



Engineering and evaluating an unsupervised predictive maintenance solution: a cold-forming press case-study

Apostolos Giannoulidis¹ · Anastasios Gounaris¹ · Athanasios Naskos² · Nikodimos Nikolaidis² · Daniel Caljouw³

Received: 23 August 2023 / Accepted: 19 February 2024
© The Author(s) 2024

Abstract

In real-world industries, production line assets may be affected by several factors, both known and unknown, which dynamically and unpredictably evolve so that past data are of little value for present ones. In addition, data is collected without assigned labels. How can someone use run-to-failure data to develop a suitable solution toward achieving predictive maintenance (PdM) in this case? These issues arise in our case, which refers to a cold-forming press. Such a setting calls for an unsupervised solution that can predict upcoming failures investigating a wide spectrum of approaches, namely similarity-based, forecasting-based and deep-learning ones. But before we decide on the best solution, we first need to understand which key performance indicators are appropriate to evaluate the impact of each such solution. A comprehensive study of available evaluation methods is presented, highlighting misconceptions and limitations of broadly used evaluation metrics concerning run-to-failure data, while proposing an extension of state-of-the-art range-based anomaly detection evaluation metrics to serve PdM purposes. Finally, an investigation of pre-processing, distance metrics, incorporation of domain expertise, and the role of deep learning shows how to engineer an unsupervised solution for predictive maintenance providing insightful answers to all these problems. Our experimental evaluation showed that judicious design choices can improve efficiency of solutions up to two times.

Keywords Anomaly detection · KPIs · Predictive maintenance · Cold forming press

Introduction

Predictive maintenance (PdM) is a key technology for every modern industry, where among others it promises increasing operation time of equipment, safer working environment, better quality products and reducing maintenance expenses.

PdM involves monitoring the condition of assets through sensor devices to predict and prevent failures, typically employing advanced data-driven techniques. Although there are many PdM proposals in the literature, there exists no generic solution because each case study has its own unique characteristics Korveis et al. (2018), Zhang et al. (2019). A common challenge is to take advantage of as much expert knowledge as possible without putting a burden to these experts. Furthermore, some additional challenges that arise in the field are training data availability since in real-world scenarios, most commonly, data has no labels, data scarcity because important failures are not that common, dealing with a dynamic environment and necessity of context-awareness. Such challenges constitute the main motivation of our work and are highlighted in other works as well, e.g., Dalzochio et al. (2020). Moreover, evaluation using correct Key Performance Indicators (KPIs) is not trivial. As implied above, asking experts to manually label monitoring data to circumvent some of these problems is seldom an option.

✉ Apostolos Giannoulidis
agiannous@csd.auth.gr

Anastasios Gounaris
gounaria@csd.auth.gr

Athanasios Naskos
naskos@atlantis-engineering.com

Nikodimos Nikolaidis
nikolaidis@atlantis-engineering.com

Daniel Caljouw
Daniel.Caljouw@philips.com

¹ Aristotle University of Thessaloniki, Thessaloniki, Greece

² Atlantis Engineering, Thessaloniki, Greece

³ Philips Consumer Lifestyle, Drachten, The Netherlands

Our motivating case refers to the operation of a cold-forming press in an industrial plant in Europe. In this setting, we characterize the operation environment as dynamic because different kinds of everyday changes, such as materials, temperature, and so on, lead to altered behaviour in the collected data. Briefly, the operation takes place in a dynamic environment in the sense that, in each production episode, the equipment may be configured and behave differently. This essentially rules out supervised PdM solutions because equipment configurations change frequently and past data do not provide insights into the behavior of the currently configured operation, as evidenced by the proof-of-concept experiment below.

As will be described in more detail later, our dataset comprises operation episodes from different modules and time periods, some resulting in failure and others being replaced without failure. We justify our claim that supervised techniques are inappropriate through conducting the following experiment (only for motivation purposes). We collect seven out of the twenty operational episodes, which pertain to the same module of the press, in order to build a labeled dataset. Although the data initially arrive without labels, we can assign labels by utilizing a period before each failure, in which, if alarms are raised, there is an opportunity to prevent failures. By classifying the data within that period as positive examples and data from other periods as negative, we can train a classifier to predict the positive instances. For this experiment, we choose the well-known XGBoost classifier Chen and Guestrin (2016), but any other classifier can be employed.

The key concept of this experiment lies in handling data either with or without real-world constraints. Specifically, we explore two approaches for selecting training and testing data as depicted in Fig. 1. The first approach completely disregards the time ordering of events in real-world cases. We gather all the data into a single set and perform shuffling. Then, we split the data into two equally sized parts, one for training and another for testing. In the results, we observe an F1 score of 0.81, which appears to indicate very high performance. How exactly this F1 score can be measured requires special attention, given that we are interested in alarms occurring only in a specific period, but we defer the detailed discussion on this topic to the main part of this work. In the second approach, we collect the first four episodes: three with failures and one without failure. These episodes are grouped together for training, while the remaining three episodes are used for testing. The reported F1 score for this approach is only 0.02 because past data cannot help in detecting failures in future episodes. This extreme disparity between the two approaches is solely attributed to the fact that the data evolve over time, although they stem from the same source. In the first case, the algorithm has knowledge about all episodes, including all failure cases, enabling it to

distinguish between negative and positive examples. However, this paradigm is not feasible in the real world, where we utilize data that have already been collected for training and apply the model to new data on the fly.

According to the results above, unsupervised techniques are inherently more suitable. Since such techniques typically rely on computing anomaly (i.e., dissimilarity) scores, their main challenge is to ensure that high scores are either faults or, even better, early warning signs for such faults without any explicitly provided training set. There are three main approaches to unsupervised anomaly scoring that are commonly explored: a) using deep learning (DL) techniques, b) using forecasting techniques, and c) using similarity-based techniques. We provide concrete evidence that the latter approach is more appropriate for our case, similarly to previous results in a similar setting Giannoulidis et al. (2022). However, given also the scarcity of AutoML works for unsupervised anomaly detection Bahri et al. (2022), we face the following research questions:

1. What are the KPIs that should be considered for the evaluation of a PdM methods, and thus the choice of the final solution using run-to-failure data?
2. How to engineer an appropriate similarity-based unsupervised anomaly detection PdM solution?
3. How to leverage the domain expertise?
4. What is the role of deep learning in such a setting, and how to use a DL model in an unsupervised fashion?

