	TPR	FNR	ACC	PRC	REC	F1	BACC	DD
Clean	0.9948	0.0051	-	-	-	-	-	-	-	164
F1	0.9979	0.0020	0.9936	0.0063	0.9943	0.9995	0.9936	0.9966	0.9958	1
F2	0.9979	0.0020	0.9812	0.0187	0.9840	0.9995	0.9812	0.9903	0.9896	14
F3	0.9979	0.0020	0.0058	0.9941	0.1712	0.9360	0.0058	0.0117	0.5019	280
F4	0.9979	0.0020	0.1175	0.8824	0.2642	0.9965	0.1175	0.2103	0.5577	5
F5	0.9979	0.0020	0.2298	0.7701	0.3578	0.9982	0.2298	0.3736	0.6139	2
F6	0.9979	0.0020	1.0	0.0	0.9996	0.9995	1.0	0.9997	0.9989	1
F7	0.9979	0.0020	1.0	0.0	0.9996	0.9995	1.0	0.9997	0.9989	1
F8	0.9979	0.0020	0.9644	0.0355	0.9700	0.9995	0.9644	0.9817	0.9812	16
F9	0.9979	0.0020	0.0062	0.9937	0.1715	0.9394	0.0062	0.0124	0.5021	281
F10	0.9979	0.0020	0.2154	0.7845	0.3458	0.9981	0.2154	0.3543	0.6067	100
F11	0.9979	0.0020	0.3218	0.6781	0.4345	0.9987	0.3218	0.4868	0.6599	8
F12	0.9979	0.0020	0.9743	0.0256	0.9782	0.9995	0.9743	0.9867	0.9861	7
F13	0.9979	0.0020	0.9373	0.0626	0.9474	0.9995	0.9373	0.9674	0.9676	40
F14	0.9979	0.0020	0.9984	0.0015	0.9983	0.9995	0.9984	0.9990	0.9982	2
F15	0.9979	0.0020	0.0074	0.9925	0.1725	0.9489	0.0074	0.0148	0.5027	275
F16	0.9979	0.0020	0.0651	0.9348	0.2206	0.9938	0.0651	0.1223	0.5315	113
F17	0.9979	0.0020	0.7761	0.2238	0.8131	0.9994	0.7761	0.8737	0.8870	32
F18	0.9979	0.0020	0.9259	0.0740	0.9379	0.9995	0.9259	0.9613	0.9619	45
F19	0.9979	0.0020	0.0147	0.9852	0.1785	0.9733	0.0147	0.0289	0.5063	94
F20	0.9979	0.0020	0.2967	0.7032	0.4136	0.9986	0.2967	0.4575	0.6473	44

Table 3. Illustration of the detection results on the clean and the 20 anomalous datasets using the RNN based autoencoder. The clean dataset doesn't contain anomalous points, thus only the TNR and FPR are shown. Additionally, the anomalous datasets contain both clean and anomalous data points.

control within the plant, The authors identifying 9 faults (i.e., F1, F2, F5-F8, F10, F12 and F13) as quality-related faults, and 5 faults (i.e., F3, F4, F9, F11 and F15) as quality-unrelated faults.

In terms of detection speed, the overall shortest delay times were obtained by the MLP approach, with instantaneous detection on the F4, F5, F6, F7 and F14 faulty datasets. The slowest detection was obtained on F15 with a DD value of 65 samples. The recurrent based approaches yielded overall worst results, with

