Towards an explainable GNN based approach for lifetime prediction using multivariate time-series data

A. P. Vedurmudi*, S. M. H. Zaidi and S. Eichstädt

Department 9.4 - Metrology for the Digital Transformation, Physikalisch-Technische Bundesanstalt,
Berlin, 10587, Berlin
**E-mail: anupam.vedurmudi@ptb.de*
www.ptb.de

Deep learning is a versatile tool for condition monitoring that provides accuracy at the cost of explainability. In this contribution we explore the use of graph neural networks (GNN) for predicting the remaining useful lifetime (RUL) of an electromechanical cylinder. Through a pre-processing scheme based on *a priori* knowledge of the constituent sensors and the corresponding dataset, the time-series data is embedded into a graph structure. The preprocessed data is used to train a GNN to correctly predict the RUL of the cylinder. Explainable AI (xAI) methods are then used to assess the importance of nodes and edges in making predictions for the specific case of a wrongly predicted value.

Keywords: Condition monitoring, explainable AI, graph neural networks

1. Introduction

A graph neural network (GNN) [1] is a type of artificial neural network designed to operate on and process data that can be represented by graphs. Representative examples of such data with an inherent graph-like structure can be found in social networks and biomolecules. GNNs improve transparency and explainability by allowing a graph-based visualization of their decision-making process and of relationships between its constituent elements. The present work explores the application of GNNs on multivariate time-series data for a condition-monitoring task. The component being monitored is an electromechanical cylinder (EMC) with a spindle drive [2] and the associated experiments were conducted at the Zentrum für Mechatronik und Automatisierungstechnik gGmbH (ZeMA) in Saarbrücken, Germany. The EMC is based on a ball screw drive and serves as a mechanical linear actuator that enables a rotational to linear translation of motion.The main purpose of the experiments was to develop methods to assess the behavior of the EMC while operating under high-loads and high-speeds for the pur-

poses of condition monitoring. In previous research[3], a series of machine learning methods for the estimation of the remaining useful lifetime (RUL) of the EMC were implemented as part of an automated toolbox. In the present work, we motivate the use of GNNs by the desire to improve the human interpretabity in condition monitoring tasks using time-series data by exploiting the inherent transparency and explainability of graph-based visualizations. In particular, the advantage of the graph representation will be emphasized by combining GNNs with two common explainable AI (xAI) methods namely, saliency maps and integrated gradients in order to attribute the results of a prediction to the input data.

2. Methods

2.1. *Experimental Setup and Dataset*

The setup in question, illustrated in Fig. 1, consists of an EMC subject to a constant axial force of 7 kN via a pneumatic cylinder and lateral forces by means of a fluidic muscle. The tests were carried out until the operational failure of the EMC. Furthermore, the setup was outfitted with eleven sensors continuously measuring various physical properties during the operation of the EMC. The operational EMC velocity is around 200 mms^{-1} with a stroke range af 100-350 mm. Moreover, it accelerates and decelerates at the rate of around 5 ms^{-2}. The combination of a high axial load, a high motion speed, and a high acceleration leads to a fast wear progression[3]. The drag error was used as failure criterion, i.e. the lifetime test was considered to have failed when a deviation larger than 30 mm between the set-position and the actual EMC end position was recorded[4]. A schematic illustration of the setup, showing the three-phase motor driven by a external power source as well as the approximate placement of the sensors, is illustrated in Fig. 2. The physical attributes measured by the sensors during the experiments are as follows

(1) The active current (ACu) input to the EMC,
(2) The three motor currents (MC1-3) corresponding to the phases of the electric motor driving the EMC,
(3) The acoustic output near the EMC measured by a microphone (Mic),
(4) The pneumatic pressure exerted by the EMC (Pre),
(5) The velacity (Vel) of the EMC,
(6) The accelarations of the plane bearing (APB), ball bearing (ABB) and the plane ring (APR) and,

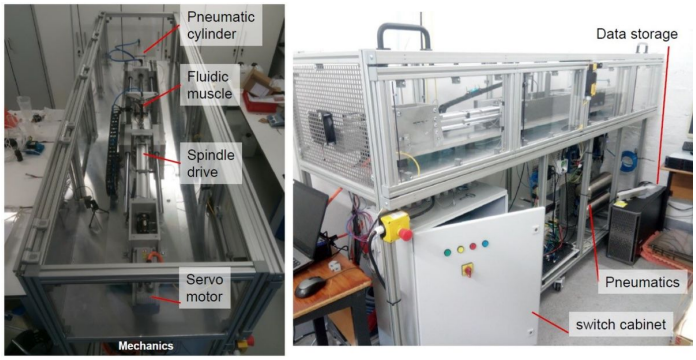

Fig. 1. The test-bed used to carry out the experiment and generate the dataset used in this work at the Zentrum für Mechatronik und Automatisierungstechnik gGmbH (ZeMA) in Saarbrücken, Germany.