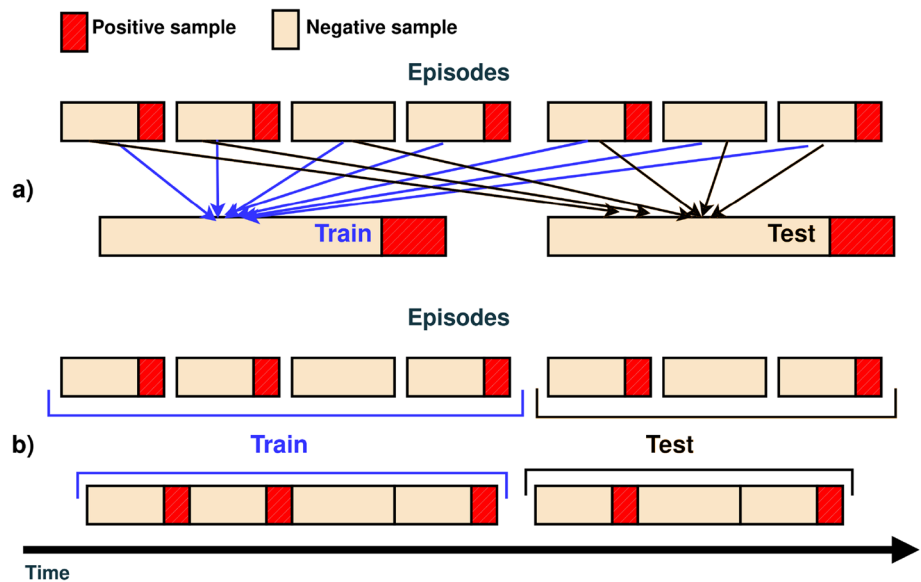
Our contribution lies in the fact that we provide concrete answers to all these questions. Figure 2 presents a flowchart of our study, which shows the engineering pipeline we follow, from understanding the problem to the final solution. We propose a novel evaluation metric (a.k.a. KPI) for anomaly detection-based PdM solutions. We also discuss in detail issues related to distance function selection, feature engineering and incorporating domain expertise, while we explain how DL and supervised techniques can be applied. Also, we thoroughly discuss the specific characteristics of our case, which are not encountered in any common PdM benchmark thus far (e.g., AI4I 2020 PdM dataset,¹ Azure² and Genesis³ for which supervised learning techniques are shown to be efficient Shi and Zhang (2023)); this allows other practitioners to understand whether their cases are similar to ours or not. Finally, we provide both our codebase and the dataset. In summary, we discuss a PdM scenario and dataset that are

¹ <https://archive.ics.uci.edu/ml/datasets/AI4I+2020+Predictive+Maintenance+Dataset>

² https://www.kaggle.com/datasets/arnabbiswas1/microsoft-azure-predictive-maintenance?select=PdM_telemetry.csv

³ https://www.kaggle.com/inIT-OWL/genesis-demonstrator-data-for-machine-learning?select=Genesis_pressure.csv

Fig. 1 a Splitting the data disregarding the time order. b Splitting data in episode level respecting time order



significantly different from existing cases already published and we provide details about all steps required in order to craft a fully operational solution in a manner that these steps can be transferred to other scenarios as well.

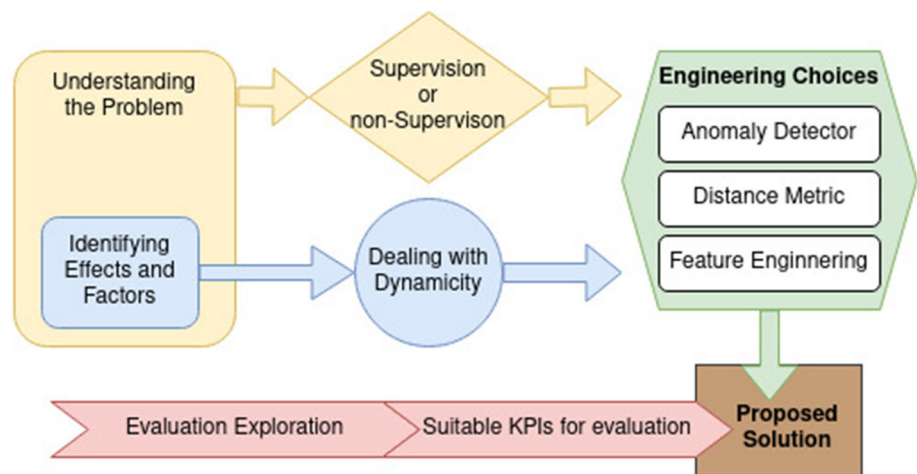
Interestingly, our approach to relying on such an unsupervised technique is in line with the state-of-the-art in time series anomaly detection Paparrizos et al. (2022), Schmidl et al. (2022). More specifically, Paparrizos et al. (2022) is a state of the art benchmarking of time series anomaly detection techniques, using more than 10,000 time series to test the performance of unsupervised detectors, while Schmidl et al. (2022) provides a comprehensive evaluation considering classical machine learning, DL (both reconstruction and forecasting approaches), stochastic learning, statistics, outlier detection and data mining techniques, in settings of both univariate and multivariate data. The thorough evaluation in these studies shows that unsupervised techniques are not inferior to supervised techniques even when training can be performed and DL-based solutions do not necessarily lead to better performance. The research in this work is aligned with such directions.

The remainder of our work is structured as follows. Next, we discuss the related work. In Sect. “Problem setting description”, we describe the setting, its challenges and the baseline PdM technique. We present novel PdM-oriented KPIs in Sect. “Which KPIs are actually good for data-driven PdM?”. In Section “Engineering a PdM solution in practice”, we discuss 4 main engineering problems when deploying PdM solutions in practice. We conclude in Sect. “Conclusion and future work”.

Related work

In PdM, when labels are available, then the prediction of failures does not differentiate from any problem that uses a supervised model. For example, the authors in Li et al. (2018) use a convolution neural network (CNN) to predict the Remaining Useful Lifetime (RUL) of assets, by transforming multivariate time series in a two-dimension matrix and leveraging one dimension convolutional filter to catch sequential information. Similarly authors in Zhu et al. (2023) use a DL architecture to predict the RUL in turbofan simulation data. In more detail their proposal is the use of attention mechanism combined with a CNN, and fused with raw data after they passed a fully connected layer before extracting the RUL. In the work Li et al. (2023) a CNN model is chosen too, to perform for fault identification. Authors here before applying the raw data to the model, explore how to produce more informative features and propose an improved an improved transformation of signals to images method. Similarly, in a PdM case Chakroun et al. (2024), where the objective is to estimate the degradation of power transmitters in robots, the authors perform a massive data analysis before applying a supervised learning solution. After that, a Discrete Bayesian Filter for estimating degradation is used to enhance scheduling maintenance in smart factories. In addition, when dealing with labeled data, plenty of autoML solutions are applicable Tornede et al. (2020), which mitigate the need for fine-tuning. When supervised learning is employed, it is common to use datasets like Azure’s one, as mentioned in the introduction, e.g., Zonta et al. (2022). Such datasets are suitable for classification problems based on telemetry data from several machines; on the contrary, our contribution targets cases where supervised learning is not fruitful.

Fig. 2 Flowchart of our study



The survey in Zhang et al. (2019) provides an overview of the current state-of-the-art data-driven methods in PdM of industrial equipment. One of the conclusions stated in this work is that unsupervised learning with unlabeled data is considered a significant research direction for the future since the current achievements of AI are mainly focused on supervised learning using labeled data. The need for unsupervised techniques is also stated in a survey of DL models for PdM Serradilla et al. (2022), along with XAI. Additionally, in the survey (Theissler et al., 2021), the challenges of PdM in the automotive industry are discussed. One of them is the underlying assumption that the models are trained using representative data, which is something that is not neither fulfilled in the automotive industry nor in our case. Note that in the case where several forms of normal data are collected, then semi-supervised PdM solutions become applicable, e.g., as in Hundman et al. (2018), Silva Arantes et al. (2021), Munir et al. (2019), Zhang et al. (2019).

When we need to resort to unsupervised techniques, as in our case, things get more complicated in terms of design choices and configuration selection. Although there are suggestions for unsupervised anomaly detection AutoML systems Zhao et al. (2021), Zha et al. (2020), these do not address the challenges encountered in our setting. More specifically, Meta-OD Zhao et al. (2021) selects the outlier detector that best performs on existing datasets that are similar to a target dataset, where the similarity is calculated according to meta-features. The main limitation that prevents the use of such a solution in our case is the need for the whole target dataset to decide the best detector, while we are interested in real-time detection. The same severe limitation, that is no support for real-time analysis, holds for the Meta-AAD solutions (Zha et al., 2020), too. This method uses reinforcement learning to train a model, using different labeled datasets, to decide which instances to provide to an analyst as potential anomalies, given a new dataset.

