Method for Treating Anomalies in Multivariate Time Series

Thiago Moeda[1], Mariza Ferro[1], Eduardo Ogasawara[2], Fabio Porto[1]

[1]*Laboratório Nacional de Computação Científica (LNCC)*
Ave. Getulio Vargas, 333 - Quitandinha, Petrópolis - RJ, 25651-076, Brazil
tmoeda@lncc.br, mariza@lncc.br, fporto@lncc.br
[2]*Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ)*
St. Gen. Canabarro, 485 - Maracanã, Rio de Janeiro - RJ, 20271-204, Brazil
eogasawara@ieee.org

Abstract. In recent years, the classification of time series has gained great relevance in significant sectors and segments of society. Machine Learning Techniques make it possible to interpret the behavior of anomalous phenomena in multivariate datasets. This work proposes a study of three methods from the perspective of their ability to provide relevant information for the detection, validation and prediction of anomalous events in time series data. To achieve this goal, a case study was carried out exploring algorithms based on neural networks and inductive symbolic learning applied to a real problem of detecting anomalies associated with the oil well drilling process. The main results indicate that this method can be a promising way to treat anomalies.

Keywords: Machine learning, Self-organizing map, Decision tree, Recurrent neural network, LSTM.

1 Introduction

During oil well drilling operations, the rig may become immobilized within the column and cannot be moved for different reasons. This event is a problem that can be caused by different reasons such as, for example, differential pressure, that is, when a part of the drill string is embedded in a layer of mud; problems of geological instability in drilling wells, where a portion of the well opening does not maintain its size and shape and/or its structural integrity or by excessive accumulation of gravel in the annular space caused by inadequate well cleaning [1]. Early detection of these anomalous events, called spine arrest, is extremely important as it causes an unpredictable downtime in production. To approach this problem, we propose a data flow model, through Machine Learning methods, with the objective of detecting, validating and predicting the occurrence of column arrest events. Thus, studies of data-oriented methods are carried out, such as the Self-organizing map (SOM) neural network, the Decision tree (DT) algorithm and the Long short-term memory (LSTM) recurrent neural network. The SOM algorithm is used, under the data mining aspect, to distinguish patterns and provide information about anomalies, enabling the detection of column arrest events. The DT symbolic classification method is applied in order to provide information about the quality of annotations of detected events. Finally, a LSTM neural network is implemented due to the capacity of its units to learn long sequential patterns of data behavior, making it possible to represent them ahead in time. Thus, the data flow model is a composition of these three methods implemented under the aspects of detector, classifier and predictor of anomalies. The validation of the proposed modeling was made using datasets referring to the drilling process of seven oil wells made available by a Brazilian oil and gas company. The main results indicate that the LSTM models, when applied to different datasets, present promising results in relation to the prediction of the occurrence of anomalies, corroborating the detection determined by the SOM method.

2 Background

Machine learning algorithms can be defined as computer programs that absorb a new experience as a function of some class of tasks, if their performance improves as a function of a certain performance measure [2]. SOM is a neural network, based on the functioning of neurons in the human brain's cortex, where, in practice, the learning absorbed by the model is obtained through a process of competition among its neurons, depending on each instance of given to the model. The result is a 2D topological map that visually provides the multivariate correlation of the data, through its grouping or distance, due to a certain similarity function [3]. The algorithm was implemented with the *Sompy* library [4]. DT is defined as an algorithm for the classification task. It belongs to the symbolic inductive learning paradigm, where the learning process occurs through the specific to the general, through decision rules based on attribute values, where the objective is to adjust a model that is capable of predicting, with a given accuracy, the value of another target attribute, that is, the training dataset instances are partitioned into groups of uniform classes [5]. The easy understanding of the result of the binary structure of the tree's branch enables the extraction of knowledge about the problem domain, in addition to the construction of models. In the structure of a DT, each data instance follows a unique path for its classification, depending on a set of comparison rules, of the *SE-Then* type, which are performed in each node of the tree, determining whether the result should proceed left or right, thus building the DT [6]. The algorithm used in this work is an optimized version of the *CART* algorithm available in the *Scikit-Learn* library [7], whose criterion for dividing the nodes of the tree aims to minimize the entropy [8], that is, in machine learning is a quantity that measures information clutter. The LSTM architecture is an enhancement of recurrent neural networks, which are a class of neural networks designed to analyze the behavior of data sequences over time, but have a gradient disappearance problem. As stated in Hochreiter and Schmidhuber [8], LSTM addresses the problem of gradient disappearance, incorporating functions (valves) into its state dynamics to maintain or discard information. The original LTSM formulation features three gates: *input*, *forget* and *output*. The algorithm was implemented with the *Keras* [9] in *Python* 3.8 language.

