Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 2915–2920

Flotation Process Fault Detection and Isolation using Neural ODE for generation of vector-field features

Luis Enciso-Salas * **Gustavo Pérez-Zuñiga** *
Javier Sotomayor-Moriano *

* *Engineering Department, Pontifical Catholic University of Peru, Av. Universitaria 1801, San Miguel, Lima 15088, Perú (e-mail: {lenciso, gustavo.perez, jsotom}@pucp.edu.pe).*

Abstract: Flotation in the mining industry is of vital importance for obtaining the right quality of product with efficiency and represents a critical process where possible failures must be monitored at all times. In this paper, complete fault detection and isolation system (FDI) based on the Neural Ordinary Differential Equations (NODE) framework is proposed; the NODE is employed to represent the dynamics of the studied plant based on the measured variables and inputs. Then, a classifier can be used to identify the faults based on the projections of the derivatives or local vector field generated by the NODE using the estimations and actual measurements. The proposed approach is applied to a controlled mining flotation process that has perturbations. The solution is compared with other known machine learning techniques showing better performance metrics. Moreover, it is demonstrated with t-SNE representation that features generated from the NODE model improve the classification.

Keywords: Fault diagnosis, Neural ODE, Flotation process, Deep learning

1 Introduction

In the mining industry, the recovery efficiency of the mineral is a key factor; the whole process generally consists of a series of steps such as crushing, grinding, flotation and concentration. A vital process for the resulting efficiency is the froth flotation process, where the concentrate is separated from the tailings by strategically using a series of tanks, also referred to as a flotation circuit, to maximize the mineral's recovery percentage (Xu et al., 2012).

The flotation circuit directly impacts the quality of the produced mineral, and unnecessary stoppage conditions due to failures can severely impact production and yield losses. For this reason, the design of a system for the detection and isolation of equipment failure in the flotation circuit is considered. The flotation process has been assessed in the literature mainly in terms of model-based approaches. Such as in (Jamsa-Jounela et al., 2003), where a model and control system for the levels in the cells of the flotation process is proposed to avoid the impact of inflow changes. In (Bask and Johansson, 2003), an observer-based residual generation is used to diagnose the state of clogging in the valves of the flotation process, and also in (Bask and Johansson, 2004), where the residuals are generated by the observers designed for a linearized system, and then a robust analysis is performed for the threshold definition. In (Zhao et al., 2017), a system for flooding and sinking recognition is proposed based on visual information using quantitative trend analysis. In previous work (Pérez-Zuñiga et al., 2019), a model-based approach was proposed based on distributed architecture computing diagnosis tests via structural design for a flotation process model composed of a series of tanks. Meanwhile, machine-learning methods are generally applied in the froth diagnosis based on visual imagery (Luo et al., 2021; Zhang et al., 2019a; Jemwa and Aldrich, 2006).

The use of deep learning in FDI systems is very extended, although few approaches can be employed as a standard methodology for several applications (Jia et al., 2016; Vásquez et al., 2020). Moreover, machine learning techniques are usually more frequently used in combination with other processing techniques or with the model-based approach (Zhang et al., 2019b). Hence, deep learning techniques are utilized for improved accuracy (Jia et al., 2016), as well as in complex systems where modeling is not doable or for the generation of unique features such as in vision and vibration signals. Although many architectures are available in deep learning, in this paper, we focus on the neural ordinary differential equations (NODE) proposed in (Chen et al., 2018); NODE gives the possibility of working with continuous hidden states that are generated by an ODE. Therefore, it provides the advantage of having a direct representation of the system dynamics and can consider external inputs with further development. Besides, the NODE framework is a well-defined tool that has been extended for other applications, such as latent-ODE (Rubanova et al., 2019) and stochastic differential equations (Kidger et al., 2021).

In this article, an FDI system for the flotation process is proposed; this system is based on using a NODE model that generates a continuous set of hidden states that are trained to follow the plant behavior utilizing the input and measured signals of the process. Then for the fault

10.1016/j.ifacol.2023.10.1412

detection mechanism, a comparison between both dynamics, the plant dynamics and the NODE dynamics, are used to obtain a set of projections of the derivatives; such projections are grouped by intervals or several steps to obtain a local vector field representation at each step and compare their trends; thus an efficient mechanism for detecting and identifying the fault is obtained. This work is a continuation of the work presented in (Enciso-Salas et al., 2021, 2022). Therefore, it keeps the concept of minimizing the pre-processing phase; however, it improves the application of the methodology for a complex system by adding a new way to obtain the features. This feature generation expands the idea of using the difference between measured and estimated vector fields in a local vicinity by considering an interval of projections; furthermore, an analysis of the results is performed to show the richness of the generated features for performing classification.

