 doi:10.20944/preprints202109.0099.v1

Article

Predictive Maintenance: An Autoencoder Anomaly-based Approach for a 3 DoF Delta Robot

Kiavash Fathi [1], Hans Wernher van de Venn [2]* and Marcel Honegger [3]

[1] Institute of Mechatronic Systems, Zurich University of Applied Sciences, Switzerland; fath@zhaw.ch
[2] Institute of Mechatronic Systems, Zurich University of Applied Sciences, Switzerland; vhns@zhaw.ch
[3] Institute of Mechatronic Systems, Zurich University of Applied Sciences, Switzerland; honr@zhaw.ch
* Correspondence: vhns@zhaw.ch; Tel.: +41-58-934-77-89

Abstract: Performing predictive maintenance (PdM) is challenging for many reasons. Dealing with large datasets which may not contain run-to-failure data (R2F) complicates PdM even more. When no R2F data are available, identifying condition indicators (CIs), estimating the health index (HI), and thereafter, calculating a degradation model for predicting the remaining useful lifetime (RUL) are merely impossible using supervised learning. In this paper, a 3 dof delta robot used for pick and place task is studied. In the proposed method, autoencoders (AEs) are used to predict when maintenance is required based on the signal sequence distribution and anomaly detection, which is vital when no R2F data is available. Due to the sequential nature of the data, non-linearity of the system, and correlations between parameter time series, convolutional layers are used for feature extraction. Thereafter, a sigmoid function is used to predict the probability of having an anomaly given CIs acquired from AEs. This function can be manually tuned given the sensitivity of the system or optimized by solving a minimax problem. Moreover, the proposed architecture can be used for fault localization for the specified system. Additionally, the proposed method is capable of calculating RUL using Gaussian process (GP), as a degradation model, given HI values as its input.

Keywords: Predictive maintenance; Anomaly detection; Autoencoder; Gaussian processes; Deep learning; Data-driven maintenance

1. Introduction

PdM is an optimal approach for performing condition based maintenance of systems, based on real-time monitoring and identifying CIs which result in a failure when deteriorated [1]. In fact, by using CIs, it is possible to detect fault or degradation, when any of the CIs depicts a symptom of a near future malfunctioning of the system. Moreover, after the anomaly detection and localization steps, the next task would be to estimate the RUL of the studied system [2,3]. In comparison with other classical methods like periodic and preventive maintenance, such an approach can significantly reduce the maintenance costs, since almost the entire useful life of the system can be utilized. One approach for performing PdM is data-driven maintenance planning framework [4]. Data-driven models have proven to be effective when the complexity of the studied system prevents using model-based approaches [5]. In fact, data-driven models tend to reveal the correlations between collected sensor data and thus are capable of calculating corresponding system formation such as RUL [6]. Moreover, given the improvements in data acquisition and availability of data, the aforementioned data-driven PdM has even caught much attention lately [7,8]. It is worth noting that, the data-driven models are divided into two groups. The first group tries to find mappings from multi-sensory data to the RUL. The second group however, as the proposed method, tries to transform the available high dimensional data into a one dimensional HI and calculate the RUL based on this new value. Moreover, the second group has

 doi:10.20944/preprints202109.0099.v1

shown better generalization and higher precision on the available public datasets [6]. Additionally, the possibility of gathering data from multiple sensors in Industry 4.0 environments, has also lead to the increased use of PdM [9]. It can be said that, once enough data from all parts of the studied process can be gathered, PdM can be employed. As a result, maintenance costs and downtime will be reduced, and the overall productivity will increase in an Industry 4.0 setting [10]. Nonetheless, one important obstacle is normally neglected in Industry 4.0 and Digital twin cases. Usually, no historical R2F dataset is available in industry [11] and it is very unlikely that one will find examples in the relevant literature that fit a current problem in detail. Furthermore, one of the main challenges of data-driven PdM is tackling with large, high dimensional datasets. Additionally, identifying CIs and extracting informative features from the available data can be tedious, when manual feature engineering is deployed. For addressing the above-mentioned problems, machine learning and deep learning approaches can be used [12–14]. There has been an extensive study on different machine learning and deep learning structures such as support vector machines, convolutional neural networks, long short-term memory (LSTM) and AEs in the domain of PdM [15–19]. Consequently, the problem with traditional machine learning approaches such as support vector machines is that these methods do not capture temporal dependencies in the available data and also neglect anomalies that are time dependent. Moreover, features derived from deep learning models tend to preserve useful information in the original data compared to the output of feature engineering required in machine learning models [20]. Therefore, for attaining better results from trained models, the extensive use of deep neural networks is vital [21]. However, it is worth noting that a deep feed forward neural network will not be effective either, given that this structure requires temporal dependencies in advance [22]. Furthermore, LSTM models are not sufficiently accurate when the output dimension is large [23,24]. On the other hand, convolutional neural networks have been proven to be effective for extracting features from raw data. Additionally, this structure was designed for high dimensional data of images. This characteristic of this type of neural network makes it perfect for feature extraction in PdM even with high dimensional data [25]. Likewise, AEs have been successfully implemented for feature extraction, dimensionality reduction, anomaly detection and time-series prediction [26–28]. Given the performance of AEs and convolutional neural networks in handling high dimensional data and extracting features, it has been decided to use these architectures in the proposed method. What is generally missing from conducted studies is a hybrid architecture that benefits from different capabilities of the available data-driven models for anomaly detection and using this information for calculating RUL given that no R2F data is available. The previous studies targeting PdM, normally consider a complete data set of R2F data [26]. However, some other are only limited to feature extraction and anomaly detection of their system in hand [29]. In the proposed method, an AE with convolutional layers is trained to find CIs and calculate the corresponding HI using a minimax optimized sigmoid function, and finally determine when maintenance is required using a GP as a degradation model. Essentially, the proposed method tries to cluster data based on the distribution of signal sequences of different tasks. The aforementioned signal sequences are gathered from a 3 dof delta robot, used for picking and placing parts in a smart factory. This architecture combines the semi-supervised training and feature extracting of AEs with the power of convolutional neural network for finding spatial information in the available signal sequences. Furthermore, reconstruction error in the proposed method is used to come up with a CI which is then passed to a minimax optimized sigmoid function to calculate the HI. Additionally, HI is interpreted as the probability of the current system failing given the formerly-mentioned CI. The HI can also be interpreted as an anomaly probability value where by $HI = 0$, the system is operating as expected [30]. It is worth noting that, the proposed HI is the complement of HI introduced in some literature [6]. After acquiring the HI, a.k.a. anomaly probability values, a GP as a degradation model is trained to predict the time-series of HI values. Based on the predicted values, it is possible to calculate the RUL of the studied system given different previously chosen thresholds [11,31]. In short, the main contributions of the proposed method are the following:

 doi:10.20944/preprints202109.0099.v1

1. Given the semi-supervised nature of the method, there is no need for failure data for training the model, which is vital when gathering such data can be dangerous or economically infeasible.
2. The proposed method does not require hand designed features for training the models, which makes the training stage easier when no or too little domain knowledge about the system is available or when a large, high dimensional dataset has to be tackled with.
3. The proposed method can also be used to classify the task and determine which task is going to fail, based on the similarity of distribution of the signal sequence.
4. By having the output of the proposed method, a probability can be assigned, which describes how probable it is that the system is going to fail. This probability includes a slack variable which determines how much deviation from reference values is allowed. Additionally, it is also possible to determine the sensitivity and rate of changes of the proposed method to deviation from the reference values or find optimal values by solving a minimax problem.
5. Given the studied system, the proposed method is capable of pinpointing where the problem originates from, regardless of the number of motors misbehaving. Moreover, the output of fault localization shows that the trained model has learned some correlations between different parameters of the studied system.
6. The trained GP does not require numerous data points to predict the anomaly values. Moreover, the dedicated GP for regression does not require a long time to be (re)trained. These characteristics make GP ideal for online prediction of anomaly values.

In essence, the hypothesis is that, by having enough data points from the delta robot's error free operating condition, it is possible to train a deep neural network, capable of determining the distribution of error free signal sequences of the studied system. Afterwards, based on the deviation from the determined distributions, it is possible to calculate a new CI. This new CI is then transformed into HI which describes the probability of having an anomaly. Afterwards, this anomaly value is predicted using GP as a degradation model. This prediction of this value can be used for calculating RUL of the studied system based on some given thresholds.

