2023 IEEE 21st International Conference on Industrial Informatics (INDIN) | 978-1-6654-9313-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/INDIN51400.2023.10217897

Using differential equation inspired machine learning for valve faults prediction

Benjamin Uhrich*, Nikolai Hlubek†, Tim Häntschel*, Erhard Rahm*
* *Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig, Leipzig University, Germany*
† *Bürkert Fluid Control Systems, Dresden, Germany*
Email: uhrich@informatik.uni-leipzig.de, nikolai.hlubek@burkert.com

***Abstract*—In an industrial plant it is necessary to monitor the operation of the equipment. Deviations from normal operation should be detected as early as possible to avoid production failures or to keep their impact as small as possible. Valves are core components of plants and critical damage can occur if they are set to switch but in reality do not switch. Therefore, plant operators would like to detect such failures as early as possible. Valves with electrodynamic actuators offer a unique diagnostics option that allows for the early detection of valve malfunctions by their intrinsic properties alone. In particular, we measure the inrush current of the valve as a function of time. From the resulting signal, we derive the information, if the valve has switched or not. Machine learning (ML) methods can be used to accurately predict these valve errors. In this paper, a long-short term memory (LSTM)-convolutional neural network (CNN) and two differential equation-inspired neural networks are presented for encoding valves function anomalies to ensure the production capability of an industrial plant. For this purpose, derivatives of the inrush currents are considered and integrated into the deep neural networks (DNNs). The implementation of this diagnostics option is cheap and robust as no extra sensor on the valve is required.**

***Index Terms*—machine learning, diagnostics, solenoid valve, electrodynamic valve**

I. Introduction

Imagine you are operating an analytical device for high-throughput DNA sequencing in a clinical environment. Key process steps of the sequencing involve adding specified amounts of reagents to the sample in a reaction chamber. In this scenario pressure time dosing could be your method of choice to fill the correct amount of reagent into the wells of microplates. In pressure time dosing you keep the liquid under a constant pressure. The opening time of a dosing valve controls that the correct amount of reagent is delivered with high accuracy. If the dosing valve is broken and only partially opens, a wrong amount of reagent is added. This will falsify your result as you cannot know if you had no reaction because of the properties of your sample or because of an incorrectly dosed reagent. Since the dosing mechanism is enclosed inside the device and the reaction volumes are too small to allow for a quick optical inspection a malfunction can go unnoticed. If the valve malfunction persists for some time or is detected at a later process step chances are high that a lot of your analysis batch must be discarded. The repetition of your analysis will cost you extra time and precious resources.

To limit such errors you can install independent diagnostics equipment. Such diagnostic equipment could be a flow meter next to the dosing valve to independently monitor that the valve operation was successful or a photoelectric sensor. The disadvantage of independent diagnostics equipment is, that it means additional cost, requires extra construction space and requires maintenance because it might also fail. An easier way to detect this type of failure is by using the dosing valve as a sensor. This is possible with electrodynamic valves. These special kind of valves use a moving coil actuator (also called voice coil actuator), which creates a motional electromotive force that is visible on the current signal. Hence, by monitoring the valve inrush current during its operation we get information about the movement of the actuator and thus about the valve stroke. It is an inexpensive and robust way to detect partial valve openings. It can either be used as only means of diagnosis for low cost applications or as an additional independent diagnostics tool to supplement external sensors.

Solenoid valves usually are driven by reluctance force actuators which have a fixed coil and a moving magnetic core. Diagnostics using the electric signals of such systems have been studied for some time [1], [2]. However, the diagnostics are difficult, because there is no motional electromotive force due to the moving magnetic core and hence no direct link to the actuator stroke. In this article, we look at a different valve design, that uses a moving coil actuator. In the context of valves, such actuators have only been studied as replacement for the camshaft of internal combustion engines, where they are used without a return spring, which makes diagnostics of the electrical signals

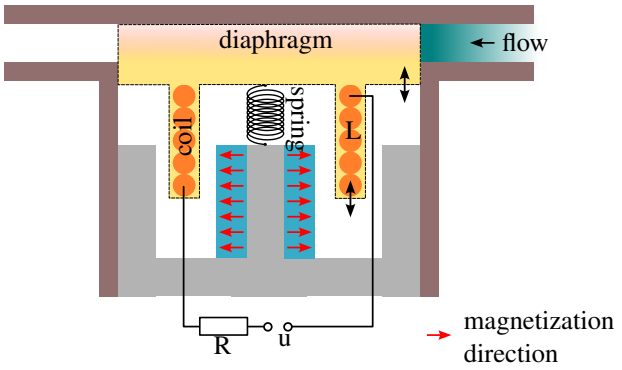

Fig. 1. Schematic showing the components and operation principle of an electrodynamic valve

straight forward [3]. We consider a constructional safe, normally closed valve and try to diagnose insufficient valve movement. Data-driven ML techniques can be used to detect such anomalies in the functionality of a valve, especially time attentive models for analyzing time signals. Recent progress in combining the numerics of ordinary and partial differential equations (ODE/PDE) with the design of DNN architectures have also demonstrated significant advantages in terms of interpretability, robustness, and stability. Therefore, we propose to use such DNN structures for the prediction of valve faults. To the best of our knowledge, this has not been done before. In particular this is the first application of differential equation-inspired neural networks and LSTM-CNNs to predict valve failures based solely on electrical signals and their derivatives. The contributions of our work are as follows:

- We propose three approaches for the prediction of valve fault conditions based on the measured time series of inrush currents: A combination of an LSTM-CNN model, a novel interpretable ODE-inspired network and a robust CNN framework.
- The results are evaluated by comparing them to state of the art ML models.
- We show how the learning process can be accelerated and improved by incorporating derivatives of the measured data and extending the dimension of the input data.