(7) The axial force (AxF) exerted by the EMC.

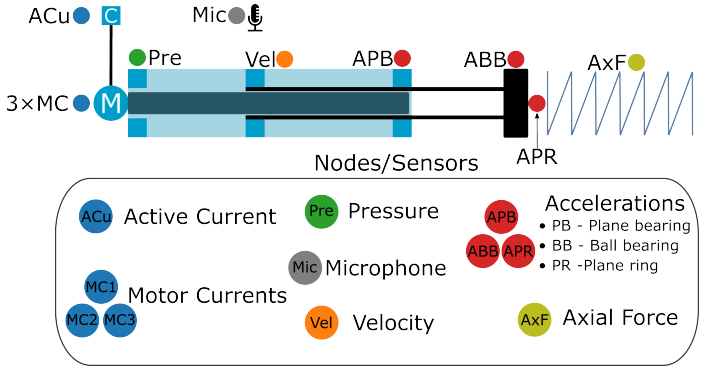

Fig. 2. A schematic representation of the EMC testbed and the associated sensors shown in Fig. 1. Adapted from Dorst et al.[5]

The dataset under consideration[6] contains the outputs of eleven sensors monitoring the EMC and consists of 4766 samples where each sample corresponds to 11 time-series with 2000 elements. Corresponding to a sampling rate of 2 kHz, each time-series is one second in duration and represents the measurements taken during one return stroke of the EMC. Moreover, each sample is provided with an associated remaining useful lifetime (RUL) estimate as a percentage. A sample from the aforementioned dataset given

by the time-series output of the eleven sensors with corresponding to an RUL of 70% is shown in Fig. 3.

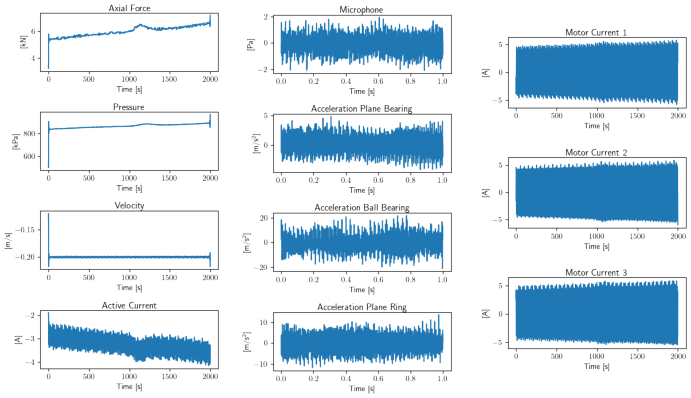

Fig. 3. A single sample from the dataset corresponding to the time-series output from the eleven sensors with an associated RUL of 70%

2.2. *Preprocessing and graph embedding*

In order to be able to the use GNNs on time-series, an appropriate graph representation of the data needs to be chosen. One way to do so is by interpreting the individual sensors as nodes and choosing an appropriate interrelation between them as the basis to set edges[7]. Both the resulting edges and nodes can be provided with data attributes. We do so by embedding the multivariate time-series into a graph-like structure such that each sample results in a graph consisting of eleven nodes representing the sensors with their individual (normalized) time-series as node-attributes. Furthermore, the edges are weighted by the correlation between the time-series of connected nodes which in turn correspond to the edge attributes. Edges with a weight below the average of the absolute correlations between all nodes (55 possible edges) are discarded. In doing so, we have effectively used the *a priori* information about the setup and the sensors to pre-process the data. An example of this procedure for the time-series dataset from Fig. 3 is illustrated in Fig. 4. In the figure, the nodes have been labeled according to their corresponding sensors as shown in Fig. 2. The edges have been labeled by their weights, i.e. the correlation between

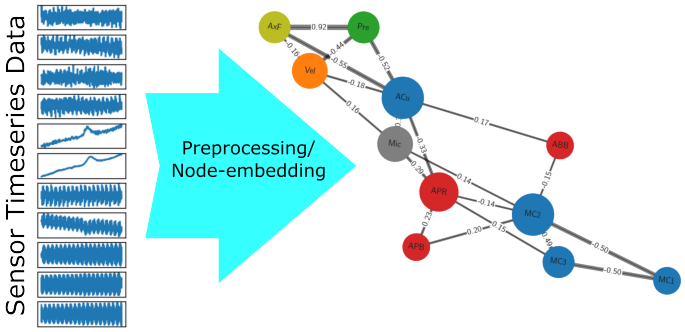

Fig. 4. Illustration of the preprocessing step in which the time-series data is restructured as a graph with eleven nodes.

the sensor nodes. The thickness of the edges in the figure has been scaled in the same manner.