2. Material and Methods

In this section different Deep learning architectures used in the proposed method along with the dynamics and parameters of the delta robot are introduced.

2.1. Autoencoder (AE)

AEs try to replicate their input at their output through an internal representation. The aforementioned internal representation is in fact a bottleneck in a higher or lower dimensional space from which the decoded output is calculated. The only constraint is to have a sufficiently complex space which is able to learn the distribution of the provided data. AEs are made of the following parts:

- encoder f
- internal representation z
- decoder g

Given an input vector x, the encoder part of AE, which can be a single or a multi-layer structure, finds an internal representation of the provided input. The internal representation is in fact the latent space, in a higher or lower dimensional space, capable of representing the provided input [32]. Therefore, it can be claimed that

$$z = f(x) \tag{1}$$

Afterwards, by having the internal representation, the decoder part of the AE, tries to estimate the formerly given input, denoted as $\hat{x}$, based on given z. The aforementioned can be formulated as

$$\hat{x} = g(z) = g(f(x)) \tag{2}$$

 doi:10.20944/preprints202109.0099.v1

The reconstruction error is the difference between the input and the output of the AE [33]. In the conducted study, the loss function of the neural network is the mean squared error (MSE).

Generality of the trained model, plays a significant role in the performance of the proposed method. By having AEs which merely copy the input at their output, no insight is provided whether a signal sequence is error free or if maintenance is required. In respect to AEs, for increasing the generality of the trained model, it has been decided to use deep AEs [34]. Additionally by using convolutional layers, instead of having deep dense neural networks, locality of the features are also taken into account for finding the underlying distribution of error free signal sequence.

In this paper, the predictive maintenance of a delta robot without any R2F data is studied. Moreover, the delta robot is used to perform pick and place of barrels and springs in a pen production line. The barrel and spring of a pen are sent to the station where the delta robot is located, with a tray containing different parts of the customized pen. More details about the delta robot are provided in *Section 2.3*. In the proposed method, two datasets from barrel and spring pick and place movement are gathered. These two distributions of data describe a error free signal sequence from 11 parameter estimations of the delta robot. It can be claimed that if the attained signal sequence from the delta robot whilst performing either of these tasks does not match these two distributions, there exists anomaly in the data. Thus, there is a problem with delta robot and maintenance is required. The gathered data from the delta robot are 11 time series from different parameters. Therefore, it is not efficient to use dense neural network in encoding and decoding parts of the AE to learn the underlying distribution of the provided data. In order to address the aforementioned problems, convolutional layers are used in the structure of the AEs.

2.2. Convolutional layer

Given the sequential nature of the gathered data, it is decided to used a convolutional layer to capture spatial information and relation in the gathered data. The aforementioned sensor signal sequences are concatenated and a new dataset is acquired from these separate senor signals. The acquired features from the convolutional layer are calculated as follows. First, several convolutions are performed to compute linear activations as

$$S(i,j) = (X * K)(i,j) = \sum_m \sum_n X(m,n)K(i-m, j-n) \tag{3}$$

where X is the concatenated signal sequences and K is the two-dimensional kernel. Afterwards, these linear activations are passed through nonlinear activations functions (detector stage). Finally, a pooling function can be applied. It is decided not to use pooling, since it will complicate the neural network in AEs. It can be claimed that the convolutional layer captures all the spatial information required in the new dataset [33], [35]. Afterwards, the AEs find a representation for these so-called error free data points and finally the distribution of these datasets are acquired. Based on these data distributions, a new CI is calculated. Moreover, the aforementioned CI in fact contains information about the distance to data distributions of the error free working conditions of the delta robot [30]. Thereafter, a probability is assigned to any new signal sequence which describes how probable it is that the delta robot is having a problem given the formerly-mentioned CIs.

2.3. Delta robot dynamics and parameters

The studied 3 dof delta robot is a fully parallel robot with closed loop mechanism. This robot is used for fast and accurate pick and place task. In the conducted study, the delta robot is part of a learning factory, which assembles customized pens according to the requests of the user . The delta robot station is depicted in Figure 1.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

Figure 1. Delta robot as a part of the smart factory.

By using three parallelograms the moving platform remains at a fixed orientation with respect to the robot base. The movement of the Tool Center Point (TCP) is achieved by the three direct drive motors. The links only transmit force along TCP and are made of carbon fiber tubes. Given the fact that dynamic equation of the studied system is linear in its dynamic parameters, the parameters of the system can be acquired to determine whether the system is operating error free or not. This can be done using the following equation

$$\tau = \Psi p \tag{4}$$

where p is the dynamic parameter of the system, Ψ is the matrix with dynamic equations of the system and τ is the applied torque. Essentially, the implemented adaptive feed forward controller (AFFC) in the 3 dof delta robot, learns the parameters of the system during motion. The block diagram of AFFC can be seen in Figure 2.

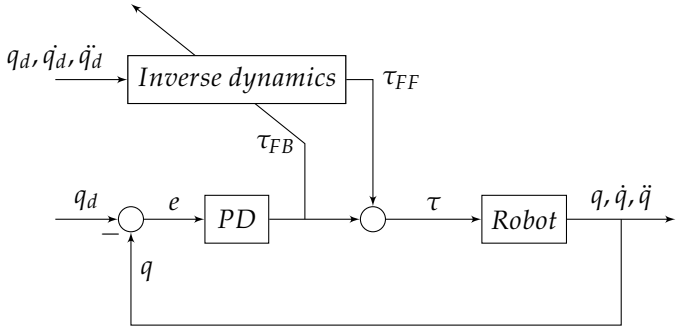

Figure 2. AFFC block diagram

Since there are some simplifications in the attained model, in order to improve the performance of the system, the unmodeled dynamics is compensated with a feedback controller. The control law of the AFFC is calculated as

$$\tau = \Psi(q_d, \dot{q}_d, \ddot{q}_d)\ p + \tau_{FB} \tag{5}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

Parameter	Learning rate
Motor inertia $[gm^2]$	0.0005
Gravity $[mN]$	0.08
Mass $[g]$	0.09
Spring offset $[mNm]$	0.07
Coulomb friction 0 $[mNm]$	0.1
Coulomb friction 1 $[mNm]$	0.1
Coulomb friction 2 $[mNm]$	0.1
Spring constant $[mNm/rad]$	0.2
Viscose friction 0 $[mNms/rad]$	0.01
Viscose friction 1 $[mNms/rad]$	0.01
Viscose friction 2 $[mNms/rad]$	0.01

Table 1. Parameters and the corresponding learning rates

where q_d is the desired joint angle and τ_{FB} is the torque calculated from the feedback controller, formulated as

$$\tau_{FB} = K_p\, e + K_d\, \dot{e} \tag{6}$$

The calculated error e is the difference between desired and actual joint positions. The terms K_p and K_d are the proportional and derivative gains of the feedback controller respectively. Whilst the delta robot is moving, the dynamic parameters of the system (p) are learned by the AFFC using the tracking error function. The aforementioned tracking error function is calculated as follows

$$E = (\tau - \Psi p)^2 \tag{7}$$

The implemented learning algorithm is gradient descent and results in the following

$$p_{new} = p_{old} + \alpha\, \Psi^T \tau_{FB} \tag{8}$$

where α is a positive definite matrix which determines the learning rates for different parameters. It is worth noting that the aforementioned values only are estimation of the real values [36–38]. The aforementioned parameters and the corresponding learning rates in the studied system are shown in table 1.

2.4. Gathering and preprocessing data from delta robot

In order to read data from the delta robot and also to send commands to it, a local area connection is established with the system and the corresponding HTML page containing all the data in XML format is scraped. Beautiful Soup package from Python is used to scrape the aforementioned page.

After gathering datasets for different tasks of the delta robot, the first step is to normalize the data to remove the effects of scaling in the data. Data normalization can be calculated as follows

$$\tilde{X} = \frac{X - \mu}{\sigma} \tag{9}$$

where μ and σ are the mean and the variance of the gathered data, and $\tilde{X}$ is the normalized data, calculated from the original data points X [39]. Another step before feeding the data to the 2d convolutional layer of the proposed method is augmenting another dimension to the data which is similar to different channels of an image.