This article is organized as follows: Section 2 deals with the NODE presentation and the algorithm for its training considering external inputs are presented, section 3 details the model and the data-set characteristics such as available signals and faults considered, section 4 describes the training procedure for the NODE model and later for the whole FDI-NODE system, section 5 is dedicated to the evaluation and discussion of results, and finally section 6 give the conclusions of this work.

2 Neural Ordinary Differential Equations and External Inputs

NODE was presented in (Chen et al., 2018) with the concept of including an ODE solver for the continuous evolution of the internal states in a neural network $\mathbf{F}(\theta)$, where θ represent the parameters of the network.

To deal with the complexity of using ODE solvers in a back-propagation mechanism, (Chen et al., 2018) proposed the use of the Adjoint sensitivity method, thus for the network $\mathbf{F}(z(t), t, \theta)$ the following loss function is defined:

$$\begin{aligned} L(z(t_1)) &= L\left(z(t_0) + \int_{t_0}^{t_1} \mathbf{F}(z(t), t, \theta)\, dt\right) \\ &= L(ODE(z(t_0), \mathbf{F}, t_0, t_1, \theta))\,, \end{aligned} \tag{1}$$

where $z(t)$ is the continuous hidden state, t_0 and t_1 are the initial and final time respectively.

To facilitate the backward propagation, the adjoint state, $a(t) = \partial L/\partial z(t)$, is employed in the calculation of the propagation gradients, thus relation (2) can be proven (Chen et al., 2018),

$$\frac{\partial a(t)}{\partial t} = -a(t)^\top \frac{\partial \mathbf{F}(z(t), t, \theta)}{\partial t}. \tag{2}$$

Also, computation of the gradient change with respect to the parameters requires solving relation (3) backward in time,

$$\frac{\partial L}{\partial \theta} = -\int_{t_1}^{t_0} a(t)^\top \frac{\partial \mathbf{F}(z(t), t, \theta)}{\partial t}. \tag{3}$$

Equations (1)–(3) are implemented as augmented dynamics that permit to integrate for z, a, and $\frac{\partial L}{\partial \theta}$ jointly in a single call to the ODE solver.

Also, in order to have the effect of the external inputs in the dynamic of the system, as was proposed in previous works (Enciso-Salas et al., 2021, 2022), a nonlinear state-space representation is used:

$$\dot{x}(t) = f(x) + g(u)\,, \tag{4}$$

$$y(t) = h(x)\,, \tag{5}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $f(.)$ is a function $\mathbb{R}^n \to \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$ the input vector, $g(.)$ is a function $\mathbb{R}^m \to \mathbb{R}^n$, $y(t) \in \mathbb{R}^p$ the output vector, and $h(.)$ is an invertible function $\mathbb{R}^n \to \mathbb{R}^p$.

Thus, to include the known external inputs in the mechanism, a new neural network is employed as in (6).

$$U = g(\bar{u}, t, \beta), \tag{6}$$

where $\bar{u}$ represent external inputs in the integration interval, and β represent the parameters of g.

Therefore the new loss function is given by (7):

$$\begin{aligned} L(z(t_1)) = L\Bigg(&z(t_0) + \int_{t_0}^{t_1} f(z(t), t, \theta)\, dt \\ &+ \int_{t_0}^{t_1} g(\bar{u}, t, \beta)\, \mathrm{dt}\Bigg) \\ &= L(ODESolve(z(t_0), \mathbf{f}, \mathbf{g}, t_0, t_1, \theta, \beta))\,, \end{aligned} \tag{7}$$

note that, in this expression, the gradients with respect to θ, β, and t are to be specified.

However, it can be shown (Enciso-Salas et al., 2022; Chen et al., 2018) that the gradients for θ and t will remain given by equations (2) and (3) respectively, and that the gradient with respect to parameters β are given by:

$$\frac{\partial L}{\partial \beta} = -\int_{t_1}^{t_0} a(t)^\top \frac{\partial \mathbf{g}(\bar{u}, t, \beta)}{\partial t}. \tag{8}$$

Algorithm 1 summarizes the adjoint procedure.