In the following, we first give an overview of the related work to the ML techniques we employ (section II). Afterwards, we explain the physical details of an electrodynamic valve in section III-A. We then describe how we acquired our data set in section III-B. Finally, we describe our proposed models (section IV-A) and evaluate, which ML technique is most suited to predict if a valve has switched in section IV-B.

II. Related Work

In recent years, ML has become an increasingly important area of research in engineering, with applications in many different fields. Some of these key areas are product optimization, predictive maintenance, process optimization and quality control. In the following subsections, we will introduce several ML methods that are relevant to the present work.

A. Convolutional Neural Networks

CNNs are a type of DNNs originally designed for supervised learning tasks on images, including image classification, object recognition and segmentation. The core functionality is based on convolutional operations to extract useful features from the input data, reduce dimensionality and ultimately solve a classification task. CNNs are not restricted to two-dimensional data and have been successfully applied to one-dimensional time series data [4]. An overview of efficient convolutional neural networks combined with hardware acceleration is given by Ghimire *et al.* [5]. Li *et al.* gave a good overview of applications and perspectives of CNNs [6]. A novel attention controlled joint learning convolutional network for condition monitoring of mechanical equipment was proposed by Wang *et al.*. They integrated fault diagnosis and signal denoising tasks into a continuous CNN architecture [7].

B. Long Short-Term Memory

LSTM models are a powerful system for analyzing and predicting time series based on recurrent neural networks (RNN). The network is capable of storing and retrieving information over an extended period of time, especially for time series with long-term dependencies. These models are able to handle exploding or vanishing gradients. An overview of LSTM models is given by van Houdt *et al.* [8]. Zha *et al.* shows a monthly forecast of gas field production using a combination of LSTM and CNN models [9]. Another application of LSTMs was presented by Zhang *et al.*, which deals with the accurate prediction of water quality [10]. An LSTM model that combines knowledge of the physical behavior of gears with the goal of fault detection is shown by Chen *et al.* [11].

C. ODE/PDE-Inspired Neural Networks

Despite the obvious success and resulting popularity of ML in academia and industry, in many cases, there is no good interpretation of the internal processes within the network, that eventually lead to the final prediction. In this sense, DNNs are often considered a black box. In general, it is possible to understand the abstract task of a DNN, but the internal processes cannot be interpreted. Therefore, it is often difficult, to predict the appropriateness and behaviour of a neural network in the context of a particular task. Thus, selecting a network design can be difficult and often requires a lot of trial and error. Moreover, there are generally no guarantees about the stability of the trained model. It has been shown that even networks that appear to work well for a particular task can be unstable. To this end, Weinan was the first author to introduce the bridge between deep residual networks (Resnets) [12] and ODEs [13]. He *et al.* showed the use of ODE-inspired network design for single image super-resolution. The authors proposed several network designs based on Runge-Kutta methods [14]. More specific DNN architectures are presented by Ruthotto *et al.* motivated by PDEs. The classes of PDEs (parabolic, hyperbolic and elliptic) form the basis for the network design presented [15]. Alt *et al.* also transfered the rich set of numerical foundations from the world of PDEs to DNNs [16].

III. Technical background

A. Electrodynamically actuated valve

Fig. 1 shows the basic setup of a normally closed (i.e. closed when de-energized) electrodynamic valve in its closed state. When no electrical current is supplied, a spring keeps the diaphragm pressed against the valve seat. The flow of the medium is blocked. A coil is immersed inside a magnetic

field generated by a strong permanent magnet and its pole pieces. The coil is attached to the diaphragm. When the coil is supplied with an electrical current, it moves in the magnetic field due to the Lorentz force. This lifts the attached diaphragm as well and opens the valve to the flow of the medium. To describe the dynamics of the system we need to consider its mechanical and electrical parts. [17]–[19]

The mechanical part given by (1) is modeled as a driven and damped harmonic oscillator with mass M_{CD} (coil and diaphragm), spring constant K_{SDA} (spring, diaphragm, stiffness due to air compression in the enclosure) and a damping coefficient R_{SD} (spring, diaphragm). The driving force is approximated by the Lorentz force on a single current carrying wire (3) in a magnetic field.

The electrical part is modeled by Kirchhoff's voltage law (2) as given by the equivalent circuit indicated schematically in Fig. 1. The parts of the equation are the supply voltage u_S, the Ohmic resistance of the coil R_C, the self-inductance L_C and the counter-electromotive force or motional electromotive force u_L (4).

The ODEs (1) and (2) describe the dynamics of the system. Equations (3) and (4) are their electromechanical coupling terms. In the following we will refer to these four equations as system equations.

$$M_{CD}\frac{\mathrm{d}^2 x_D\left(t\right)}{\mathrm{d}t^2} = f_{\mathrm{L}}\left(t\right) - R_{SD}\frac{\mathrm{d}x_D\left(t\right)}{\mathrm{d}t} - K_{SDA}x_{\mathrm{D}}\left(t\right) \quad (1)$$

$$u_S\left(t\right) = u_{\mathrm{L}}\left(t\right) + L_C\frac{\mathrm{d}i_C\left(t\right)}{\mathrm{d}t} + R_C i_C\left(t\right) \quad (2)$$

$$f_L\left(t\right) = i_{\mathrm{C}}\left(t\right)\cdot\ell\cdot B \quad (3)$$

$$u_L\left(t\right) = \frac{\mathrm{d}x_D\left(t\right)}{\mathrm{d}t}\cdot B\cdot\ell \quad (4)$$

The range of x is limited by the closed position of the valve seat and the opened position of the valve seat, i.e. $x \in [x_{\mathrm{close}}, x_{\mathrm{open}}]$.

