



Anomaly detection in oil-producing wells: a comparative study of one-class classifiers in a multivariate time series dataset

Wander Fernandes Jr.^{1,2} · Karin Satie Komati² · Kelly Assis de Souza Gazolli²

Received: 2 January 2023 / Accepted: 26 September 2023 / Published online: 8 November 2023
© The Author(s) 2023

Abstract

Anomalies in oil-producing wells can have detrimental financial implications, leading to production disruptions and increased maintenance costs. Machine learning techniques offer a promising solution for detecting and preventing such anomalies, minimizing these disruptions and expenses. In this study, we focused on detecting faults in naturally flowing offshore oil and subsea gas-producing wells, utilizing the publicly available 3W dataset comprising multivariate time series data. We conducted a comparison of different anomaly detection methods, specifically one-class classifiers, including Isolation Forest, One-class Support Vector Machine (OCSVM), Local Outlier Factor (LOF), Elliptical Envelope, and Autoencoder with feedforward and LSTM architectures. Our evaluation encompassed two variations: one with feature extraction and the other without, each assessed in both simulated and real data scenarios. Across all scenarios, the LOF classifier consistently outperformed its counterparts. In real instances, the LOF classifier achieved an F1-measure of 87.0% with feature extraction and 85.9% without. In simulated instances, the LOF classifier demonstrated superior performance, attaining F1 measures of 91.5% with feature extraction and 92.0% without. These results show an improvement over the benchmark established by the 3W dataset. Considering the more challenging nature of real data, the inclusion of feature extraction is recommended to improve the effectiveness of anomaly detection in offshore wells. The superior performance of the LOF classifier suggests that the boundaries of normal cases as a single class may be ill-defined, with normal cases better represented by multiple clusters. The statistical analysis conducted further reinforces the reliability and robustness of these findings, instilling confidence in their generalizability to a larger population. The utilization of individual classifiers per instance allows for tailored hyperparameter configurations, accommodating the specific characteristics of each offshore well.

Keywords Oil well monitoring · 3W dataset · LSTM (Long short-term memory) · One-class Support Vector Machine (OCSVM) · Local Outlier Factor (LOF) · Elliptical envelope

List of symbols

Latin symbols

b	Input vector of bias in Autoencoder neural network, Dimensionless
b'	Output vector of bias in Autoencoder neural network, Dimensionless
c	Vector of bias in Autoencoder neural network, Dimensionless
\tilde{c}_t	Is a temporary variable that contains the relevant information in the current timestep t
f	Forgetting gate of LSTM, Dimensionless
h	State of the hidden units of the network, Dimensionless
i	Input gate of LSTM, Dimensionless
k	Number of neighbors to be considered in LOF algorithm, Dimensionless

✉ Karin Satie Komati
kkomati@ifes.edu.br

Wander Fernandes Jr.
wanderfj@gmail.com

Kelly Assis de Souza Gazolli
kasouza@ifes.edu.br

¹ Petrobras, Av. Nossa Sra. da Penha, 1688 - Barro Vermelho, Vitória, ES 29057-550, Brazil

² Graduate program in Applied Computing (PPComp), Instituto Federal do Espírito Santo, Campus Serra, Av. dos Sabiás, 330 - Morada de Laranjeiras, Serra, ES 29166-630, Brazil

M	Length of multivariable time series, Dimensionless
n	Subset of observations in Elliptical Envelope algorithm, Dimensionless
o	Output gate of LSTM, Dimensionless
p	For the Wilcoxon test, a p -value is the probability of getting a test statistic as large or larger assuming both distributions are the same
P	Precision, Dimensionless
R	Recall, Dimensionless
t	Value of threshold in Elliptical Envelope algorithm selected by “fastmcd” algorithm, Dimensionless
T	Length of univariate time series, Dimensionless
U	Weight matrices for output layer in RNN, Dimensionless
V	Weight matrices for hidden layer in RNN, Dimensionless
W	Input weight matrices for input layer in RNN, Dimensionless
W'	Output weight matrices for input layer in RNN, Dimensionless
x_i	Real value in univariate time series, unit of measure
X	Input data of an autoencoder neural network, unit of measure
X'	Output data of an autoencoder neural network, unit of measure
X^i	Univariate time series, unit of measure
y_t	Output of softmax function in RNN
Z	Reduced or latent dimension in Autoencoder neural network. Dimensionless

Greek symbols

γ	OCSVM parameter that influences the radius of the Gaussian hypersphere, Dimensionless
μ	Mean, unit of measure
ν	OCSVM parameter that is used to control the sensitivity of support vectors, Dimensionless
ρ	A statistical measurement used to validate a hypothesis against observed data, Dimensionless
σ	Sigmoid function or Standard deviation, unit of measure
σ^2	Variance, square of unit of measure
Σ	Covariance matrix, Dimensionless
φ	Activation function in Autoencoder neural network, Dimensionless

Abbreviations

ABOD	Angle-based Outlier Detection
AI	Artificial intelligence
BSW	Basic sediment and water
CBM	Condition-based monitoring
CKP	Choke for production
DHSV	Down Hole Safety Valve
DLSTM	Deep long short-term memory
ELM	Extreme learning machine
F1	F1 score or the F1 measure
FN	False negative
FP	False positive
LOF	Local Outlier Factor
LSTM	Long short-term memory
MCD	Minimum Covariance Determinant
MEMD	Multiple empirical mode decomposition
ML	Machine learning
OCSVM	One-class Support Vector Machine
PDG	Permanent downhole manometer
P-JUS-CKGL	Fluid pressure downstream of gas lift
P-MON-CKP	Fluid pressure upstream to valve CKP
P-PDG	Fluid pressure at PDG
P-TPT	Fluid [Pressure at TPT
QGL	Flow of gas lift
RNN	Recurrent neural network
SVM	Support Vector Machine
T-JUS-CKGL	Fluid temperature downstream of gas lift
T-JUS-CKP	Fluid temperature downstream to CKP valve
TN	True negative
TP	True positive
TPT	Temperature transducer
T-TPT	Fluid temperature at TPT

Introduction

During oil and gas production, unwanted events called anomalies can cause significant financial impacts. Considering the average production per well in the pre-salt of 18 Mbbbl/d (thousands of barrels per day) (ANP 2020) and the average oil price of US\$110.93 (MacroTrends 2022), the loss of revenue in the event of an anomaly that interrupts the production of a well is on the order of US\$1 million dollars per day in Brazil. Additionally, vessels that carry out repairs in damaged wells (called rigs) have high costs that reach US\$500 thousand dollars per day.

Condition-based monitoring (CBM) is a strategy that verifies the condition of a system or equipment during its continuous operation (Marins et al. 2021). Understanding the behavior of fluid flow in porous media, including the phenomena of channeling and fault effects, is crucial in CBM oil and gas applications. Fluid flow in porous media

refers to the movement of fluids, such as liquids or gases, through a porous material, such as soil, rock, or sediment (Soltanmohammadi et al. 2021). Channeling, in the context of fluid flow in porous media, refers to the preferential flow of fluid through specific pathways or channels within the porous material. Instead of uniformly spreading throughout the porous media, the fluid concentrates its flow in certain areas, resulting in uneven distribution and potentially reduced effectiveness of fluid transport or extraction processes. Faults are fractures or planes of weakness in the Earth's crust where the movement has occurred. They can affect fluid flow in several ways. Firstly, faults can act as conduits or barriers for fluid movement, either facilitating or obstructing the flow of fluids through the porous media. Secondly, fault zones can introduce permeability variations, leading to preferential fluid flow along the fault surfaces or altered flow patterns within the porous media. Lastly, faults can also influence the overall structure and geometry of the porous media, affecting the flow dynamics and distribution of fluids.

Automatic monitoring of the oil production process could detect and prevent anomaly events. Detecting anomalies is a hard task because it does not have a set of characteristics or rules that aggregate them and there are a lot of challenges in fault detection for process signals of traditional methods (Alrifayy et al. 2021). An anomaly can be occasional, a single extreme value (such as a temperature) above a threshold can be enough to characterize an anomaly. A sudden change in temperature during an industrial process can be regarded as abnormal, even if the initial and final values of the change are not atypical in isolation (Chandola et al. 2009). In many industrial processes, we seek to detect irregular patterns, in which most observations refer to typical situations, and the minority, to rare situations that we want to identify (Santos and Kern 2016).

Anomaly detection is a binary classification problem (as normality and abnormality) (Castro et al. 2021). A possible solution to predict an anomaly is the application of multivariate statistics (Soriano-Vargas et al. 2021) and machine learning (ML) methods (D'Almeida et al. 2022). In industrial processes, the input data for monitoring come from several sensors indexed by time, that is, they are multivariate time series. As written by Fawaz et al. (2019), time series classification has been considered one of the most challenging problems in data mining. The detection of new patterns (novelty detection) can be done with classifiers of a single class, in which only data associated with the common class (normality) are used in the training (Khan and Madden 2014). The challenges associated with petroleum time series data, as mentioned by (Sagheer and Kotb 2019), include excessive noise, defects, anomalies, high dimensionality, non-stationarity, variable trends, and the nonlinear and heterogeneous nature of reservoir properties.

This work applied and compared machine learning techniques to detect anomalies in oil-producing wells, using the 3W dataset composed of multivariate time series. The present work used the benchmark for anomaly detection proposed by Vargas et al. (2019) and extended it with more classifiers Local Outlier Factor (LOF), Elliptical Envelope, and neural networks of the type Autoencoder with layers feedforward and recurrent LSTM type (Long Short-Term Memory), besides the hyperparameter calibration step.

Tariq et al. (2021)'s work provided a detailed and comprehensive review of data science and ML roles in different petroleum engineering. The work of Tariq et al. (2021) also brings a discussion about the limitations of the ML model, and one of the limitations is the availability of data addressed through the use of a public and annotated database. Vargas et al. (2019) made the 3W database public, which is composed of multivariate time series. The 3W dataset contains 1984 time series instances of the production of surge-type offshore oil wells (wells that manage to flow the produced fluids to the platform with their pressure). These instances are separated into normal conditions and anomalies, and the anomalies are organized into eight classes. This base can be used both for detecting and classifying anomalies in oil wells. In addition to the base with real production data, Vargas et al. (2019) also developed two specific benchmarks, one for evaluating the impact of using simulated and hand-drawn instances and another for detecting anomalies. In the benchmark for anomaly detection, the Isolation Forest (with F1 metric of 0,727) and OCSVM - One-class Support Vector Machine (with 0,47 of F1 metric) techniques were used.

Experiments are carried out with and without the feature extraction step. In the experiments with feature extraction, the median, mean, standard deviation, variance, maximum, and minimum are extracted for each variable. In the experiments without feature extraction, the time series themselves are the input for the classifiers.

Friedman's and Wilcoxon's statistical tests assess whether the tested classifiers generate performance metrics whose average is different from the others (Demšar 2006).

The main contributions of the present work are:

1. Anomaly detection in oil wells using one-class classifiers;
2. Comparison of one-class classifiers with and without feature extraction to assess their performance;
3. Comparison of multiple one-class classifiers in anomaly detection using the 3W database, including five classifiers for experiments with feature extraction and six for experiments without feature extraction;
4. Evaluation of the effectiveness of different one-class classifiers in detecting anomalies in both real-world and simulated scenarios. This approach enabled us to examine the impact of feature extraction on detection

performance across datasets with diverse characteristics; and

5. Utilization of statistical tests at the 5% significance level by Wilcoxon's statistical tests with Bonferroni adjustment to validate the results and ensure their generalizability to a larger population.

This paper is organized as follows: Section 2 describes related works; Section 3 details the material (3W database) and methods for each step of the experimental methodology; Section 4 presents the results and analyzes the performance of the proposed system, and Section 7 closes the paper emphasizing its conclusions.