	Detection Results LSTM Autoencoder									
	TNR	FPR	TPR	FNR	ACC	PRC	REC	F1	BACC	DD
Clean	0.9950	0.0049	-	-	-	-	-	-	-	171
F1	0.9982	0.0017	0.9940	0.0059	0.9947	0.9996	0.9940	0.9968	0.9961	5
F2	0.9982	0.0017	0.9813	0.0186	0.9841	0.9996	0.9813	0.9904	0.9898	15
F3	0.9982	0.0017	0.0055	0.9944	0.1710	0.9405	0.0055	0.0110	0.5019	314
F4	0.9982	0.0017	0.1124	0.8875	0.2601	0.9968	0.1124	0.2021	0.5553	1
F5	0.9982	0.0017	0.2333	0.7666	0.3608	0.9984	0.2333	0.3783	0.6158	1
F6	0.9982	0.0017	1.0	0.0	0.9997	0.9996	1.0	0.9998	0.9991	1
F7	0.9982	0.0017	1.0	0.0	0.9997	0.9996	1.0	0.9998	0.9991	1
F8	0.9982	0.0017	0.9648	0.0351	0.9704	0.9996	0.9648	0.9819	0.9815	20
F9	0.9982	0.0017	0.0060	0.9939	0.1714	0.9453	0.0060	0.0121	0.5021	319
F10	0.9982	0.0017	0.2220	0.7779	0.3514	0.9984	0.2220	0.3632	0.6101	110
F11	0.9982	0.0017	0.3361	0.6638	0.4464	0.9989	0.3361	0.5029	0.6671	8
F12	0.9982	0.0017	0.9750	0.0249	0.9789	0.9996	0.9750	0.9872	0.9866	8
F13	0.9982	0.0017	0.9376	0.0623	0.9477	0.9996	0.9376	0.9676	0.9679	43
F14	0.9982	0.0017	0.9985	0.0014	0.9985	0.9996	0.9985	0.9991	0.9983	1
F15	0.9982	0.0017	0.0072	0.9927	0.1723	0.9534	0.0072	0.0143	0.5027	295
F16	0.9982	0.0017	0.0686	0.9313	0.2235	0.9948	0.0686	0.1283	0.5334	137
F17	0.9982	0.0017	0.7856	0.2143	0.8210	0.9995	0.7856	0.8797	0.8919	33
F18	0.9982	0.0017	0.9260	0.0739	0.9381	0.9996	0.9260	0.9614	0.9621	52
F19	0.9982	0.0017	0.0154	0.9845	0.1792	0.9777	0.0154	0.0305	0.5068	160
F20	0.9982	0.0017	0.3084	0.6915	0.4233	0.9988	0.3084	0.4712	0.6533	50

Table 4. Illustration of the detection results on the clean and the 20 anomalous datasets using the LSTM based autoencoder. The clean dataset doesn't contain anomalous points, thus only the TNR and FPR are shown. Additionally, the anomalous datasets contain both clean and anomalous data points.

detection delays up to 281 and 319 for the RNN and LSTM approaches. Tables 2 - 3 illustrate the remainder of the detection delay results.

Fig. 2 - 5 display the reconstruction residuals of three specifically chosen features: D Feed Stream, Reactor pressure, and Reactor temperature. Additionally, the figures show the detection decision made by the OCSVM model concerning the validity of the data points. It is important to note that the OCSVM processes a 52-dimensional data point, encompassing the reconstruction residuals for all 52 features. However, for visualization purposes, only three of these features were

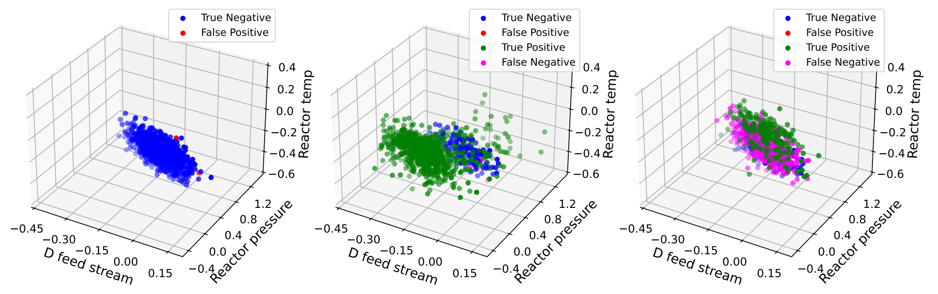

Fig. 2. Illustration of the detection results through the application of the MLP-based AE and the OCSVM, on the Clean (left), F1 (middle) and F20 (right) datasets.