3 Related Work

Several studies used methods based on metric learning and deep learning to address the problem of detecting and classifying anomalies in time series. However, after extensive bibliographical research, no works were found that use these approaches for the oil and gas area and for the specific problem addressed in this article. For example, Tian [10] proposed using the SOM algorithm to identify the closest neighbors of the principal neurons and defined as an anomaly indicator the minimum quantization error produced by the SOM for each data window provided to the model. In recent work, Zhang [11] also used deep learning through an LSTM network to classify Parkinson's disease subtypes, which are associated with several clinical manifestations and are heterogeneous. Thus, an LSTM model was trained with the patients' medical records in order to provide integrated representations of multivariate sequences. Thus, it was possible to use it to define similarities among patients, making it possible to discern subtypes of disease progression. A framework was developed by Salles et al. [12] for event detection. In this work, the authors performed combinations of results obtained by different detection methods enabling a better understanding of the nature of the events.

4 Description of Data

Information about the different states during the drilling operation of a certain oil well is registered by sensors. Seven wells were selected, with each well corresponding to a single unsupervised dataset. For each set of data, the measures presented refer to ten numerical attributes, being *Operation Mode*, *Bit Depth*, *Weight on Hook*, *Weight on Bit*, *Standpipe Pressure*, *Hole Depth*, *Rotary Speed*, *Torque*, *Block Position* and *Fluid Flow*.

5 Pre-processing and Exploratory Analysis

The pre-processing consisted of data normalization and the interpolation of missing values through the mean value. In this first study, there was no treatment for removing noise in the data, as this operation had a negative influence on the anomaly detection process. The normalization of the input data for an interval between [0,1] was obtained through the method *MinMaxScaler* [7]. Working with data at the same scale is a good practice when it comes to distance calculation algorithms and it also speeds up the backpropagation chain calculation processing in deep neural network models. Given the peculiarity of the theme of this study, it is natural that doubts arise in relation to the behavior of attribute values, which in turn, are not self-explanatory. To address these issues, the data mining step was essential to extract some knowledge about the application domain. To assist in this step, procedures such as differentiation, grouping, statistical analysis and stratification were used. To evaluate the training of prediction models, the main machine learning metrics were used, such as mse, rmse, mae, accuracy, ROC, precision, sensitivity and f1-score. The Table 1 synthesizes the main information extracted through this approach:

Table 1. Summary of datasets and their main characteristics

Dataset	Total Instances	Total Operating Time (days)	Total drill depth (meters)	Absolute Drilling Interval (meters)	Longest downtime (hours)
A	54554	25.14	1777.37	[3073-4850]	50.70
B	47494	17.55	2045.92	[2810-4856]	273.65
C	23427	9.11	4487.69	[0350-4837]	18.87
D	60985	11.16	1914.28	[3494-5409]	26.40
E	33427	68.60	2044.27	[3088-5132]	1379.48
F	32651	4.00	716.12	[2665-3381]	1.30
G	25442	118.82	2988.03	[2761-5749]	2476.36