Algorithm 1 NODE training by the adjoint sensitivity method with the addition of external inputs

Require: Parameters θ and β, initial time t_0, end time t_1, final state z_{t_1}, loss $\partial L/\partial z_{t_1}$, dynamics $f(z, t, \theta)$ and $g(\bar{u}, t, \theta)$ ▷ Inputs for each step

function $\bar{f}([z_t, a_t,], t, \theta, \beta)$ ▷ Augmented dynamics

  $v = f(z_t, -t, \theta) + g(\bar{u}_t, -t, \beta)$

  return $[-v, a_t \partial v/\partial z, a_t \partial v/\partial \theta, a_t \partial v/\partial \beta]$

end function

$$\begin{bmatrix} z_{t_0} \\ \partial L/\partial z_{t_0} \\ \partial L/\partial \theta \\ \partial L/\partial \beta \end{bmatrix} = ODE\left(\begin{bmatrix} z_{t_1} \\ \partial L/\partial z_{t_1} \\ 0_p \\ 0_p \end{bmatrix}, \bar{f}, -t_1, -t_0\right)$$

return $[\partial L/\partial z_{t_0}, \partial L/\partial \theta, \partial L/\partial \beta]$

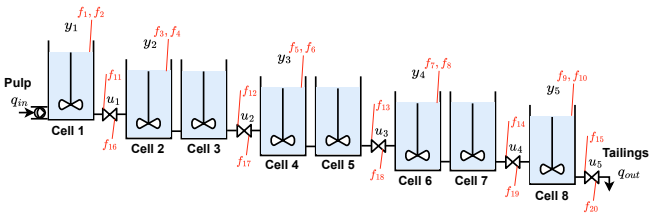

Fig. 1. Flotation process under study.

3 Flotation level process and faults considered

3.1 Flotation process

The system considered is presented in Figure 1; the flotation circuit is composed of eight cells connected in series, typical in copper plants. The flotation process under study has five levels and is composed of 41 equations; there is a set of 5 measurements y_1 to y_5 and a set of 5 control valves u_1 to u_5 (Pérez-Zuñiga et al., 2019).

Parameters of the plant under study are shown in Table 1. Additionally, the faults considered for this work are presented in Table 2.

The system has to operate in closed loop to maintain adequate levels in the tanks. Therefore the levels in each cell are controlled independently by PID controllers.

Table 1. Parameters of the plant under study

Parameter	Value	Units
Feed rate of first cell, q	0.3	m^3/s
Valve flow coefficient, C_v	0.668	$m^{5/2}/s$
Discharge coefficient, C_d	0.5	$m^{5/2}/s$
Gravity, g	9.81	m/s^2
Transversal section	20	m^2

Table 2. Possible faults in the flotation level process.

Fault	Type	Range
Sensor bias y_i for $i = 1 \cdots 5$	Additive	≥ 0.5 cm
Valve bias i for $i = 1 \cdots 5$	Additive	≥ 25.0 %
Valve error u_i for $i = 1 \cdots 5$	Multiplicative	≤ 0.3

4 Training and evaluation of the FDI system

4.1 NODE training

The dataset is composed of noisy samples of inputs and outputs generated from the model in section 3 that are also described in Table 3; all these signals are normalized and used as inputs to the NODE network.

The NODE model is then trained with the following setup:

- The dataset for the NODE training is composed of 1000 samples.
- The dataset is divided into two sets: 80% for training and 20% for validation.
- Then the system is trained with the loss function $\hat{x} - h^{-1}(y)$ for 1000 iterations.
- Learning rate is adapted during the training, starting with 10^{-2} during the first 100 iterations, then 10^{-3} until the 500 iterations, and finally to 10^{-4} for the next iterations.