To make the physics stand out we made several simplifications in the system equations. We ignored the force applied on the valve by the pressure of the medium. This force would be an additional term in (1), consisting of pressure and area. Furthermore, we assumed that the quantities in equations (3) and (4) are perpendicular to each other. If this would not be the case, we would need to treat them as vector quantities in the equation's and evaluate their the cross products. Additionally we approximated the coil as a current carrying wire. In the case of an actual coil, we would need to solve Faraday's law of induction to calculate an expression for $U \approx N\cdot \mathrm{d}\Phi_W/\mathrm{d}t$.

Figure 2 shows a numerical solution of the system equation. Plotted are position, velocity and current as a function of time. The parameters of the system equation were set to typical values of electrodynamic valves. We can infer several properties from this solution: All curves start at zero for a closed valve and end at a final steady state after the valve has

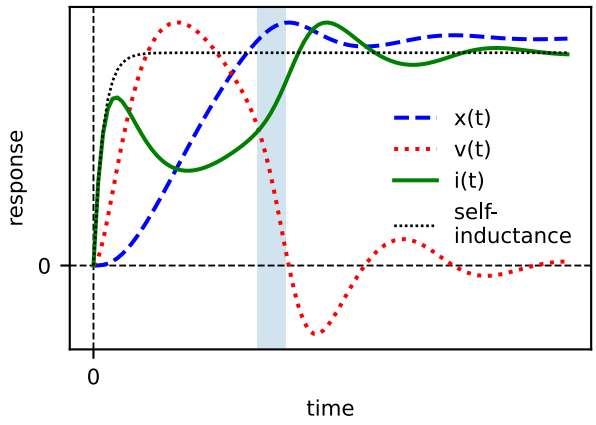

Fig. 2. Solution to the system equations (1), (2), (3), (4) for typical parameters of electrodynamical valves.

opened. The final steady state of the current curve is given by the Ohmic resistance of the coil. This can be seen from equation (2). The velocity of the actuator is zero in the final steady state therefore the motional electromotive force $u_L\left(t\right)$ is zero. The magnetic field has built up and the coil draws a constant current therefore the self-inductance is zero as well. This gives us equation (5) which is just Ohm's law:

$$i_{\mathrm{FSS}} = \frac{U_s}{R_C}. \quad (5)$$

At some point of the dynamic behavior shown in 2 the velocity becomes negative and undergoes a global minimum. This happens when the actuator hits the valve seat of the open position and bounces back. At the point where the actuator hits the valve seat the velocity is zero. The counter-electromotive force at this point is zero as well so that the overall current matches the current of the final steady state. This contact bounce is visible on the position curve and on the current curve by the damped oscillating behavior towards the final steady state.

When comparing the velocity and the current curve, it seems that the first maximum of the velocity is strongly related to the local minimum of the current. This is further corroborated when solving equation (2) without the counter-electromotive force to check the influence of self-induction. This is shown as dashed line in 2. If we ignore the self-induction in equation (2), assume a constant supply voltage and movement perpendicular to the magnetic field we end up with equation (6). For this equation we approximated the current by an Ohmic part and a part based on the counter electromotive force. We know the Ohmic part from the final steady state. Therefore we can extract the counter electromotive force from the time dependent current curve alone. An example of this is shown in the right part of Figure 3. Since the electromotive force is dependent on the velocity of the coil we can approximately say that the drop in current during switching of the valve is proportional to the velocity of its actuator. Hence we can infer

that the integral of the drop in current is proportional to the distance the valve actuator has travelled and thus to the stroke of the valve.

$$i\left(t\right) \approx i_{\mathrm{L}}\left(t\right) + i_{\mathrm{FSS}} = \frac{u_{\mathrm{S}}}{R_{\mathrm{C}}} - \frac{B\ell}{R_{\mathrm{C}}} \frac{\mathrm{d}x_{\mathrm{D}}\left(t\right)}{\mathrm{d}t} \tag{6}$$

In physical terms this time integral over an electrical power represents the work performed by the actuator.

In a practical customer application, solving these system equations requires an extensive knowledge of the fluidic system to fill all the parameters in the equation with meaningful values. These parameters are typically not know. They could be determined experimentally. This however requires time, effort and knowledge. Three quantities a customer usually is not willing to spent or does not have. Even if one invests in a functional model this only helps to detect deviations if all external parameters are captured by sensors which is expensive. Therefore, in the following we describe different supervised ML approaches to derive information from the inrush current and provide the customer with diagnostics information.

B. Data acquisition

The curve shape of the induction current depends on internal parameters such as the valve diaphragm material and external parameters such as the environmental temperature and a potential fault state. In order to understand all these factors of influence, a multivariate data set has to be acquired where all relevant parameters are varied in a controlled manner. We measured a data set of time resolved inrush current curves where we changed the environmental temperature, the coil temperature, the diaphragm type, the pressure and the voltage over the range allowed by the valve data sheet. Additionally we introduced fault states by blocking the actuator at certain positions. We used this multivariate data set to train our classifiers. Fig. 3 shows an overview of the data set.