After preprocessing the gathered data, the next step is to create a training dataset for the proposed method. Given the dynamics of the delta robot in performing different tasks, a window of size 5 data points with stride of 1 is used to create the training dataset. It is worth noting that once the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

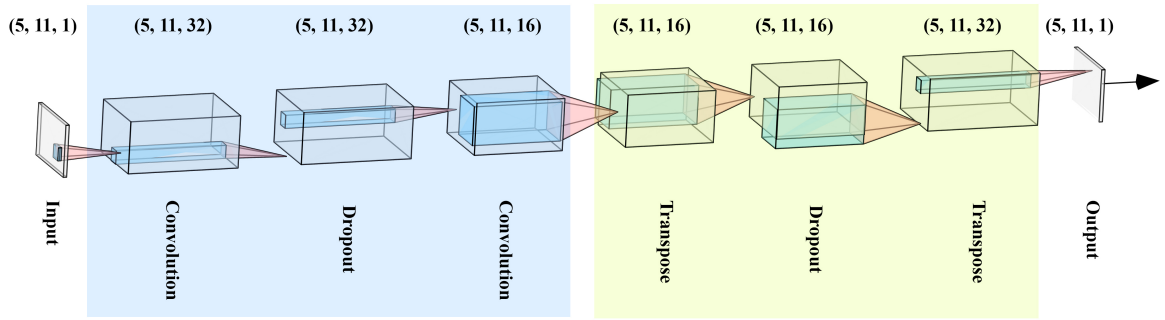

Figure 3. Architecture of the proposed method

sampling frequency increases, the size of the aforementioned window has to increase to capture the same dynamics from the data.

2.5. Architecture of the proposed method

The architecture of the proposed method can be seen in Figure 3 [40]. Given the fact that, signal sequences from delta robot sensors can be concatenated, it has been decided to use convolutional layers to capture the spatial information in the time series. By using the AE there is no need to do manual feature engineering and the deep neural network will learn a representation of the distribution of the error free signal sequences. By replicating the input at the output of the AE, the reconstruction error is used to determine whether a signal sequence is error free. In the conducted study, the AE is made of the following layers:

- Convolution with 32 filters and kernel size of 4×4
- Dropout
- Convolution with 16 filters and kernel size of 12×12
- Transpose (a.k.a deconvolution) with 16 filters and kernel size of 12×12
- Dropout
- Transpose with 32 filters and kernel size of 4×4

It is worth noting that, the activation function in the convolutional layers is Rectified Linear Unit (ReLU) calculated as follows

$$ReLU(x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

The formerly-mentioned activation function can address the vanishing gradient problem, improve the learning speed and lastly, result in positive response of neurons to representations from the features [41–43]. It is worth noting that, the selected loss function for the neural network is the MSE.

Based on the dynamics of the system and the sampling rate, the convolution window and the depth of the neural network can be tampered with to come up with the optimal solution for a given system. In the conducted study, two AEs were trained: one for the spring pick and place movement and one for barrel pick and place movement. Based on the reconstruction error the movement can be identified, and it can also be determined whether the delta robot is having a problem and maintenance is required.

2.6. GP

In the conducted study, GP is used for performing regression. GPs are non-parametric models which try to assign higher probabilities to functions which have closer values to the available

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

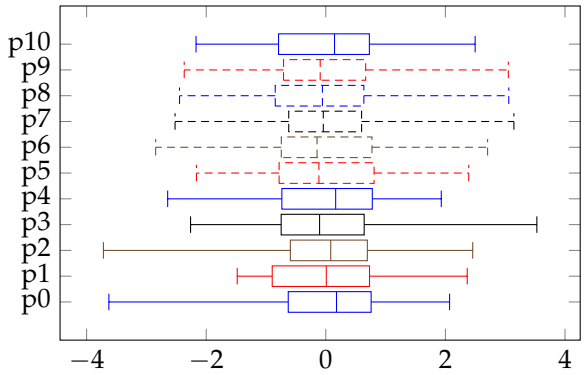

Figure 4. Statistical information of normalized sensor readings of barrel movement

data points or in other words to the functions which fit the dataset better. GPs are in fact an infinite-dimensional multivariate distribution where every selection of random variables has a multivariate Gaussian distribution. Formally, GPs are denoted as $GP(m(x), k(x, x^{'}))$, where $m(x)$ is the mean function and $k(x, x^{'})$ is the covariance function. When x and $x^{'}$, according to the kernel function, have similar values, the output of the model at these two points will also be similar. Moreover, a collection of random variables $\{h(x) : x \in X\}$ is drawn from GP with mean function m and covariance function k if the following for $x_1, x_2, ..., x_n \in X$ holds

$$\begin{bmatrix} \mathbf{h(x_1)} \\ \vdots \\ \mathbf{h(x_n)} \end{bmatrix} \sim N\left\{ \begin{bmatrix} \mathbf{m(x_1)} \\ \vdots \\ \mathbf{m(x_n)} \end{bmatrix}, \begin{bmatrix} \mathbf{k(x_1, x_1)} & \cdots & \mathbf{k(x_1, x_n)} \\ \vdots & \ddots & \vdots \\ \mathbf{k(x_n, x_1)} & \cdots & \mathbf{k(x_n, x_n)} \end{bmatrix} \right\} \tag{11}$$

Once a prior is assigned to the parameters of the model w, the posterior can be calculated after observing the available data as follows

$$p(w|y, X) = \frac{p(y|X, w)p(w)}{p(y|X)} \tag{12}$$

where $p(y|X, w)$ is the likelihood, $p(w)$ is the prior and p(y | X) is the marginal likelihood. Thereafter, for a new data point x^*, the output of the model can be calculated as shown below

$$p(f^*|x^*, y, X) = \int_w p(f^*, x^*, w)p(w|y, X)dw \tag{13}$$

3. Results

In this section the simulation results are presented for continuous and binary implementation of the proposed method.

3.1. Setting up the model

The delta robot performed spring and barrel pick and place for 900 seconds, and it was sampled every 100 milliseconds. As an example, the statistical description of the gathered dataset of barrel movement after preprocessing is shown in Figure 4.

The aforementioned datasets are fed to the two AEs for barrel and spring pick and place movement. For demonstration purposes, the learning curves of the deep neural net for the spring movement is shown in Figure 5.

After training AEs, the training dataset is fed to the models and the mean absolute error (MAE) for data points is acquired. Moreover, the maximum MAE is used as the threshold for classifying

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

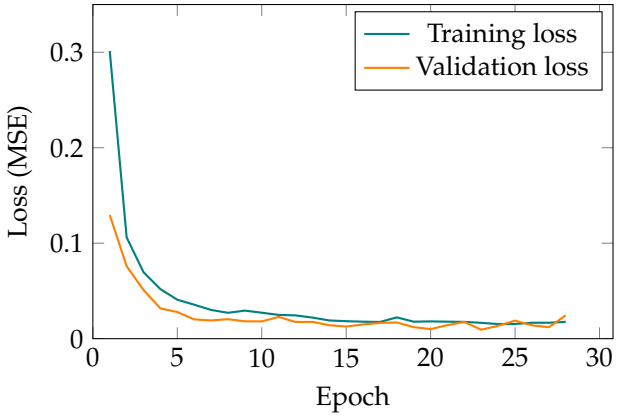

Figure 5. Learning curve of the AE for the spring movement

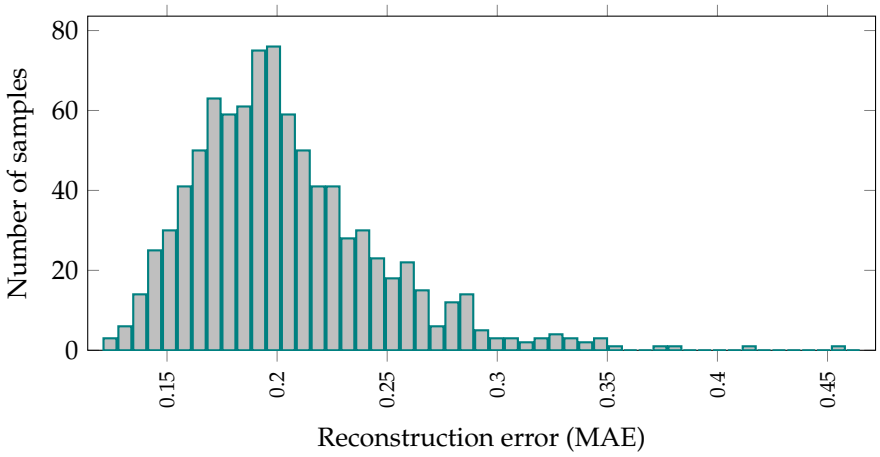

Figure 6. Distribution of reconstruction error given the normalized input data

a signal sequence as anomaly in binary classification case. The simulation results for sample MAE reconstruction errors can be seen in Figure 6.