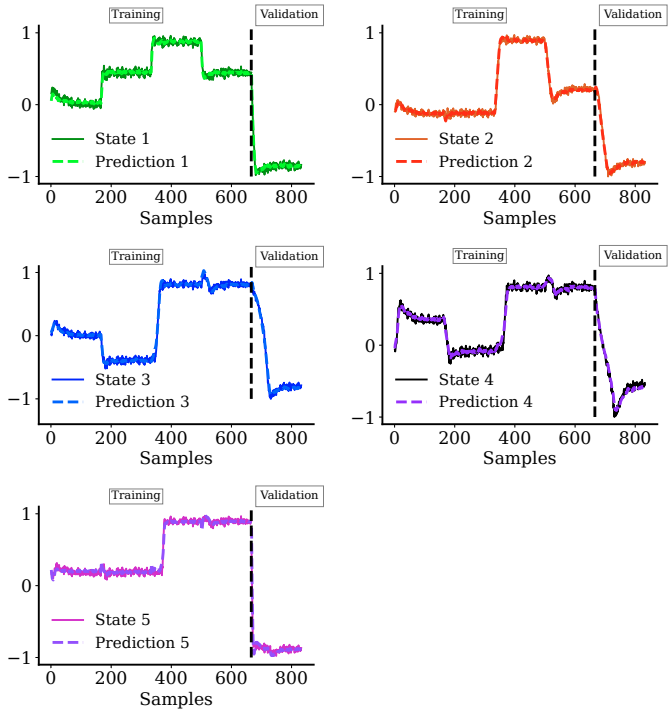

Fig. 2. Training and validation of the NODE for 5 measured signals in the flotation process.

The results are shown in Figure 2 for the training and validation of the NODE model with a good fit in each case.

Table 3. Signals in the dataset.

Signal	Description
q	Feed rate of the first cell
y_i for $i = 1 \cdots 5$	Sensor measurement of level i (see Figure 1)
r_i for $i = 1 \cdots 5$	Set point for level i (see Figure 1)

4.2 Fault dataset and pre-processing

The dataset for the fault diagnosis training is also generated using the model in section 3 for the cases considered in Table 4. The only pre-processing considered is the normalization of the input signals.

4.3 Classifier by fully-connected neural network

For fault detection, a classifier based on two fully-connected layers is added to the NODE system, as presented in Figure 4 (Phase 2), then the following settings are used for training:

- The faulty dataset is divided into 75% for training and 25% for validation.
- The learning rate is set at 10^{-2} and is updated in the 1000 step to 10^{-3} for the rest of the training.

Table 4. Classes considered in relation to failures in the float level process.

Fault condition	Class
Sensor bias y_i rising edge for $i = i \cdots 5$	Fault 1,3,5,7,9
Sensor bias y_i falling edge for $i = i \cdots 5$	Fault 2,4,6,8,10
Pressure drop in valve i for $i = i \cdots 5$	Fault 11 to 15
Valve friction u_i for $i = i \cdots 5$	Fault 16 to 20

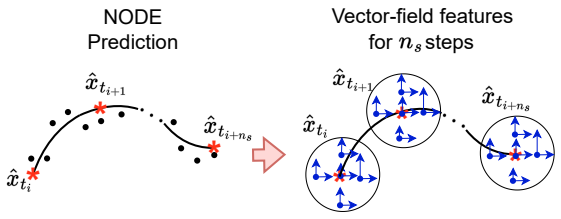

Fig. 3. NODE prediction and vector-field features generation for a sequence of n_s steps.

- Training is performed in 1000 iterations.

The optimization mechanism is described in the algorithm 2. First, the number of batches and their size is defined in lines 1 and 2, then in line 3, the optimizer used for the training is set and configured; for example, as Adam type. A loop is used to repeat the training procedure iteratively, where vectors u_v, y_v^- and y_v are created in lines 6, 7 and 8, and then the prediction $\hat{x}_v$ is calculated using the NODE model in line 9. The local vector fields features $\dot{\hat{y}}$ and $\dot{\tilde{y}}$ are computed in lines 10 and 11, operator $Uniform$ is used to specify the use of a uniform distribution to generate the vector field around the predicted points, in this case a total of five samples are used. The generated features are used by the classifier to elaborate a decision in line 12. And finally, the loss is calculated using cross-entropy method in line 13 and the error used for back-propagation in line 14. A visual representation of the process to generate the features is presented in Figure 3, the process can be described in two stages.