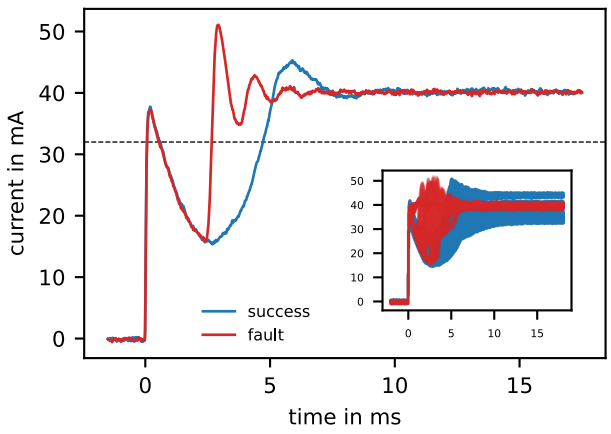

Fig. 3. Comparison of a successful valve switching (100% stroke, blue color) and a faulty switching (50% stroke, red color). The inset shows an overview of the complete dataset highlighting that when taking all parameters into account, the two classes overlap and are not seperable by using thresholds.

IV. Machine Learning

A. Methods

1) LSTM-CNN: LSTM networks represent an improved time sequential model class of RNNs that use memory blocks to solve the vanishing and exploding gradient problem. Generally, an LSTM network consists of a combination of three phases, such as an input gate, a forget gate, and an output gate. The functionality of LSTM states can be formalised as follows:

$$f_t = \sigma(W_f h_{t-1} + W_f u_t + b_f) \tag{7}$$

$$i_t = \sigma(W_i h_{t-1} + W_i u_t + b_i) \tag{8}$$

$$c_t = f_t c_{t-1} + i_t \tilde{c} \tag{9}$$

$$\tilde{c} = tanh(W_i h_{t-1} + W_i u_t + b_i) \tag{10}$$

$$o_t = \sigma(W_o h_{t-1} + W_o u_t + b_o) \tag{11}$$

$$h_t = o_t \tanh(c_t) \tag{12}$$

where u_t represent the current input vector, c_{t-1}, c_t are the previous and current cell state, respectively, $\tilde{c}$ is the current moment activation vector, h_t, h_{t-1} are the current and previous outputs, f_t, i_t, o_t are the forget gate-,input gate-,output gate activation vectors. W and b are the trainable weight matrices and bias vectors of the three different gates. σ is the sigmoid function. The weights and biases are updated through backpropagation. For the second part of the LSTM-CNN model, the 1D convolutional operation for input with two channels can be described:

$$y_j = (s * p) = \sum_{c=0}^{M-1} \sum_{m=0}^{N-1} s_{c,m+j} p_{c,m} \tag{13}$$

where s is the input vector, p denotes a 1D kernel of size N, M describes the size of the input channels and y_j gives the 1D feature vector. The convolutional operation is described by s*p. The 1D feature vector is the foundation for the ultimatly final classification decision.

2) PDE-Inspired Neural Network: The following classes of neural networks are inspired by the numerics of PDEs, as mentioned in Section II.C. To study PDEs systematically, it is helpful to classify them according to their structure. There are three classes of PDEs, namely elliptic, parabolic and hyperbolic. In this paper we will focus on hyperbolic and parabolic PDEs. A parabolic time dependent PDE requires second order differentials. The time differential can be discretized in the form of a difference quotient:

$$\partial_t u(x,t) = \frac{u(x, t+\delta_t) - u(x,t)}{\delta_t} \tag{14}$$

To model this in a DNN, a classical convolutional Resnet architecture can be used. In this case, the resulting discretized sequential model can be linked to the explicit forward Euler method to solve PDEs numerically. For given input data u^0, the one-dimensional CNN model is a sequence of layers of the form:

$$u^{t+1} = u^t + \delta_t(-k(\theta^t) * \sigma(k(\theta^t) * u^t)) \tag{15}$$

where u^t is the current network layer, θ are trainable parameters, σ is the activation function and $\delta_t = 1$. We set $f(\theta^t, u^t) = -k(\theta^t)^T \sigma(k(\theta^t)u^t)$. Due to the definition of $f(\theta^t, u^t)$, the resulting dynamics can be considered parabolic, the convolution kernel being interpreted as a differential operator. For each function value of the input time series, the convolution operation essentially computes the linear combination of its neighbourhood with coefficients from the kernel. There is an analogy between discrete spatial differentials and convolution kernels. To model hyperbolic PDEs as DNNs, it is necessary to introduce a second-order time derivative equivalent. For this purpose, the following DNN relies on the leapfrog method for solving PDEs. Recall that a discrete approximation for the second order time derivative is as follows:

$$\delta_t^2 u(x,t) = \frac{u(x,t-\delta_t) - 2u(x,t) + u(x,t+\delta_t)}{\delta_t^2} \tag{16}$$

Thus we can approximate a dynamic by calculating:

$$u(x,t+\delta_t) = 2u(x,t) - u(x,t-\delta_t) + \delta_t^2 u(x,t) \tag{17}$$

A neural network architecture that corresponds to hyperbolic PDEs can therefore be build by the rule:

$$u^{t+1} = 2u^t - u^{t-1} + \delta_t^2 f(\theta^t, u^t) \tag{18}$$

The method is called Leapfrog because it refers not only to the values of the current time step, but also on the values of a previous time step, as can be seen in Figure 4.