As pointed out earlier, the maximum reconstruction error is used as a threshold in a binary classification model. In fact, when the reconstruction error is less than this threshold, the given signal sequence will be classified as error free. It is also possible to come up with a continuous value which describes the probability that a signal sequence is not error free given the reconstruction error as a CI. As an example, a sigmoid function can be used to map the aforementioned CI to a probability of anomaly which is then used as the HI of the delta robot. The aforementioned sigmoid function is calculated as follows

$$\phi(d) = \frac{1}{1 + e^{-c_1 \times (d - c_2)}} \tag{14}$$

where d is reconstruction error, a.k.a. the CI, which contains information about the deviation from the error free working condition data distribution of the delta robot. Furthermore, c_1 is the slope of the sigmoid function and finally, c_2 determines the point where the sigmoid function is equal to 0.5. Based on the sensitivity of the system, the parameters c_1 and c_2 can be chosen. In the conducted study, the threshold for binary classification of anomaly in the barrel and spring movement are 0.46 and 0.30 respectively.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

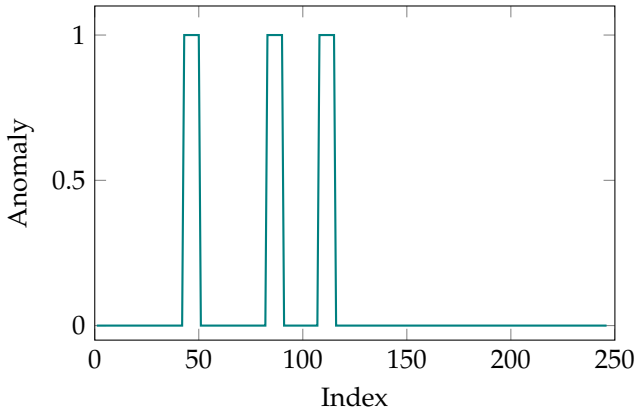

Figure 7. Indices of detected anomalies

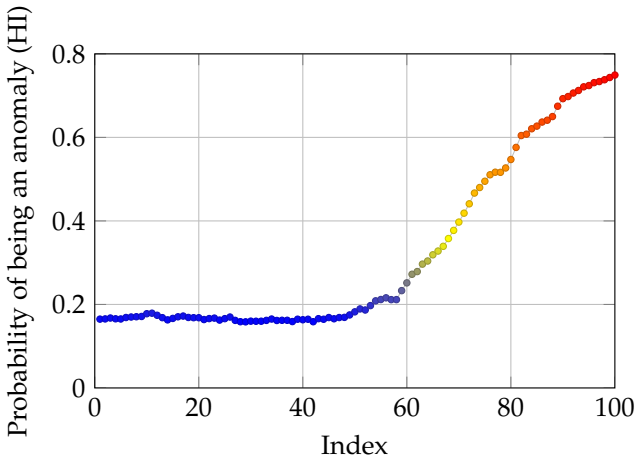

Figure 8. Probability of a signal sequence being an anomaly given the CI

3.2. Binary classification and predicting anomaly probability

After training the AEs, a sample dataset is used to test the proposed method. Three sequences of impaired data is created by adding the corresponding signal sequences in the error free dataset an attenuated noise (maximum amplitude of the noise is 0.1). As an example, an impaired signal sequence of length 4 is created at 45, 85 and 110 indices from the barrel movement dataset. Afterwards, the sample dataset is fed to the AE trained for the barrel movement. It can be seen in Figure 7 that the AE has successfully found the impaired signal sequences. It is worth noting that the binary classification was used in this simulation. Hence, the data points are either denoted as error free or impaired which are equivalent to 0 and 1 respectively. Moreover, smoother HI values for another anomalous signal sequence from the CI using a sigmoid function with values $c_1 = 1.4$ and $c_2 = 1.6$ are depicted in Figure 8.

It is worth noting that peak value and also the slope can be tuned to get a more case specific output. For example, when the studied system is very susceptible to changes, then a steeper slope and a lower threshold can be used. However, when no domain knowledge about the system is available it is best to use the optimization method introduced in *Section* 3.5. The pseudo-code of the proposed method can be seen in Figure 9.

3.3. Robustness testing

Lastly, to test the robustness of the proposed method, another dataset is acquired from the parameters' matrix p while its values are converging. As it was described in *Section* 2.3, the estimated

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

Obtain data from test runs
Compute new dataset by preprocessing and augmenting
Train AEs for different tasks
while *System operating* **do**
  Read signal from system
  Compute reconstruction error (CI) from AEs
  if *any CI is below threshold* **then**
    Display error free system operation
    Display task
  else
    Display anomaly detected
    MAINTENANCE REQUIRED
  end
end

Figure 9. Pseudo-code of the binary classification

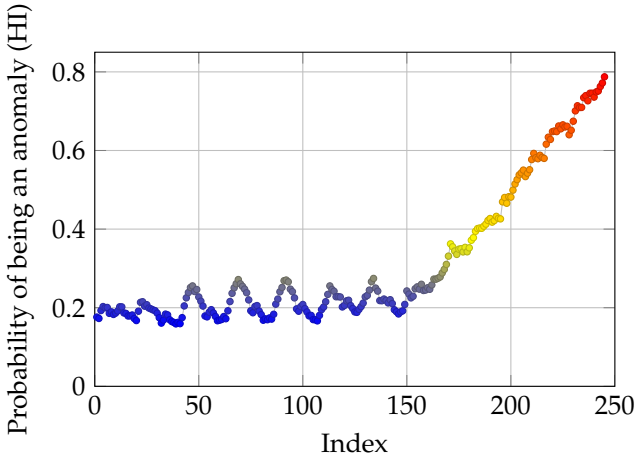

Figure 10. Results of simulation from incremental disturbance

parameters are updated using the tracking error. Therefore, at the early stages when the system has just started working, the values of parameters are not good estimates of the parameters. The aforementioned parameters are converging to the actual value with a predefined learning rate α. The spring AE is trained on this low quality dataset. Afterwards, another dataset which contains converged values of parameters is tampered with. The length of the test time series is 245, and from the 150^{th} data point, the corresponding values of the three Coulomb frictions are added with three separate disturbances which increases steadily from 0 until they reach 60% of the maximum value of corresponding friction values. Subsequently, after attaining the new test dataset, this dataset is fed to the spring AE. The simulation results can be seen in Figure 10.