Algorithm 2 : FDI system

Require: u, y^-, t, $labels$ ▷ Initial data
1: $N \leftarrow n\_batch$ ▷ number of sub-batches
2: $S \leftarrow size\_batch$ ▷ size of sub-batches
3: $optimizer = Adam(lr = 0.001)$
4: **for** $i \in epochs$ **do**
5: **for** $batch(N, S) \in training\_DS$ **do**
6: $u_v = u[k - n_s : k]$
7: $y_v^- = y[k - n_s - 1 : k - 1]$
8: $y_v = y[k - n_s : k]$
9: $\hat{x}_v \leftarrow NODE(u_v, h^{-1}(y_v^-))$
10: $\dot{\hat{y}} \leftarrow NODE.step(Uniform(\hat{x}_v))$
11: $\dot{\tilde{y}} \leftarrow NODE.step(Uniform(h^{-1}(y_v)))$
12: $pred \leftarrow Classifier\_NN((\dot{\hat{y}}, \dot{\tilde{y}}))$ ▷ Decision
13: $loss \leftarrow CrossEntropyLoss(pred, label)$
14: $loss.backward()$
15: **end for**
16: **end for**

Additionally, for the evaluation, the following structures were tested:

- **Support Vector Machine - SVM** ($n_s = 20$): The available signals (5 measurements and 5 inputs) grouped for $n_s = 20$ steps. The SVM kernel was a RBF (radial basis function), parameter $C = 10$, and the class weights were 10 for failure and 1 for no failure.
- **Random Forest - RF** ($n_s = 20$): The 5 measurements and the 5 available inputs for $n_s = 20$ steps are used as inputs, an entropy type criterion is used, depth of 10 and 160 estimators were used.

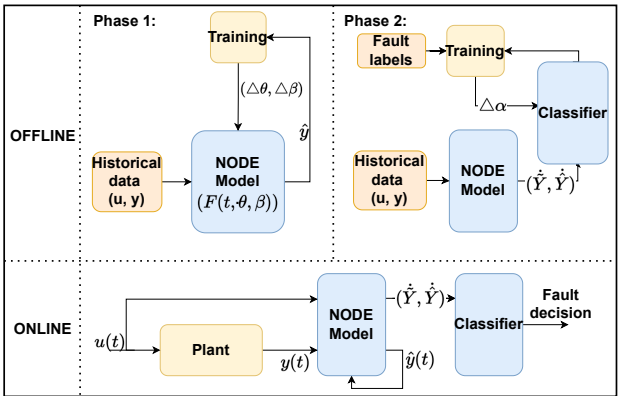

Fig. 4. Scheme of offline and online procedure for the approach.

- **FDI-NODE + SVM** ($n_s = 10$): Diagnostic system based on NODE with $n_s = 10$ steps and SVM classifier with same characteristics as before.
- **FDI-NODE + RF** ($n_s = 10$): Diagnostic system based on NODE with $n_s = 10$ steps and RF classifier with same characteristics as before.
- **FDI-NODE + NN** ($n_s = 2$): Diagnostic system based on NODE with $n_s = 2$ steps and dense neural network (NN) classifier.
- **FDI-NODE + NN** ($n_s = 5$): Diagnostic system based on NODE with $n_s = 5$ steps and NN dense classifier.
- **FDI-NODE + NN** ($n_s = 10$): Diagnostic system based on NODE with $n_s = 10$ steps and NN dense classifier.
- **FDI-NODE + NN** ($n_s = 15$): Diagnostic system based on NODE with $n_s = 15$ steps and NN dense classifier.

A summary of the implementation is shown in Figure 4. In the offline stage, two phases are considered; in the first, the training of NODE is performed, and then in the second phase the classifier is trained to isolate the fault. Later, in the online case, the NODE receives the inputs and measurements and produces the projections of the local vector field derivatives for each instant. The classifier uses these derivatives for fault detection and isolation.

5 Results and evaluation

In practice, any fault dataset is an unbalanced set that contains more data representing the normal operating conditions than fault conditions; for that reason, the *Recall* and the *F1-score* are chosen as metrics to evaluate the system.

$$Recall = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}}, \tag{9}$$

$$\text{F1-score} = 2\frac{Recall * Precision}{Recall + Precision}, \tag{10}$$

where precision is given in the following manner:

$$Precision = \frac{\text{True positives}}{\text{True positives} + \text{False positives}}. \tag{11}$$

Table 5. Mean metrics for different techniques.