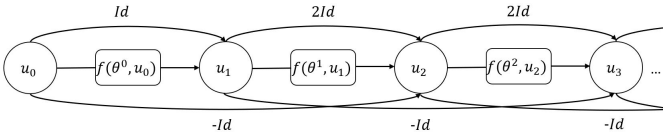

Fig. 4. Hyperbolic PDE-inspired CNN architecture

3) ODE-inspired Neural Network: There is another possibility to design a DNN inspired by the ODE-system of the valve dynamics itself. By approximation for some small time step δ_t, we obtain the discrete system of equation (1), (2), (3), (4) in the explicit scheme :

$$K = M_{CD} + \frac{\delta_t}{2} R_{SD} \tag{19}$$

$$P = \frac{\delta_t}{2} R_{SD} - M_{SD} \tag{20}$$

$$Q = 2M_{CD} - \delta_t^2 K_{SDA} \tag{21}$$

$$S = L_c - \delta R_c \tag{22}$$

$$Kx_D(t+\delta_t) = \delta_t^2 (B \cdot l) i_c(t) + P x_D(t-\delta_t) + Q x_D(t) \tag{23}$$

$$L_c i_c(t+\delta_t) = v(x_D(t+\delta_t) - x_D(t)) + S i_c(t) - \delta_t u_s(t) \tag{24}$$

Based on a time series i_c the valve type can be identified. It is assumed that a valve type can be associated with the set of (hidden) associated physical parameters ($M_{CD}, R_{SD}, Lc, ...$). The network is informed by the equations (19), (20), (21), (22), (23), (24) to determine appropriate parameters for the current flows of a given valve type. In general, the method works similar to a standard recurrent layer, but there are some special features. The output and signal values used as input to a cell depend on the formula for the explicit scheme (the recurrent layer uses one value per cell). Only one-dimensional cell outputs are used, since the current is real-valued and the valve moves only along a spatial direction. Certain parameters are constrained according to the coefficients in the differential equations, and physical assumptions can be derived from the coefficients to initialize the trainable parameters. Last, the activation function is omitted, in order to preserve the linearity of the equations. In this approach, a model is trained for iteratively predicting the current flow over time from previous observations. The model numerically computes values for i_C according to (24), where the hidden valve parameters are trainable weights. Estimates for the hidden values of x_D are provided by (23). Using Θ to denote model weights, the calculations performed by the model are given by:

$$\hat{x}_{t+1} = i_t \Theta_i^{(x)} + \hat{x}_{t-1} \Theta_{x_{t-1}}^{(x)} + x_t \Theta_{x_t}^{(x)} \tag{25}$$

$$\hat{i}_{x+1} = i_t \Theta_i^{(i)} + (\hat{x}_{t+1} - \hat{x}_t) \Theta_x^{(i)} + \Theta_{const.}^{(i)} \tag{26}$$

Here, $\hat{x}$ and $\hat{h}$ denote predicted values, whereas h denotes input data. $x[0]$ and $x[1]$ are initialized as 0, reflecting that before the valve stroke, the diaphragm is resting at its initial position. Since (23) contains valve-specific parameters, we train a number of different versions of (25), according to the number of different valve types. For the subsequent classification, the model returns the predicted values $\hat{h}$ according to the trained valve parameters, as well as the inputs h. The underlying assumption of this architecture is that given the right valve specific parameters and starting values, the equations (23) and (24) can predict the current flow in the future. Comparing the predictions to the actual measurements, the model should be able to discriminate whether the valve has the implied parameters. The optimisation of the weights is accelerated by two loss functions, where the first measures the prediction quality (categorical cross-entropy loss) on the training data, while the second forces the fitting of a smooth function to protect against over-fitting (gradient regulariser):

$$\mathcal{L}_1(f) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \sum_{j=1} -ln(f(x)_j) y_j \tag{27}$$

$$\mathcal{L}_2(f) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \sum_{j=1} |\nabla f(x)_j|^2 \tag{28}$$

B. Results

We try to explain the data set in Fig. 3 using the techniques presented in section IV-A. In order to see how good they perform, we evaluate them against typical ML classifiers and a simple baseline model. An overview of the performance is given in TABLE I.

For the baseline model, we counted the time the current is below a certain threshold $i_\mathrm{t} = 32\mathrm{mA}$. This threshold is

TABLE I
OVERVIEW OF THE QUALITY OF THE CLASSIFIERS.

NoMC[a]	Acc[b]	Precision[b]	Recall[b]	Pre[c]	Method
0	100	100	100	snv	Hyperbolic-CNN
0	100	100	100	snv	ODE-CNN
0	100	100	100	snv	LSTM-CNN
4	99	99	100	max	kNN
10 ± 2	99	99 ± 0.4	99 ± 0.0	snv	gradient boosting
15 ± 6	99	98 ± 2	99 ± 1	snv	1D transformer
16 ± 3	99	98 ± 0.5	98 ± 0.5	snv	1D CNN
17	99	97	99	snv	random forest
20	99	98	98	snv	svm
22	98	96	99	raw	PCA+kmean
26	98	99	95	snv	PLS
27 ± 13	98 ± 1	97 ± 4	97 ± 2	raw	FFN
51	96	96	91	raw	baseline

[a]Total number of misclassifications from our data set of 1364.
[b]Given as a percentage.
[c]Scaling method applied during data preprocessing.

shown in 3 as dashed line. If this time was large enough, we counted this as a successful switching. The reason for this is that a valve that does not switch at all reaches the final steady state current almost immediately as shown in 2. A valve that switches produces a motional electromotive force which reduces the current over the duration of the movement. Therefore for a successful switching the current is expected to be below our current threshold for some time.

From the staple of shallow learning techniques, we used a k-nearest neighbors classifier (kNN) with a k of 13 [20], [21], partial least squares regression (PLS) with 7 components [22], a random forest classifier with 100 estimators [23], [24] and a gradient boosting classifier with 100 estimators [25]. We transformed the PLS regressor to a classifier by using a threshold on the regression value. Additonally, we tried a combination of principal components analysis and k-means clustering using 4 components and 5 clusters.