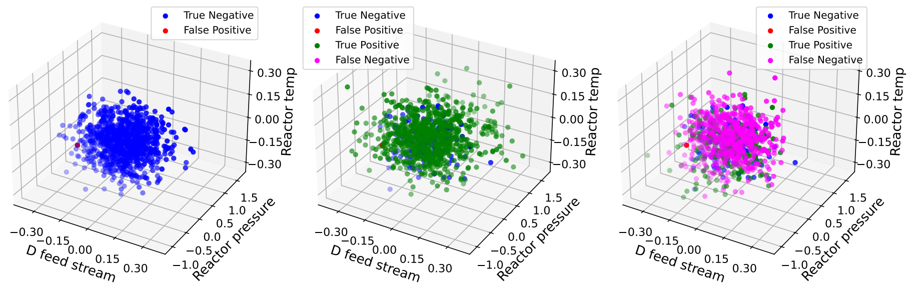

Fig. 3. Illustration of the detection results through the application of the LSTM-based AE and the OCSVM, on the Clean (left), F1 (middle) and F20 (right) datasets.

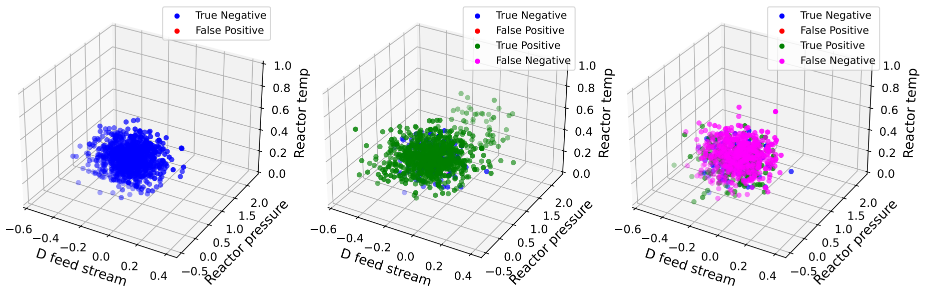

Fig. 4. Illustration of the detection results through the application of the RNN-based AE and the OCSVM, on the Clean (left), F1 (middle) and F20 (right) datasets.

plotted. As observed in Fig. 2 - 5, there is a clear difference between the residuals for the F1 scenario, where the anomalous points distance themselves from the training points. However, for F20 the points are more compact, thus increasing the separation difficulty of OCSVM and resulting in higher FNR.

Continuing with the comparative analysis of our methodology, we present the findings in Table 5, alongside results obtained from alternative detection approaches. Specifically, we compare our results with the GPR approach proposed by Maran et al. [20] and the supervised SVM approach introduced by Onel et al. [21].

	AE-OCSVM			GPR [20]			SVM [21]		
	FPR	TPR	DD	FPR	TPR	DD	FPR	TPR	DD
Clean	0.0024	-	171	0.2020	-	-	-	-	-
F1	0.0100	0.9962	2	0.0875	0.9963	4	0.00	0.9980	5
F2	0.0100	0.9874	10	0.1000	0.9838	15	0.00	0.9710	8
F3	0.0100	0.0145	70	0.0875	0.1913	17	0.00	0.0040	814
F4	0.0100	0.7667	1	0.0750	1.00	1	0.00	1.00	1
F5	0.0100	0.2339	1	0.0750	0.9925	1	0.00	1.00.	1
F6	0.0100	1.00	1	0.0500	1.00	1	0.00	1.00	1
F7	0.0100	1.00	1	0.1330	0.9963	1	0.00	1.00	1
F8	0.0100	0.9725	17	0.1625	0.9688	19	0.00	0.9340	18
F9	0.0100	0.0149	71	0.3625	0.1538	1	0.4940	0.5590	5
F10	0.0100	0.2156	42	0.0500	0.6113	84	0.075	0.7760	24
F11	0.0100	0.6089	10	0.0500	0.8075	6	0.00	0.9510	6
F12	0.0100	0.9817	6	0.1063	0.9888	12	0.00	0.9930	5
F13	0.0100	0.9439	34	0.0500	0.9575	37	0.00	0.8480	30
F14	0.0100	0.9992	1	0.0750	0.9988	1	0.00	1.00	1
F15	0.0100	0.0166	65	0.0625	0.1988	567	0.0720	0.0110	545
F16	0.0100	0.1092	37	0.2750	0.6450	29	0.00	0.8740	3
F17	0.0100	0.8554	25	0.1250	0.9025	24	0.00	0.9150	70
F18	0.0100	0.9330	38	0.0438	0.9038	85	0.00	0.9530	20
F19	0.0100	0.1182	8	0.0375	0.4213	326	0.00	0.9960	6
F20	0.0100	0.4163	39	0.0438	0.6588	75	0.00	0.9120	33