One way to accomplish PdM via anomaly detection is by leveraging *forecasting* techniques to predict the next data points and measure the deviation from the real value. Examples of such techniques are Hundman et al. (2018), Munir et al. (2019), Makridis et al. (2020). In these solutions, a training set consisting of normal data is required. Regarding Hundman et al. (2018), a DL model is used to forecast future telemetry values and measure the error from the real data to calculate the anomaly score. Similarly, in Munir et al. (2019), a convolution neural network is proposed to predict the next value of time series data, and calculate the anomaly score. Additionally, the authors in Zhang et al. (2021), Thiagarajan et al. (2020) use an automatic Prophet model Taylor and Letham (2018) as a forecasting model. Prophet is a forecasting technique developed to automatically fit nonlinear trends of data seasonality. We take advantage of the relative easy applicability of Prophet and use it to test the forecasting error in our case as well.

Furthermore, another anomaly detection methodology consists of *reconstruction* techniques. The authors in Zhang et al. (2019) use the normal data to train a convolution network to reconstruct signatures of the data. The signatures are computed using the correlation of data in a window, and anomalies can be spotted in poorly reconstructed signatures. Using the reconstruction error of data for a PdM application, when sufficient amount of normal data are collected, can be done using state-of-the-art methods like Audibert et al. (2020), Xu et al. (2018), Tuli et al. (2022) too. We pick the Tuli et al. (2022) to test the usage of a (transformer) deep reconstruction model in our PdM case given also that is proposed for non-supervised settings that are similar to our problem case and its evaluation shows superior performance compared to alternatives.

Similarity-based techniques are also applied to measure the degree of data anomalies. An unsupervised algorithm to detect failures in bridge joints is proposed in Diez et al. (2016), where normality is modeled using clustering algo-

rhythms or one class support vector machine Diez-Olivan et al. (2018). We also employ a distance-based baseline solution. Another unsupervised approach for vehicle fleet maintenance is proposed in Rögnvaldsson et al. (2018), where authors use the wisdom of the crowd to define normality, using the hypothesis that most of the vehicles are in normal condition on each time point and have common characteristics. The *Profile-Based (PB)* Giannoulidis et al. (2022) solution is used to detect early warnings on the same data as in our case. This algorithm detects the degradation (a.k.a. wear) of a component by measuring the distance between new data and a small dataset collected in the initial operating phase of the component. We use PB as our baseline model to accomplish PdM in our case. Similarly, in Silva Arantes et al. (2021), the authors use windows to calculate features across data and use the training data to calculate optimized thresholds, after cleaning them from outliers. The main difference from PB lies in the form of data: in Silva Arantes et al. (2021), windows are used to produce time series and calculate features, while PB handles fixed sequences of data, where time domain features can be calculated without a windowing technique.

Table 1 summarizes the presented related work, where the methodology limitations and goal of each technique family is shown. In our work, we aim to engineer a PdM solution considering and assessing the potential of all main representatives, namely reconstruction Tuli et al. (2022), forecasting Taylor and Letham (2018) and similarity-based techniques Giannoulidis et al. (2022).

Finally, the engineering problem we face is not focused solely on the technique itself, but also on engineering choices, which may boost its performance. In Paparrizos et al. (2020), Paparrizos et al. evaluate the choice of distance metrics and discuss several relevant aspects, while they suggest SBD (see Section 5) as a robust choice. However, to the best of our knowledge, no such work has been conducted for unsupervised anomaly-based techniques tailored to a PdM problem.

Problem setting description

This work focuses on a cold-forming press that is a part of the production line for cutters of the Philips factory; it is inspired and motivated by the promising results in Giannoulidis et al. (2022). The aim is to monitor the state of the multiple components that assemble the press. Specifically, the press is composed of six different modules placed side-by-side. Each different module has its die, which is designed to morph a metal piece to be ready as an input for the next component of the production line as shown in Fig. 3. A limitation is that there is no means to monitor directly the condition of the modules (i.e. if they are scratched or worn) other than stopping the press from operating and inspecting the module. To

overcome this limitation piezoelectric sensors are installed in each module that collect the force signature of press strikes. These sensors produce 500 values for a single strike, for each different module, where each value corresponds to a specific angle as the press performs a single punch. These signals are collected at periodic intervals, which are not fixed. As a result, we have access to only one strike force signal in each period, even though the press may have executed many strikes. Based on these data, we try to estimate the degradation of modules and decide when is the right time to take action and prevent costly failures using anomaly detection techniques. However, in PdM, the objective is slightly different than merely detecting anomalies in time series. There are no multiple anomalous periods, in which we are interested but a single one corresponding to a restricted time frame before the failure, called prediction horizon (PH). The time frame should be restricted so that too early alarms are punished. Moreover, any correct anomaly detection should not be extremely close to the actual failure to allow for actions to be taken, so we need to account for a buffer period (a.k.a. lead time) between the end of the PH and the failure.

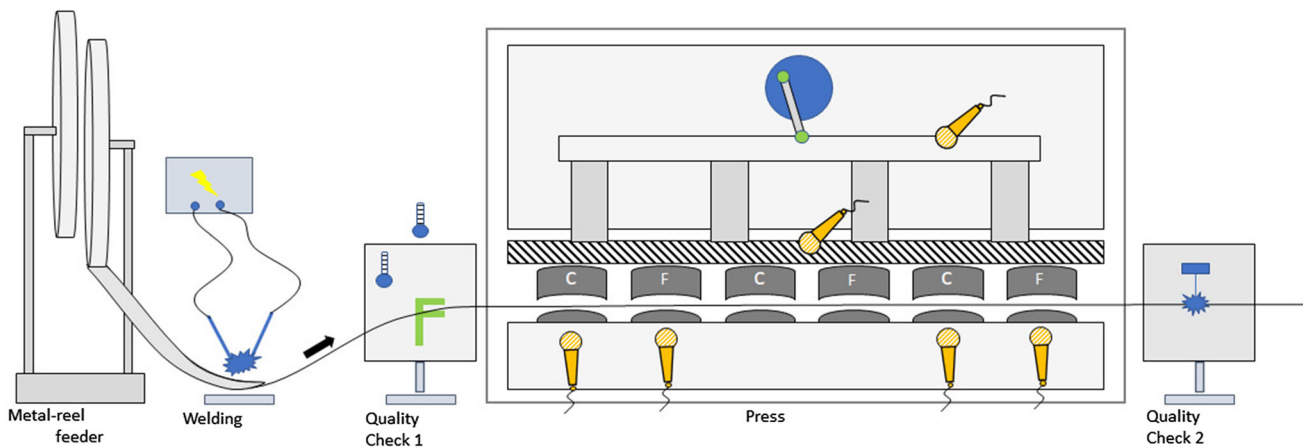
Although RUL models are widely used for PdM, we cannot apply this approach to our run-to-failure data due to several reasons. Firstly, our run-to-failure data come from different systems and components that may change over time and do not exhibit smooth degradation. As a result, we have limited data availability that corresponds to the same setting, i.e., the same component, to build reliable models. Additionally, there are many types of failure modes, and for most of them, early warning signals are not evident in the data. Moreover, in our case, similar data may correspond to a healthy state in one episode and a near-failure state in another episode. This prevents us from using a supervised algorithm to estimate the remaining useful lifetime based on the collected data.

As in many real-world problems, many factors might impact the behavior and state of the press, some of which remain unknown. Below we note some factors that affect the performance of the press.