S.No.	Training set			Validation set		
	Recall	F1-score	Acc.	Recall	F1-score	Acc.
SVM ($n_s = 20$)	0.28	0.37	85.0	0.37	0.41	88.0
RF ($n_s = 20$)	0.90	0.70	96.0	0.37	0.38	89.0
NODE+RF ($n_s = 10$)	0.99	1.00	100.0	0.57	0.66	92.0
NODE+SVM ($n_s = 10$)	0.94	0.96	97.0	0.76	0.76	90.0
NODE+NN ($n_s = 2$)	0.96	0.97	98.1	0.81	0.82	94.8
NODE+NN ($n_s = 5$)	0.99	0.99	99.5	0.93	0.90	95.7
NODE+NN ($n_s = 10$)	0.99	0.99	99.6	0.93	0.90	96.7
NODE+NN ($n_s = 15$)	1.00	1.00	99.8	0.97	0.95	97.6

The results for each case are presented in Table 5, which represents an average of the metrics obtained for the failures by each algorithm. As shown, the FDI-NODE algorithms obtained better metrics; in addition, the cases with 10 and 15 steps outperformed the cases with 2 and 5 steps. Also, both in the case of SVM and RF algorithms, the metrics improve when using features generated with NODE. However, in the case of RF, the validation can sometimes under-perform, indicating over-fitting to the training set. Thus, the most adaptable methodology seems to be using dense layers for the classifier since they can be trained to produce the best results both in the training set and in the validation set.

For the selection of the most suitable method, a more specific comparison can also be made with the FDI-NODE + NN algorithms in their performance on specific failures considered in the validation set; this is shown in Table 6, it is evident that the results on average are better for the latter case; however, it can also be improved in some cases, such as in faults 2 and 3, where the case that considers 5 steps being the one that obtains a better result, this indicates that a fusion of features could be performed to find a better classifier if necessary.

Finally, the results of the fault isolation are presented graphically in Figure 5 for a sequence of faults in the validation set and using $n_s = 15$ steps. It can be seen that a few false positives are mostly a sub-representation of the appropriate failures by other failures with similar behavior but in a short period of time. Therefore this result shows that the correct fault will be identified in each case with greater probability over a long period of time.

5.1 Discussion

The proposed method showed better capabilities to identify all the faults; this is due to the inclusion of the NODE as a representation of the plant for detecting anomaly behavior in the flotation process; however, to appreciate the advantages of the use of NODE a visual representation is helpful. For that reason, the t-SNE (T distributed Stochastic Neighbor Embedding) (Van Der Maaten and Hinton, 2008) representation is used in this section; the t-SNE is a helpful tool to visualize higher dimensional data in a lower dimension setup; it is based on the optimization of the probabilities of pertinence to a class, considering the mapping from the higher dimension into the lower dimension.

Table 6. Metrics for different failures with different n_s for projections using FDI-NODE in the validation set.

Class	$n_s = 5$		$n_s = 10$		$n_s = 15$	
	Recall	F1-score	Recall	F1-score	Recall	F1-score
Normal	0.99	0.99	0.99	0.99	0.98	0.99
Fault 1	1.00	1.00	1.00	1.00	1.00	1.00
Fault 2	1.00	0.97	1.00	0.97	1.00	0.86
Fault 3	0.93	0.54	0.93	0.54	1.00	0.91
Fault 4	1.00	0.65	1.00	0.65	1.00	1.00
Fault 5	0.87	0.93	0.87	0.93	1.00	1.00
Fault 6	1.00	1.00	1.00	1.00	1.00	1.00
Fault 7	1.00	1.00	1.00	1.00	1.00	1.00
Fault 8	1.00	1.00	1.00	1.00	1.00	1.00
Fault 9	0.93	0.93	0.93	0.93	1.00	1.00
Fault 10	1.00	1.00	1.00	1.00	1.00	0.88
Fault 11	0.96	0.91	0.96	0.91	0.95	0.90
Fault 12	0.90	0.94	0.90	0.94	0.95	0.96
Fault 13	0.75	0.86	0.75	0.86	0.97	0.97
Fault 14	0.90	0.93	0.90	0.93	0.90	0.92
Fault 15	0.97	0.96	0.97	0.96	0.96	0.95
Fault 16	0.99	0.95	0.99	0.95	0.97	0.96
Fault 17	0.47	0.58	0.47	0.58	0.89	0.89
Fault 18	0.80	0.88	0.80	0.88	0.80	0.88
Fault 19	0.97	0.98	0.97	0.98	0.96	0.97
Fault 20	0.99	0.84	0.99	0.84	0.99	0.88