Related works

This section presents two subsections. The first subsection describes articles that use anomaly detection techniques in several applications and the second, articles that used the 3W dataset.

Anomaly detection

The work of Chandola et al. (2009) is an important survey article on anomaly detection. It presents contributions and discussions about the concept of anomaly, and its different aspects in each application domain, providing a structured overview, grouping existing techniques into different categories, and identifying the advantages and disadvantages of each one. It also discusses the computational complexity of the techniques. While Chandola et al. (2009) extensively discuss the concept of anomalies and provide a structured overview of existing techniques across various application domains, our research focuses specifically on the detection of faults in naturally flowing offshore oil and subsea gas-producing wells. By narrowing the scope, we aim to evaluate the performance of different one-class classifiers in this specific domain. Unlike previous studies, our work compares the performance of these classifiers with and without feature extraction, examines their effectiveness in both simulated and real instances, and conducts rigorous statistical analysis to validate the results and ensure their generalizability.

Barbariol et al. (2019) proposed an anomaly detection approach in metering modules. This equipment is an important tool in the oil and gas sector, as it simultaneously provides real-time data on the flows of oil, gas, and water. The Cluster-Based Local Outlier Factor and Isolation Forest algorithms were used to detect quality changes in the measurements performed, using a semi-synthetic dataset. In contrast to their work, our research addresses the detection of faults in naturally flowing offshore oil and subsea gas-producing wells. The application is different but used

the same one-class classifiers, LOF and Isolation Forest, as this work.

Chan et al. (2019) performed anomaly detection in programmable logic controllers that make up supervisory control and data acquisition systems. Such equipment manages sensor-based industrial equipment operations and is exposed to cyber threats. A case study involving a traffic light simulation was carried out which demonstrated that anomalies are detected with high precision using One-class SVM. The application is different but used the same one-class classifiers, OCSVM, as this work.

Khan et al. (2019) applied anomaly detection techniques in unmanned aerial vehicles. In the experiments, the Aero-Propulsion System Simulation database was used and then, more experiments were carried out in a real vehicle. The anomaly detection technique was the Isolation Forest algorithm. The application is different but uses the same one-class classifier, Isolation Forest, as this work.

Tan et al. (2020) compared the performance of several classifiers for anomaly detection in marine vessel machines. The safety and reliability of navigation depend on the performance of these machines, and intelligent condition monitoring is essential for maintenance activities. A dataset from a ship's gas turbine propulsion system was the input in the experiments. They investigated the performance of single class classifiers: OCSVM, Support Vector Data Description, Global K-Nearest Neighbors, LOF, Isolation Forest, and ABOD (Angle-based Outlier Detection). The OCSVM algorithm obtained the best accuracy results. As this work, Tan et al. (2020) specifically investigated the performance of one-class classifiers. Their findings revealed that the OCSVM algorithm achieved the best accuracy results, which is used in this experiment.

Grashorn et al. (2020) describe the use of neural networks to detect anomalies in the operation of the Columbus module of the International Space Station. It is a scientific laboratory that transmits around 17,000 telemetry parameters per second to Earth. The Columbus Control Center operations team, in collaboration with Airbus, monitors these parameters and uses autoencoder-type algorithms with LSTM-type cells to support the detection of anomalies during the center's workflow of control. This research highlights the successful application of deep learning techniques in anomaly detection. We also evaluate the performance of LSTM architecture.

Said Elsayed et al. (2020) used a combination of neural network structure autoencoder with LSTM type cells, together with the OCSVM algorithm, to model the normal data flow in a computational network. The experiments showed that the proposed model can efficiently detect the anomalies presented in the network traffic data. As the work of Said Elsayed et al. (2020), we also evaluate the performance of the OCSVM algorithm and LSTM type cells.

ML applied to multivariate time series in the oil industry

Tariq et al. (2021)'s work is a comprehensive state-of-the-art on artificial intelligence (AI) applied in the oil industry, which we highly encourage anybody to read. In this subsection, we will focus on papers that used ML techniques on time series.

The objective of the paper of Takbiri-Borujeni et al. (2019) is to implement an artificial intelligence technique to develop a model for more accurate and robust real-time drilling performance monitoring and optimization. They have proved that the data-driven model built using the multilayer perceptron (backpropagation, feed-forward neural network model) technique can be successfully used for drilling performance monitoring and optimization, especially in identifying the bit malfunction or failure.

Sagheer and Kotb (2019) proposed an approach based on deep long-short-term memory (DLSTM) architecture tested in two case studies from the petroleum industry domain that are carried out using the production data of two actual oil-fields. A genetic algorithm is applied in order to optimally configure DLSTM's optimum architecture. The performance of the proposed approach is compared with several standard methods, and the empirical results show that the proposed DLSTM model outperforms other standard approaches. Besides, the authors list the challenges of samples of petroleum time series data: (i) often contain excessive noise, defects, anomalies, and high dimensionality; (ii) they are non-stationary and may exhibit variable trends by nature; and (iii) the rock and fluid properties of the reservoirs are highly nonlinear and heterogeneous.

The paper of Li and Fan (2020) presents a novel method combining extreme learning machine (ELM) and multiple empirical mode decomposition (MEMD) to identify flow patterns of oil–water two-phase flow. In the proposed method, they employ the MEMD to decompose the multivariate conductance signal of oil–water two-phase flow, select the normalized energy of the high-frequency components as the eigenvalue, and utilize the trained ELM. The experimental results show that the proposed method has high accuracy in identifying five typical flow patterns of oil–water two-phase flow.

In contrast to these previous studies, our research focuses on anomaly detection in naturally flowing offshore oil and subsea gas-producing wells using one-class classifiers. Additionally, we investigate the impact of feature extraction on detection performance and evaluate the effectiveness of these classifiers in both simulated and real instances. By addressing the specific context of fault detection in oil wells and comparing various one-class classifiers, our work provides a valuable contribution to the existing literature on anomaly detection in the oil industry.

3W dataset

Marins et al. (2021) have studied different approaches to the automatic detection and classification of faulty events in the 3W dataset¹ with random forest. Seven fault classes are considered, with distinct dynamics and patterns, as well as several instances of normal operation. Three different classification scenarios are devised: fault detector, single-fault detector/classifier, and multi-fault classifier. In experiment 1, they investigate a single classifier to discriminate only between normal and faulty operations, as this work. They balanced the dataset and kept the same class proportions in the training set throughout every experiment. They obtained 97.1% accuracy in classification with random forest. As the results are balanced and the F1 measures are not provided, they will not be comparable to the results in this work.

In experiment 1, which is similar to our work, they investigated a single classifier to discriminate between normal and faulty operations. However, there are differences between their approach and ours. Firstly, they balanced the dataset and maintained the same class proportions in the training set throughout every experiment, whereas our study did not explicitly mention balancing the dataset. Secondly, they did not use instances of the “Incrustation in CKP” class in their work.

Turan and Jäschke (2021) have used a sliding window with feature extraction, followed by standardization, grid search, feature selection, and k-fold cross-validation. Their study applies multi-class classification for 7 classes, except for scaling in CKP. They compared 7 classifiers: Logistic regression, Support Vector Classifier, Linear and Quadratic Discriminant Analysis, Decision Trees, Random Forest, and AdaBoost. A decision tree classifier reached an F1 score of 85% on test data. Although the steps of the proposal are similar to those of this work, the objective is different, as Turan and Jäschke (2021)'s work seeks to classify failure events into their different classes. Nevertheless, the results of Turan and Jäschke (2021) are compared with this work.

Although the methodology steps of Turan et al.'s work share similarities with our paper, the objectives, and research questions are different. Turan and Jäschke (2021) aimed to classify failure events into different classes, whereas our study focused on anomaly detection in oil wells using one-class classifiers. Our research question was centered around identifying the most effective approach for detecting faults in oil and gas-producing wells using the publicly available 3W dataset. While the objective and focus of our work differ from Turan and Jäschke (2021), it is still valuable to compare their results with ours. Their decision tree classifier achieved an F1 score of 85% on test data. Comparing their

¹ https://github.com/ricardovargas/3w_dataset.

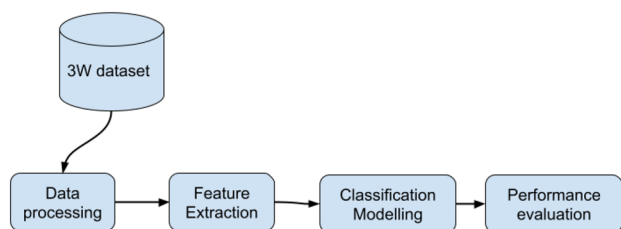


Fig. 1 Block diagram of the proposed system

results with our findings provides insights into the relative performance of different classification methods and adds to the broader understanding of anomaly detection and fault classification in related domains.

Methodology

The anomaly detection in the oil wells process has five stages: data collection, pre-processing, feature extraction, classification, and performance evaluation. The dimensionality reduction step after feature extraction does not exist in the approach proposed in this work. Fig. 1 shows the system pipeline, and the data collection, in this work, is based on the existing 3W dataset. The following subsections describe each of the steps after data collection. The source code is publicly available on the GitHub website.²

Data collection

An oil well is a structure drilled into the ground in stages that form an inverted telescope (diameters decrease as depth increases) and is equipped with devices and sensors that allow the flow of oil and gas from the oil reservoir rock to the surface (Guo et al. 2007). To enable the production of oil and gas in the maritime environment, the wells are connected to systems composed of subsea equipment installed on the seabed and lines that allow the control of the well and the oil flow to production, storage, and transfer platform (Gerwick 2007).

Fig. 2 shows a simplified schematic of an oil production system, including the well, the subsea system, and the platform. Oil and gas flow from an oil reservoir rock through the production column and then through a production line to a platform. Valves installed on the seabed are remotely operated by an electro-hydraulic umbilical. There are sensing devices that assist in monitoring: a permanent down-hole manometer (P-PDG), a temperature transducer (T-TPT), and a pressure transducer (P-TPT). The DHSV (Down Hole

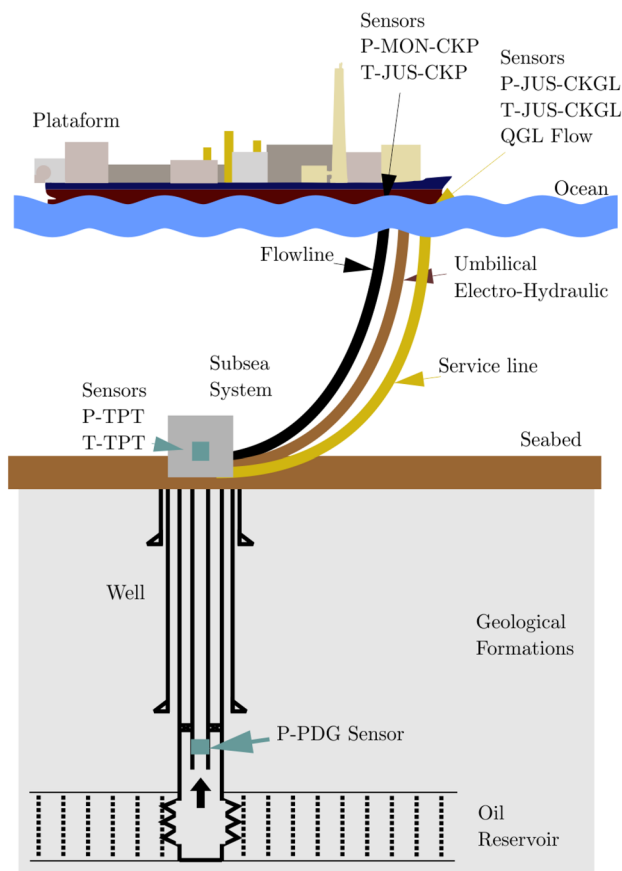


Fig. 2 Simplified schematic of an oil subsea production system

Safety Valve) is a safety valve installed in the well production column. Its purpose is to ensure well closure in a situation where the production unit and well are physically disconnected or in an emergency or catastrophic failure of surface equipment. The CKP valve (Choke for Production) is located on the platform and is responsible for controlling the opening of the well. It has temperature (T-JUS-CKP) and pressure (P-MON-CKP) sensors. The gas lift line on the platform has flow (flow QGL), temperature (T-JUS-CKGL), and pressure (P-JUS-CKGL) sensors.