- *Cold start and speed*: in general, each time the press starts after a pause period, it has a lower temperature and lower speed (which are correlated). At these circumstances, it is common to observe an increasing amplitude as the press starts operating and gaining higher speed.
- *Material change*: the material used as input may affect the operation of modules, since characteristics like thickness cause change in the derived force signal from the modules. Moreover, when material change takes place, a welded area between the old and new material batches may produce instantaneous outlier punches.
- *Type of Modules*: the different types of the same module may have small differences, which can be seen from the

Table 1 PdM related work summary

| Techniques | Method | Supervision | Goal | limitation |
|--|--------------------------|-----------------|------------------------|-----------------------------|
| Li et al. (2023), Li et al. (2018), Zhu et al. (2023), Zonta et al. (2022) | Regression | supervised | RUL | need for vast historic data |
| Chakroun et al. (2024) | Discrete Bayesian Filter | supervised | Degradation estimation | need for vast historic data |
| Tornede et al. (2020) | Auto ML | supervised | RUL | need for vast historic data |
| Hundman et al. (2018), Munir et al. (2019), Makridis et al. (2020), Taylor and Letham (2018) | Forecasting | semi-supervised | Anomaly detection | need many healthy data |
| Zhang et al. (2019), Audibert et al. (2020), Xu et al. (2018), Tuli et al. (2022) | Reconstruction | semi-supervised | Anomaly detection | need many healthy data |
| Silva Arantes et al. (2021) | Similarity-Based | semi-supervised | Anomaly detection | need many healthy data |
| Zhao et al. (2021), Zha et al. (2020) | Auto-ML | unsupervised | Anomaly Detection | no real-time |
| Diez et al. (2016), Diez-Olivan et al. (2018), Silva Arantes et al. (2021), Rögnvaldsson et al. (2018), Giannoulidis et al. (2022) | Similarity-based | unsupervised | Anomaly Detection | tuning-sensitive |

**Fig. 3** A cold-forming press

resulting force signal when they operate. Therefore, the different types of modules may affect the operation of another module downstream, e.g., the module 2-type A operating with module 1-type C has different behavior than operating with module 1-type B. In general, due to the ordered placement of modules, operation of preceding modules may affect the operation of succeeding modules. For example if a preceding cutting module does not cut well the metal strip, the downstream module is affected.

Profile-based anomaly detection: our baseline technique

Previous work has shown that the Profile-Based (PB) algorithm is effective in detecting component degradation Giannoulidis et al. (2022). This algorithm detects the degradation of a component by measuring the distance between new data and a small dataset collected in the initial operating phase of the component. The algorithm is separated into two main steps: 1) Construction of profile and 2) Anomaly detection. With regards to the first step, a profile is constructed from continuous time points that have small distance between them. More accurately, these continuous time points form a set of data-points. If the maximum distance between the data of this set is less than a threshold, then we accept this set as a profile. Essentially, the profile is a homogeneous representation of the initial state of the component. To eliminate the need of a threshold to construct the profile, we use the following methodology: instead of checking if the set has maximum inner distance smaller than a threshold, we just check if its distance is smaller than the next candidate set. If that is true, then we accept the set as a profile. After a profile is constructed, we move to the second step for anomaly detection. Whenever a new point arrives, we calculate its distance from the profile, which is determined from its closest point in the profile. Each time the anomaly score is calculated as the median of distances of the last n instances. This is done to observe the trend of distances from upcoming points, while in case of $n = 1$, only the last point distance is used as anomaly score. To produce alarms, we use a threshold value that can be calculated using several techniques as described in (we choose the self-tuning one since, in Giannoulidis et al. (2022), evidence regarding its improved efficiency is provided).

The second and the third research questions mentioned in the introduction refer to engineering the PB solution, in order to render it (more) effective. The first research question deals with the assessment of such efficiency, while the fourth research question explores other non-similarity-based techniques.

Effect of factors in practice

To understand the complexity and the dynamicity of the problem in practice, we discuss some indicative cases, called episodes, as shown in Fig. 4. In the figure, in episode 7, we can identify a change point in anomaly scores produced by PB that is aligned with a change in module 1 due to preventive maintenance. This phenomenon supports previous indications of the impact of different types of the same module on the press' operation, i.e. different types of the same module may have varying effects on the other modules of the press.

In the episode 15 of module 3 depicted on the top-right part of the figure, a sharp increase is observed after a material change in the middle of the episode, which is correlated with a sudden change in the behavior of module 6 too. This change is depicted as a step increase in the amplitude of the force signal of module 6 around the 180-degree angle, as shown in Fig. 5. Note that seemingly small changes have a big impact. We can see that the force amplitude changed only around the 180-degree angle, while the remaining signal remains at the same amplitude. In addition, by looking at the same punch signals of module 3 in Fig. 6, we can observe that after the material change (with grey color), we have higher amplitude in the two peaks of the force signal and lower amplitude in the remaining signal than before the material change (black color). Interestingly, the error sharply dropped after a change in module 2 due to maintenance. The case here is the opposite of the previous one, where module 3 returns to a similar state to that before the change of material. More specifically, looking at the signals of Module 3 before (black) and after (grey) the change of module 2 in Fig. 7, we observe that the force amplitude increases in the peaks and drops in the remaining signal. After discussing with engineers, at the time of writing, we cannot justify the reason for that effect, therefore we simply notice it.

In another run-to-failure episode, episode 16 (see Fig. 4), we notice a generally increasing error. The produced error shows a big rate of increase which is followed by a sharp increase after a "gap" time (where no data are collected). Here, we observe how "cold starting" affects the produced error. The cold start can be observed also from the amplitude of the signals where we have a gradual increase in force. The duration of the cold start can vary (maybe depending on factors such as temperature of the press and material). The effect of cold start is depicted in Fig. 8, where force signals of Module 2 in the start of the episode are plotted. The episode 17 depicted in Fig. 4 of module 2 is one of the biggest episodes in our dataset, which ends with a failure on the die. Examining the big picture, we see an increasing trend of error with fluctuations over time and a sharp increase due to the change of Module 1. As described earlier, the different type of the modules impacts on the behavior of the other modules