The goal is to visualize the clusters generated by the t-SNE for each fault class using the data generated by the projection generated by NODE and then compare them to the clusters in the original dataset or raw data. The t-SNE for the raw data is shown in Figure 6(a) shows the dispersion of the samples; only some small clusters appear to be identifiable, yet they are interlocked which makes it very difficult to distinguish or classify them. Hence, fault isolation, even by means of an advanced classifier is very complex at this point and explains the low efficiency of the SVM and RT algorithms on the raw data. Meanwhile, the t-SNE for the features generated using NODE with $n_s = 10$ steps is shown in Figure 6(b) showing the formation of well-defined clusters; such clusters can be explained because the projection via NODE highlights abnormal transitions of the local vector fields, therefore, allowing better isolation.

6 Conclusions

The proposed approach was designed for a flotation process where the fault diagnosis system was tested for sensor and actuator failures of different sources. The solution presented is based on the adaptability of the NODE algorithm to model the plant dynamics; besides, extensive pre-processing of the data is not required and can use the measurements and inputs of the system directly. In an offline stage, the NODE and classifier are trained based on historical data; moreover, the NODE is used to provide the features in the form of local vector fields for the classifier to isolate the fault cases. Later, the NODE and classifier can be used in an online operation to detect and isolate the faults in real-time. Results show the better performance of the system in each of the faulty conditions and are also demonstrated utilizing t-SNE visualization showing that the approach permits the creation of well-defined clusters

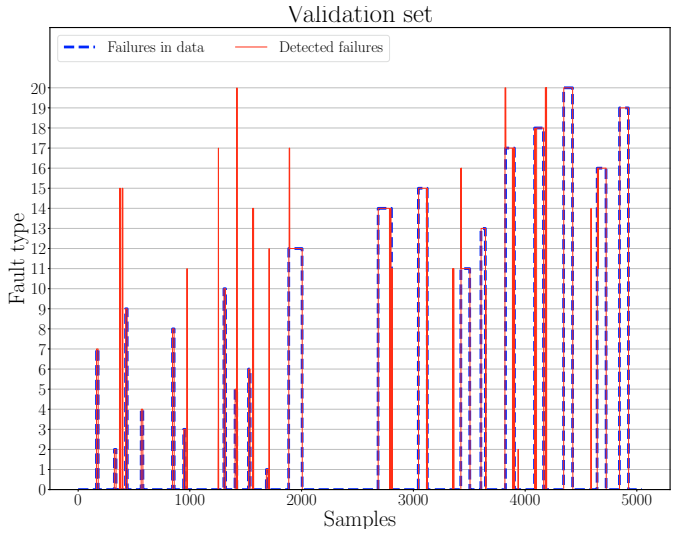

Fig. 5. NODE validation results for a sequence of faulty scenarios using $n_s = 15$ steps.

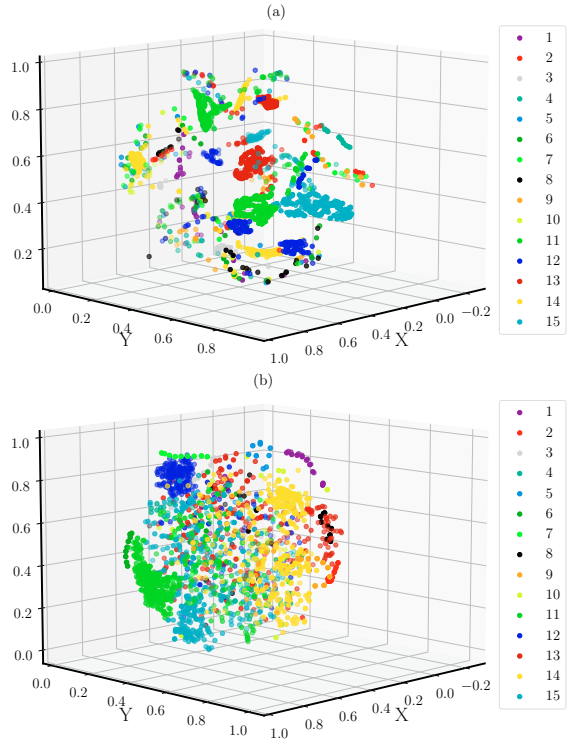

Fig. 6. (a) t-SNE for raw data-set, (b) t-SNE for features generated using $n_s = 10$, in both cases the legend displays the classes of each possible fault.

in comparison with the dispersion of the points in the raw dataset, which is beneficial for the isolation mechanism implemented by any classifier.

