Vector Reconstruction Error for Anomaly Detection: Preliminary Results in the IMOCO4.E Project

Dario Guidotti
DUMAS
University of Sassari
Sassari, Italy
dguidotti@uniss.it

Riccardo Masiero
CRIT srl
Vignola (MO), Italy
masiero.r@crit-research.it

Laura Pandolfo
DUMAS
University of Sassari
Sassari, Italy
lpandolfo@uniss.it

Luca Pulina
DUMAS
University of Sassari
Sassari, Italy
lpulina@uniss.it

***Abstract*—In recent years, the integration of artificial intelligence (AI) techniques has significantly transformed the field of predictive maintenance, enabling businesses to proactively monitor and address potential equipment failures before they occur. One critical aspect of predictive maintenance is the detection of anomalies, which can serve as early warning signs for impending faults or failures. In this paper we present some preliminary results obtained by leveraging autoencoders and the related vector reconstruction error in the scope of the IMOCO4.E Project.**

***Index Terms*—Predictive Maintenance, Anomaly Detection, Neural Networks**

I. Introduction

Maintenance strategies are vital for optimal performance and longevity of industrial systems. Traditional approaches, based on fixed schedules or reactive responses, result in inefficiencies, increased downtime, and high costs. However, predictive maintenance [1], leveraging AI and data-driven methods, has revolutionized the industry by enabling proactive identification and mitigation of potential failures.

Anomaly detection [2] is a crucial component of predictive maintenance, involving the monitoring of equipment behavior to detect deviations from normal operational patterns. Accurate identification of anomalies enables timely corrective actions, risk mitigation, optimized maintenance schedules, and improved reliability and availability of critical assets.

Autoencoders [3], a type of artificial neural network, have emerged as a promising approach for anomaly detection. Their ability to learn compact representations of data, capturing underlying patterns while suppressing noise and outliers, makes them well-suited for this task. By training autoencoders on normal operating data, minimal reconstruction errors are achieved. Consequently, when exposed to anomalous data, higher reconstruction errors occur, facilitating the detection and isolation of abnormal instances.

IMOCO4.E [4] is a Key Digital Technologies Joint Undertaking project, initiated in September 2021 with the participation of 46 partners from 13 countries. The project aims to enhance the intelligence and adaptability of mechatronic systems by integrating novel sensory information, model-based approaches, AI, ML, and industrial IoT principles. By leveraging these advanced technologies, IMOCO4.E aims to facilitate the transition of European manufacturing toward Industry 4.0, enabling the perception and management of sophisticated machinery and robotics. The project will deliver a reference architecture tested and validated in diverse industrial domains, including packaging, industrial robotics, healthcare, and semiconductors.

In this work we focus on a use case originating from the food and beverage industries: in particular, the component of interest is a cutting machine used in a packaging production line. Our aim is to build an anomaly detection system which can recognize when the cutting blade need to be replaced due to its degradation.

The rest of the paper is structured as follows. In Section II we introduce some basic concepts and definitions. In Section III and IV we present our solution for the task of interest and our experimental results respectively. Finally, in Section V we present some of our future research plans.

II. Background

A. Neural Networks

A *sequential* neural networks is a machine-learning model consisting of interconnected computing units, commonly known as *neurons*. Such network present a structure composed by several sequential layers whose neurons are only connected to the neurons of the immediately previous and following layers. The first and the last layer of the network are called, respectively, *input* and *output* layers.

Computationally, the l-th layer of a sequential neural network can be defined as a function $f_l : \mathcal{I}_l \rightarrow \mathcal{O}_l$ where $\mathcal{I}_l$ and $\mathcal{O}_l$ represent respectively the input and output domain of the function. Similarly, a generic sequential neural network can be defined as a function $\nu : \mathcal{I}_1 \rightarrow \mathcal{O}_p$ where $\mathcal{I}_1$ and $\mathcal{O}_p$ are, respectively the input (output) domain of both the first (last) layer of the network and of the network as a whole. In particular, the neural network function ν can be expressed as:

$$\nu(x) = f_p(f_{p-1}(...f_1(x)...)) \tag{1}$$

In this work, we are mainly focused on layers whose inputs and outputs consist of vectors of real numbers, therefore we will usually have $\mathcal{I}_l \subseteq \mathbb{R}^n$ and $\mathcal{O}_l \subseteq \mathbb{R}^m$.