2.3. *GNN Structure and Training*

The GNN consists of three GraphConv convolutional layers[8] followed by one pooling, one dropout and two alternating linear layers. The pooling layer returns batch-wise graph-level-outputs by adding node features across the node dimension. The algorithm was implemented using the PyTorch Geometric[9] library. Furthermore, the target value was modified such that the percentage of lifetime passed was binned into steps of 10%. In doing so, the RUL prediction task was reformulated as the classification of a given data sample into one of 10 categories. The negative log-likelihood (NLL) function was chosen as the target loss for the classification problem. From the resulting set of 4766 graphs, 4290 are used to train the GNN, while the remaining are set aside for testing.

3. Results

The performance of the GNN classification is shown in Fig. 5 in terms of a training and test accuracy as well as the training loss as a function of the training epochs. The NLL loss after 30 epochs was around 0.33%. Furthermore, after the same number of epochs a training accuracy of 97% and a test accuracy of 92% was achieved. As an example, the case of a correct prediction made by the GNN for the graph from Fig. 4 is illustrated

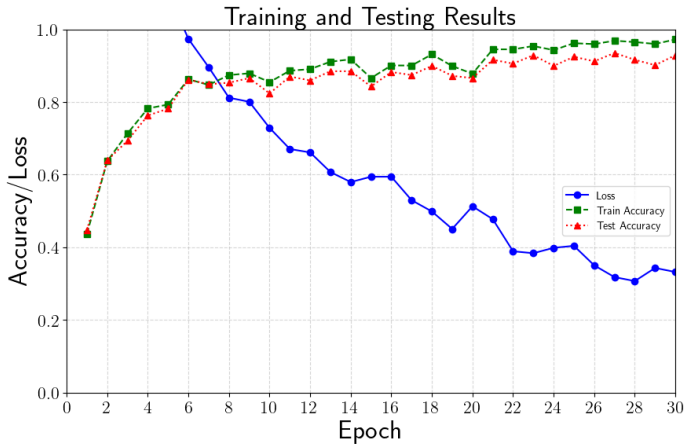

Fig. 5. The training and test accuracy as well as the loss of the GNN against the training epochs.

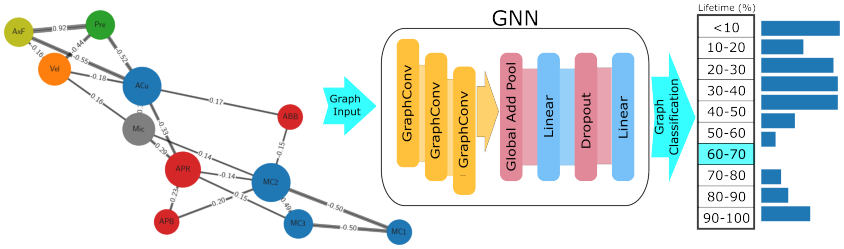

Fig. 6. An example from the test dataset where the GNN makes a correct prediction (blue). The bar plot indicates the negative log-likelihood associated with each prediction.

in Fig. 6. In order to demonstrate the advantage of the combination of xAI and GNNs, we consider another sample from the test dataset where the network makes a false prediction. In Fig. 7, we see that for the given graph input, the GNN predicts an RUL of 50-60% instead of the true value of 60-70%. In this case, it would be advantageous to attribute this discrepancy to the sensor inputs.