6 Proposed Method

Our dataflow model consists of three distinct machine learning modules: the SOM neural network for anomaly detection; a DT to validate the results of the detection stage; the LSTM deep learning neural network to generate prediction models. Thus, the model is centered on updating the dataset, in which the adjustment of anomaly annotations occurs through the detection and validation modules of the annotations. Then, once the best possible classification is obtained, the temporal prediction module is used to predict the trends of column arrest events, by learning the sequential representations of the data associated with the annotations. In each module, several hypotheses were generated in order to reach the best adjustments. More specifically, the *SOM Detection* module, has its parameter set x_1, and is intended to provide information about anomalies from the multidimensional dataset A. The *DT Validation* module, has its set of parameters x_2, and has the objective of evaluating the quality of the labeling of anomalies in annotated form. And the *LSTM prediction* module, has its parameter set x_3, and has the purpose of estimating the trends of learned behaviors and representing them through a classification interval between *0* and *1*. In detail, the process flow begins with the definition of the parameters x_1 for the *SOM Detection* module, and the parameters x_3 for the *LSTM prediction* module, depending on the multidimensional dataset A. The parameters x_2 for the *DT Validation* module remained fixed throughout the process. Next, as input to the *SOM Detection* module, the multidimensional dataset (without labels) A, a parameter vector x_1 and a vector Y for annotations of the labels with the anomalies, with Y being initialized with 0, are provided as input to the module. The A dataset provided to the SOM has been segmented into windows of size n, in which $n \in x_1$, to be processed incrementally. Thus, each subsequent map generated by the SOM uses, in addition to the information from its lot, the information provided by the previous map, in order to form the map solution set S. Afterwards, the SOM parameters, also contained in x_1, were defined. The first stage corresponds to the initialization of the map, where the dimensions of the mesh (x, y), the shape of the topology and the neighborhood function are defined. The second stage consists of training the map, and for this, it is necessary to define the number of iterations, the learning rate and the operating radius of the neighborhood function around the selected neuron. Then the solution set of generated maps S is analyzed qualitatively and the detected anomalies, $y_1, .., y_i$, with i being the total of maps contained in the solution set S, are annotated in Y, as continuous substrings labeled 1, in the dataset A, changing the state from A to $A^{'}$. The descriptive method established for the analysis of SOM maps to detect a shape anomaly consists of interpreting the amplitudes of the displacement rates of the distances between the spatial components x, y and z in relation to the sequence of maps $s_i \in S$. A shape anomaly is detected and annotated in Y if there is a discrepant amplitude of the component z, correlated to (x, y), whose displacement varies in a discrepant way in relation to the adjacent maps s_{i-1} and s_{i+1}, where (x, y) represents the topographic ordered pair in the map and the component z the topographical error. Afterwards, the multidimensional dataset (with labels) $A^{'}$ is selected and submitted for processing by the *DT Validation* module according to its parameters x_2. The parameter values were kept constant throughout the process, and consists of the parameter *criterion* which was defined as *entropy*, where the type of measure used specify the level of clutter of the dataset, and the parameter *random_state* defined with the value of *100*, which controls the randomness of the estimator, so that the attributes are always randomly permuted in each division. If the results obtained after processing the DT are not satisfactory, the process is restarted with a new parameter configuration profile $x_1^{'}$ for the *SOM Detection* module. Otherwise, the process continues with the call to the *LSTM Prediction* module with the parameters $A^{'}$ and x_3 for training an LSTM model. Afterwards, the model is tested and evaluated. If the obtained results are not satisfactory, the process is restarted with a new parameter configuration profile $x_3^{'}$ for the *LSTM prediction* module. Otherwise the model is saved and the process ends.