Each instance of the 3W dataset represents an operational condition of a well. It is composed of eight variables (Eight-time series), as described in Table 1, from sensors of oil production systems, according to the approximate physical location shown in Fig. 2. There is an additional variable that is an observation-level label vector that establishes up to three periods in each instance of any type: normal, anomaly transient, and anomaly steady state. All collected instances have a fixed sampling rate (1 Hz).

According to Vargas et al. (2019), some inherent difficulties of the 3W dataset are the missing variables and frozen variables. The database has 4947 missing variables (values unavailable due to problems in sensors or communication

² <https://github.com/wanderfernandesjunior/mestrado>.

Table 1 Description of time series variables present in 3W dataset - adapted from Vargas et al. (2019)

Variable	Description	Unit
P-PDG	Fluid pressure at PDG	Pa
P-TPT	Fluid pressure at TPT	Pa
T-TPT	TPT fluid temperature	°C
P-MON-CKP	Fluid pressure upstream to valve CKP	Pa
T-JUS-CKP	Fluid temperature downstream to CKP valve	°C
P-JUS-CKGL	Fluid pressure downstream of gas lift	Pa
T-JUS-CKGL	Fluid temperature downstream of gas lift	°C
QGL	Flow of gas lift	m ³ /s
Label vector	Numerical value that indicates the state of each anomaly along the time series: normal period, anomaly transient and anomaly steady state	-

networks or because the variable is not applicable to the instance) that represent 31.17% of all 15,872 variables from all 1,984 instances. It also has 1535 frozen variables (values that remain fixed due to problems with sensors or communication networks) that represent 9.67% of all 15,872 variables from all 1,984 instances.

Besides, from 1984 instances of the base 3W dataset, 597 are normal and 1397 are anomalies. One relevant aspect of the dataset is its natural class imbalance for all events. This is a major challenge to the application of machine learning methods. The 3W dataset is a collection from 3 (three) different sources, as described in the items below:

- Real instances: historical and real data that occurred in producing wells;
- Simulated instances: data created by a dynamic multiphase simulator, in which models were calibrated by specialists in the petroleum area;
- Hand-drawn instances: data digitized from paper forms, in which experts have hand-drawn specifying their attributes such as magnitude (variable), scales, and type of event (anomaly).

The 3W dataset categorizes anomalies into eight classes, which are described in Table 2. All faults are combined into a unique class which is compared against the normal-operation class. The different types of anomalies also have different dynamics in terms of the speed of occurrence. For example, for an anomaly of the “Incrustation in CKP” class the average size of the occurrence time window is up to 72 h, while for the “Flow Instability” class this window is 15 min. Class 3 and Class 4 are not used in this work.

Figure 3 presents an example of the class “Abrupt Increase of BSW” with graphs of P-PDG, P-TPT, T-TPT, P-MON-CKP, and T-JUS-CKP. It is possible to verify that the well was initially in normal operation (green), then changed to the transient anomaly (red) until reaching the stable state of anomaly (orange). In this example, the steady

anomaly state was reached after about 8 h and 30 min from the onset of the anomaly. It also noticed the existence of a frozen variable (P-PDG) possibly caused by operational problems.

Figure 4 presents an example of the class “DHSV Spurious Closure” with the curves of the variables P-PDG, P-TPT, T-TPT, P-MON-CKP, and T-JUS-CKP. The well was initially in normal operation (green), then passing through the anomaly transient (oscillation of pressure, temperature, and flow values in the sensors) in red, until reaching the anomaly steady state (orange). The well took about 35 min to enter the anomaly steady state (orange).

Data pre-processing

The data pre-processing step deals with the initial preparation of the collected dataset. In the case of time series, this step includes data analysis, generation of graphs to understand the data, removal of null and/or frozen values, and

Table 2 Breakdown of events in the 3W database (Vargas et al. 2019)

Class #	Type of event	Real	Simulated	Hand-drown	Total
0	Normal	597			597
1	Abrupt increase in BSW (Basic sediment and water)	5	114	10	129
2	DHSV Spurious closure	22	16		38
3	Severe intermittence	32	74		106
4	Flow instability	344			344
5	Fast productivity Loss	12	439		451
6	Fast restriction in CKP	6	215		221
7	Incrustation in CKP	4		10	14
8	Hydrate in production line	3	81		84

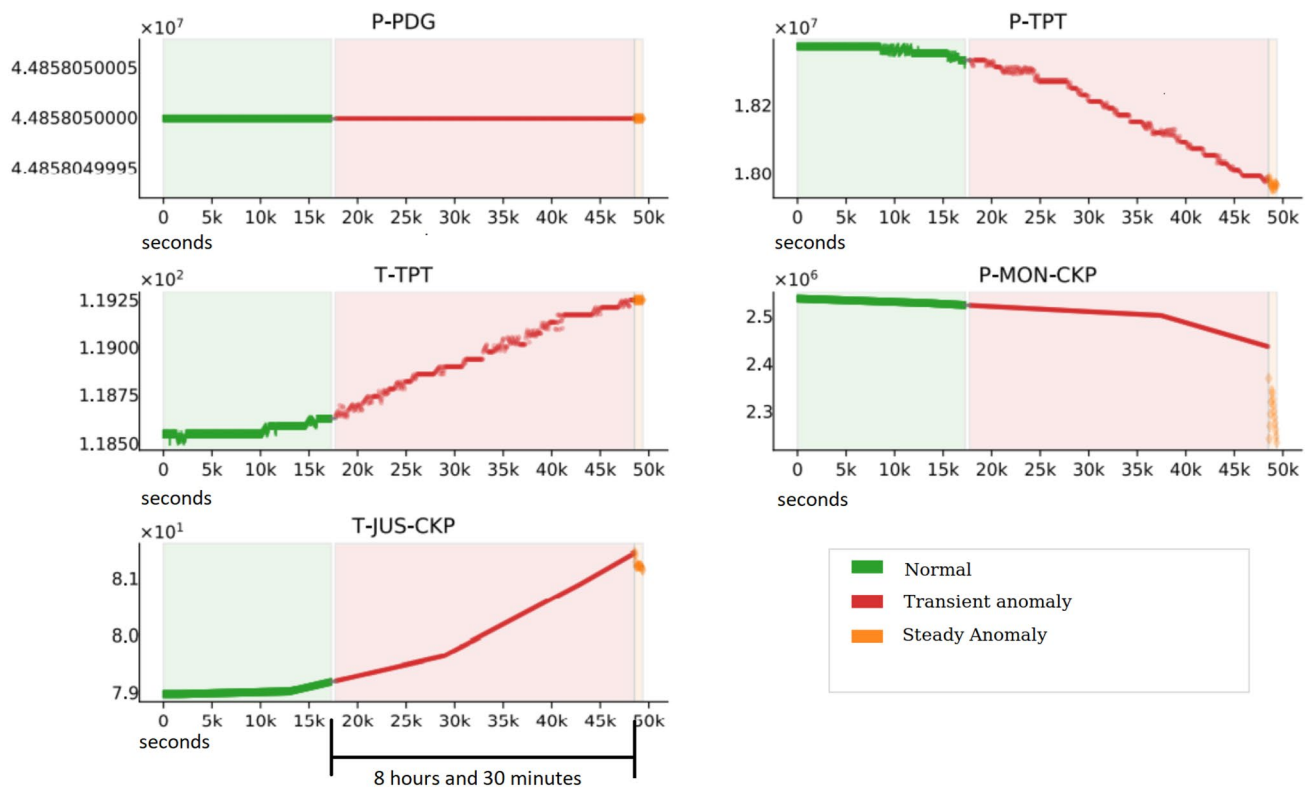


Fig. 3 Instance of class 1 “Abrupt Increase of Basic Sediment and Water” from 3W dataset

re-sampling of time-series observations to balance the database (Pal and Prakash 2017).

In the pre-processing stage, instances were sampled with a sliding window, generating up to 15 samples with 180 observations each, as illustrated in Fig. 5a. For normal periods, the first 60% of observations are for training, and the remaining are for validation (40%). The anomaly periods are used only for validation, as illustrated in Fig. 5b.

The variables of the samples (generated with 180 observations each) with missing values above a threshold (10%) or that have standard deviations below a threshold (1%) are discarded. The variables of the training samples are normalized using the mean and standard deviation. The validation data are normalized by the mean (μ) and standard deviation (σ) values (Fig. 5c).

Feature extraction

A univariate time series $x = [x_1, x_2, \dots, x_T]$ is an ordered set of real values. The length of x is equal to the number of real values T . The time series is multivariable $X = [X^1, X^2, \dots, X^M]$ when it consists of M different univariate time series with $X^i \in \mathbb{R}^T$ (Fawaz et al. 2019).

For each time series sample, the median, mean, standard deviation, variance, maximum, and minimum are extracted as characteristics following Vargas et al. (2019) (Fig. 6). For

this process, the library **tsfresh**³ (Time series feature extraction) Christ et al. (2018) was used with minimum parameters configuration, with the aim of reproducing the initial results collected by Vargas et al. (2019).

Classification

One-class classifiers detect rare patterns. Only the common class (normality) is used in training (novelty detection) and in tests, there is a mixture of normal and abnormal instances (Khan and Madden 2014). In the experiments, the following state-of-the-art classifiers are OCSVM, Isolation Forest, LOF, Elliptical Envelope, and neural networks (autoencoder feedforward and autoencoder LSTM). In the benchmark of 3W dataset for anomaly detection, the Isolation Forest and OCSVM techniques were used by Vargas et al. (2019). To carry out the experiments, the algorithms implemented in the libraries Scikit-Learn⁴ developed by Pedregosa et al. (2011b) and Tensorflow⁵ developed by Abadi et al. (2015).

³ <https://tsfresh.readthedocs.io>.

⁴ <https://scikit-learn.org/>.

⁵ <https://www.tensorflow.org/>.