It is worth noting that corresponding c_1 and c_2 values are 1.8 and 1.9 respectively. It can be seen that, as the friction values diverge from their normal values (error free), the AE produces CIs which further deviate from the normal values which results in higher HI values. From a PdM point of view, a threshold can be chosen based on the sensitivity of the system and maintenance can be performed based on the calculated HI. Simply, by changing the c_1 parameter of the sigmoid function, the rate of increase or decrease in the returned probability values can be tuned. On the other hand, by tampering with c_2, it can be decided which values of attained anomaly probability have to be ignored. In fact, c_2 is like a slack variable which allows divergence from the reference values. Overall, it can be claimed that, these two parameters can be fine-tuned according to the studied system and suitable models for PdM can be acquired accordingly. Nonetheless, in the following sections a *minimax* optimization based method will also be introduced which provides optimal values for c_1 and c_2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

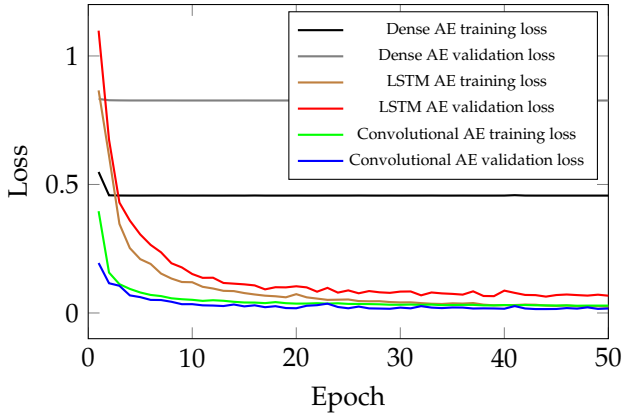

Figure 11. Learning curve of the different AEs for the barrel movement

3.4. Comparison of the convolutional, LSTM and dense AE

As it was discussed in the introduction, the convolutional layer in the proposed method, significantly enhances the model. Namely, the captured spatial information in the attained signal sequences, and also the acquired features improve the performance of the AE for PdM. The impact of the structure of AE on its performance is studied by training three different AEs:

- AE with dense feed-forward layers: This architecture has the lowest performance and simply is not capable of handling sequential data. Even by increasing the model complexity, this architecture does not show any improvement in the performance. The number of trainable parameters in this case is 475, 200. Moreover, the minimum validation loss for this structure is 0.8266.
- AE with LSTM layers: This model has a much better performance than the previous scenario. However, this model cannot outperform the results of the AE with convolutional layers. This simulation results proves that as expected, this structure is capable of handling sequential data, but not as good as the AE with convolutional layers. In this case, the number of trainable parameters is 254, 347. It is worth noting that, the minimum validation loss for AE with LSTM layers is 0.0637.
- AE with convolutional layers: The best performing amongst the test models with the fewest parameters. The proposed method has successfully found all the spatial information and acquired features for reconstructing the given signal sequences. The number of parameters in this case is 125, 665. Lastly, the minimum validation loss for the chosen AE structure is 0.0281

The learning curve of the aforementioned models is depicted in Figure 11.

As shown, there is quite a gap between validation loss of the different methods, as an indicator of generalization power of different architectures.

3.5. Optimization of sigmoid function

In this part, in order to improve the interpretability of the output of the model, the sigmoid function which determines the probability of being an anomaly (HI) is optimized. It is worth noting that this optimization is performed on the test run data which contains no anomaly in different signal sequences. The main idea is to find minimum values of c_1 and c_2 whilst keeping the output sensitive to deviations by maximizing the sigmoid cost function. In other words, a *minimax* optimization is performed on the sigmoid function, mapping deviation from references, a.k.a reconstruction errors, to probability of having a problem in the studied system. The optimization problem is formulated as follows

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

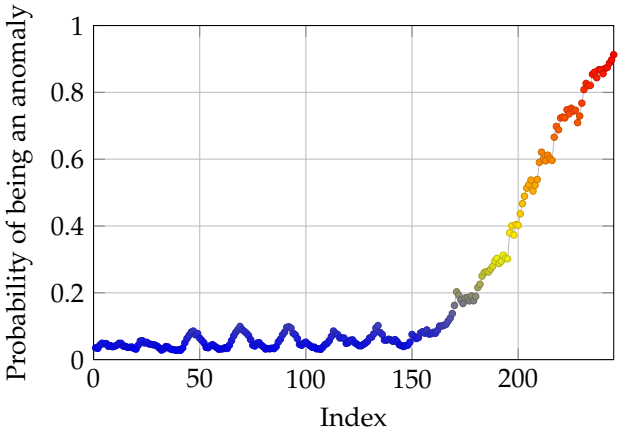

Figure 12. Results of simulation from optimized sigmoid function

$$
\begin{aligned}
C^* &= \min_{c_1,c_2} \max_i f_i,\ i \in 1,2 \\
f_1 &= \sum_d^D \frac{1}{1+e^{-c_1(d-c_2)}} \\
f_2 &= std(\frac{1}{1+e^{-c_1(D-c_2)}})
\end{aligned}
\tag{15}
$$

where C^* is the optimal parameters c_1 and c_2, given the aforementioned optimization problem and initial values of c_1 and c_2. Additionally, D is the vector of calculated d for all the data points in the test run signal sequences. Furthermore, std is the standard deviation of applied sigmoid function on D, with c_1 and c_2 parameters. The maximization part of the above optimization is for ensuring that the output of the optimization problem is not a flat signal of zero with large values of c_1 and c_2. Moreover, the minimization forces the c_1 and c_2 parameters to converge to values which make the sigmoid function have lower bias and a gentle slope. The results of optimization is depicted in Figure 12.

A ramp disturbance is added to the available spring movement from the 150^{th} data point. The data used in this part of the study are the same as the previous section, as shown in Figure 10, where the robustness of the model was tested. As it can be seen in Figure 12, the optimized probability function, tries to keep the probability of anomaly for the first 150 data points zero, and just as the disturbance and deviation is added to the original data, the probability of anomaly increases steadily. By requiring more variance in the optimization problem, a more informative mapping from deviation to anomaly probability is acquired. Moreover, c_1 and c_2 values are 3.56 and 1.96 respectively.

3.6. Fault localization

After finding out the fact that, there is anomaly in the studied system and therefore, a need for maintenance, the next task is fault localization. For doing so, after classifying the current task through the signal sequence distribution with AEs, the contribution of different reconstruction errors is analyzed to determine where the problem originates from. In the conducted study, there are three motors for moving the TCP of the delta robot and the goal is to find out which of these motors is not operating well. Different scenarios are studied to show the capability of the proposed method for fault localization. In order to do so, different ramp disturbances are added to Coulomb and Viscose friction of the motors in the spring movement dataset from the 150^{th} data point. The final value of these disturbances is 60% of the maximum value of the friction values. The CI for different motors is determined by the sum of reconstruction errors in Coulomb and Viscose friction passed through the sigmoid function. As it can be seen in Figure 13, Figure 14 and Figure 15, the model is able to pinpoint

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

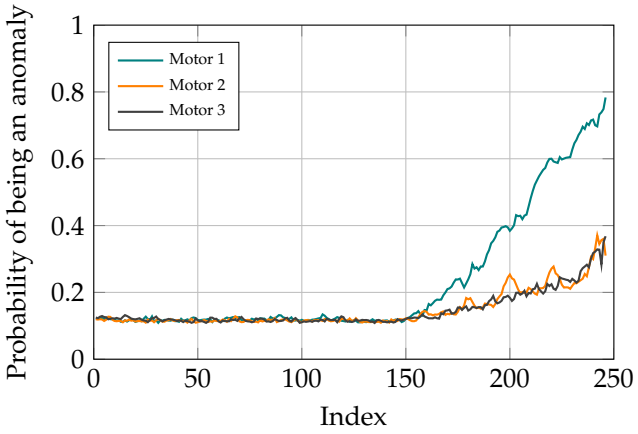

Figure 13. Simulation results with distortion added to motor 1

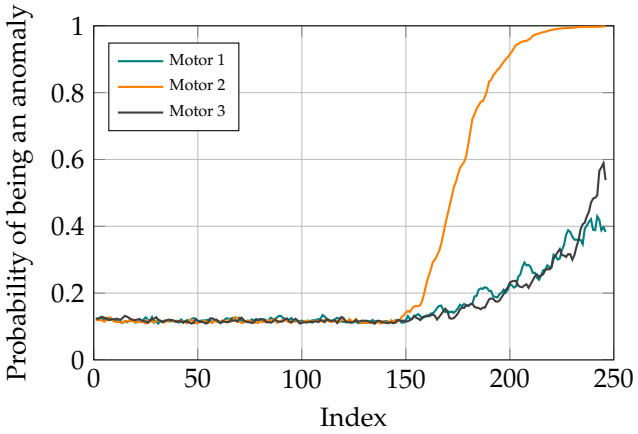

Figure 14. Simulation results with distortion added to motor 2

where the problem derives from. Noticeably, the anomaly probability increases steadily from the 150^{th} data point for all the scenarios.