7 Acknowledgments:

The authors wish to thank the National Fund for Scientific, Technological Development and Technological Innovation (FONDECYT) through its national program PROCIENCIA (160-2020-FONDECYT) for providing the means and resources for this research and development.

References

Bask, M. and Johansson, A. (2003). Model-based supervision of valves in a flotation process. In *IEEE Conf. on Decision and Control*, 744–749. Hawaii, USA.

Bask, M. and Johansson, A. (2004). Robust time-varying thresholds for supervision of valves in a flotation process. In *IEEE Conf. on Decision and Control*, 4305–4310. Atlantis, Bahamas.

Chen, R.T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural Ordinary Differential Equations. In *NeurIPS 2018*, volume 109, 31–60.

Enciso-Salas, L., Pérez-Zuñiga, G., and Sotomayor-Moriano, J. (2021). Fault Diagnosis via Neural Ordinary Differential Equations. *J. Applied Sciences*, 11(3776).

Enciso-Salas, L., Pérez-Zuñiga, G., and Sotomayor-Moriano, J. (2022). Fault Detection and Isolation for UAVs using Neural Ordinary Differential Equations. *IFAC-PapersOnLine*, 55(6), 643–648.

Jamsa-Jounela, S.L., Dietrich, M., Halmevaara, K., and Tiili, O. (2003). Control of pulp levels in flotation cells. *Control Engineering Practice*, 11(1), 73–81.

Jemwa, G.T. and Aldrich, C. (2006). Kernel-based fault diagnosis on mineral processing plants. *Minerals Engineering*, 19(11), 1149–1162.

Jia, F., Lei, Y., Lin, J., Zhou, X., and Lu, N. (2016). DLN: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mech. Sys. and Signal Proc.*, 72-73, 303–315.

Kidger, P., Foster, J., Li, X., and Lyons, T. (2021). Efficient and Accurate Gradients for Neural SDEs. (34), 18747–18761.

Luo, J., Member, S., Tang, Z., Zhang, H., and Member, G.S. (2021). LTGH : A Dynamic Texture Feature for Working Condition Recognition in the Froth Flotation. *IEEE Trans. On Inst. and Meas.*, 70.

Pérez-Zuñiga, C.G., Sotomayor-Moriano, J., Chanthery, E., Travé-Massuyés, L., and Soto, M. (2019). Flotation Process Fault Diagnosis Via Structural Analysis. *IFAC PapersOnLine*, 52(14), 225–230.

Rubanova, Y., Chen, R.T., and Duvenaud, D. (2019). Latent ODEs for irregularly-sampled time series. *Advances in Neural Information Processing Systems*, 32.

Van Der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9, 2579–2625.

Vásquez, J., Perez-Zuñiga, G., Muñoz, Y., and Ospino, A. (2020). Simultaneous occurrences and false-positives analysis in discrete event dynamic systems. *J. Comput. Sci*, 44, 101162.

Xu, C., Gui, W., Yang, C., Zhu, H., Lin, Y., and Shi, C. (2012). Flotation process fault detection using output PDF of bubble size distribution. *Min. Eng.*, 26(1), 5–12.

Zhang, J., Tang, Z., Xie, Y., Ai, M., and Gui, W. (2019a). Flotation Fault Diagnosis Method Using Statistical Approaches. In *Conf. CBD 2019*, 266–271.

Zhang, S., Zhang, S., Wang, B., and Habetler, T.G. (2019b). ML and DL algorithms for bearing fault diagnostics - a comprehensive review. *arXiv*.

Zhao, L., Peng, T., Xie, Y., Yang, C., and Gui, W. (2017). Recognition of flooding and sinking conditions in flotation process using soft measurement of froth surface level and QTA. *Chemometrics and Intelligent Lab Sys.*