As standard deep learning techniques we used a feed forward network [26], a 1D CNN [4] and a 1D transformer network [27]. The feed-forward networks had [128 32] neurons in fully connected layers. The design of the 1D CNN is done as in [4] with two sets of Convolution, MaxPooling and Dropout layers. The filters were [128 64] the kernel size was 5. The transformer network uses the example code from Keras [28] without hyperparameter tuning. To estimate the model error of the neural networks we trained each network 10 times and derived statistics from the deviations. The best transformer network had 9 misclassifications. Those are shown in Fig. 5 in addition to closely matching successful classifications. From this figure it is obvious that the networks have a problem to detect the inflection point in the data correctly. Therefore, we designed a LSTM-CNN that work with the raw data and the first derivative. In order to improve the information each cell obtains, we numerically calculate the differential in each time step and pass it as an additional input. Since we know that the valves follow specific differential equations, we would assume that this can improve the classification. The LSTM-CNN architecture is shown in Algorithm 1, based on section IV-A1.

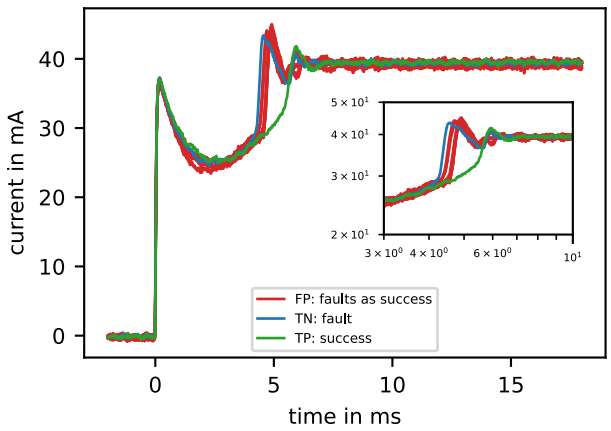

Fig. 5. All misclassifications of the best transformer network (shown in red) were false positives (FP) - i.e. unsuccessful switching classified as successful switching. Additionally, a true positive (TP) data set and a true negative (TN) data set are shown. The inset shows the transition region in logarithmic scale.

Algorithm 1 General Model (x - input signal, n - input dimension, c - #channels, k - #categories, θ - convolution kernel, W - classifier weights)

function MODEL($x_{[1\times n]}$):
  $\Delta x_{[1\times n]} \leftarrow x_{[1\times n]} - (0, x_1, x_2, \ldots, x_{n-1})$
  ▷ numerical differentiation
  $x'_{[2\times n]} \leftarrow \text{CONCATENATE}(x_{[1\times n]}, \Delta x_{[1\times n]})$
  $x'_{[2\times (n/20)]} \leftarrow \text{AVGPOOL}(x_{[2\times n]}, 20)$
  $u_{[c\times (n/20)]} \leftarrow \text{TIMEATTENTIVELAYER}(x'_{[2\times n/20]})$
  ▷ either ODELAYER or LSTMLAYER
  $u'_{[c\times (n/80)]} \leftarrow \text{AVGPOOL}(u_{[c\times (n/20)]}, 4)$
  $v_{[c\times (n/80)]} \leftarrow u'_{[c\times n/80]} * \theta_{[(c\times c)\times 4]}$
  ▷ convolutional embedding
  $v'_{[c\times (n/400)]} \leftarrow \text{MAXPOOL}(v_{[c\times (n/80)]}, 5)$
  $f_{[1\times (c\cdot n/400)]} \leftarrow \text{CONCATENATE}((v'_{1,.}, v'_{2,.}, \ldots, v'_{c,.}))$
  ▷ flattening
  $f'_{[1\times (c\cdot n/400)]} \leftarrow \text{TANH}(f_{[1\times (c\cdot n/400)]})$
  $p_{[1\times k]} \leftarrow f'_{[1\times (c\cdot n/400)]} W_{[(c\cdot n/400)\times k]}$
  ▷ linear classification
  return SOFTMAX($p_{[1\times k]}$)
end function

In this way, it was possible to correctly classify all current curves and thus predict the fault conditions of the valves with 100 percent accuracy. Although the LSTM-CNN model produces excellent results, it is not interpretable and stability estimates are also not possible. To address these two issues, two further architectures have been developed, inspired by the foundations of numerical analysis of differential equations described in IV-A2 and IV-A3, as shown in algorithms 2 and 3. The training of the developed methods is controlled by minimizing the loss functions (27), (28). We evaluate the performance of the different networks by model inference on a randomly sampled test dataset, that was not used for training.

Algorithm 2 ODE-inspired Layer (h - input signal, n - input dimension, c - #channels, $\Theta^{(\cdot)}_{\cdot}$ - weights)

function ODELAYER($h_{[1\times n]}$)

▷ approximate valve position from current flow time series, trainable weights are physical parameters of ODE system, multiple channels for variety of diaphragm

  for $i = 1, \ldots, c-1$ **do**

    $\hat{x}_0^{(i)},\ \hat{x}_1^{(i)} \leftarrow 0,\ 0$

    for $t = 1, \ldots, n-1$ **do**

      $\hat{x}_{t+1}^{(i)} \leftarrow h_t \cdot \Theta_h^{(i,x)} + \hat{x}_{t-1}^{(i)} \cdot \Theta_{x_{t-1}}^{(i,x)} + \hat{x}_t^{(i)} \cdot \Theta_{x_t}^{(i,x)}$

    end for

  end for

▷ predict current flow one step into the future using approximated valve positions, one prediction for each trajectory

  for $i = 1, \ldots, c-1$ **do**

    for $t = 1, \ldots, n-1$ **do**

      $\hat{h}_{t+1}^{(i)} \leftarrow h_t \cdot \Theta_h^{(h)} + \left(\hat{x}_{t+1}^{(i)} - \hat{x}_t^{(i)}\right) \cdot \Theta_x^{(h)} + \Theta_{const.}^{(h)}$

    end for

  end for

  $u_{[c\times n]} \leftarrow$ CONCATENATE$(\hat{h}_{[(c-1)\times n]}, h_{[1\times n]})$

  return $u_{[c\times n]}$

end function

Algorithm 3 Hyperbolic CNN (x - input signal, n - input dimension, k - #categories, θ - convolution kernel, W - classifier weights)

function HYPERBOLIC-MODEL($x_{[1\times n]}$):