Similarly, when two or more friction values are manipulated, the model is able to determine the fault in the system. Moreover, the anomaly probabilities higher than a predetermined threshold depict which motors are having problem and need maintenance. Two examples are shown in Figure 16 and Figure 17, where the friction values of the first and third motor and all the motors are manipulated respectively.

The probability of being an anomaly given the CI, is calculated similar to the previous sections, with the c_1 and c_2 values of 1.8 and 1.3 respectively. Given the dynamic of parallel robots, when there is a disturbance in the friction values of a motors, it can be seen that all the reconstruction errors in different motors increase. This shows that, the trained model has learned the correlation between different parameter values and thus, is not purely an identity function which tries to replicate the input at its output. This characteristic of the model can significantly help increase the interpretability of the trained model.

3.7. RUL

After acquiring a HI which determines the probability of having an anomaly given the CI, the next and last step is to calculate RUL. In fact, by using the AEs, raw parameter readings from the delta robot are converted to new CIs. Afterwards, by calculating the deviation from reference values for the aforementioned CIs, the probability of having an anomaly given CI is calculated as the HI of

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

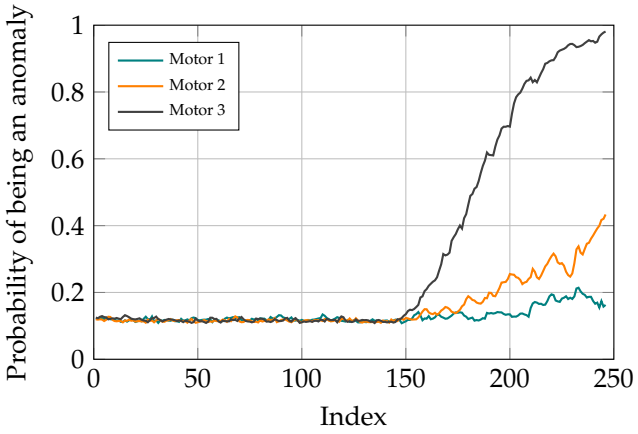

Figure 15. Simulation results with distortion added to motor 3

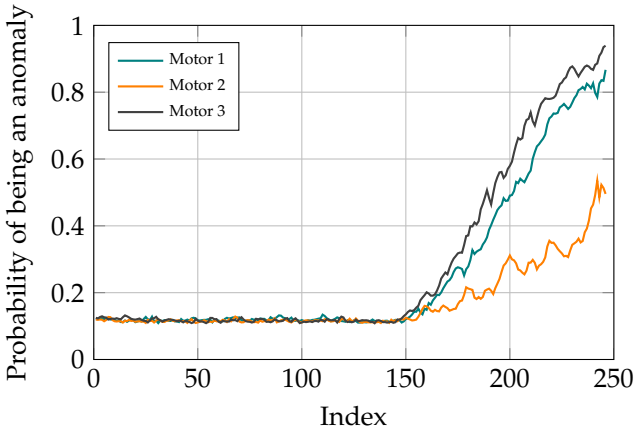

Figure 16. Simulation results with distortion added to motors 1 and 3

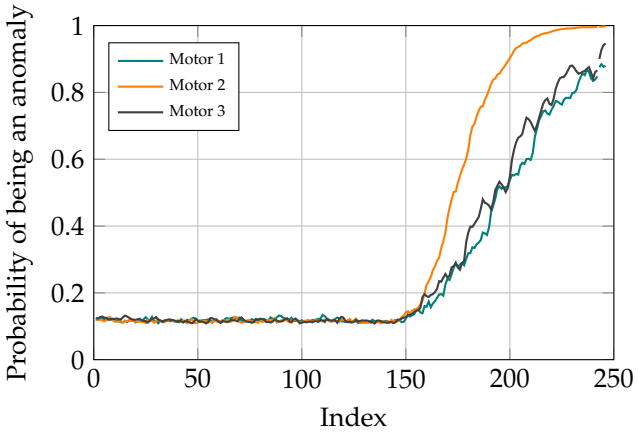

Figure 17. Simulation results with distortion added to motors 1, 2 and 3

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

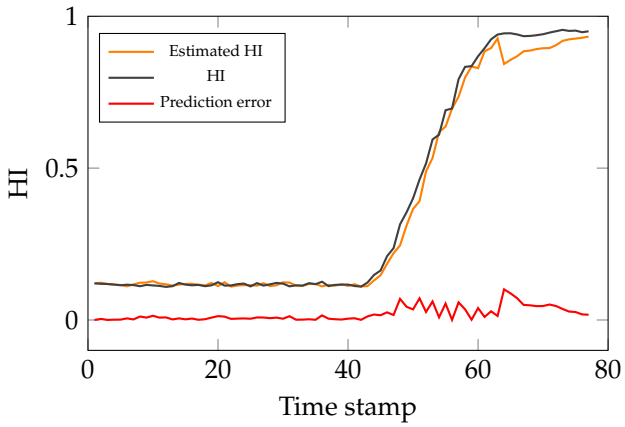

Figure 18. Simulation results for HI estimation using the degradation model

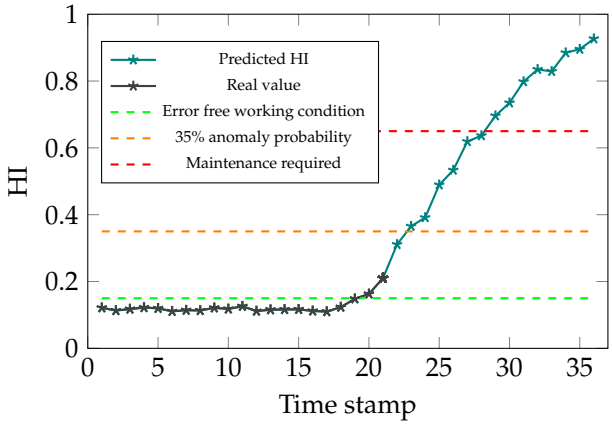

Figure 19. Simulation results for RUL estimation

the studied system. Moreover, in the last step, this value is predicted to estimate RUL of the studied system. Additionally, given the fact that no R2F data from the system is available, threshold values for the anomaly probability values can be used to predict RUL. In fact, by having the aforementioned thresholds, RUL is calculated as the difference between the current time stamp and the time stamp when the HI is higher than the threshold. Furthermore, for calculating the HI in future time stamps, a GP for modelling degradation is trained to predict the anomaly probability values. The input for the GP are 20 past HI data points and afterwards, 15 future HI data points are predicted. It is worth noting that the kernel is designed only once and the GP is (re)trained once the new data points are available. Moreover, for acquiring better results and also reducing calculations, it is recommended to use an appropriate stride value for retraining the GP. In the conducted study, the CIs are used to train the GP with a stride of 6. In this case, in every step the GP required on average 0.02936 seconds to be retrained. The results of the conducted study for the degradation model and RUL calculation can be seen in Figure 18 and 19.

As shown in Figure 18, the GP is capable of predicting the future HI values of the system. It should be noted that the aforementioned figure contains plots of the HI of the system though time and also the predicted value from the degradation model. The GP regressor has a MSE of 0.00092. Moreover, the explained variance score of the GP is 99.41% which is calculated as follows:

$$explained\ variance(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}} \tag{16}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0099.v1

where y is the HI of the system, $\hat{y}$ is the predicted value using the GP and Var is the variance of the formerly-mentioned values. As it can been seen in Figure 19, the probability of having an anomaly, has steadily increased from the 18^{th} time stamp. Additionally, the predictions from the degradation model suggest that in 8 steps the system will reach a critical condition and therefore, maintenance is required prior to the 29^{th} time stamp. Thus, it can be claimed that the RUL of the system is 7 time stamps given the assigned threshold, which is anomaly probability value of 65%, and the applied degradation model. It is worth noting that, given the stride value of the GP, one time stamp is equivalent to 6 sampling times for reading the parameters from the system. Moreover, the used kernel for the GP is made of the following parts:

1. Exp-Sine-Squared kernel
$$k(x, x^{'}) = exp(-\frac{2sin^2(\pi d(x, x^{'})/p)}{l^2}) \tag{17}$$
2. Radial basis function
$$k(x, x^{'}) = exp(-\frac{d(x, x^{'})^2}{2l^2}) \tag{18}$$
3. Matern
$$k(x, x^{'}) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\sqrt{2\nu}\frac{d(x, x^{'})}{l}\right)^{\nu} K_{\nu}\left(\sqrt{2\nu}\frac{d(x, x^{'})}{l}\right) \tag{19}$$
4. Constant kernel
$$k(x, x^{'}) = constant\ value\ \forall x, x^{'} \in X \tag{20}$$

where $d(x, x^{'})$ is the Euclidean distance between two data points x and $x^{'}$. Additionally, p is the periodicity parameter, l is the length scale, ν controls the smoothness of the kernel, K_{ν} is a modified Bessel function and finally, Γ is the gamma function.