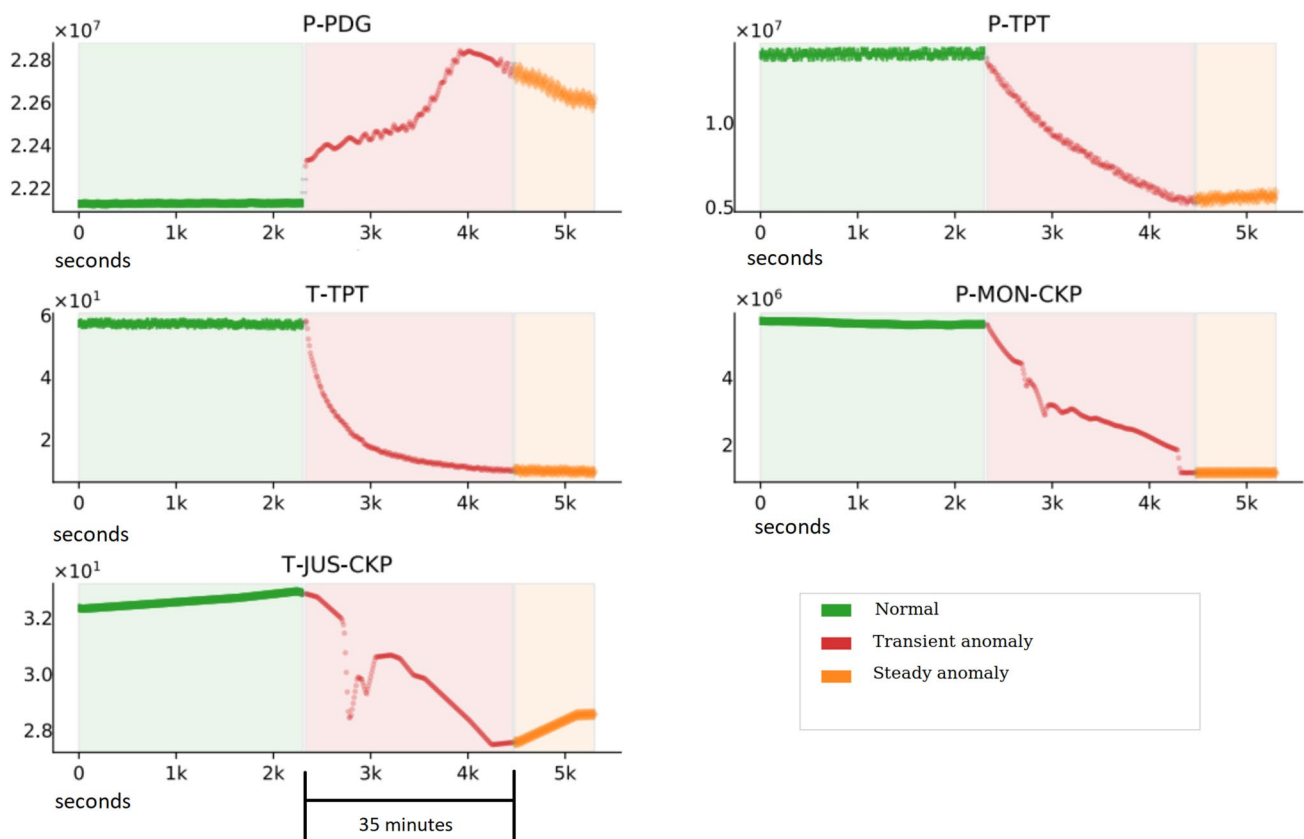


Fig. 4 Instance of class 2 “Down Hole Safety Valve Spurious Closure” from 3W dataset

One-class Support Vector Machine (OCSVM)

One-class SVM was introduced by Schölkopf et al. (2001), and it aims to separate the dataset from the source and build a hypersphere that encompasses all normal instances in a space. One new instance is classified as an anomaly when it does not fit within the space of this hypersphere. Fig. 7 shows a hypersphere containing the target data having center c and radius r . Objects on the boundary are support vectors (yellow), and two objects (red) that lie outside the boundary are anomalies, and inside are normal (green).

The OCSVM hyperparameters are the kernel (linear, polynomial, radial), γ (gamma), and ν (nu). The γ parameter influences the radius of the Gaussian hypersphere that separates normal instances from anomalies - large values of γ result in a smaller hypersphere and a “harder” model that finds more discrepancies. The fraction ν is used to control the sensitivity of support vectors and defines the percentage of the dataset that is an outlier and helps create tighter decision boundaries (Misra et al. 2020). Kernels are nonlinear mathematical functions that map the feature space into a space of higher dimensionality in which the separability between classes tends to be greater (Géron 2019).

Isolation forest

Search-based decision trees are built based on rules inferred from the attributes. Liu et al. (2012) called an isolation tree when in each node an attribute is selected randomly and then, the dataset is divided in two from a random threshold value (between the minimum and maximum values). The dataset is gradually split until all instances are isolated (Géron 2019). The anomalies are usually in less populated regions of the dataset, and fewer random partitions are usually needed to isolate them at nodes in the tree (Misra et al. 2020).

The method ensemble Isolation Forest seeks to create a structure of random trees to isolate the anomalies of the instances. As illustrated in Fig. 8, anomalies (red) are more susceptible to isolation and are closer to tree roots, while normal points (green) are difficult to isolate and are usually at the deepest end of the tree. Leaves that are not normal or anomalies are considered unusual (blue). The average path lengths across multiple trees are used to score and rank the instance (Chen et al. 2016).

The Isolation Forest implementation, available in the Scikit-learn library (Pedregosa et al. 2011a), accepts the following hyperparameters: number of estimators (number

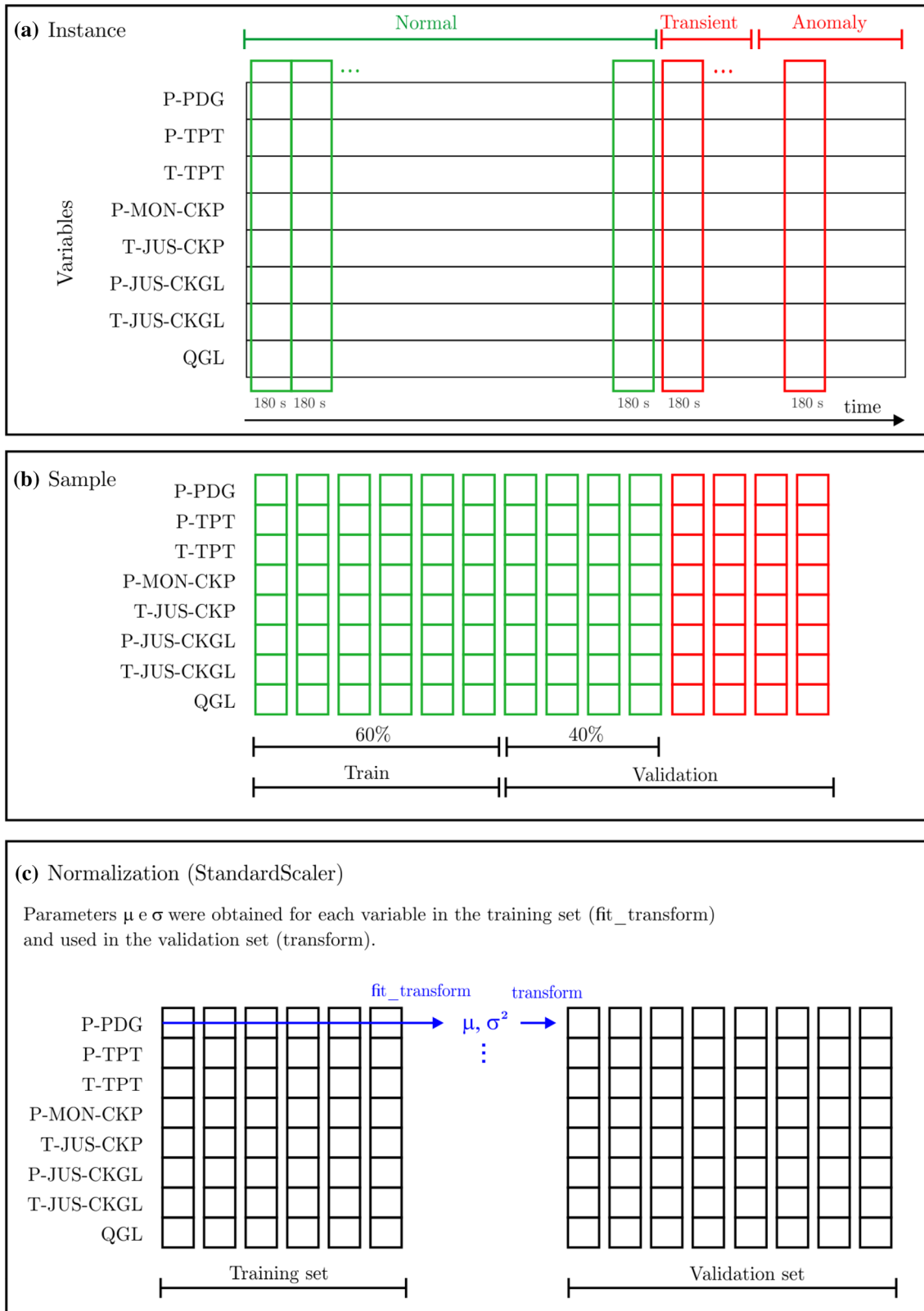


Fig. 5 Representation of the pre-processing process

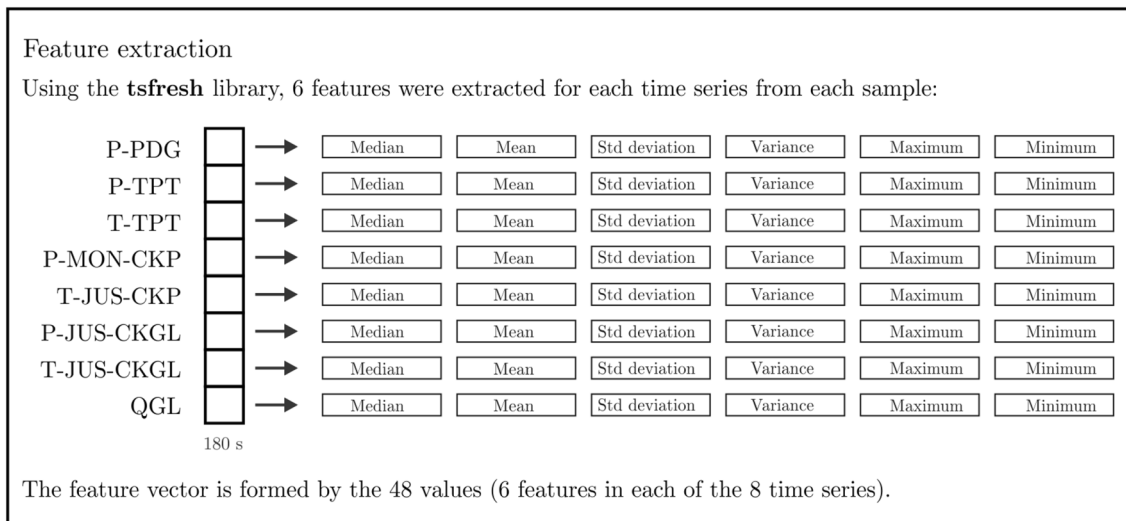


Fig. 6 Representation of the feature extraction process

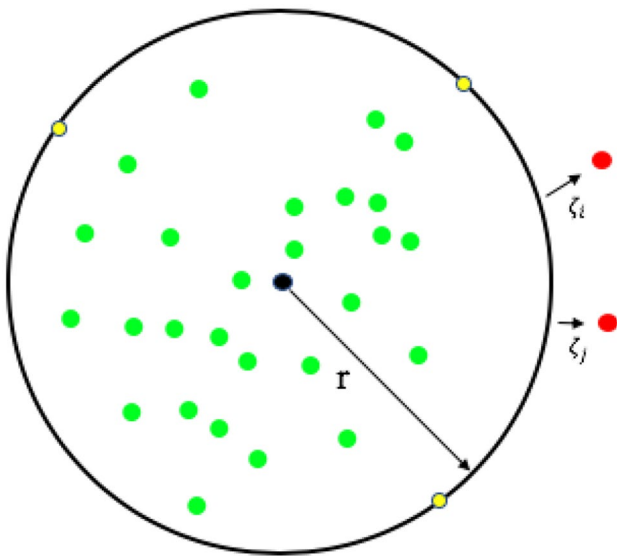


Fig. 7 One-class Support Vector Machine

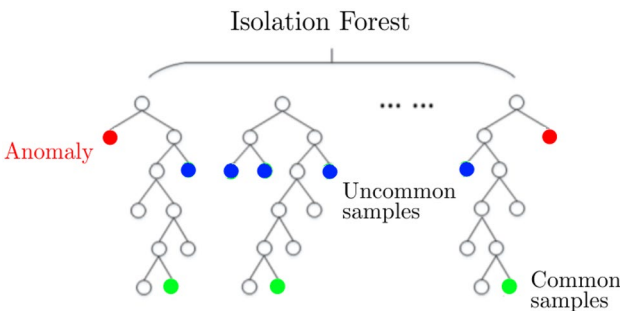


Fig. 8 Example of Isolation Forest structure

of trees), the maximum number of samples used per tree, number of characteristics used for each tree, and contamination in the dataset (estimate of outliers) (Misra et al. 2020).