Table 5. Illustration of the performance compassion between our proposed solutions and other recent techniques. Our proposed solution is showcased in the first column. This table presents the best results obtained by each detection method.

Compared to the unsupervised GPR approach, our method obtained better results in terms of FPR, on the clean dataset, with a 19.96% difference, and also on all the faulty datasets. In terms of TPR, our method outperformed the GPR approach on the F2, F7, F8, F14 and F18 datasets, and obtained a similar score on the F1 dataset. The detection delay comparison reveals similar results on the

F4, F5 ,F6, F7 and F14 datasets. Overall, in most of the cases, our approach obtained faster detection times.

Compared to the supervised SVM approach, our method achieved better results in terms of FPR on the F9, F10, and F15 datasets, with the most significant improvement observed on the F9 dataset, showing a 48% decrease in the FPR. The TPR results reveal that our method outperformed the supervised SVM approach on the F2, F3, F8, F13 and F15 datasets. The measured detection delays are identical and instantaneous, on the F4, F5, F6, F7 and F14 datasets, for both methods. Overall, the two methods obtained similar results in terms of detection delays, with two exceptions, on F3 and F15, where our model yielded shorter delays.

6 Conclusions and Future Work

This paper introduced an unsupervised outlier detection approach designed for continuous nonlinear systems, incorporating both an AE and an OCSVM model. The AE plays a crucial role in modeling the monitored system by compressing and reconstructing all system variables at each time-step. The resulting reconstruction residuals of the signals are then utilized as multidimensional datapoints, fed into the OCSVM model, which provides binary decisions of clean or outlier for each time-step. The study evaluated three different AE architectures, namely LSTM, RNN, and MLP.

For the experimental assessment, we selected the TEP dataset, which is a representation of an industrial chemical process, extensively used for exploration, design, and evaluation of process control technologies, outlier detection and fault analysis. To measure the performance of the proposed solutions, nine performance metrics were utilized, including detection delay and Balanced Accuracy for accurate performance evaluation in unbalanced datasets.

The proposed detection solutions were tested on both clean and 20 anomalous datasets (F1 – F20). All models yielded the best results on the F1, F2, F4, F8, F11,F12, F13, F14, F17, F18 and F20 anomalous datasets, and encountered difficulties on the F3, F9, F15 and F19 datasets. Out of the three models, the MLP based approach yielded the best results in terms of detection scores, while the LSTM and RNN approaches obtained lower false alert rates. As highlighted by other authors, the detection of F3, F9 and F15 is particularly challenging due to the fact that there is no observable change in the mean, variance, and higher-order variances of the signals. In terms of detection delays, the proposed methods achieved notable results overall.

Compared to other outlier detection methods, our proposed approaches outperformed unsupervised GPR techniques, in terms of FPR, on all the datasets, with differences upwards of 19%. In terms of detection delays, our method outperformed the GPR technique in all the tested scenarios.

In comparison to supervised SVM solutions, where the models were trained with anomalous data as well, our method obtained better FPR results in three scenarios, with differences up to 48%. In the rest of the scenarios, the SVM outperformed our method with a 1% difference. In terms of TPR and detection delay, our method obtained comparable results in multiple scenarios, even outperforming the supervised approach in some of them.