4. Discussion

In this paper, a semi-supervised hybrid deep neuron network structure was introduced for finding anomaly in signal sequences acquired from sensor readings of a delta robot for PdM. The delta robot performed pick and place of barrels and springs and was part of a smart factory for assembling customized pens. The main notion of the proposed method is to find anomaly in the data sequence and assign an anomaly probability accordingly. The proposed structure did not require any R2F signal sequence samples of the system for different tasks. The only requisite of this method was to have enough data from different tasks for the AEs to find the underlying distribution of the error free signal sequences. This characteristic is vital when there are not enough samples of failure available due to high costs or the danger it causes. In fact, as the system gets more complicated, it is not possible to claim that all the different signal sequences leading to different failures are captured and stored. Therefore, it is better to predict error free working conditions of the system and afterwards, flag the signal sequences which do not correspond to the error free data sequences as anomaly. In every step, the AEs were given the acquired signals from sensor readings and, they tried to replicate these signals at their output. In addition, in the binary classification case, if any of these AEs had a reconstruction error less than the previously assigned threshold, the provided signal sequence would be classified as error free. It was also possible to determine the task, given the fact that only the AE, trained on that specific task, would have a reconstruction error less than the threshold. In fact, the reconstruction error was used as a new CI for the studied system, which was afterwards used to calculate the HI of the system. In addition, the simulation results showed that the trained AEs had successfully learned the distribution of spring and barrel pick and place movement. Moreover, when the error free signal sequence was disturbed with noise, the AEs were able to find the indices of data which were tampered with. Additionally, by using a sigmoid function and assigning appropriate parameters with regard to the studied system, a continuous value, as an anomaly probability value, defined as the HI of the system, could be acquired. The HI then could be used to determine whether

the system required maintenance given the thresholds. Furthermore, the aforementioned sigmoid function could be optimized by solving a minimax problem, and the acquired parameters were in fact the saddle point of this optimization problem. Additionally, the aforementioned approach was effective when not enough domain knowledge about the system was available. It is worth noting that this function, could also be manually tuned based on the sensitivity of the studied system and the needs of the user. In addition, it was also shown that the most effective architecture based on AEs for performing PdM was a combination of convolutional layers for extracting features from the available high dimensional raw data and afterwards, finding the signal sequence of different tasks accordingly. Thereafter, the trained deep neural networks were used for fault localization. The results of this part of the study showed that the trained model was capable of pinpointing where the problem originated from, assuming that distortions originated from the internal parts of the system. Moreover, when two or more motors were misbehaving in the delta robot, the proposed method was able to distinguish which motors were not working well. Additionally, the anomaly signal sequences in fault localization depicted the fact that, the trained model has learned some correlation between signal sequences. Moreover, the results suggested that the trained AEs were not merely an identity function trying to replicate their input at the output. Lastly, the recently calculated HIs were used to predict the RUL, using a GP regressor as a degradation model. In fact, the aforementioned GP was retrained after acquiring new parameter readings from the delta robot. Moreover, once new parameter readings were available, the corresponding CIs were calculated and finally the HI of the system given the CIs were acquired. Thereafter, the new sequence of HIs were sent to the degradation model to be retrained. Afterwards, the degradation model predicted the future HI values of the system. As an example, in the conducted studies 15 future time stamps were predicted by having the last 20 HI values of the system. Subsequently, it was possible to use a threshold data to assign a time in future for performing PdM given the predictions of the degradation model. It is vital to remember that no R2F data was available in the conducted study and thus the formerly explained approach was used for calculating HI and RUL of the system. It is worth noting that, this approach is most likely to work on systems similar to the delta robot introduced in this paper. Whereas, for more complicated systems, which show highly nonlinear behavior in their degradation value patterns, this model is most likely to fail. In such cases, it is important to have enough R2F data to compensate for the complexity of the degradation model of such systems. In addition, the proposed architecture is superior to analytical models given the following reasons. By using autoencoders, a movement dependent anomaly detection model can be trained. In addition, the proposed model relies merely on the read data from the system and thus can be fine-tuned given new parameter readings. Moreover, the stochastic nature of the architecture helps to model the uncertainties in the real system and the acquired data. In fact, by employing soft computing, it is possible to account for both small deviations from the acceptable parameter readings and also deviations that will lead to an anomaly. Furthermore, unlike the proposed method, for an accurate estimation of RUL, a very precise analytical model is required, which can be hard to obtain as the system gets more complex. However, it can be claimed that by employing an analytical model along with an AE architecture, a relatively simpler analytical model can be used for estimating the RUL. Moreover, by using an analytical model, a considerable amount of domain knowledge is contained in the analytical model. Thereafter, the AE can be used to determent deviations from the error free parameter sequences given the available domain knowledge from the analytical model. In short, given a complex system, a combination of an analytical model and the AE architecture can determine the RUL of the system. Whereas, for a simpler system, similar to the studied delta robot, a combination of AE and GP can determine the RUL. Moreover, as the HI deteriorates, the impact of the precision of the analytical model decreases, due to the fact that, at extreme the HI is not far from point of failure. Lastly, in the case of minor changes in the physical system, the proposed method only has to be retrained given the new data acquired from the system and no further updates in the remaining parts of the proposed model is required.

Author Contributions: Conceptualization, K.F., H.W.v.d.V. and M.H.; methodology, K.F., H.W.v.d.V. and M.H.; software, K.F.; validation, K.F., H.W.v.d.V. and M.H.; formal analysis, H.W.v.d.V. and M.H.; investigation, K.F.; resources, K.F., H.W.v.d.V. and M.H.; data curation, K. F. and M.H.; writing—original draft preparation, K.F.; writing—review and editing, K.F., H.W.v.d.V. and M.H.; visualization, K.F.; supervision, H.W.v.d.V. and M.H.; project administration, H.W.v.d.V. and M.H. All authors have read and agreed to the published version of the manuscript.

Funding: "This research received no external funding"

Conflicts of Interest: "The authors declare no conflict of interest."

Abbreviations

The following abbreviations are used in this manuscript:

PdM	Predictive maintenance
R2F	Run-to-failure
CI	Condition indicator
HI	Health index
RUL	Remaining useful lifetime
AE	Autoencoder
GP	Gaussian process
LSTM	Long short-term memory
MSE	Mean squared error
AFFC	Adaptive feed forward controller
MAE	Mean absolute error

References

1. Levitt, Joel, *Complete guide to preventive and predictive maintenance*, Industrial Press Inc., 2003.
2. Goyal, Deepam and Pabla, BS, Condition based maintenance of machine tools—A review, *IRP Journal of Manufacturing Science and Technology*, Elsevier, **2015**, 24–35.
3. Lughofer, Edwin and Sayed-Mouchaweh, Moamar, *Predictive maintenance in dynamic systems: advanced methods, decision support tools and real-world applications*, Springer, 2019.
4. Jack C.P. Cheng, Weiwei Chen, Keyu Chen, Qian Wang, Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms, *Automation in Construction*, **2020**, *112*, 87–103.
5. Baptista, Marcia and Sankararaman, Shankar and de Medeiros, Ivo P and Nascimento Jr, Cairo and Prendinger, Helmut and Henriques, Elsa MP, Forecasting fault events for predictive maintenance using data-driven techniques and ARMA modeling, *Computers & Industrial Engineering, Elsevier*, **2018**, *115*, 41–53.
6. Yu, Wennian and Kim, II Yong and Mechefske, Chris, An improved similarity-based prognostic algorithm for RUL estimation using an RNN autoencoder scheme, *Reliability Engineering & System Safety, Elsevier*, **2020**, *199*.
7. Da Xu, Li and He, Wu and Li, Shancang, Internet of things in industries: A survey, *IEEE Transactions on industrial informatics*, **2014**, *10*, 2233–2243.
8. Li, Xiaomin and Li, Di and Wan, Jiafu and Vasilakos, Athanasios V and Lai, Chin-Feng and Wang, Shiyong, A review of industrial wireless networks in the context of industry 4.0, *Wireless networks, Springer*, **2017**, *23*, 23–41.
9. Jihong Yan, Yue Meng, Lei Lu and Lin Li, Industrial Big Data in an Industry 4.0 Environment: Challenges, Schemes, and Applications for Predictive Maintenance, *IEEE Access*, **2017**, *5*, 23484–23491.
10. Kiangala, Kahiomba Sonia and Wang, Zenghui, Initiating predictive maintenance for a conveyor motor in a bottling plant using industry 4.0 concepts, *The International Journal of Advanced Manufacturing Technology, Springer*, **2018**, *97*, 3251–3271.
11. Cattaneo, Laura and Macchi, Marco, A Digital Twin Proof of Concept to Support Machine Prognostics with Low Availability of Run-To-Failure Data, *IFAC-PapersOnLine, Elsevier*, **2019**, *2*, 37–42.