3.1. *Explainability*

Explainable AI or xAI refers to techniques used to improve the human-interpretability of machine learning methods[11]. Deep learning (DL) algo-

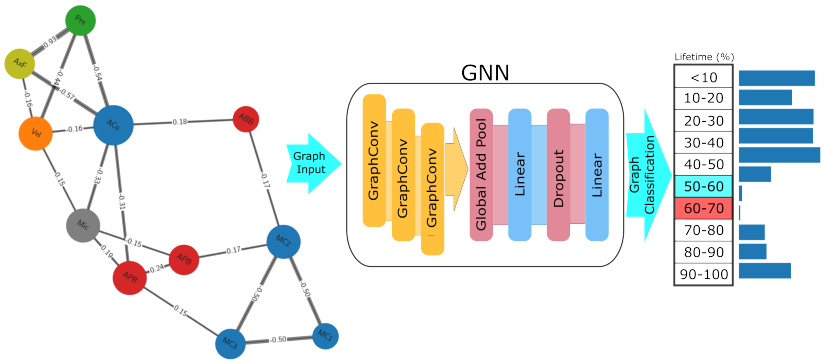

Fig. 7. An example from the test dataset where the GNN makes a false prediction (red) instead of the true prediction (blue). The bar plot indicates the negative log-likelihood associated with each prediction.

rithms in particular have a tendency to behave as 'black boxes', i.e., the explanation for a specific decision or result taken by the algorithm is opaque even to the designers. In some cases explainability is achieved by using ML methods that are inherently human-interpretable, such as linear regression or decision trees, while in other cases supplementary methods need to be used in addition to the implemented ML algorithm. Our goal is to use the graph structure to identify the nodes (sensors) or interactions (edges) that significantly contributed to the wrong decision taken by the GNN as shown in Fig. 7. To do so we apply two commonly used methods to improve the interpretability of DL algorithms by providing numerical attributions to the input features for a given output - saliency maps[12] and integrated gradients[13]. Both methods require a knowledge of the implemented algorithm, which in our case refers to the weights and parameters of the GNN learned via training. Saliency maps correspond to a simple gradient of the output of a neural network with respect to a particular feature. For an output C and an input feature x_i, the saliency E_{grad} is expressed as

$$E_{\text{grad}} = \left.\frac{\partial F_C}{\partial x}\right|_{x=x_i} . \tag{1}$$

The attribution can be attained at both the edge and node levels. The result of this computation for the classification shown in Fig. 6 is illustrated in Fig. 8. Typically, calculating saliency maps is computationally intensive and not feasible for large datasets. On the other hand, integrated gradients require the computation of the path integral of the gradient of the output

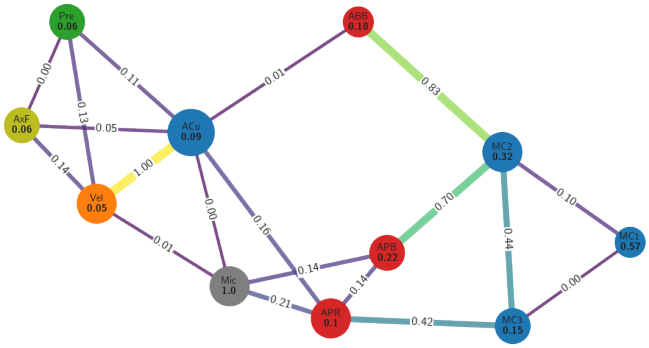

Fig. 8. Graph depicting node- and edge-level attributions for a false prediction generated using saliency maps.

with respect to a given feature and a baseline input value x_i'. In this case, the baseline corresponds to the GNN receiving a zero input for all nodes and edges. The path integral computed in this way is given by

$$I_{\text{grad}} = (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial F_C(x_i' + \alpha(x_i - x_i'))}{\partial x}\bigg|_{x=x_i} d\alpha \ . \tag{2}$$

The corresponding attributions at a node and edge level obtained via integrated gradients is shown in Fig. 9. In contrast to saliency maps, the computation of integrated gradients in this manner requires fewer computational resources.

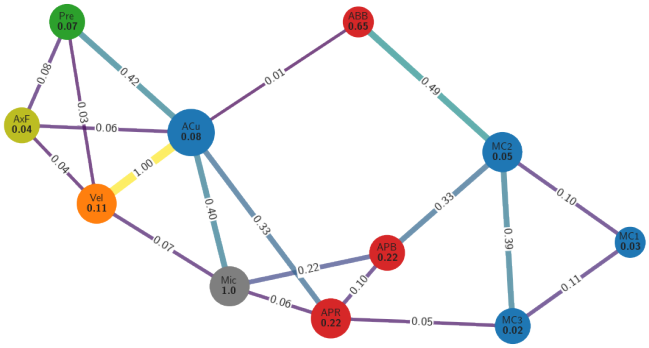

Fig. 9. Graph depicting the node- and edge-level attributions generated using integrated gradients for a false prediction.