The structure defined by the various layers of a neural network, their connections, and their hyper-parameters is commonly known as the network *architecture*. In this paper we

focus on sequential neural networks presenting *linear layers* and *non-linear activation functions*. A linear layer applies to its input a simple affine transformation $y = Wx + b$, where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ are respectively the input and output vector and W and b are the learnable weight matrix and bias vector. The size m of the output is determined by the number of hidden neurons in the layer. An activation function layer, in general, applies a certain activation function to all the element of the vector provided as input. In this work, we consider only ReLU activation functions.

B. Autoencoders

Autoencoders are a class of neural networks widely used in unsupervised learning tasks, including data compression, dimensionality reduction, and anomaly detection. They consist of an encoder and a decoder, which work together to learn a compact representation of input data. The encoder maps the input data into a lower-dimensional latent space representation, often referred to as the bottleneck layer or code. This encoding process captures the most salient features and patterns of the input data, effectively reducing its dimensionality. The decoder then reconstructs the original input data from the encoded representation. During training, autoencoders aim to minimize the reconstruction error between the input and output data. By optimizing this reconstruction loss, the autoencoder learns to capture the underlying structure of the data, enabling it to generate accurate reconstructions. One key advantage of autoencoders is their ability to learn robust representations by extracting meaningful features from the input data. This is achieved through the use of bottleneck layers, which force the network to capture the most important information needed for reconstruction. As a result, autoencoders can effectively denoise input data and suppress outliers, making them particularly well-suited for anomaly detection tasks.

III. Materials and Methods

A. Dataset

The dataset of interest is the One Year Industrial Component Degradation[1] Dataset. It comprises measurements obtained during one year of operation of an OCME Vega shrink-wrapper. The data is collected in sampling sessions lasting 8 seconds, with a time resolution of 4 milliseconds. Each session consists of 2048 samples, and the number of sessions varies across different months. In total, there are 519 sessions, amounting to 1,062,912 samples. The dataset includes measurements from various sensors on the machine, with a primary focus on the blade's operations for cutting the packaging plastic film. Specifically, the measurements considered in this study encompass the motor torque, blade and film position, blade and film speed, lag between expected and actual positions of the blade and film, and a performance evaluation value. The objective is to detect instances when the cutting blade component becomes worn out, prompting its replacement with a new one.

[1] https://www.kaggle.com/datasets/inIT-OWL/one-year-industrial-component-degradation

B. Autoencoder Architecture

In this initial analysis, we examine four different architectures for our autoencoders. These architectures share a similar structure, consisting of three hidden linear layers, each followed by a layer of ReLU activation functions. The output layer does not present an activation function. Specifically, the considered autoencoders have different numbers of neurons in each hidden layer: 32, 50, 64, and 128 in the first layer, 8, 10, 16, and 32 in the second layer, and 32, 50, 64, and 128 in the third layer, respectively. As expected in autoencoders, there is an expansion of the output space in the first and third hidden layers and a contraction in the second hidden layer. For simplicity, we label these autoencoders as $A1$, $A2$, $A3$, and $A4$ respectively. Currently, we do not explore activation functions other than ReLU, as it remains one of the most commonly used activation functions in practice.

C. Training Process

We utilized pyNeVer [5], a neural network management, training, and verification tool, to train the networks of interest. pyNeVer employs PyTorch [6] as a back-end and provides a user-friendly custom training loop.

During training, we employed the Adam optimizer with a learning rate of 0.001. The mean square error was used as the loss function. We utilized a batch size of 512 for training, 128 for validation, and the same value for testing. The networks were trained for 50 epochs. The validation set constituted 30% of the training data, while 20% of the dataset was reserved for testing. These learning parameters were consistent across all the networks trained.

It is important to note that for anomaly detection tasks, the autoencoders must be trained on a subset of the dataset where anomalies are absent. Therefore, in our training and testing processes, we considered only the first 200,000 samples of the dataset. This choice is based on the reasonable assumption that significant degradation of the blade would not occur within the initial period of operations.

D. Anomaly Detection with Vector Reconstruction Error

As discussed in Subsection II-B, the trained autoencoders generate an internal compact representation of the input data and then reconstruct the original data based on this representation. However, the output of the autoencoders alone does not provide a measure of the anomaly level in a specific input data.

To quantify the degree of anomaly, we utilize the *vector reconstruction error*, following a similar approach as proposed in [7]. The vector reconstruction error measures the discrepancy between the reconstructed output and the corresponding target data. By assuming that the autoencoder has been properly trained, this measurement enables the recognition of anomaly presence and intensity. The underlying principle is that if the autoencoder fails to accurately reconstruct the input data, it indicates that the input data is anomalous in some way.