  $\Delta x_{[1\times n]} \leftarrow x_{[1\times n]} - \left(0, x_1, x_2, \ldots, x_{n-1}\right)$

▷ numerical differentiation

  $x'_{[2\times n]} \leftarrow$ CONCATENATE$(x_{[1\times n]}, \Delta x_{[1\times n]})$

  $x_{0_{[2\times n]}} \leftarrow$ BATCHNORMALIZATION$(x'_{[2\times n]})$

  $x_{0_{[64\times n]}} \leftarrow x_{0_{[2\times n]}} * \theta_{[2\times 5\times 64]}$

  $x_{0_{[64\times n]}} \leftarrow$ MAX$(0, x_{0_{[64\times n]}})$

  $x_{1_{[64\times n]}} \leftarrow x_{0_{[64\times n]}}$

  $i = 2$

  while $i \leq n-1$ **do**

    $x_{i_{[64\times n]}} \leftarrow u_{i-1_{[64\times n]}} * \theta_{[2\times 5\times 64]}$

    $x_{i_{[64\times n]}} \leftarrow$ MAX$(0, x_{i_{[64\times n]}})$

    $x_{i_{[64\times n]}} \leftarrow u_{i-1_{[64\times n]}} * \theta_{[2\times 5\times 64]}$

    $x_{i_{[64\times n]}} \leftarrow 2u_{i-1_{[64\times n]}} + u_{i_{[64\times n]}} - u_{i-2_{[64\times n]}}$

    $x_{i_{[64\times n]}} \leftarrow$ MAX$(0, x_{i_{[64\times n]}})$

    $i \leftarrow i + 1$

  end while

  $v'_{[64\times(n/8)]} \leftarrow$ AVGPOOL$(x_{n-1_{[64\times n]}}, 8)$

  $f_{[1\times(64\cdot n/8)]} \leftarrow$ CONCATENATE$((v'_{1,\cdot}, v'_{2,\cdot}, \ldots, v'_{c,\cdot}))$

▷ flattening

  $p_{[1\times k]} \leftarrow f'_{[1\times(c\cdot n/8)]} W_{[(c\cdot n/8)\times k]}$

▷ linear classification

  return SOFTMAX$(p_{[1\times k]})$

end function

We measure accuracy, precision and recall for each model. The results are presented in TABLE I.

V. DISCUSSION

In the context of diagnostics for electrodynamical valves we typically deal with systems that are operated using microcontrollers. Therefore the computational resources for ML are limited. We did not restrict ourselves in this regard in our initial overview of ML techniques as shown in TABLE I. However, for a practical application the resource consumption of a prediction model is a critical parameter. In particular, this means that kNN, gradient boosting and random forest approaches end up with models that are way too large for microcontrollers.

Additionally we note that although the overall number of misclassifications is small, regardless of the technique used, for the best customer acceptance the number of misclassifications should actually be zero in our laboratory setup.

All methods from section IV-A fulfil this requirement. They are therefore well suited for predicting valve failures. In order to develop a more general approach, the approaches need to be evaluated on more data and other valve test setups, especially on valves with different functions. The transferability of the approaches to other valves in industrial plants has not yet been proven. The integrability of the models on microcontrollers has not been sufficiently addressed in this work and should be further investigated. All presented models achieve a accuracy of 100 percent with no misclassifications, which is only possible by including information about the derivative of the current. Furthermore, the approaches are competitive and can outperform state of the art ML models. The LSTM-CNN model is capable of classifying 100 percent of valve fault conditions, but interpretability and stability estimation are not possible. For this purpose, advantageous properties of the ODE-system describing the valve dynamic are integrated into the DNN design. This allows not only the prediction of valve errors, but also the discovery of physically interpretable parameters of the ODE system. The overfitting issue of DNNs is protected using a second loss function, which penalize high gradients to ensure a smooth objective function within the classes. The number of trainable parameters is relatively small (LSTM-CNN:247, ODE-CNN:130) compared to the data samples of 1729 time series, indicating that there is no overfitting. Moreover, valuable stability properties of the Leapfrog method can be transferred to the hyperbolic CNN architecture. These include forward stability and the decrease in error as time progresses. There is also the expectation that problems such as chaotic behavior (small disturbances at the input can have large effects on the output) are avoided.

VI. CONCLUSION

Valves with electrodynamic actuators can use the actuator as a sensor by measuring the time dependence of the current during their operation. In particular it can be determined if the valve opened properly or not. We presented three techniques

that use the raw data and the first derivative to achieve a perfect classification. It should be possible to estimate further parameters such as type of valve diaphragm, pressure of the medium, viscosity of the medium and environmental temperature as well. We plan to verify the quality of the estimation in further studies [29]. Our initial application example was a clinical laboratory that is filling reaction chambers with reagents during high-throughput DNA sequencing using pressure time dosing. We have shown that such a setup can benefit greatly from the intrinsic diagnostics offered by electrodynamic valves. It is possible to ascertain proper valve operation without the need for external sensors. The hardware for this kind of diagnostics is inexpensive compared to an additional sensor. All the intelligence is in the smart algorithm tuned to the specific type of valve. To use this technology a time resolved current measurement and a microcontroller for analysis are required. If the application already has a microcontroller the analysis can run there. No external sensors on the valve are needed. This makes this mode of diagnostics robust as there are a small number of parts that can break. This also makes it easy to add to systems where construction space is limited or valves are tightly clustered.