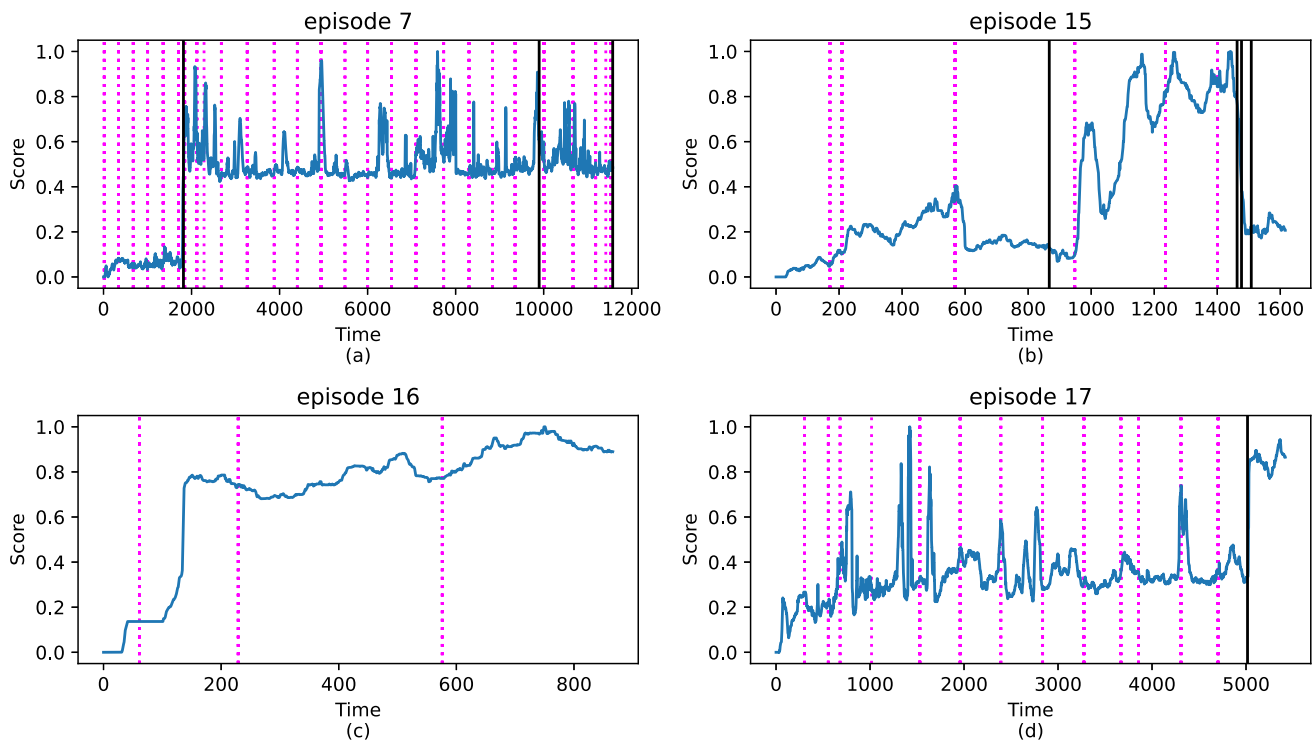


Fig. 4 Results of PB in four run-to-failure episodes using euclidean distance. Vertical dotted lines show change of the metal strip (and welding), and solid lines show maintenance in other modules

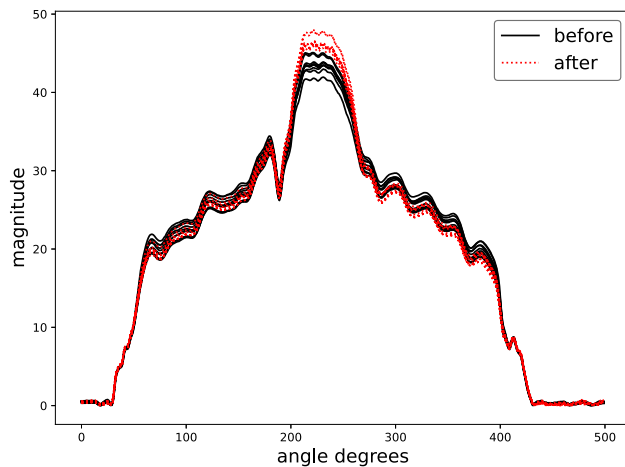


Fig. 5 Episode 15: Module 6 before and after the change material

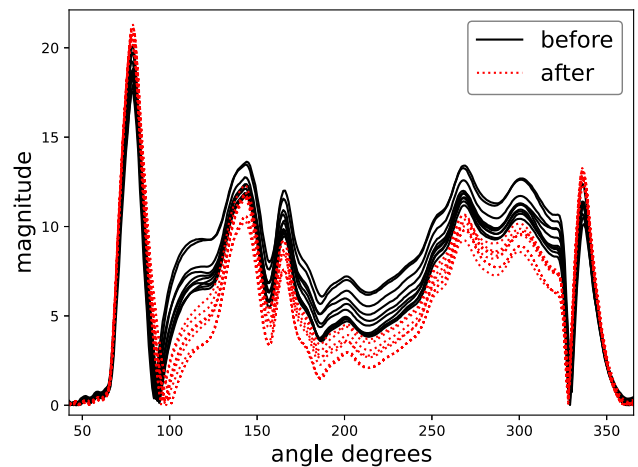


Fig. 6 Episode 15: Module 3 before and after the change material

too. By plotting the signals before and after the change of module 1 type, we see a known behavior change of Module 2. In Fig. 9, we see that before the Module 1 changes, the signal shows a small step around 160 degrees (black color), which cannot be observed after the module 1 change (red color).

Our approach to dealing with the Dynamicity of the environment

In the previous section, we have discussed volatility aspects of the operation environment, which render early warning-oriented anomaly detection particularly challenging. In our solution, we propose a novel pipeline, which introduces profile reset in the baseline PB method, while detailed engineering aspects are dealt with later in this work. This pipeline generalizes to other anomaly detection methods.

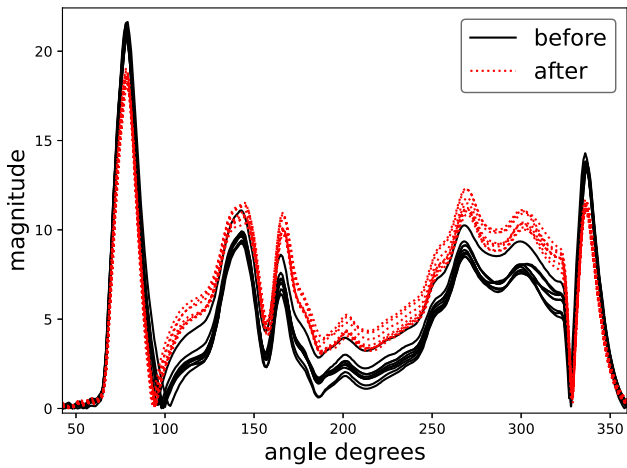


Fig. 7 Episode 15: Module 3 before and after the change of Module 2 die

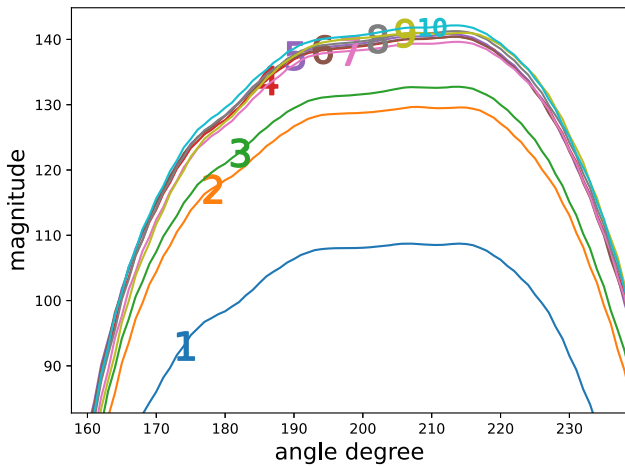


Fig. 8 Episode 16: Cold start of Module 2. Increasing number indicates time order