As highlighted by the experimental results, there are several limitations to unsupervised approaches, and not all the outliers can be instantly detected. However, considering the fact that the anomalous data-points were seen only during detection, the models yielded notable results. Moreover, in real-life scenarios, it is often difficult to generate or to gain access to faulty or anomalous datasets, and supervised models are also limited to detecting only the scenarios that were seen during training.

As identified by others as well [20, 21], a crucial metric to consider is the detection delay. In certain situations, quickly detecting an outlier might be more crucial than the overall number of true detections over an extended period. This emphasizes the importance of timely and efficient outlier detection methods, which can be better addressed by unsupervised approaches.

Future work could explore ways to enhance the proposed solutions through a more meticulous feature selection process. However, this might result in increased overhead and reduced ease of implementation. Another option worth considering is employing ensemble approaches by integrating multiple smaller AE or prediction models, as suggested in [56]. Additionally, the outlier detection solution could be further improved by incorporating an extra module dedicated to pinpointing the source of outliers.

As previously stated, supervised approaches yield great results but are limited to detecting only seen outlier patterns. Furthermore, obtaining labeled data for all the possible scenarios might be difficult or unrealistic in certain scenarios [57]. Therefore, another interesting future direction worth exploring is the development of semi-supervised outlier detection approaches. These approaches involve training models with a subset of outlier patterns and tested against new, unseen, ones. Regarding the TEP dataset, semi-supervised models might be trained with clean and a reduced subset of faulty observations, and tested against all the available faulty scenarios.

Acknowledgments

The article is supported by the Institution Organizing University Doctoral Studies (I.O.S.U.D.), the Doctoral School of Letters, Humanities and Applied Sciences, George Emil Palade University of Medicine, Pharmacy, Science, and Technology of Târgu Mureş, 540139 Târgu Mureş, Romania.

References

1. C.-D. Cheng, B. Tian, Y.-X. Ma, T.-Y. Zhou, and Y. Shen, "Pfaffian, breather, and hybrid solutions for a (2+ 1)-dimensional generalized nonlinear system in fluid mechanics and plasma physics," *Physics of Fluids*, vol. 34, no. 11, 2022.
2. J. R. Hauser, *Numerical methods for nonlinear engineering models*. Springer, 2009.

3. A. F. Villaverde *et al.*, “Observability and structural identifiability of nonlinear biological systems,” *Complexity*, vol. 2019, 2019.
4. J. P. Higgins, “Nonlinear systems in medicine.,” *The Yale journal of biology and medicine*, vol. 75, no. 5-6, p. 247, 2002.
5. T. A. A. Elgohary, *Novel computational and analytic techniques for nonlinear systems applied to structural and celestial mechanics.* Texas A&M University, 2015.
6. T. Clark and A. D. Luis, “Nonlinear population dynamics are ubiquitous in animals,” *Nature ecology & evolution*, vol. 4, no. 1, pp. 75–81, 2020.
7. S. Sabzpoushan, “A flexible nonlinear model for simulating growth systems,” *Communications in Nonlinear Science and Numerical Simulation*, vol. 82, p. 105009, 2020.
8. A. Perrusquía and W. Yu, “Identification and optimal control of nonlinear systems using recurrent neural networks and reinforcement learning: An overview,” *Neurocomputing*, vol. 438, pp. 145–154, 2021.
9. P. Dhivya and A. Bazilabanu, “Deep hyper optimization approach for disease classification using artificial intelligence,” *Data & Knowledge Engineering*, vol. 145, p. 102147, 2023.
10. V. Tra, M. Amayri, and N. Bouguila, “Outlier detection via multiclass deep autoencoding gaussian mixture model for building chiller diagnosis,” *Energy and Buildings*, vol. 259, p. 111893, 2022.
11. B. Margalef-Bentabol, M. Huertas-Company, T. Charnock, C. Margalef-Bentabol, M. Bernardi, Y. Dubois, K. Storey-Fisher, and L. Zanisi, “Detecting outliers in astronomical images with deep generative networks,” *Monthly Notices of the Royal Astronomical Society*, vol. 496, no. 2, pp. 2346–2361, 2020.
12. C. C. Aggarwal and C. C. Aggarwal, *An introduction to outlier analysis.* Springer, 2017.
13. K. G. Mehrotra, C. K. Mohan, H. Huang, K. G. Mehrotra, C. K. Mohan, and H. Huang, *Anomaly detection.* Springer, 2017.
14. R. Bolboacă and P. Haller, “Performance analysis of long short-term memory predictive neural networks on time series data,” *Mathematics*, vol. 11, no. 6, 2023.
15. L. Zhang, J. Lin, and R. Karim, “Adaptive kernel density-based anomaly detection for nonlinear systems,” *Knowledge-Based Systems*, vol. 139, pp. 50–63, 2018.
16. M. Lazar and O. Pastravanu, “A neural predictive controller for non-linear systems,” *Mathematics and Computers in Simulation*, vol. 60, no. 3-5, pp. 315–324, 2002.
17. I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* Adaptive computation and machine learning, Cambridge, Massachusetts: The MIT Press, 2016.
18. S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
19. J. J. Downs and E. F. Vogel, “A plant-wide industrial process control problem,” *Computers & chemical engineering*, vol. 17, no. 3, pp. 245–255, 1993.
20. A. Maran Beena and A. K. Pani, “Fault detection of complex processes using nonlinear mean function based gaussian process regression: application to the tennessee eastman process,” *Arabian Journal for Science and Engineering*, vol. 46, pp. 6369–6390, 2021.
21. M. Onel, C. A. Kieslich, and E. N. Pistikopoulos, “A nonlinear support vector machine-based feature selection approach for fault detection and diagnosis: Application to the tennessee eastman process,” *AIChE Journal*, vol. 65, no. 3, pp. 992–1005, 2019.