ACKNOWLEDGMENT

We thank N. Kurzay for the countless hours he spent in the laboratory carefully performing the measurements. The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany and by the Sächsische Staatsministerium für Wissenschaft Kultur und Tourismus in the program Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification number: ScaDS.AI

REFERENCES

[1] G. Tod, G. Mazaev, K. Eryilmaz, A. P. Ompusunggu, E. Hostens, and S. V. Hoecke, "A Convolutional Neural Network Aided Physical Model Improvement for AC Solenoid Valves Diagnosis," in *Prognostics and System Health Management Conference*, 2019, pp. 223–227.

[2] M. Utah and J. Jung, "Fault state detection and remaining useful life prediction in AC powered solenoid operated valves based on traditional machine learning and deep neural networks," *Nuclear Engineering and Technology*, vol. 52, no. 9, pp. 1998–2008, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1738573319308435

[3] T. Guo, S. Chang, Z. Chen, H. Huang, and J. Xu, "Fault monitoring and diagnosis of actuators in electromagnetic valve-train based on neural networks optimization algorithm," *IEEE Access*, vol. 7, pp. 110 616–110 627, 2016.

[4] S. Kiranyaz, T. Ince, R. Hamila, and M. Gabbouj, "Convolutional Neural Networks for patient-specific ECG classification," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015, pp. 2608–2611.

[5] D. Ghimire, D. Kil, and S.-h. Kim, "A survey on efficient convolutional neural networks and hardware acceleration," *Electronics*, vol. 11, no. 6, p. 945, 2022.

[6] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2022.

[7] H. Wang, Z. Liu, D. Peng, and Z. Cheng, "Attention-guided joint learning cnn with noise robustness for bearing fault diagnosis and vibration signal denoising," *ISA transactions*, vol. 128, no. Pt B, pp. 470–484, 2022.

[8] G. van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5929–5955, 2020.

[9] W. Zha, Y. Liu, Y. Wan, R. Luo, D. Li, S. Yang, and Y. Xu, "Forecasting monthly gas field production based on the cnn-lstm model," *Energy*, vol. 260, p. 124889, 2022.

[10] Y. Zhang, C. Li, Y. Jiang, L. Sun, R. Zhao, K. Yan, and W. Wang, "Accurate prediction of water quality in urban drainage network with integrated emd-lstm model," *Journal of Cleaner Production*, vol. 354, p. 131724, 2022.

[11] Y. Chen, M. Rao, K. Feng, and M. J. Zuo, "Physics-informed lstm hyperparameters selection for gearbox fault detection," *Mechanical Systems and Signal Processing*, vol. 171, p. 108907, 2022.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[13] W. E, "A proposal on machine learning via dynamical systems," *Communications in Mathematics and Statistics*, vol. 5, no. 1, pp. 1–11, 2017.

[14] X. He, Z. Mo, P. Wang, Y. Liu, M. Yang, and J. Cheng, "Ode-inspired network design for single image super-resolution," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 1732–1741.

[15] L. Ruthotto and E. Haber, "Deep neural networks motivated by partial differential equations," *Journal of Mathematical Imaging and Vision*, vol. 62, no. 3, pp. 352–364, 2020.

[16] T. Alt, K. Schrader, M. Augustin, P. Peter, and J. Weickert, "Connections between numerical algorithms for pdes and neural networks," *Journal of Mathematical Imaging and Vision*, vol. 65, no. 1, pp. 185–208, 2023.

[17] Falaize, Antoine and Hélie, Thomas, "Passive modelling of the electrodynamic loudspeaker: from the Thiele-Small model to nonlinear port-Hamiltonian systems," *Acta Acust.*, vol. 4, no. 1, p. 1, 2020. [Online]. Available: https://doi.org/10.1051/aacus/2019001

[18] Kallenbach, Eberhard, Eick, Rüdiger, Ströhla, Tom, Feindt, Karsten, Kallenbach, Matthias, and Radler, Oliver, *Elektromagnete: Grundlagen, Berechnung, Entwurf und Anwendung*. Springer Vieweg, 2018.

[19] Wang, Y., Megli, T., Haghgooie, M., Peterson, K., and Stefanopoulou, Anna, "Modeling and Control of Electromechanical Valve Actuator," *SAE Technical Paper*, vol. 01, 03 2002.

[20] Fix, Evelyn and Hodges, J.L., "Discriminatory analysis, nonparametric discrimination," *unpublished*, 1951.

[21] Cover, T. and Hart, P., "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[22] H. Wold, "Path Models with Latent Variables: The NIPALS Approach," in *Quantitative Sociology*, ser. International Perspectives on Mathematical and Statistical Modeling, Blalock, H.M., Aganbegian, A., Borodkin, F.M., Boudon, Raymond, and Capecchi, Vittorio, Eds. Academic Press, 1975, pp. 307–359.

[23] Ho, Tin Kam, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, pp. 278–282 vol.1.

[24] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[25] Friedman, Jerome H., "Greedy function approximation: A gradient boosting machine." *The Annals of Statistics*, vol. 29, no. 5, pp. 1189 – 1232, 2001.

[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[27] Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N., Kaiser, Łukasz, and Polosukhin, Illia, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[28] Ntakouris, Theodoros, "Timeseries classification with a transformer model," https://keras.io/examples/timeseries/timeseries_transformer_classification/, 2021.

[29] Hlubek, Nikolai, Baumann, Michael, Heinze, Sebastian, and Ostermaier, Florian, "Using Machine Learning for Diaphragm Prediction in Solenoid Valves," in *IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2022, pp. 1–4.