7 Experiments and Results

The detection module comprises the visual interpretation of the results provided by the SOM algorithm. So that the underlying anomalies could be observed in relation to the whole. Each dataset was partitioned and provided to the algorithm to be processed in incremental windows, with each window containing a range of instances. Thus, each map generated corresponds to the result of processing a window, so that each subsequent window increments 500 instances. The hyperparameters assigned to the algorithm for generating the maps were: for each window, 300 random hypotheses were generated; the size of the generated map has dimensions of 60x90 and this corresponds to the number of neurons arranged in the grid with hexagonal topology; the function responsible for the adjustment of the reference vectors of the neighborhood in relation to the neurons was the Gaussian one; the number of iterations was set to 100; the alpha parameter, which corresponds to the learning rate, was started with a value of 0.45; the radius parameter, which corresponds to the training area around the active neuron, was started with a value of 3. Of the 300 models generated with their respective maps for each window, the one with the smallest *quantization error* was selected, that is, the average of the distances between the reference neurons and their neighbors. Through the visual analysis of the maps, it was possible to detect, in a segmented way, behavior patterns such as groupings and dispersion of data over time, in the topographic space produced by the algorithm. Thus, the anomalies were noted in a binary way, with the value ***zero*** corresponding to the trivial state and the value ***one*** corresponding to the detected anomaly. In the Figure 1, you can see the results for the first module, where each dataset is represented by a well in relation to the effective depth that each drilling rig reached, as well as the anomalies detected qualitatively.

In order to validate the accuracy of the classification performed in the previous step, seeking to reduce subjectivity, a DT algorithm was applied and the result evaluated for each dataset. The annotated dataset was divided into 70% for training and 30% for testing. In this module, the configuration for the values of the model's parameters needed to be as generic as possible so that only the entropy for the information gain parameter was defined. We evaluated the labels annotated in the datasets by analyzing the test of the models fitted with such data, using the ROC curve, precision, recall and F1-score metrics. The results obtained are shown in Table 2. They characterize that regardless of the degree of generalization that the classifier has reached.

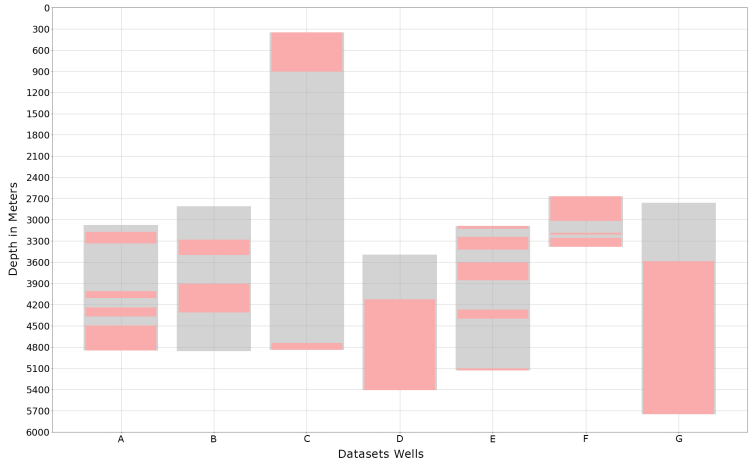

Figure 1. Detected anomalies, in red, in relation to the depth of the wells.

Table 2. Summary of test results from the datasets annotated using the DT.

Dataset	ROC	Precision	Sensitivity	F1-score
A	0.9979	1.00/1.00	1.00/1.00	1.00/1.00
B	0.9974	1.00/1.00	1.00/1.00	1.00/1.00
C	0.9270	0.98/0.99	1.00/0.87	0.99/0.93
D	0.9996	1.00/1.00	1.00/1.00	1.00/1.00
E	0.9993	1.00/1.00	1.00/1.00	1.00/1.00
F	0.9958	0.99/1.00	0.99/1.00	0.99/1.00
G	0.9975	1.00/1.00	1.00/1.00	1.00/1.00

As the last module, an LSTM neural network model was trained, for each dataset, in order to estimate the trends of known behaviors and represent them, for a given interval of instances ahead in time, through the rating between [0.1]. As in the second module, the data instances were also distributed in 70% used for training and 30% for testing. An *Embedding* layer was used in order to make the data streams continuous and dense. The network architecture was defined with a single inner layer with ten LSTM units. The *Adam* optimizer was applied, with a learning rate of 0.001, the number of iterations defined as 500 epochs and the batch with size 256. The loss function used was the *binary_crossentropy*. The value of 180 was determined for the length of the string that returns to the model. In order to reduce *overfitting*, two *Dropout* layers with values of 0.6 and 0.2 were used. Finally, an output layer of a unit was defined with the function *Sigmoid* providing the result of the prediction in an interval between [0, 1]. In Table 3 shows the results regarding the training and testing of the models. Finally, each model trained with a given dataset was validated against the other datasets. Several factors contributed to the heterogeneity of the results, such as the adjustment of initialization parameters of the models, the use of the same LSTM network topology for training all models and the visual labeling performed in the first module. The results are shown from Table 4 to Table 7.