Local Outlier Factor (LOF)

LOF was developed by Breunig et al. (2000) and compares the density of instances around a given instance with the density around its neighbors. The LOF algorithm operates under the assumption that anomalies are data points that have a significantly different density compared to their neighbors. In other words, anomalies are instances usually more isolated than their closest neighbors. Element ‘A’ has a much lower density than its neighbors in Fig. 9. The algorithm calculates an observation’s LOF score as the ratio of the average local density of its nearest *k* neighbors to its own local density.

Samples with a local density similar to that of their neighbors are considered normal instances, while those with a lower local density are considered anomalies (Misra et al. 2020). LOF considers an outlier not as a binary property, but as the relative degree to which the object is isolated from its surrounding neighborhood. For objects deep inside a cluster, the LOF value is approximately 1.

To calculate the LOF score for a data point, the algorithm compares the density of its local neighborhood to the densities of the neighborhoods of its neighbors. If a data point has a much lower density than its neighbors, it will have a higher LOF score, indicating a higher likelihood of being an anomaly. The LOF hyperparameters are the number *k* of neighbors to be considered, distance metric, and contamination in the dataset. The *novelty = True* parameter allows the

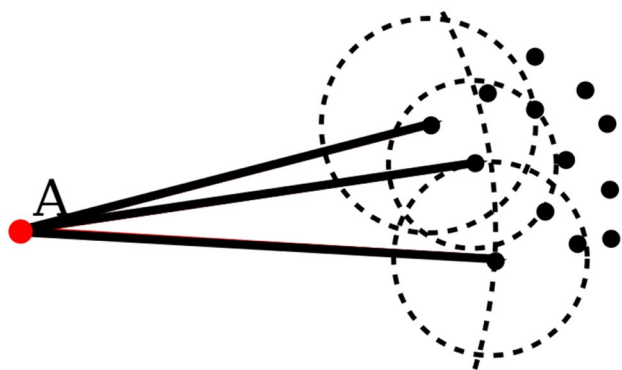


Fig. 9 Local Outlier Factor (LOF): comparison of the local density of a point with the densities of its neighbors. Element 'A' has a much lower density than its neighbors

technique to be applied to new data. Typically, larger values of k capture a broader context, while smaller values focus on more local relationships.

The LOF algorithm has the advantage of being able to detect anomalies in datasets with complex distributions and varying densities. It is especially useful in cases where anomalies are not easily characterized by specific rules or patterns. LOF is also robust to the presence of noise and able to handle datasets with high dimensionality. The performance of the LOF algorithm can be influenced by the choice of parameters, such as the value of k and the distance metric used. It is important to tune these parameters appropriately for each specific dataset and application.

Elliptical envelope

The Elliptical Envelope implements the Minimum Covariance Determinant (MCD) technique which assumes that normal instances are generated from a single Gaussian distribution. This distribution provides an estimate of the elliptical envelope from which anomalies can be identified (Géron 2019). Eq. (1) represents the multivariate cases of the Gaussian distribution, respectively, where μ is the mean, σ^2 is the variance and Σ is the covariance matrix.

$$f(x) = \frac{1}{\sqrt{(2\pi)^n \Sigma}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right). \quad (1)$$

The distance of each observation must be calculated in relation to some measure of centralization of the data, which is considered an anomaly in the observation whose distance is greater than some predetermined value (Hardin and Rocke 2004). The distance of an observation from the distribution can be calculated using the Mahalanobis distance. Estimates such as the Mahalanobis distance are considered part of

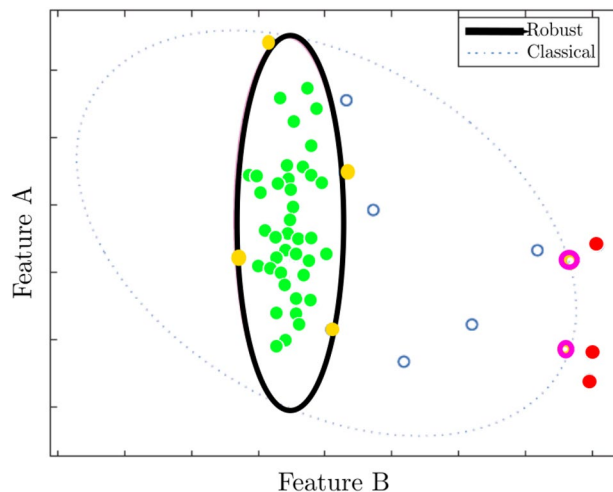


Fig. 10 Elliptical Envelope

classical statistics and are greatly affected by outliers (called the masking effect). To obtain better results, Rousseeuw and Driessen (1999) proposed the use of robust estimators, whose characteristic is to be less influenced by deviations caused by the presence of outliers in the data set. Fig. 10 illustrates two elliptical envelopes generated using classical (dotted line) and robust (black line) statistics (Hubert and Debruyne 2010). It is possible to verify that the robust approach has a smaller area and encompasses the elements in a higher-density space. In robust mode, normal objects are green, those on the boundary are yellow, and all outside the edge are anomalies. In classic mode, normal objects are green and blue, boundary objects are pink and anomalies are red.

The robust distance estimator method uses a subset of observations n , where $t < n$, that allows obtaining an estimate resistant to outliers in the dataset (Hubert and Debruyne 2010). The selection of the value of t is done according to the “fastmcd” algorithm proposed by Rousseeuw and Driessen (1999), which finds the value that minimizes the determinant of the covariance matrix.

The algorithm implemented in the Scikit-Learn library by Pedregosa et al. (2011a) allows the adjustment of parameters that influence the robust distance calculation with MCD: centering, support fraction, and contamination in the dataset.

Autoencoder

An autoencoder is a neural network structure in which the number of nodes in the output layer is the same as in the input layer and the architecture is symmetrical. Fig. 11 illustrates an example of a fully connected autoencoder. The main objective of this concept is that both encoder and decoder are trained together and the discrepancy between

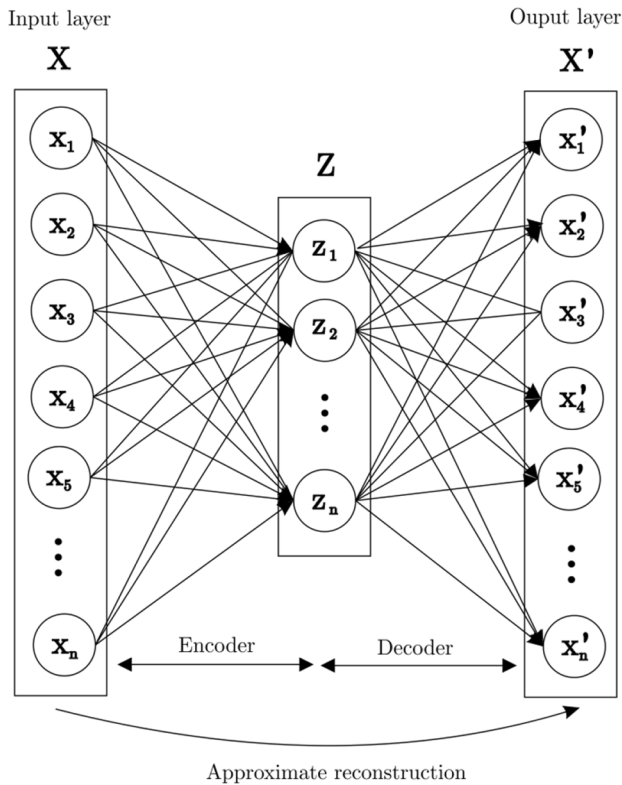


Fig. 11 Example of a Autoencoder neural network

the original data and its reconstruction, based on some cost function, is minimized (Kwon et al. 2019).

In the encoding step (encoder), the dimensions of the X input data are reduced according to Eq. (2a).

$$Z = \varphi(WX + b) \tag{2a}$$

$$X' = \varphi'(W'Z + b') \tag{2b}$$

where Z is the reduced or latent dimension, φ is the activation function, W is the weight matrix and b is the vector of bias. Likewise, in the decoding step (decoder), trained according to Eq. (2b), the weights are calibrated so that the output data are as similar as possible to the original data.

For anomaly detection, the autoencoder network has been used in anomaly detection (Chen et al. 2017; Said Elsayed et al. 2020; Ranjan 2020). One of the first studies that involved the autoencoder for anomaly detection was proposed by Hawkins et al. (2002). The basic idea of this model is that anomalies will be more difficult to reconstruct than normal conditions and, consequently, they will present greater reconstruction errors when submitted to the neural network.

The goal is to train the output to reconstruct the input as closely as possible. Nodes in the middle layers are smaller

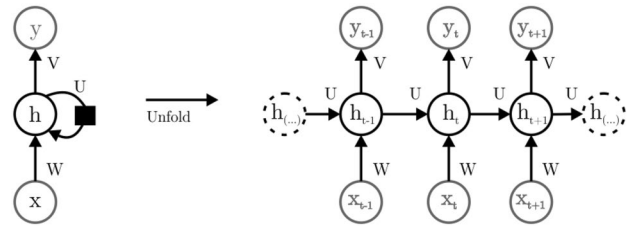


Fig. 12 Basic structure of a recurrent neural network (RNN) - adapted from Goodfellow et al. (2016)

in number and so the only way to reconstruct the input is to learn weights so that the intermediate outputs of the nodes in the middle layers are scaled-down representations of the input data.

Long short-term memory (LSTM)

When working with sequential data, previous observations can have an effect on future observations. The recurrent neural network (RNN) class, initially developed by Rumelhart et al. (1986), allows the use of this sequential memory to classify temporal patterns, as the output data are fed back into its inputs. The neural network is called feedforward when each layer receives input only from the previous layer and provides input only to the subsequent layer. When neural networks are extended to include feedback connections, neural networks are called recurrent. The basic idea of this feedback is to allow the sharing of parameters (Goodfellow et al. 2016).