12. Rengasamy, Divish and Jafari, Mina and Rothwell, Benjamin and Chen, Xin and Figueredo, Grazziela P, Deep Learning with Dynamically Weighted Loss Function for Sensor-Based Prognostics and Health Management, *Sensors, Multidisciplinary Digital Publishing Institute*, **2020**, *20*.
13. Rengasamy, Divish and Morvan, Hervé P and Figueredo, Grazziela P, Deep learning approaches to aircraft maintenance, repair and overhaul: a review, *2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE*, **2018**, 150–156.
14. Jogin, Manjunath and Madhulika, MS and Divya, GD and Meghana, RK and Apoorva, S and others, Feature extraction using Convolution Neural Networks (CNN) and Deep Learning, *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE*, **2018**, 2319–2323.
15. Reddy, Kishore K and Sarkar, Soumalya and Venugopalan, Vivek and Giering, Michael, Anomaly detection and fault disambiguation in large flight data: A multi-modal deep auto-encoder approach, *Annual Conference of the Prognostics and Health Management Society*, **2016**.
16. Sarkar, Soumalya and Reddy, Kishore K and Giering, Michael and Gurvich, Mark R, Deep learning for structural health monitoring: A damage characterization application, *Annual Conference of the Prognostics and Health Management Society*, **2016**, 176–182.
17. Sarkar, Soumalya and Reddy, Kishore K and Giering, Michael and Gurvich, Mark R, Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network, *2016 IEEE International Conference on Aircraft Utility Systems (AUS), IEEE*, **2016**, 135–140.
18. Susto, Gian Antonio and Schirru, Andrea and Pampuri, Simone and McLoone, Seán and Beghi, Alessandro, Machine learning for predictive maintenance: A multiple classifier approach, *IEEE Transactions on Industrial Informatics, IEEE*, **2014**, *11*, 812–820.
19. Kiangala, Kahiomba Sonia and Wang, Zenghui, An Effective Predictive Maintenance Framework for Conveyor Motors Using Dual Time-Series Imaging and Convolutional Neural Network in an Industry 4.0 Environment, *IEEE Access, IEEE*, **2020**, *Volume 8*, 121033–121049.
20. Fan, Cheng and Sun, Yongjun and Zhao, Yang and Song, Mengjie and Wang, Jiayuan, Deep learning-based feature engineering methods for improved building energy prediction, *Applied energy, Elsevier*, **2019**, *240*, 35–45.
21. Rutagarama, Maxime, Deep Learning for Predictive Maintenance in Impoundment Hydropower Plants, **2019**.
22. Sutskever, Ilya and Vinyals, Oriol and Le, Quoc V, Sequence to sequence learning with neural networks, *Advances in neural information processing systems*, **2014**, 3104–3112.
23. Que, Zhiqiang and Liu, Yanyang and Guo, Ce and Niu, Xinyu and Zhu, Yongxin and Luk, Wayne, Real-time Anomaly Detection for Flight Testing using AutoEncoder and LSTM, *2019 International Conference on Field-Programmable Technology (ICFPT), IEEE*, **2019**, 379–382.
24. Hundman, Kyle and Constantinou, Valentino and Laporte, Christopher and Colwell, Ian and Soderstrom, Tom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, **2018**, 387–395.
25. Yang, Jianbo and Nguyen, Minh Nhut and San, Phyo Phyo and Li, Xiaoli and Krishnaswamy, Shonali, Deep convolutional neural networks on multichannel time series for human activity recognition, *AIjcai*, **2015**, *15*, 3995–4001.
26. Mishra, Krishna Mohan and Krogerus, Tomi R and Huhtala, Kalevi J, Fault detection of elevator systems using deep autoencoder feature extraction, *2019 13th International Conference on Research Challenges in Information Science (RCIS)*, **2019**, 1–6.
27. Sakurada, Mayu and Yairi, Takehisa, Anomaly detection using autoencoders with nonlinear dimensionality reduction, *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, **2014**, 4–11.
28. Essien, Aniekan and Giannetti, Cinzia, A deep learning model for smart manufacturing using convolutional LSTM neural network autoencoders, *IEEE Transactions on Industrial Informatics, IEEE*, **2020**, *16*, 6069–6078.
29. Wu, Jingyao and Zhao, Zhibin and Sun, Chuang and Yan, Ruqiang and Chen, Xuefeng, Fault-attention generative probabilistic adversarial autoencoder for machine anomaly detection, *IEEE Transactions on Industrial Informatics, IEEE*, **2020**, *16*, 7479–7488.

30. Chandola, Varun and Banerjee, Arindam and Kumar, Vipin, Anomaly detection: A survey, *ACM computing surveys (CSUR)*, **2009**, *41*, 1–58.
31. Yan, Jihong and Koc, Muammer and Lee, Jay, A prognostic algorithm for machine performance assessment and its application, *Production Planning & Control, Taylor & Francis*, **2004**, *8*, 796–801.
32. Charu, C Aggarwal, *Neural Networks and Deep Learning: A Textbook*, Springer, 2018.
33. Goodfellow, Ian and Bengio, Yoshua and Courville, Aaron and Bengio, Yoshua, *Deep learning*, MIT press Cambridge, Volume 1, Number 2, 2016.
34. Hinton, Geoffrey E and Salakhutdinov, Ruslan R, Reducing the dimensionality of data with neural networks, *Science, American Association for the Advancement of Science*, **2006**, *313*, 504–507.
35. Brownlee, Jason, Deep learning for time series forecasting: Predict the future with MLPs, CNNs and LSTMs in Python, *Machine Learning Mastery*, 2018.
36. Arrazate, Rodrigo Torres, Development of a URDF file for simulation and programming of a delta robot using ROS, **2017**.
37. Honegger, Marcel and Codourey, Alain and Burdet, Etienne, Adaptive control of the hexaglide, a 6 dof parallel manipulator, *Proceedings of International Conference on Robotics and Automation, IEEE*, **1997**, *1*, 543–548.
38. Codourey, Alain and Burdet, Etienne, A body-oriented method for finding a linear form of the dynamic equation of fully parallel robots, *Proceedings of International Conference on Robotics and Automation, IEEE*, **1997**, *2*, 1612–1618.
39. Raschka, Sebastian and Mirjalili, Vahid, Python machine learning, *Packt Publishing Ltd*, 2017.
40. LeNail, Alexander, Nn-svg: Publication-ready neural network architecture schematics, *Journal of Open Source Software*, **2019**, *4*, 747.
41. Ide, Hidenori and Kurita, Takio, Improvement of learning for CNN with ReLU activation by sparse regularization, *2017 International Joint Conference on Neural Networks (IJCNN), IEEE*, **2017**, 2684–2691.
42. Nair, Vinod and Hinton, Geoffrey E, Rectified linear units improve restricted boltzmann machines, *ICML*, **2010**.
43. Glorot, Xavier and Bordes, Antoine and Bengio, Yoshua, Deep sparse rectifier neural networks, *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, **2011** , 315–323.