In both cases, the accelerations of the plane ring (APR) and plane bearing (APB) seem to contribute significantly to the decision taken by the GNN. In the case of the saliency map, the correlation between APB and the second motor current (MC2) also has a high contribution, while the use of integrated gradients identifies a high attribution of the interaction between APR and APB. In contrast to integrated gradients, a high impact of the motor currents can be seen in the explanation via saliency maps, while the microphone output has a significant contribution in both cases. While neither the active current (ACu) nor its interaction with other sensors was given a high numerical attribution for the given example, the interaction of ACu and the velocity (Vel) seem to significantly contribute to the output in both cases.

4. Conclusions

In the present work, we have demonstrated the use of graph neural networks for a condition monitoring task involving the prediction of the remaining useful lifetime (RUL) of an electromechanical cylinder. Time-series data collected by means of a series of high-load and high-speed driving tests was first embedded into a graph structure using the time-series data and the correlation between the sensors as *a priori* information. A graph neural network was trained using this preprocessed data to predict the RUL, which was reformulated as a simplified classification task. Two common xAI methods - integrated gradients and saliency maps - were used in conjunction with the GNN to provide node- and edge-level attributions for a given false prediction. In doing so, GNNs were shown to offer increased explainability/interpretability for condition monitoring, while the graph structure enabled us to account for the effects of both individual sensors and their interactions. Future research will explore the applicability of model agnostic attribution methods like SHAP and LIME to graph neural networks and consider the integration of semantic information about the sensors into the graph structure.

References

1. J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li and M. Sun, Graph neural networks: A review of methods and applications, *AI Open* **1**, 57 (2021).
2. Festo S. E. & Co. K. G., *Electric cylinders ESBF, with spindle drive.*

Available: https://www.festo.com/cat/en-gb_gb/data/doc_ENUS/PDF/US/ESBF_ENUS.PDF. [Accessed: 16 Nov 2023].
3. T. Dorst, Measurement Uncertainty in Machine Learning – Uncertainty Propagation and Influence on Performance (Doctoral Dissertation, Universität des Saarlandes), August 2023. doi:10.22028/D291-40173.
4. N. Helwig, Zustandsbewertung industrieller Prozesse mittels multivariater Sensordatenanalyse am Beispiel hydraulischer und elektromechanischer Antriebssysteme. (Doctoral Dissertation, Universität des Saarlandes), December 2018. doi:10.22028/D291-27896.
5. T. Dorst, M. Gruber, B. Seeger, A. P. Vedurmudi, T. Schneider, S. Eichstädt and A Schütze, Uncertainty-aware Data Pipeline of Calibrated MEMS Sensors Used for Machine Learning. *Measurement: Sensors* **22**, 100376 (2022).
6. T. Dorst, *Sensor data set of 3 electromechanical cylinder at ZeMA testbed (2kHz)*, 5, Zenodo, 2019. [HDF5]. Available: https://doi.org/10.5281/zenodo.3929385. [Accessed: 26 June 2023].
7. Wang, Y., Duan, Z., Huang, Y., Xu, H., Feng, J. & Ren, A. MTHetGNN: A heterogeneous graph embedding framework for multivariate time series forecasting. *Pattern Recognition Letters* **153**, 151 (2022),
8. C. Morris, M. Ritzert, M. Fey, W. Hamilton, J. Lenssen, G. Rattan & M. Grohe, Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks. *AAAI Conference On Artificial Intelligence*, (New Orleans, USA, 2018).
9. M. Fey, & J. Lenssen, Fast Graph Representation Learning with PyTorch Geometric. *ICLR 2019 Workshop On Representation Learning On Graphs And Manifolds*. (New Orleans, USA, 2019).
10. H. Yuan, H. Yu, S. Gui and S. Ji, Explainability in Graph Neural Networks: A Taxonomic Survey, *IEEE Transactions on Pattern Analysis & Machine Intelligence* **45**, 5782 (2023).
11. C. Molnar, *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable* (2nd ed.), (2022). Available: https://christophm.github.io/interpretable-ml-book/cite.html.
12. K. Simonyan, A. Vedaldi, and A. Zisserman, Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, in *Proc. 2nd International Conference On Learning Representations (ICLR'14)* (Banff, AB, Canada, 2014).
13. M. Sundararajan, A. Taly and Q. Yan, Axiomatic Attribution for Deep Networks, in *Proc. 34th International Conference On Machine Learning (ICML'17)*, (Sydney, Australia, 2017).