Table 3. Summary of training results from LSTM models.

Dataset	A	B	C	D	E	F	G
Training time (s)	8392	7239	3198	9799	4661	4533	3520
Mse	0.1654	0.0851	0.1250	0.2304	0.2078	0.1706	0.0686
Rmse	0.4067	0.2918	0.3535	0.4800	0.4371	0.4131	0.2620
Mae	0.3317	0.1701	0.2503	0.4596	0.3356	0.3406	0.1368
Summary of test results for LSTM models.							
accuracy	0.7710	0.9078	0.8541	0.6284	0.7566	0.7675	0.8810
ROC	0.7350	0.8054	0.5091	0.6178	0.7331	0.7141	0.9046
Precision (0/1)	0.73/0.93	0.90/0.99	0.85/0.00	0.76/0.59	0.76/0.95	1.00/0.72	0.99/0.57
Sensitivity (0/1)	0.97/0.49	1.00/0.57	1.00/0.00	0.33/0.91	0.99/0.59	0.43/1.00	0.87/0.94
F-score (0/1)	0.83/0.64	0.94/0.73	0.92/0.00	0.46/0.72	0.76/0.72	0.60/0.84	0.93/0.71

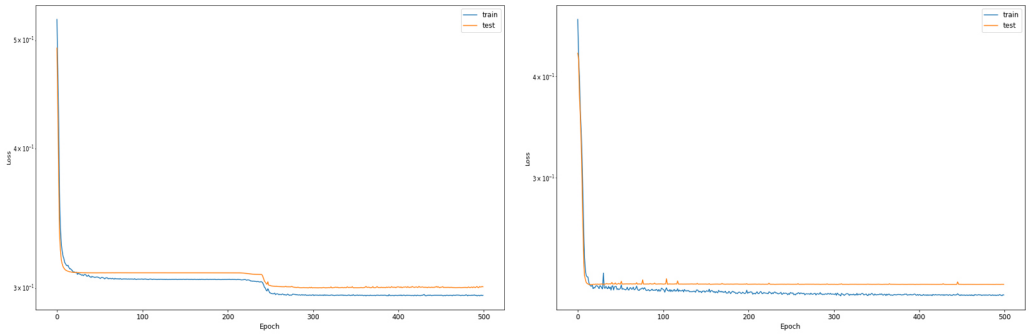

Figure 2. Error plot: mse versus epochs for the models trained with the B (left) and G (right) datasets.

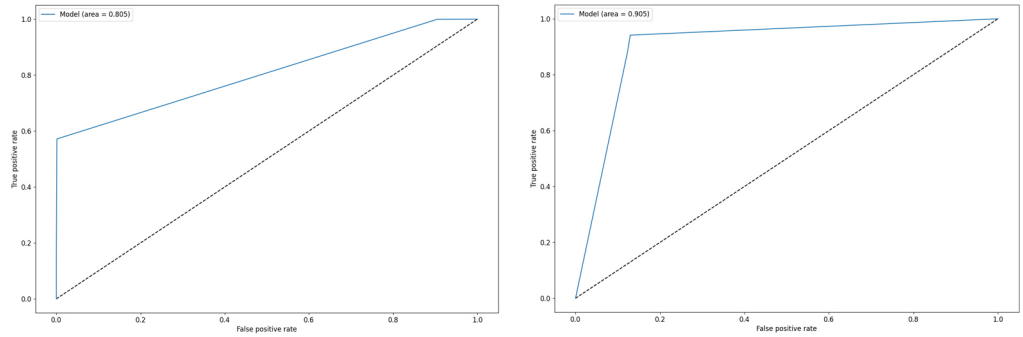

Figure 3. ROC plot of trained models with B (left) and G (right) datasets.