In our experimental evaluation, we employ the mean square error between the output and the corresponding target as the vector reconstruction error.

IV. Results

The experiments were performed on a MacBook Air laptop with 24 GB of RAM and an Apple M2 CPU. The operating system used was macOS Ventura 13.4. We utilized MPS for training the autoencoders. The code required to replicate our experiments can be found at https://github.com/darioguidotti/imoco4e-pilot3-etfa2023.

All the considered autoencoders achieved a reasonable level of performance during the training process. Specifically, the mean square errors computed on the test set were $7.54 \cdot 10^{-7}$, $3.60 \cdot 10^{-6}$, $5.72 \cdot 10^{-7}$, $8.05 \cdot 10^{-6}$ for $A1$, $A2$, $A3$, and $A4$ respectively. Notably, increasing the number of neurons in the hidden layers did not result in a corresponding improvement in the accuracy of the networks.

As mentioned in Subsection III-D, the test set loss alone is insufficient to fully assess the performance of our autoencoders on the task of interest. Consequently, we computed the vector reconstruction error as the mean square error (MSE) between the output of the autoencoders and the corresponding target for all the samples in the dataset.

In Figure 1, we present scatter plots illustrating these measurements for our autoencoders. Notably, an anomaly is observed in the data during the 4th month, which is most effectively detected by $A1$. Specifically, even the smallest MSE value computed during the anomaly is significantly greater than the maximum MSE of the non-anomalous data points.

It is worth mentioning that while $A1$ appears to be the most adept at identifying the anomaly, all the autoencoders demonstrate the ability to correctly detect it. Following $A1$, the second, third, and fourth best performers were $A3$, $A4$, and $A2$, respectively. As previously observed regarding the accuracy of the networks, the size of the autoencoders does not appear to be directly correlated with their performance in anomaly detection.

V. Conclusions and Future Works

In this paper we investigated how to leverage autoencoders and the vector reconstruction error to identify anomalies in a real-world dataset originating from a predictive maintenance task. In our experimental evaluation we considered various network architectures and we evaluated both their accuracy in reconstructing the original data and their sensitivity to the presence of anomalous data. Surprisingly, our experimental results reveal that the less complex autoencoder outperforms the others in detecting anomalous data.

At present, we are working both on extending our experimental evaluation and on deepening our investigation. In particular, our experimental results show that, in addition to the anomaly during the 4th months, other minor anomalies seems to be present during the operation of the cutting blade. We believe that understanding the underlying reasons for these anomalies is an intriguing avenue for exploration. To extend our experimental evaluation, we plan to extend the network architectures considered with other activation functions and different numbers of hidden layers and neurons. Finally, we would like to investigate if the reliability of our autoencoders may be certified, and eventually increased, using formal verification techniques [8], [9].

Acknowledgement

This work was supported by the H2020 ECSEL JU grant agreement No. 101007311 "Intelligent Motion Control under Industry4.E" (IMOCO4.E) project. The support is gratefully acknowledged.

References

[1] C. Krupitzer, T. Wagenhals, M. Züfle, V. Lesch, D. Schäfer, A. Mozaffarin, J. Edinger, C. Becker, and S. Kounev, "A survey on predictive maintenance for industry 4.0," *CoRR*, vol. abs/2002.08224, 2020.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, 2009.

[3] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Unsupervised and Transfer Learning - Workshop held at ICML 2011*, ser. JMLR Proceedings, vol. 27. JMLR.org, 2012, pp. 37–50.

[4] M. Cech, A. Beltman, and K. Ozols, "Digital twins and AI in smart motion control applications," in *Proceedings of ETFA 2022*. IEEE, 2022, pp. 1–7.

[5] D. Guidotti, L. Pulina, and A. Tacchella, "pynever: A framework for learning and verification of neural networks," in *Proceedings of ATVA 2021*, ser. Lecture Notes in Computer Science, vol. 12971. Springer, 2021, pp. 357–363.

[6] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Proceedings of NIPS 2019*, 2019, pp. 8024–8035.

[7] H. Torabi, S. L. Mirtaheri, and S. Greco, "Practical autoencoder based anomaly detection by using vector reconstruction error," *Cybersecur.*, vol. 6, no. 1, p. 1, 2023.

[8] D. Guidotti, F. Leofante, L. Pulina, and A. Tacchella, "Verification and repair of neural networks: A progress report on convolutional models," in *Proceedings of AI*IA 2019*, ser. LNCS, vol. 11946. Springer, 2019, pp. 405–417.