Figure 12 illustrates the basic structure of a recurrent neural network, highlighting the node h in which the feedback occurs and its unfolded representation. Equations (3) and (4) represent the expressions for a sequence of values x, where the variable h represents the state of the hidden units of the network, b and c are the vectors of bias, and W, V and U are the input weight matrices for the hidden layer, hidden layer for output and hidden layer for the hidden layer, respectively. Different types of activation functions are possible for the hidden and output layers, and the functions tanh and softmax were represented in the equation for the hidden and output layers, respectively.

$$h_t = \tanh(Wx_t + Uh_{t-1} + b) \tag{3}$$

$$y_t = \text{softmax}(Vh_t + c) \tag{4}$$

RNNs are a powerful class of computational models capable of learning arbitrary dynamics, as they are able to maintain this memory of patterns in sequential orders, but their main limitation is the inability to learn long-term memories due to the vanishing gradient problem. The gradient is used

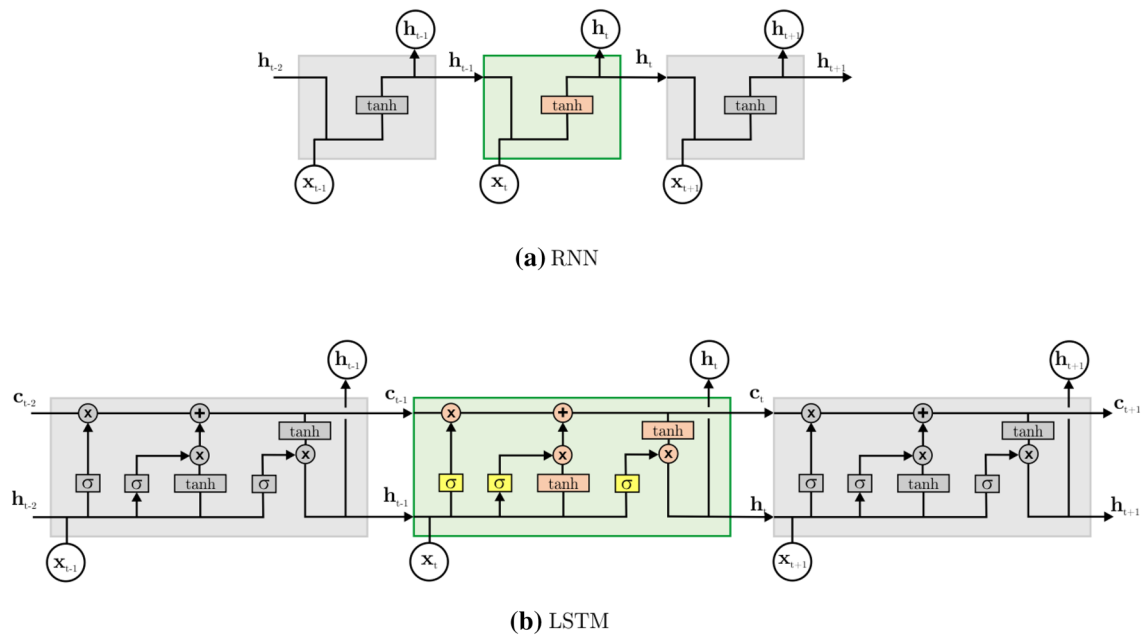


Fig. 13 Flow of information in the internal structures of the RNN (Recurrent Neural Network) and LSTM (Long Short-Term Memory) networks - adapted from Ranjan (2020)

update the values of the learned model weights. However, if the gradient is too small (vanishing gradient), the model does not learn efficiently. This problem occurs in RNN networks since backpropagation operations are performed on the recurring entries and units h . To minimize this problem, the Long short-term memory (LSTM) network was developed, originally described in the work of Hochreiter and Schmidhuber (1997), which introduced the concept of cell states that store long-term and short-term memories. Fig. 13 illustrates the flow of information along with the RNN and LSTM networks.

LSTM is a special type of recurrent neural network (RNN) that is stable, and powerful enough to be able to model long-range time dependencies and overcome the vanishing gradient problem. Its additional state cell allows gradients to flow for long periods of time without going to zero (no vanishing gradient effect). A mechanism of ports is used to regulate the flow of LSTM information, as illustrated in Fig. 14. The cell consists of three gates, input (i), output (o), and forgetting (f), with sigmoid activation (σ) shown in yellow boxes. The cell obtains relevant information through the activation functions \tanh shown in orange boxes. The cell takes the previous states (c_{t-1}, h_{t-1}), runs them through the ports and extracts information to produce the updated state (c_t, h_t) (Ranjan 2020). The mathematical expressions involved are described in Eq. (5).

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (5a)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (5b)$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (5c)$$

$$c_t = f_t c_{t-1} + i_t \tilde{c}_t \quad (5d)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (5e)$$

$$h_t = o_t \tanh(c_t) \quad (5f)$$

where: i_t , o_t , and f_t are input, output, and forgetting ports; σ is the sigmoid function; \tilde{c}_t is a temporary variable which contains the relevant information in the current timestep t ; c_t and h_t are the cell state and outputs; and $W_{i,o,f,c}$, $U_{i,o,f,c}$ and $b_{i,o,f,c}$ are the parameters of weights and bias of input (i), output (o), forget (f) and cell memory (c) ports, respectively.

Initially, the forgetting gate (forget gate), calculated according to Eq. (5a) and represented in Fig. 14a, determines which data will be discarded and which will be kept in the state of the cell. Through a sigmoid activation function σ the forget gate keeps a fraction of the information from previous states (Greff et al. 2016). At the same time, as illustrated in Fig. 14b, the decision of which data will be stored in the cell state is made in two steps: by the input gate (input gate), which is responsible for obtaining data from the current timestep and the previous state, and by determining the value (\tilde{c}_t), which evaluates the values

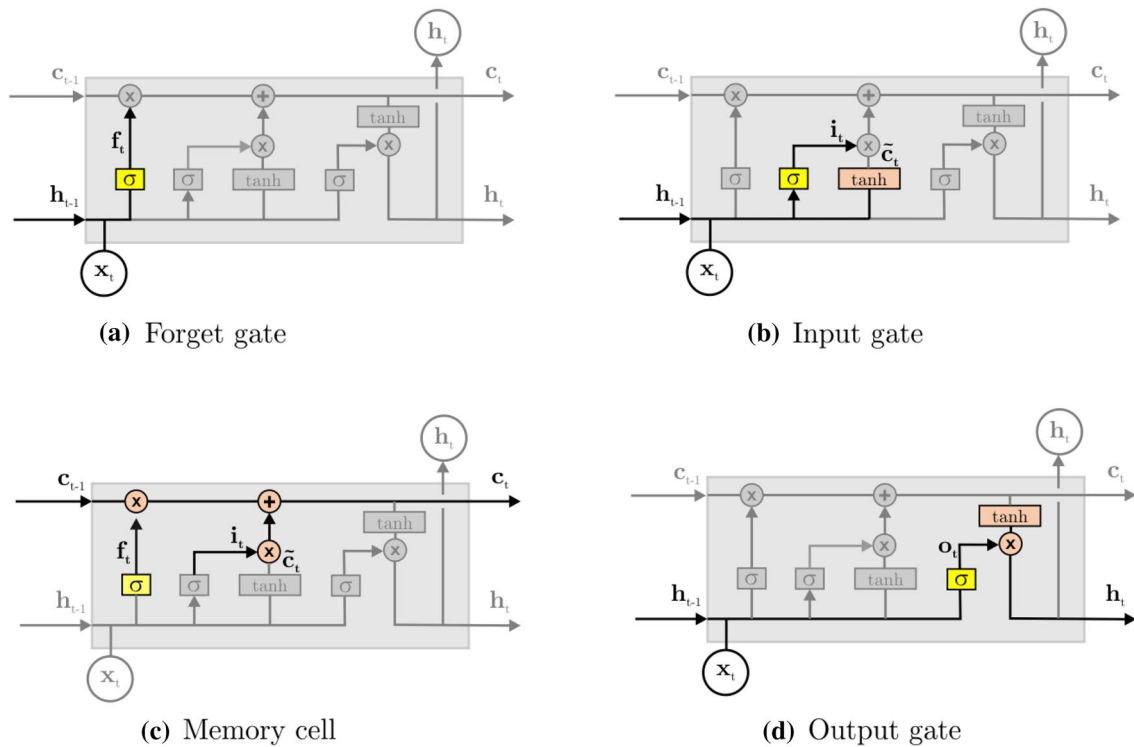


Fig. 14 Details of the parts present in the internal structures of the LSTM (Long Short-Term Memory) cell - adapted from Ranjan (2020)

to be added to the state (candidates for state cells). The expressions are represented in Eqs. (5b and c).

Then, the memory cell is updated (current state of the cell), illustrated in Fig. 14c. This calculation is based on the state of the previous cell, on the forget gate, on the input gate, and on the candidate state cell (\tilde{c}_t), as per Eq. (5d). Finally, the cell output o_t (output gate) and the new hidden state h_t are calculated, according to Eqs. (5e and f) and as illustrated in Fig. 14d.

The algorithms implemented in the Tensorflow library developed by Abadi et al. (2015) allow the performance of experiments with several variations of parameters in neural networks, such as type, number, and size of layers; activation functions, weight matrix initialization parameters, learning rate, among others.

Performance evaluation

In the final step of the classification process, it is necessary to evaluate the performance of the model. According to Kowsari et al. (2019), this evaluation is usually done through metrics such as revocation, precision, or through the measure-F1 indicator. These metrics are obtained from a confusion matrix, in which there is a true positive (TP), false positive (FP), true negative (TN), and false negative (FN) values for each class.

The recall (R), represented in Eq. (6), is defined as the percentage of samples properly classified among those belonging to a certain class, that is when it really belongs to the K class, how often the classifier hits this class. Precision (P) represents, among those classified as correct, how many were actually correct, and it is represented in Eq. (7). Finally, the F1 score or the F1 measure (Kadhim 2019) represents the harmonic mean between the revocation and the precision.

$$R = \frac{TP}{TP + FN} \tag{6}$$

$$P = \frac{TP}{TP + FP} \tag{7}$$

$$F1 = \frac{2 \times P \times R}{P + R} \tag{8}$$

In this work, the performance analysis is based on the F1 measure. The proposition of the F1 score is a more realistic metric than accuracy to avoid misleading conclusions about the performance of classifiers in an unbalanced dataset.

Statistical tests verify if multiple classifiers can generate performance metrics whose averages are equal to each other. The Friedman test is a statistical test to detect differences between various experiments. The Wilcoxon test is a

hypothesis test used when you want to compare two related samples to assess whether the means differ from each other and makes no assumption about the underlying distribution of the error difference (Demšar 2006). The verification of which classifiers can generate F1 metrics whose means are equal to the mean obtained by the other classifiers was carried out with the Wilcoxon Test proposed by Wilcoxon (1992) and the Bonferroni correction mentioned in Dunn (1961). The tests failed to reject the null hypothesis at the 5% level.

Results and discussion

This section presents the results of the experiments carried out according to the benchmark for anomaly detection proposed by Vargas et al. (2019). These experiments were performed at the instance level in two ways: with and without the feature extraction step.

Experiments were carried out following the next rules established in the benchmark for anomaly detection proposed by Vargas et al. (2019):

- Only real instances with anomalies of types that have normal periods (1 - BSW Abrupt Increase, 2 - DHSV Spurious Closure, 5 - Fast Productivity Loss, 6 - Fast Restriction in CKP, 7 - Incrustation in CKP, and 8 - Hydrate in Production Line) greater than or equal to twenty minutes were used;
- Multiple training and testing rounds are performed, the number of rounds being equal to the number of instances. In each round, the samples for training or testing are extracted from only one instance. Part of the normality samples is used in training and the other part in the test. All abnormality samples are used in the test only (single-class learning technique). The test set consists of the same number of samples from each class (normality and abnormality);
- In each round, precision, recall, and F1 measure are computed (mean value and standard deviation of each metric), with the mean value of the F1 measure presented in this section for comparison with the previous work of Vargas et al. (2019).

The number of runs of each algorithm for tuning was: Elliptical envelope (36), Isolation Forest (384), LOF (64), OCSVM (216), Autoencoder (8) and LSTM (8).

Experiments with feature extraction

The experiments in which the feature extraction step was performed as described in Section 2. From each time series sample, the median, mean, standard deviation, variance,

maximum, and minimum for each variable were extracted and used as characteristics.

Prior to applying the algorithms with the benchmark rules in real instances, the classifiers were calibrated (implemented by the scikit-learn **ParameterGrid** function) in 426 simulated instances through runs in different combinations between classifiers and hyperparameters. Table 3 presents for each classifier (row), the values of hyperparameter combinations (third column, with best parameters highlighted in bold) and the values of F1 measures. The fourth column is the best average of F1 and standard deviation (in parentheses) by the algorithm (with best value highlighted in bold), for simulated instances (calibration). The last column corresponds to the choice of hyperparameters that produced the best F1 mean and standard deviation (with best value highlighted in bold) using real instances (test).