Table 4. Summary of LSTM model validation results. Accuracy metric:

Model/Datasets	A	B	C	D	E	F	G
m(A)	x	**0.8297**	**0.5205**	0.3767	0.4734	0.2387	**0.8822**
m(B)	**0.7537**	x	**0.8529**	0.3774	0.4732	0.2382	**0.8523**
m(C)	**0.5859**	**0.7795**	x	0.4851	0.3758	0.4129	**0.8519**
m(D)	0.2283	0.1705	0.4792	x	**0.5266**	**0.7612**	0.1181
m(E)	**0.8674**	**0.8124**	**0.5132**	0.2978	x	0.2301	**0.8597**
m(F)	0.2464	**0.9512**	0.1465	**0.6227**	**0.5271**	x	0.1483
m(G)	**0.6046**	**0.7043**	**0.5211**	0.4845	0.3755	0.4131	x

Table 5. Summary of LSTM model validation results. Precision metric:

Model/Datasets	A	B	C	D	E	F	G
m(A)	x	**0.88/0.88**	0.86/0.15	0.41/0.24	0.42/1.00	0.29/0.00	**0.99/0.56**
m(B)	**0.70/1.00**	x	0.85/0.00	0.41/0.24	0.42/1.00	0.29/0.00	0.85/0.17
m(C)	0.59/0.00	0.78/0.00	x	0.49/0.00	0.38/0.00	0.41/0.00	0.85/0.00
m(D)	0.07/0.27	0.38/0.12	0.85/0.14	x	0.00/0.58	**1.00/0.71**	0.44/0.01
m(E)	**0.83/0.88**	**0.84/0.63**	0.85/0.13	0.40/0.25	x	0.14/0.18	**0.99/0.56**
m(F)	0.00/0.30	0.01/0.11	1.00/0.15	**0.76/0.59**	0.00/0.58	x	1.00/0.15
m(G)	**0.60/0.69**	0.76/0.00	0.86/0.15	0.48/0.20	0.38/0.75	0.41/0.75	x

Table 6. Summary of LSTM model validation results. Sensitivity metric:

Model/Datasets	A	B	C	D	E	F	G
m(A)	x	**0.90/0.57**	**0.52/0.50**	0.68/0.10	1.00/0.16	0.58/0.00	**0.87/0.95**
m(B)	**1.00/0.58**	x	1.00/0.00	0.68/0.10	1.00/0.16	0.58/0.00	1.00/0.00
m(C)	1.00/0.00	1.00/0.00	x	1.00/0.00	1.00/0.00	1.00/0.00	1.00/0.00
m(D)	0.03/0.52	0.10/0.43	0.48/0.50	x	0.00/0.84	0.42/1.00	0.13/0.05
m(E)	**0.97/0.70**	**0.89/0.60**	**0.52/0.50**	0.64/0.11	x	0.50/0.12	**0.84/0.99**
m(F)	0.00/0.60	0.00/0.43	0.00/1.00	0.32/0.90	0.00/0.84	x	0.00/1.00
m(G)	**0.97/0.58**	0.90/0.00	**0.52/0.50**	1.00/0.00	1.00/0.00	1.00/0.00	x

Table 7. Summary of LSTM model validation results. F-score metric:

Model/Datasets	A	B	C	D	E	F	G
m(A)	x	**0.8271**	**0.5899**	0.3190	0.3901	0.1597	**0.9130**
m(B)	**0.7228**	x	**0.7859**	0.3196	0.3898	0.1592	**0.7846**
m(C)	0.4330	**0.6829**	x	0.3170	0.2053	0.2414	**0.7838**
m(D)	0.1694	0.1623	**0.5523**	x	0.4309	**0.7330**	0.1728
m(E)	**0.8547**	**0.8123**	**0.5643**	0.3092	x	0.2104	**0.8457**
m(F)	0.1636	0.0399	0.0378	**0.5870**	0.4316	x	**0.7334**
m(G)	**0.5355**	**0.6444**	**0.5906**	0.3166	0.2053	0.2417	x