[9] S. Demarchi, D. Guidotti, A. Pitto, and A. Tacchella, "Formal verification of neural networks: A case study about adaptive cruise control," in *Proceedings of ECMS 2022*. ECMS, 2022, pp. 310–316.

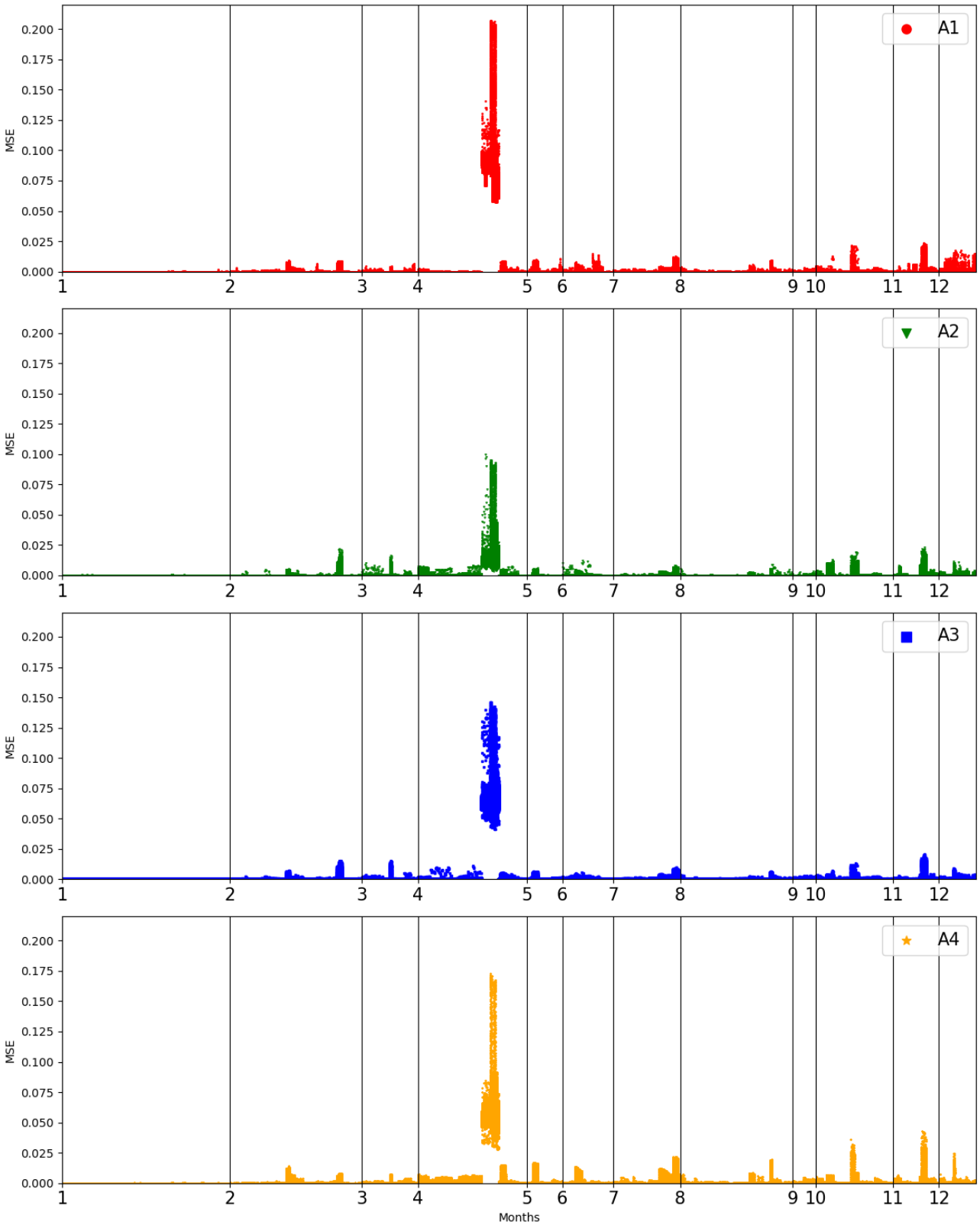

Fig. 1. Graphical representation of the results of our experimental evaluation. On the x-axis are reported the input samples of interest divided in the 12 months of measurements, whereas on the y-axis the MSE between the output of $A1$ and the target is reported. As can be seen, the number of samples for each months is subject to variation.