22. Y. Wei, J. Jang-Jaccard, W. Xu, F. Sabrina, S. Camtepe, and M. Boulic, "Lstm-autoencoder-based anomaly detection for indoor air quality time-series data," *IEEE Sensors Journal*, vol. 23, no. 4, pp. 3787–3800, 2023.
23. T. de Riberolles, Y. Zou, G. Silvestre, E. Lochin, and J. Song, "Anomaly detection for ics based on deep learning: a use case for aeronautical radar data," *Annals of Telecommunications*, vol. 77, no. 11-12, pp. 749–761, 2022.
24. M. Said Elsayed, N.-A. Le-Khac, S. Dev, and A. D. Jurcut, "Network anomaly detection using lstm based autoencoder," in *Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, pp. 37–45, 2020.
25. J. Bokor and Z. Szabó, "Fault detection and isolation in nonlinear systems," *Annual Reviews in Control*, vol. 33, no. 2, pp. 113–123, 2009.
26. Y. Tan, C. Hu, K. Zhang, K. Zheng, E. A. Davis, and J. S. Park, "Lstm-based anomaly detection for non-linear dynamical system," *IEEE access*, vol. 8, pp. 103301–103308, 2020.
27. W. Yao, D. Li, and L. Gao, "Fault detection and diagnosis using tree-based ensemble learning methods and multivariate control charts for centrifugal chillers," *Journal of Building Engineering*, vol. 51, p. 104243, 2022.
28. S. Prykhodko, N. Prykhodko, L. Makarova, and A. Pukhalevych, "Outlier detection in non-linear regression analysis based on the normalizing transformations," in *2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, pp. 407–410, IEEE, 2020.
29. J. Boone and S. Chakraborti, "Two simple shewhart-type multivariate nonparametric control charts," *Applied Stochastic Models in Business and Industry*, vol. 28, no. 2, pp. 130–140, 2012.
30. G. M. Abdella, M. R. Maleki, S. Kim, K. N. Al-Khalifa, and A. M. S. Hamouda, "Phase-i monitoring of high-dimensional covariance matrix using an adaptive thresholding lasso rule," *Computers & Industrial Engineering*, vol. 144, p. 106465, 2020.
31. S. Nidsunkid, J. J. Borkowski, and K. Budsaba, "The effects of violations of the multivariate normality assumption in multivariate shewhart and mewma control charts," *Quality and Reliability Engineering International*, vol. 33, no. 8, pp. 2563–2576, 2017.
32. G. Strang, *Linear algebra and its applications.* 2012.
33. F. Hartung, B. J. Franks, T. Michels, D. Wagner, P. Liznerski, S. Reithermann, S. Fellenz, F. Jirasek, M. Rudolph, D. Neider, *et al.*, "Deep anomaly detection on tennessee eastman process data," *Chemie Ingenieur Technik*, 2023.
34. G. L. P. Palla and A. K. Pani, "Independent component analysis application for fault detection in process industries: Literature review and an application case study for fault detection in multiphase flow systems," *Measurement*, p. 112504, 2023.
35. T. Jockenhövel, L. T. Biegler, and A. Wächter, "Dynamic optimization of the tennessee eastman process using the optcontrolcentre," *Computers & Chemical Engineering*, vol. 27, no. 11, pp. 1513–1531, 2003.
36. M. Hu, X. Hu, Z. Deng, and B. Tu, "Fault diagnosis of tennessee eastman process with xgb-avssa-kelm algorithm," *Energies*, vol. 15, no. 9, p. 3198, 2022.
37. I. Lomov, M. Lyubimov, I. Makarov, and L. E. Zhukov, "Fault detection in tennessee eastman process with temporal deep learning models," *Journal of Industrial Information Integration*, vol. 23, p. 100216, 2021.
38. S. Heo and J. H. Lee, "Fault detection and classification using artificial neural networks," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 470–475, 2018.