The classifier that obtained the best F1 measure in tests on real instances was the LOF with an F1 measure of 0.870, followed by the Isolation Forest with an F1 of 0.701. Non-parametric statistical tests are used considering a significance of 5%. The result of the Friedman Test is low, value $p = 2,432 \times 10^{-20}$, then the null hypothesis is rejected. The null hypothesis for the Friedman test is that there are no differences between the variables and it can be concluded that at least 2 of the F1 metrics are significantly different from each other.

The results of Wilcoxon's test obtained are presented in Table 4. The paired tests in the table show that the null hypothesis can be rejected for the LOF classifier in all comparisons.

The results obtained for the Isolation Forest and OCSVM classifiers did not present statistically significant improvements in relation to the results obtained by Vargas et al. (2019), as expected. The F1 measure was 0.727 for Isolation Forest, and it was 0.470 for OCSVM (sigmoid kernel), respectively. The classifiers Local Outlier Factor, Elliptical Envelope, and Autoencoder were not used in the reference work. Therefore, depending on the results presented in Table 4, it can be concluded that classifiers based on the Local Outlier Factor generate, with high probability, F1 metrics whose means are different and higher than in relation to the averages of F1 metrics obtained by the other classifiers.

Experiments without feature extraction

In order to allow comparisons in experimentation with neural networks autoencoder with LSTM layers, which demand time series as input, a simulation round of all classifiers was performed without performing the feature extraction step. A window size of 3 min (with 180 observations each) was used. The maximum value of the mean average error obtained in training was used as a threshold in the test. The same combinations of the previous item were simulated, and

Table 3 Means of the F1 measure and standard deviation (in parentheses) of the feature extraction experiment, by an algorithm (with best parameters highlighted in bold), for the best case in simulated and real instances

Classifier	Hyperparameter combinations	Values	Average of F1 and standard deviation in simulated instances	Average of F1 and standard deviation in real instances
OCSVM	Kernel	['linear', 'rbf', 'poly', ' sigmoid ']	0.572 (0.208)	0.477 (0.221)
	nu ν	[0.0001, 0.001 , 0.01, 0.1, 0.5, 1]		
	Gamma γ	['auto', 'scale', 0.0001, 0.001, 0.01 , 0.1, 0.5, 1, 5, 10]		
OCSVM - benchmark Vargas et al (2019)	No tuning	–	–	0.470 (0.201)
Isolation forest	# estimators	[50, 100, 150, 200]	0.777 (0.187)	0.701 (0.176)
	Maximum samples per tree	['auto', 0.5, 0.75 , 1.0]		
	# features per tree	[0.5, 0.75, 1.0]		
	Bootstrap	[True, False]		
	Contamination	['auto', 0, 0.05, 0.1]		
Isolation forest - benchmark Vargas et al (2019)	No tuning	–	–	0.727 (0.182)
LOF	# neighbors	[5, 10, 15, 20]	0.915 (0.035)	0.870 (0.14)
	Metric	[hamming, euclidean , manhattan, minkowski]		
	Contamination	[' auto ', 0.01, 0.05, 0.10]		
Elliptical envelope	Assume centered	[True, False]	0.577 (0.132)	0.586 (0.141)
	Support fraction	[0.95, 0.975, 0.99]		
	Contamination	[0.0001, 0.001, 0.01 , 0.05, 0.1, 0.5]		
Autoencoder	Batch size	[4, 8]	0.721 (0.184)	0.590 (0.169)
	Dropout	[0.0 , 0.2]		
	Hidden neurons	[[8,4,4,8], [4,2,2,4]]		

Table 4 p values of Wilcoxon's statistical tests with Bonferroni adjustment of the experiments with feature extraction on real instances

Local Outlier Factor	1,000000								
Floresta de Isolamento (benchmark Vargas)	0,006755	1,000000							
Floresta de Isolamento	0,000306	1,000000	1,000000						
Autoencoder (feedforward)	0,000006	0,007872	0,065352	1,000000					
Envelope Elíptico	0,000020	0,010399	0,056257	1,000000	1,000000				
Dummy	0,000005	0,000127	0,000218	0,112747	0,039093	1,000000			
One-Class SVM (benchmark Vargas)	0,000018	0,006883	0,010718	1,000000	1,000000	1,000000	1,000000		
One-Class SVM	0,000024	0,010787	0,007326	1,000000	1,000000	1,000000	1,000000	1,000000	

Table 5 Means of the F1 measure and standard deviation (in parentheses) of the experiment without feature extraction, by the algorithm (with best parameters highlighted in bold), for the best case in simulated and real instances

Classifier	Hyperparameter combinations	Values	Average of F1 and standard deviation in simulated instances	Average of F1 and standard deviation in real instances
OCSVM	Kernel	['linear', 'rbf', 'poly', ' sigmoid ']	0.569 (0.178)	0.551 (0.191)
	nu ν	[0.0001 , 0.001, 0.01, 0.1, 0.5, 1]		
Isolation forest	Gamma γ	['auto', 'scale', 0.0001, 0.001, 0.01 , 0.1, 0.5, 1, 5, 10]	0.68 (0.186)	0.616 (0.183)
	# estimators	[50 , 100, 150, 200]		
	Maximum samples per tree	['auto', 0.5, 0.75 , 1.0]		
	Features per tree	[0.5, 0.75 , 1.0]		
	Bootstrap	[True , False]		
LOF	Contamination	['auto', 0, 0.05, 0.1]	0.920 (0.027)	0.859 (0.129)
	# neighbors	[5, 10, 15, 20]		
	Metric	[hamming, euclidean, manhattan , minkowski]		
Elliptical envelope	Contamination	[' auto ', 0.01, 0.05, 0.10]	0.665 (0.160)	0.65 (0.145)
	Assume centered	[True , False]		
	Support fraction	[0.95, 0.975, 0.99]		
Autoencoder	Contamination	[0.0001 , 0.001, 0.01, 0.05, 0.1, 0.5]	0.751 (0.192)	0.579 (0.169)
	Batch size	[4, 8]		
	Dropout	[0.0 , 0.2]		
	Hidden neurons	[[16,8,8,16], [8,4,4,8]]		
LSTM	Batch_size	[4, 8]	0.668 (0.153)	0.627 (0.179)
	Dropout	[0.0 , 0.2]		
	Hidden_neurosn	[[16,8,8,16], [8,4,4,8]]		

a new classifier based on neural networks was also included. Table 5 shows the results obtained from the averages of the F1 measure and standard deviation (in parentheses) for each algorithm (with best parameters highlighted in bold), for the best case in the simulated instances (calibration) and in the instances real (test).

The classifier that obtained the best F1 measure in tests on real instances was the LOF with an F1 measure of 0.859, followed by the Elliptical Envelope with an F1 of 0.650 and the Autoencoder (LSTM) with an F1 of 0.627.

The result of the Friedman Test is low, value $p = 3.812 \times 10^{-19}$, so the null hypothesis is rejected. The results of the paired tests of Wilcoxon's test are presented in Table 5. The data in the table show that the null hypothesis can be rejected for the LOF classifier in all comparisons. The results obtained for the Elliptical Envelope, Autoencoder (LSTM), and Isolation Forest classifiers were only significantly superior to the naive classifier (Dummy). The Autoencoder (feedforward) and OCSVM classifiers were not statistically better than the naive classifier.

Based on the results presented in Table 6, it can be concluded that classifiers based on the Local Outlier Factor

generate, with high probability, F1 metrics whose means are different and higher in relation to the means of F1 metrics obtained by the other classifiers.

Discussion

In this work, the classifier that obtained the best F1 measure in tests on real instances was the LOF with an F1 measure of 0.870 with feature extraction. Even without the feature extraction step, the LOF classifier achieved the best F1 score of 0.859. The null hypothesis can be rejected for the LOF classifier in all comparisons in Wilcoxon's test.

A conjecture about the fact that the LOF has presented better results is that the definition of boundaries of normal cases in a single class, as in OCSVM and in the Elliptical Envelope, that presented the lowest values, is not well defined. Thus, even normal cases are better represented by several clusters, and that is why the F1 measure was higher for the Isolation Forest and for the LOF.

Table 7 shows the results of related works. The result of this work is better than the benchmark for anomaly detection provided by Vargas et al. Vargas et al. (2019) of F1 measure

Table 6 *p* values of Wilcoxon statistical tests with Bonferroni adjustment of experiments without feature extraction on real instances

<i>Local Outlier Factor</i>	1,000000						
<i>Envelope Eliptico</i>	0.000016	1,000000					
<i>Autoencoder (LSTM)</i>	0.000007	1,000000	1,000000				
<i>Floresta de Isolamento</i>	0.000013	1,000000	1,000000	1,000000			
<i>Autoencoder (feedforward)</i>	0.000007	0.629214	0.734375	1,000000	1,000000		
<i>One-Class SVM</i>	0.000010	0.296349	1,000000	1,000000	1,000000	1,000000	
<i>Dummy</i>	0.000003	0.000480	0.018093	0.007500	0.100479	1,000000	1,000000

Table 7 Results of related work

Author	Method	F1-score
Vargas et al. (2019)	Isolation forest	0.72
Turan and Jäschke (2021)	Decision tree	0.85
This work	LOF	0.87

of 0,727 with Isolation Forest. And also better than the F1 score of 0.85 from the work of Turan and Jäschke Turan and Jäschke (2021).

Although the quantitative results were better, the previous works dealt with the multi-class problem while this work deals with the two-class problem. In a monitoring system, an alert should be issued when a problem is detected, whatever the cause of the problem, and the two-classes classification is more compatible with this type of system.

The neural network results were not as good as the LOF. Perhaps the neural network model was not suitable for the 3W dataset or the size of the training dataset was not enough or the training method had to be different as proposed by Ergen and Kozat (2019).

Conclusions

The present work applied and quantitatively compared anomaly detection techniques in emerging oil-producing offshore wells for fault detection. The following single-class classifier techniques were compared: Isolation Forest, One-class Support Vector Machine (OCSVM), Local Outlier Factor (LOF), Elliptical Envelope, and Autoencoder with feedforward and LSTM (Long short-term memory) layers. The use of techniques based on multilayer neural networks

(also called deep learning or deep learning) for time series classification is considered an interesting but challenging solution in the data mining area (Fawaz et al. 2019). The experiments have used the 3W public database composed of multivariate time series and the benchmark for anomaly detection developed by Vargas et al. (2019). Experiments were carried out to detect anomalies with the application of algorithms to the database, and the measurement metric F1 was determined.

- The comparative analysis of anomaly detection techniques in emerging offshore oil-producing wells for fault detection revealed that the Local Outlier Factor (LOF) classifier outperformed other single-class classifiers in all scenarios, including real-world and simulated scenarios. Notably, LOF’s performance remains consistent even when considering scenarios with and without feature selection.
- The statistical tests conducted at the 5% significance level provide validation for the LOF results and ensure their generalizability to a larger population.

The superior performance of the LOF classifier suggests that defining boundaries of normal cases as a single class may not be well defined, and normal cases are better represented by multiple clusters.

In future work, we intend to test other deep neural networks; use other techniques in the feature extraction step; augment the database with instances of other types of wells; and implement the classifiers in Petrobras’ well monitoring system.

Acknowledgements The authors thank IFES, and the financial support from FAPES (Fundação de Amparo à Pesquisa e Inovação do Espírito

Santo — Brazil) and CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brazil) of the PDPG (Graduate Development Program, Strategic Partnerships in the States) – SIAFEM no 2021-2S6CD, no FAPES 132/2021. Professor Komati thanks CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) for the DT-2 Grant (no 308432/2020-7), 407742/2022-0 project and FAPES for the Research Grant (no 293/2021, SIAFEM no 2021-XLSXX) and no 1023/2022 P:2022-8TZV6 project.