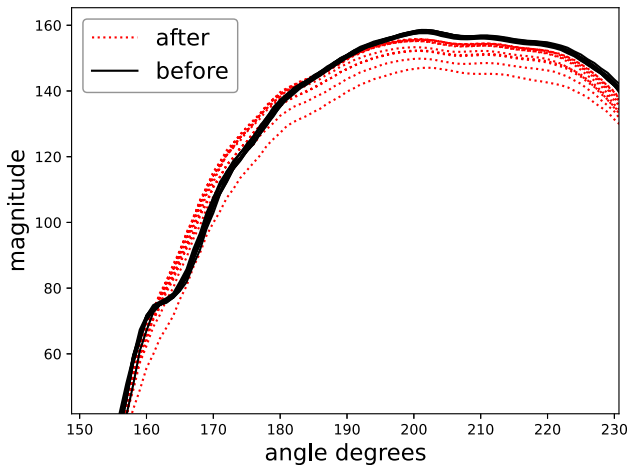


Fig. 9 Episode 17: Change in force signal of module 2 when the module 1 changes

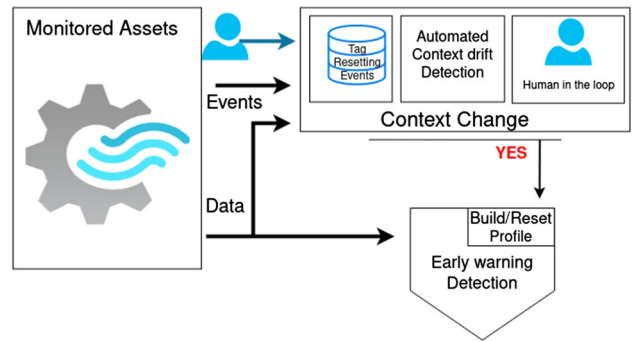


Fig. 10 An illustration of the reset policy pipeline

In a nutshell, to address dynamicity, we propose a generic readjustment policy, which is independent from the actual method and enhances the PdM solution with context awareness. Considering that the change of environment can alter the characteristics of the data, including the historic data considered normal, we perform a readjustment of the utilized method upon every noticed configuration change. A representation of such policy is depicted in Fig. 10.

The monitoring assets produce different kind of data and events. Given an early warning detection method that builds a profile based on streaming data to produce alarms, we could use a human in the loop, expert feedback or automatic process that signals the need to reset the underlying technique (and build the profile even from scratch) in order to capture the environmental changes. This can be done automatically, e.g., using a technique that detects environmental changes using data from other sources rather than using only actual monitored data. In our case, these data could be events produced from the monitored press, or data from adjustment modules in the topology of the press. Another way to accomplished that is via expert’s feedback, where the expert could provide simple rules, which indicate when to refit. Interestingly, this can implemented as a human-in-the loop process, where an expert could send a signal for refitting that models. In our case we leverage the expert’s feedback, where the expert tags a subset of events types produced from the press, as resetting ones. The tags refer to the changes or maintenance actions of the modules of the press, which are recorded.

Which KPIs are actually good for data-driven PdM?

Quantified evaluation is crucial for testing new or existing ideas and determining their capabilities and impact on existing problems. Data-driven PdM boils down to using ML solutions, which are typically evaluated based on metrics such as true/false positives/negatives. Our purpose is twofold: first to explain why previously proposed methods for evaluat-

ing PdM during run-to-failure episodes are inappropriate to a smaller or larger extent, and second, to propose a new metric that is tailored to PdM scenarios like the one described in the previous section. Note that questioning existing approaches to evaluating failure prediction is not new, e.g., it is discussed in Salfner et al. (2010), which highlights the role of parameters such as data size, prediction horizon (PH), and lead time.

Episode-level performance

This type of assessment focuses on whether an anomaly-based detection technique managed to lead to PdM actions more directly. Given a set of run-to-failure (or run-to-target event) episodes, this evaluation metric characterizes each one of the episodes as false positive (FP) or false negative (FN) or true positive (TP). Then the recall, precision, and F1 values are used to summarize the overall performance of the algorithm. As such, we characterize a whole episode as FP if an alarm is raised before the PH. In the real world, this would lead to an investigation or unnecessary maintenance of the component and the production would stop. Moreover, an episode is TP if anomalies are detected only within PH, which results in the full usage of the component. Finally, if no alarms are raised although the episode ends with a failure, we characterize the episode as FN. It is worth mentioning that in such a scenario, there is no sense in distinguishing between micro and macro calculation of F1 score. This evaluation setting was used for an aviation PdM problem Korvesis et al. (2018).

The main problem with this evaluation metric is that, in a specific setting, it cannot differentiate between a dummy method that always reports an alarm while a more preferable method starts reporting alarms just before PH begins until the end of the episode. Such an example is depicted in the Fig. 11. Both of the methods, achieve zero precision, recall, and F1 score (since all episodes are characterized as false positives). Although, for the specific example, the issue can be resolved by increasing the predictive horizon period, in general, this issue limits the ability of that evaluation metric to be used for comparison purposes, as even some isolated anomalies during the first part of the episode lead to the characterization of the episode as FP, i.e., there is extremely low flexibility.

pw-ROC metric

The pw-ROC (Preceding Window Receiver Operating Characteristic) is a new evaluation metric for PdM scenarios Carrasco et al. (2021). It is based on the ROC curve and uses a non-overlapping time window to aggregate anomaly scores. The data is split into episodes, with each episode consisting of data from one maintenance event or failure to the next. The window aggregation process starts at the end of each episode

and works backwards, using a user-specified window length w to calculate an aggregate score (e.g., mean or median) for each window. The resulting time series of aggregate scores is used to calculate TP, TN, FP and FN values. The w parameter must be between 1 and the length of the shortest episode, and can be varied depending on the problem. For example, we count a TP if the aggregate score exceeds a threshold in the last window, which coincides with the PH. Obviously, the choice of aggregation function affects the results.

The w parameter is associated with the PH. The main difference being that in the pw-ROC framework, the w parameter is used for both positives (periods labelled as anomalous) and negatives (periods labelled as normal) intervals. Although the windowing technique results in a more holistic view in such scenarios, there are two main issues. First, by evaluating aggregated results we actually evaluate something different than the anomaly detection output (we may opt for an optimal aggregate function rather than a more suitable anomaly detector), besides the fact that aggregated results may mean that we impede real-time application. Second, the applicability of the ROC in anomaly detection and PdM scenarios has been studied in depth in works such as Davis and Goadrich (2006), Saito and Rehmsmeier (2015), and is found to be problematic, whereas an optimized AUC (Area Under the Curve) - ROC (Receiver Operating Characteristic curve) does not necessarily provide an optimized AUC-PR (precision recall curve), where the latter is more suitable for imbalanced datasets. Note that in practice, when applying PdM in reality, we are interested in the performance of the specific configuration, e.g., measured as a F1 score, that we choose rather than in the potential of the technique in general measured with AUC metrics. See Fig. 12 for a counterexample. We elaborate on this in the following subsection.

On Improving PR and ROC curves

When evaluating the performance of a classification algorithm, ROC and PR curves are typically calculated using precision, recall, and false positive rate (FPR). These are well-established metrics that are widely used in machine learning and data analysis. However, when working with time-series datasets, range-based variants of precision and recall can also be used. In both cases, the PR curve is created by plotting precision and recall (or range-based precision and recall) on the x- and y-axes, respectively. A recent work Paparrizos et al. (2022) has proposed using range-based PR and ROC curves (and the volume under the surface) for more flexibility in measuring detected real anomaly ranges. More specifically, the work in Paparrizos et al. (2022) is well documented and deals with the limitations of existing evaluation metrics and in the same time provides a non-parametric evaluation metric. While typical AUC and ROC get an evaluation metric for multiple values of threshold, the main idea of