8 Analysis of Results

In the modeling process, we sought to provide a balance between the ability to learn representations of temporal behaviors, in relation to the generalization and specialization of models. Several factors contributed to the heterogeneity of the results, such as the adjustment of initialization parameters of the models, the use of the same LSTM network architecture for training all models and, mainly, the detection of events determined in the first module. The Figure 2 and Figure 3 illustrate the results obtained for the models trained with the datasets *B* and *G*. In Figure 2 it can be seen that the LSTM model trained with the dataset *G*, by having a smaller mse, provides a better ability to learn the representations of temporal behaviors in relation to the model trained with the *B* dataset. The effectiveness of the LSTM models, too, can be evaluated by their generalizability through the analysis of the ROC curve. In Figure 3, it can be seen that the model trained with the dataset *G* may be a better base model than the model trained with the dataset *B* as it has a larger area under the ROC curve. Comparisons between forecast performances are presented from Table 4 to Table Table 7. Each model trained with a given dataset was applied to the other datasets. The first column corresponds to the models generated by from the dataset in parentheses. The best results are highlighted in **bold**. Among these results, it is noteworthy that the results **underlined**, represent the models in relation to the datasets that obtained the best results regarding cross validation. For example, as shown in Table 7, the model trained with the dataset *G*, obtained favorable results when applied to the datasets *A*, *B* and *C*. Thus, due to the complementary relationship, it can be suggested that there are greater multivariate temporal similarity relationships between the variables of these wells than the other analyzed wells.

9 Conclusions

We present a data flow model based on machine learning methods capable of predicting anomalies in time series. For this, three distinct modules were defined: the Self-organizing Map neural network was implemented as an anomaly detector; the Decision Tree algorithm for validating the labeling of anomalies; and the Long Short-term Memory deep neural network to produce prediction models. Our prediction models were able to obtain results greater than 90% when applied to datasets from other wells not used during their data assimilation process. Additionally, under the aspect of data mining, through this technology, it was possible to find evidence of similarities between datasets from different wells, revealing a promising path for the extraction of knowledge. For future work, we foresee the development of strategies to improve the modules for detecting, validating and predicting anomalies in multivariate time series datasets.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] R. Mitchell and of S. Petroleum Engineers. *Petroleum engineering handbook*, volume II, 2006.
[2] T. M. Mitchell. *Machime Learning*, volume 1. McGraw-Hill Science/Engineering/Math, New York, 1997.
[3] T. Kohonen. Essentials of the self-organizing map. *Elsevier - Neural Networks*, vol. 37, pp. 52–65, 2013.
[4] Sompy. *SOMPY: numpy based SOM (Self Organizing Map) Library*, 2021.
[5] M. Ferro, L. Huei, and C. Sandro. Intelligent data analysis: A case study of the diagnostic sperm processing. *CSITeA'02 - ACIS International Conference*, 2002.
[6] S. S. Skiena. *The Data Science Design Manual*, volume 1. Springer, New York, 2017.
[7] Scikit_Learn. *Scikit-learn: Machine Learning in Python*, 2011.
[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, pp. 52–65, 1997.
[9] Keras. *LSTM layer*, 2021.
[10] J. Tian. Anomaly detection using self-organizing maps-based k-nearest neighbor algorithm. *European Conference of the Prognostics and Health Management Society*, 2014.
[11] X. Zhang. Data-driven subtyping of parkinson disease using longitudinal clinical records: A cohort study. *Nature*, 2019.
[12] R. Salles, L. Escobar, L. Baroni, R. Zorrilla, A. Ziviani, V. Kreischer, F. C. Delicato, P. F. Pires, L. Maia, R. Coutinho, L. Assis, and E. Ogasawara. Harbinger: Um framework para integração e análise de métodos de detecção de eventos em séries temporais. *SBBD - Brazilian Symposium on Databases*, 2020.