39. B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," *Advances in neural information processing systems*, vol. 12, 1999.
40. D. M. Tax and R. P. Duin, "Support vector data description," *Machine learning*, vol. 54, pp. 45–66, 2004.
41. C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.
42. A. D. Shieh and D. F. Kamm, "Ensembles of one class support vector machines," in *Multiple Classifier Systems: 8th International Workshop, MCS 2009, Reykjavik, Iceland, June 10-12, 2009. Proceedings 8*, pp. 181–190, Springer, 2009.
43. R. Zhang, S. Zhang, S. Muthuraman, and J. Jiang, "One class support vector machine for anomaly detection in the communication network performance data," in *Proceedings of the 5th conference on Applied electromagnetics, wireless and optical communications*, pp. 31–37, Citeseer, 2007.
44. F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.
45. A. Martín *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.
46. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.
47. N. Ricker and J. Lee, "Nonlinear modeling and state estimation for the tennessee eastman challenge process," *Computers & chemical engineering*, vol. 19, no. 9, pp. 983–1005, 1995.
48. C. Rieth, B. Amsel, R. Tran, and M. Cook, "Additional tennessee eastman process simulation data for anomaly detection evaluation," *Harvard Dataverse*, vol. 1, p. 2017, 2017.
49. C. Hu, Z. Xu, X. Kong, and J. Luo, "Recursive-cpls-based quality-relevant and process-relevant fault monitoring with application to the tennessee eastman process," *IEEE Access*, vol. 7, pp. 128746–128757, 2019.
50. M. Kubat and J. Kubat, *An introduction to machine learning*, vol. 2. Springer, 2017.
51. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
52. X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, 2010.
53. J. Han and C. Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *International workshop on artificial neural networks*, pp. 195–201, Springer, 1995.
54. B. Karlik and A. V. Olgac, "Performance analysis of various activation functions in generalized mlp architectures of neural networks," *International Journal of Artificial Intelligence and Expert Systems*, vol. 1, no. 4, pp. 111–122, 2011.
55. K. Fukushima, "Visual feature extraction by a multilayered network of analog threshold elements," *IEEE Transactions on Systems Science and Cybernetics*, vol. 5, no. 4, pp. 322–333, 1969.
56. R. Bolboacă, "Adaptive ensemble methods for tampering detection in automotive aftertreatment systems," *IEEE Access*, vol. 10, pp. 105497–105517, 2022.

57. Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang, and H. Zha, “Unsupervised deep learning for optical flow estimation,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.