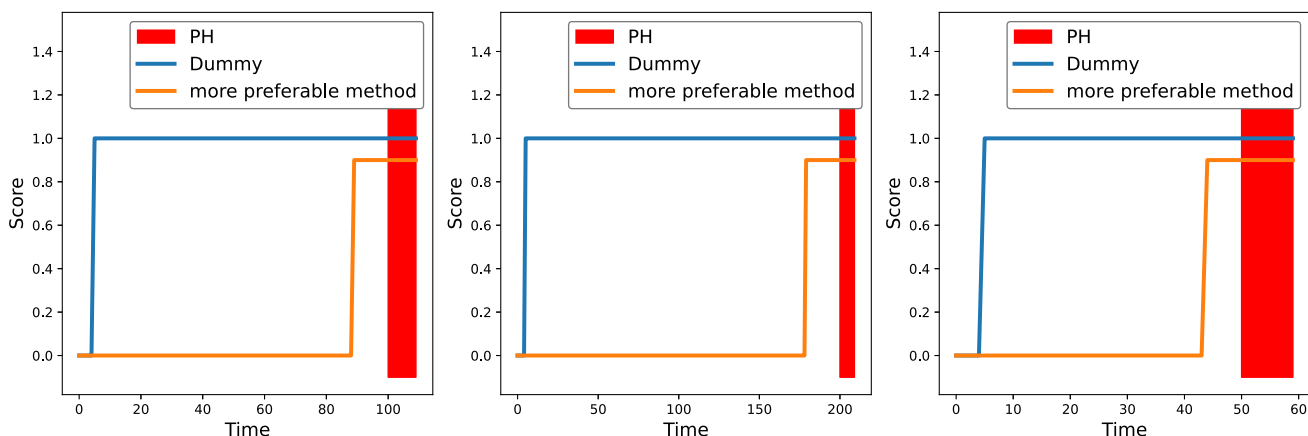


Fig. 11 Alarms of dummy and a more preferable method

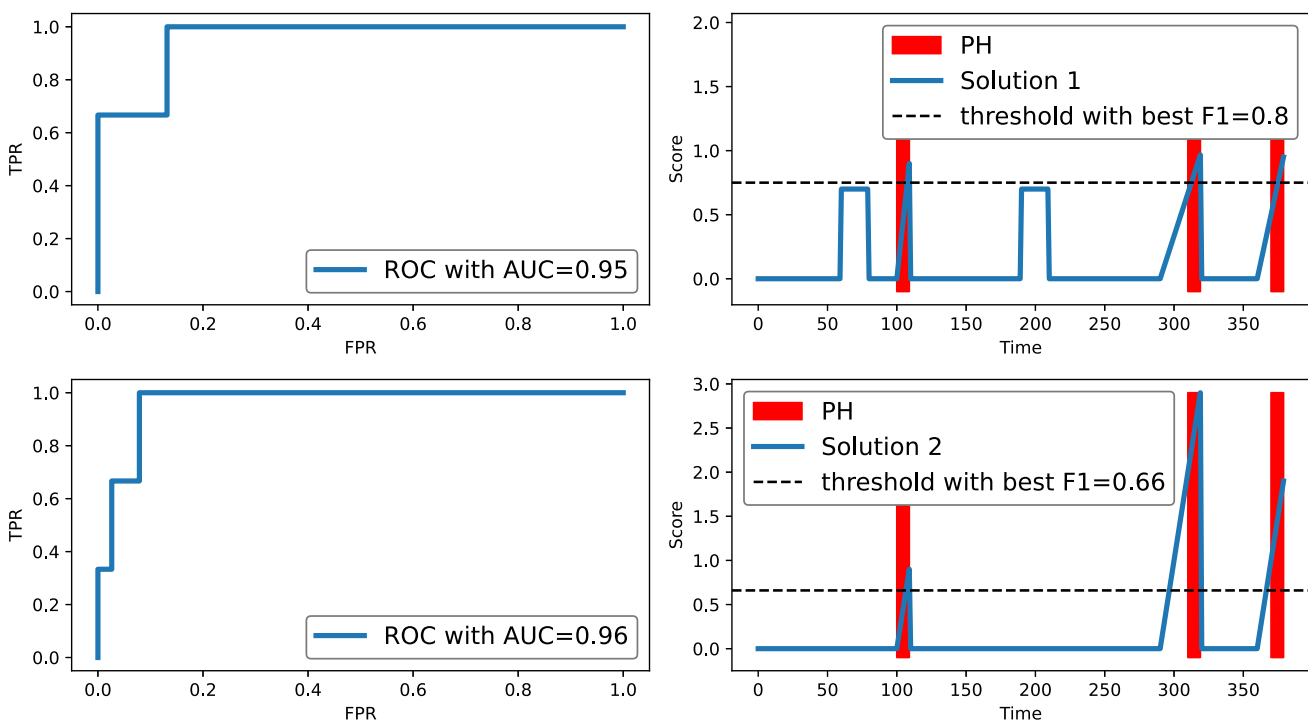


Fig. 12 The images in the right columns show the anomaly scores produced from two potential solutions, trying to detect anomaly ranges (coloured areas). In the left column depict the ROC curves for the two solutions, where, the conclusion is that best AUC-ROC doesn't means best F1 score

Paparrizos et al. (2022) is that using VUS (Volume Under the Surface) we can get an evaluation metric using multiple parametrization in more than one parameter (such as threshold and predictive horizon). In the same work, it is mentioned that "An alternative solution is to learn necessary parameters and thresholds. However, such a solution works only under supervised settings and may impact the generalizability to new datasets.". Although we agree with that statement and despite the fact that we operate in an unsupervised setting, the objective of utilizing a PdM solution is the prediction of failures in particular case. Therefore we are not interested in the general performance of our techniques in anomaly detec-

tion tasks, but in their application for a specific task. Finally we consider good practice to test multiple time parameters in evaluation such as predictive horizon, as VUS does, in cases we are not sure about the time span of anomalies, as explained later. Besides, fine-tuning unsupervised PdM solutions is a topic that there is significant progress, e.g., Giannoulidis et al. (2022) as already discussed.

In summary, for our problem of predicting forthcoming failures using run-to-failure episodes, we can compute the PR curve (range-based or not) and use it along with the area under the curve (AUC) to compare potential solutions. However, when deploying the solution to a production line, we need

a discrete output (typically in a binary form) such as alarms or safe/warning/failure indications. A common approach to this problem is to use the harmonic balance between precision and recall (F-measure) to find the parameter that results in the optimal pair of recall and precision (as depicted in the PR curve). However, this raises several questions: a) if we use the F-measure to decide on the final solution, why not use it to compare the performance of different solutions? b) Does a solution with a larger PR-AUC lead to parametrization with a larger F-measure as well? To answer the second question, we have already provided a counterexample in Fig. 12. Regarding the former question, in our work which focuses on a specific real-worlds scenario, we adopt F-measures as the main means to compare deployed solutions. However, instead of F1, the F2 metric can be used to calculate the final score, since in many industrial settings, the recall is more valuable than precision (in the sense that a failure is costlier than a needless inspection).

Fault detection alarm rate

The authors in Jiang and Yin (2019, 2017) demonstrate the capabilities of the DB-KIT Matlab toolbox for fault detection. To evaluate the performance of the proposed solution, the fault detection rate (FDR) and false alarm rate (FAR) are used. Essentially, these metrics are calculated as the numerical ratios of valid and false alarms to the total number of samples. Each episode is divided into two periods: the faulty and the healthy one. The division is based on the occurrence of a fault, with the period following the fault being the faulty period. FAR is calculated as the ratio of alarms to the total number of samples in the healthy period, and the FDR is the ratio of alarms to the total number of samples in the faulty period. The goal is to have a low FAR (i.e., minimal false alarms in a period without faults) and a high FDR (i.e., maximum alarms in a faulty period). The FDR can be thought of as a measure of recall, while the FAR aims to quantify false positives, and resembles modern approaches to evaluate time series anomaly detection, which will be discussed next. Apart from the essential difference in purpose between detection and prediction, practically, in predictive tasks, a lead time period prior to the fault occurrence shall often be employed in practice. These differences make the FAR and FDR suitable evaluation metrics only for fault detection tasks.