Author contributions All authors contributed to the study's conception, design, and analysis. Material preparation, data pre-processing, and coding were performed by WF Jr. The first draft of the manuscript was written by WF Jr, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript. On behalf of all the co-authors, the corresponding author states that there is no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Abadi M, Agarwal A, Barham P, et al (2015) TensorFlow: large-scale machine learning on heterogeneous systems. <https://www.tensorflow.org/>, software available from tensorflow.org
- Alrifaiy M, Lim WH, Ang CK (2021) A novel deep learning framework based RNN-SAE for fault detection of electrical gas generator. *IEEE Access* 9(21):433–442. <https://doi.org/10.1109/ACCESS.2021.3055427>
- ANP (2020) Boletim mensal da produção de petróleo e gás natural. <http://www.anp.gov.br/>, Accessed 19 Sept 2022
- Barbariol T, Feltresi E, Susto GA (2019) Machine learning approaches for anomaly detection in multiphase flow meters. *IFAC-PapersOnLine* 52(11):212–217. <https://doi.org/10.1016/j.ifacol.2019.09.143>
- Breunig MM, Kriegel HP, Ng RT, et al (2000) LOF: identifying density-based local outliers. In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp 93–104. <https://doi.org/10.1145/342009.335388>
- Castro AODS, Santos MDJR, Leta FR et al (2021) Unsupervised methods to classify real data from offshore wells. *Am J Op Res* 11(5):227–241. <https://doi.org/10.4236/ajor.2021.115014>
- Chan CF, Chow KP, Mak C, et al (2019) Detecting anomalies in programmable logic controllers using unsupervised machine learning. In: Peterson G, Shenoi S (eds) *Advances in Digital Forensics XV*. Digital Forensics 2019. IFIP Advances in Information and Communication Technology, Springer, vol 569. Springer International Publishing, pp 119–130. https://doi.org/10.1007/978-3-030-28752-8_7
- Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: A survey. *ACM Comput Surv* 41(3):1–58. <https://doi.org/10.1145/1541880.1541882>
- Chen J, Sathe S, Aggarwal C, et al (2017) Outlier detection with autoencoder ensembles. In: *Proceedings of the 2017 SIAM international conference on data mining*, SIAM, pp 90–98. <https://doi.org/10.1137/1.9781611974973.11>
- Chen WR, Yun YH, Wen M et al (2016) Representative subset selection and outlier detection via isolation forest. *Anal Methods* 8(39):7225–7231. <https://doi.org/10.1039/C6AY01574C>
- Christ M, Braun N, Neuffer J et al (2018) Time series feature extraction on basis of scalable hypothesis tests (tsfresh-a python package). *Neurocomputing* 307:72–77. <https://doi.org/10.1016/j.neucom.2018.03.067>
- D'Almeida AL, Bergiante NCR, de Souza Ferreira G et al (2022) Digital transformation: a review on artificial intelligence techniques in drilling and production applications. *Int J Adv Manuf Technol* 119(9):5553–5582. <https://doi.org/10.1007/s00170-021-08631-w>
- Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30
- Dunn OJ (1961) Multiple comparisons among means. *J Am Stat Assoc* 56(293):52–64. <https://doi.org/10.1080/01621459.1961.10482090>
- Ergen T, Kozat SS (2019) Unsupervised anomaly detection with LSTM neural networks. *IEEE Trans Neural Netw Learn Syst* 31(8):3127–3141. <https://doi.org/10.1109/TNNLS.2019.2935975>
- Fawaz HI, Forestier G, Weber J et al (2019) Deep learning for time series classification: a review. *Data Min Knowl Disc* 33(4):917–963. <https://doi.org/10.1007/s10618-019-00619-1>
- Géron A (2019) *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc
- Gerwick BC Jr (2007) *Construction of marine and offshore structures*. CRC Press, New York. <https://doi.org/10.1201/9780849330520>
- Goodfellow I, Bengio Y, Courville A (2016) *Deep learning*. MIT press, Cambridge
- Grashorn P, Hansen J, Rummens M (2020) How airbus detects anomalies in iss telemetry data using tfx. <https://blog.tensorflow.org/2020/04/how-airbus-detects-anomalies-iss-telemetry-data-tfx.html>, accessed 19 September 2022
- Greff K, Srivastava RK, Koutnk J et al (2016) LSTM: a search space odyssey. *IEEE Trans Neural Netw Learn Syst* 28(10):2222–2232. <https://doi.org/10.1109/TNNLS.2016.2582924>
- Guo B, Lyons WC, Ghalambor A (2007) *Petroleum production engineering: a computer-assisted approach*. Gulf Professional Pub
- Hardin J, Rocke DM (2004) Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Comput Stat Data Anal* 44(4):625–638. [https://doi.org/10.1016/S0167-9473\(02\)00280-3](https://doi.org/10.1016/S0167-9473(02)00280-3)
- Hawkins S, He H, Williams G, et al (2002) Outlier detection using replicator neural networks. In: Kambayashi Y, Winiwarer W, Ari-kawa M (eds) *International conference on data warehousing and knowledge discovery*, Springer. Springer Berlin Heidelberg, pp 170–180. https://doi.org/10.1007/3-540-46145-0_17
- Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput* 9(8):1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Hubert M, Debruyne M (2010) Minimum covariance determinant. *Wiley Interdiscip Rev: Comput Stat* 2(1):36–43. <https://doi.org/10.1002/wics.61>
- Kadhim AI (2019) Survey on supervised machine learning techniques for automatic text classification. *Artif Intell Rev* 52(1):273–292. <https://doi.org/10.1007/s10462-018-09677-1>
- Khan S, Liew CF, Yairi T et al (2019) Unsupervised anomaly detection in unmanned aerial vehicles. *Appl Soft Comput* 83(105):650. <https://doi.org/10.1016/j.asoc.2019.105650>
- Khan SS, Madden MG (2014) One-class classification: taxonomy of study and review of techniques. *Knowl Eng Rev* 29(3):345–374. <https://doi.org/10.1017/S026988891300043X>
- Kowsari K, Jafari Meimandi K, Heidarysafa M et al (2019) Text classification algorithms: a survey. *Information* 10(4):150. <https://doi.org/10.3390/info10040150>

- Kwon D, Kim H, Kim J et al (2019) A survey of deep learning-based network anomaly detection. *Clust Comput* 22(1):949–961. <https://doi.org/10.1007/s10586-017-1117-8>
- Li ZC, Fan CL (2020) A novel method to identify the flow pattern of oil-water two-phase flow. *J Pet Explor Prod Technol* 10(8):3723–3732. <https://doi.org/10.1007/s13202-020-00987-1>
- Liu FT, Ting KM, Zhou ZH (2012) Isolation-based anomaly detection. *ACM Trans Knowl Discov Data (TKDD)* 6(1):1–39. <https://doi.org/10.1145/2133360.2133363>
- MacroTrends (2022) Brent crude oil prices - 10 year daily chart. <https://www.macrotrends.net/2480/brent-crude-oil-prices-10-year-daily-chart>, Accessed 19 Sept 2022
- Marins MA, Barros BD, Santos IH et al (2021) Fault detection and classification in oil wells and production/service lines using random forest. *J Petrol Sci Eng* 197(107):879. <https://doi.org/10.1016/j.petrol.2020.107879>
- Misra S, Osogba O, Powers M (2020) Chapter 1 - unsupervised outlier detection techniques for well logs and geophysical data. In: Misra S, Li H, He J (eds) *Machine learning for subsurface characterization*. Gulf Professional Publishing, p 1–37. <https://doi.org/10.1016/B978-0-12-817736-5.00001-6>, <https://www.sciencedirect.com/science/article/pii/B9780128177365000016>
- Pal A, Prakash P (2017) *Practical Time series analysis: master time series data processing, visualization, and modeling using python*. Packt Publishing, <https://books.google.com.br/books?id=mY3HwgEACAAJ>
- Pedregosa F, Varoquaux G, Gramfort A et al (2011) Scikit-learn: machine learning in python. *J Mach Learn Res* 12:2825–2830
- Pedregosa F, et al (2011b) Novelty and outlier detection. https://scikit-learn.org/stable/modules/outlier_detection.html, Accessed 19 Sept 2022
- Ranjan C (2020) *Understanding deep learning: application in rare event prediction*. Connaissance Publishing, <https://doi.org/10.13140/RG.2.2.34297.49765>
- Rousseeuw PJ, Driessen KV (1999) A fast algorithm for the minimum covariance determinant estimator. *Technometrics* 41(3):212–223. <https://doi.org/10.1080/00401706.1999.10485670>
- Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. *Nature* 323(6088):533–536. <https://doi.org/10.1038/323533a0>
- Sagheer A, Kotb M (2019) Time series forecasting of petroleum production using deep lstm recurrent networks. *Neurocomputing* 323:203–213. <https://doi.org/10.1016/j.neucom.2018.09.082>, <https://www.sciencedirect.com/science/article/pii/S0925231218311639>
- Said Elsayed M, Le-Khac NA, Dev S, et al (2020) Network anomaly detection using LSTM based autoencoder. In: *Proceedings of the 16th ACM symposium on qos and security for wireless and mobile networks, Q2SWinet '20*, pp 37–45. <https://doi.org/10.1145/3416013.3426457>
- Santos T, Kern R (2016) A literature survey of early time series classification and deep learning. In: *Sami40 workshop at i-KNOW'16*
- Schölkopf B, Platt JC, Shawe-Taylor J et al (2001) Estimating the support of a high-dimensional distribution. *Neural Comput* 13(7):1443–1471. <https://doi.org/10.1162/089976601750264965>
- Soltanmohammadi R, Iraj S, De Almeida TR et al (2021) Insights into multi-phase flow pattern characteristics and petrophysical properties in heterogeneous porous media 2021(1):1–5. <https://doi.org/10.3997/2214-4609.202183016>, <https://www.earthdoc.org/content/papers/10.3997/2214-4609.202183016>
- Soriano-Vargas A, Werneck R, Moura R et al (2021) A visual analytics approach to anomaly detection in hydrocarbon reservoir time series data. *J Petrol Sci Eng* 206(108):988. <https://doi.org/10.1016/j.petrol.2021.108988>
- Takbiri-Borujeni A, Fathi E, Sun T et al (2019) Drilling performance monitoring and optimization: a data-driven approach. *J Pet Explor Prod Technol* 9(4):2747–2756. <https://doi.org/10.1007/s13202-019-0657-2>
- Tan Y, Tian H, Jiang R et al (2020) A comparative investigation of data-driven approaches based on one-class classifiers for condition monitoring of marine machinery system. *Ocean Eng* 201(107):174. <https://doi.org/10.1016/j.oceaneng.2020.107174>
- Tariq Z, Aljawad MS, Hasan A et al (2021) A systematic review of data science and machine learning applications to the oil and gas industry. *J Pet Explor Prod Technol* 11(12):4339–4374. <https://doi.org/10.1007/s13202-021-01302-2>
- Turan EM, Jäschke J (2021) Classification of undesirable events in oil well operation. In: *2021 23rd international conference on process control (PC)*, IEEE, pp 157–162. <https://doi.org/10.1109/PC52310.2021.9447527>
- Vargas REV, Munaro CJ, Ciarelli PM et al (2019) A realistic and public dataset with rare undesirable real events in oil wells. *J Petrol Sci Eng* 181(106):223. <https://doi.org/10.1016/j.petrol.2019.106223>
- Wilcoxon F (1992) *Individual comparisons by ranking methods*. Springer, New York, pp 196–202. https://doi.org/10.1007/978-1-4612-4380-9_16

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.