Our proposal for an evaluation metric for PdM

Following-up on the discussion above, our starting is the proposal in Jacob et al. (2021), where range-based evaluation metrics to calculate range-based precision and recall are proposed. These evaluation metrics are tunable Tatbul et al. (2018), where the parameters are selected based on the problem. In Jacob et al. (2021), four different parametriza-

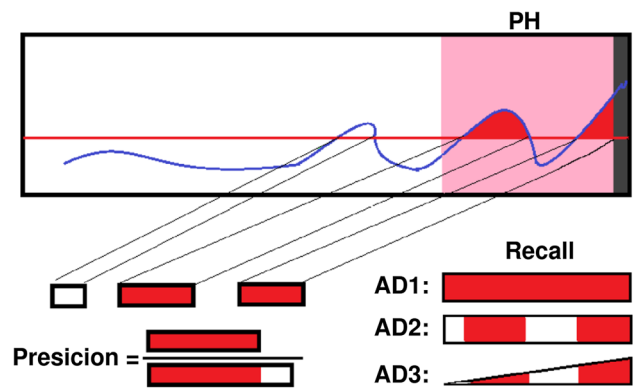


Fig. 13 Given an episode and the reported alarms, this figure shows how the Recall and Precision are calculated. The blue line represents the anomaly score and the red line the threshold. The pink part of the signal is the Predicted Horizon (PH) time window and the dark grey the buffer zone (lead time)

tions are selected where each one is stricter than the previous one, to evaluate the results of anomaly detection algorithms. Inspired by their work, we propose three range-based evaluation parametrizations suitable for PdM.

We are interested in detecting faulty behavior of a component within the PH. Any alarms raised during the buffer period are practically ignored. Finally, the length of the PH is application dependent and can range from a few hours to several days. Adapting the rationale of Jacob et al. (2021) to PdM, in each episode, we propose three anomaly detection (AD) evaluation scores regarding recall (AD_1 - AD_3), where each one adds extra requirements to the previous one, as shown in Fig. 13. Moreover, it always holds that $AD_1 \geq AD_2 \geq AD_3$. Starting with AD_1 , we get the maximum recall value, i.e., 1, if an alarm is raised within PH, which means that we “predict” the failure. AD_2 recall is proportional to the relative size of alarms made in PH ($\frac{\#alarms_in_PH}{|PH|}$). AD_3 emphasizes on the capability of the technique to detect anomalies as we move towards the end of the PH. The key difference between AD_3 and AD_2 is that the former employs a weight that is inversely proportional to the time difference between the alarm and the failure time. The precision is always computed as the ratio of the anomalies detected within the PH over all anomalies detected. The bottom part of the figure visually summarizes the precision and recall variants.

Engineering a PdM solution in practice

In Sect. 3, we pointed out several factors that influence the operation and data of the press, most of them not necessarily related with the wear of the modules. In this section, we craft a solution assembled from a suitable selection of pre-process, domain expertise and anomaly detector class (distance based,

DL and forecasting model), leading to a solution that is capable to distinguish the wear of modules with higher scores than other cases (i.e., present of factors). In other words, we build an unsupervised solution engineered to be robust to the several factors adhering to the architecture of Fig. 10. Although in the recent years, many Auto Machine Learning (AutoML) models have been developed, the majority of them are dedicated to supervised problems, while the ones dedicated to unsupervised anomaly detection are not robust yet for streaming applications Bahri et al. (2022), as pointed in Sect. “Related work”.

To evaluate engineering decisions, we use the proposed evaluation metric of Sect. 4. The test dataset contains 12 run-to-failure plus 8 episodes where no failure appeared. Three different PH periods are used, namely 8, 12 and 24 h, to measure the recall and precision. Moreover a lead time of 1 h is used since the objective is to prevent failures thus there should be some time to react. In Fig. 14, we can see an example of the PB algorithm using SBD (Shape-based distance) Paparrizos and Gravano (2016). The PH here can be seen from the light red periods, while the lead period is in grey. In some episodes, there are grey colored periods in the middle of the episodes, which indicate a period that we ignore, since there was an expected change of behaviour in the modules (irrelevant to a failure). The anomaly scores of run-to-failure episodes (first 12 episodes) are in blue, while green color is used for the episodes that end with no failure (8 bottom episodes). Finally, we provide the implementation of our methods and the AD evaluation for PdM, both implemented using Python 3.8, along with the 20 operational episodes data.⁴

All AD levels are important and each one has each own meaning as describe above. In our problem, we would like to get continuous alarms before the failure happens, so we pick the AD_2 as the basic evaluation measurement to compare the potential solutions, without ignoring the other two. Moreover we prefer a method that performs better for 8 h PH than 24 h, since it indicates that it can predict the failure without losing too much production.

Finally, for all engineering choices regarding the detection technique, when a training (or fitting to the data) is needed, we use a homogeneous representation of data (profile) exactly as described in PB anomaly detection 3.1, where the examined sizes are equal to 30, 60 and 100 samples. Moreover, we use the same smoothing of anomaly scores as described in 3.1 for all considered techniques, which is equal to 30 (i.e., each time, we use the last 30 anomaly scores to calculate the final score as their median).

Comparison of anomaly detection methodology

As mentioned in Sect. “Related work”, we can apply three methodologies to produce anomalies that act as early warnings of upcoming failures, forecasting, reconstruction-based using DL techniques and similarity-based anomaly detection. Here, we provide evidence regarding our choice for the latter though showing that the former two are inefficient.

Behavior of a forecasting technique

Regarding the forecasting technique and to illustrate the relative difficulty of our problem, we peek the well known Prophet Taylor and Letham (2018), which was used for the task of anomaly detection in several works, e.g., Zhang et al. (2021); Thiyagarajan et al. (2020). As mentioned already, Prophet is a forecasting technique developed to automatically fit nonlinear trends of data seasonality. We set the seasonality equal to one punch time in order the model to catch the periodicity of punches. Finally, we fit the model to data from the start of each episode (using profile size equal to 100). Optimizing the threshold for PH 8 h, we get 0.2, 0.3 and 0.4 F2 score using AD_2 , for 8, 12 and 24 h PH, respectively. The results are better for 8 and 12 h PH when we increase the size of the profile three times leading to 0.27, 0.32 and 0.39 F2 score, respectively. But a major limitation is that as the profile gets bigger, the time of no monitoring grows too, making this solution not feasible. We can see that the engineering decisions we take based on the discussion below on top of the PB algorithm yields much higher F2 scores, e.g., the improvement factor is more than 2X. We can say that just using naively an automatic model (which is efficient in many other tasks), does not provide necessarily a good solution.

PdM using DL for reconstruction

A valid question is whether DL outperforms any distance-based solution, such as PB. As DL models, usually Auto-Encoders-based models Tuli et al. (2022); Audibert et al. (2020) are used for the task of anomaly detection in a semi-supervised fashion. These are DL architectures trained in data considered normal, to reconstruct their input. Then, in the online phase, they use the reconstruction error (i.e. the distance between reconstructed and actual data provided as input) to produce an anomaly score.

To effectively apply a DL model in PdM settings, two main challenges are encountered. Firstly, existence of normally labeled data is rare, also due to the dynamic environment. Even if we collect data from normal operations (which end with no failures), these may not be suitable to train the DL model because of the continuous changes in monitored components, as already mentioned. We deal with such a problem by following an approach that resembles semi-supervised

⁴ <https://github.com/EngineeringUnsupervisedPdM/EngUnsPdM>

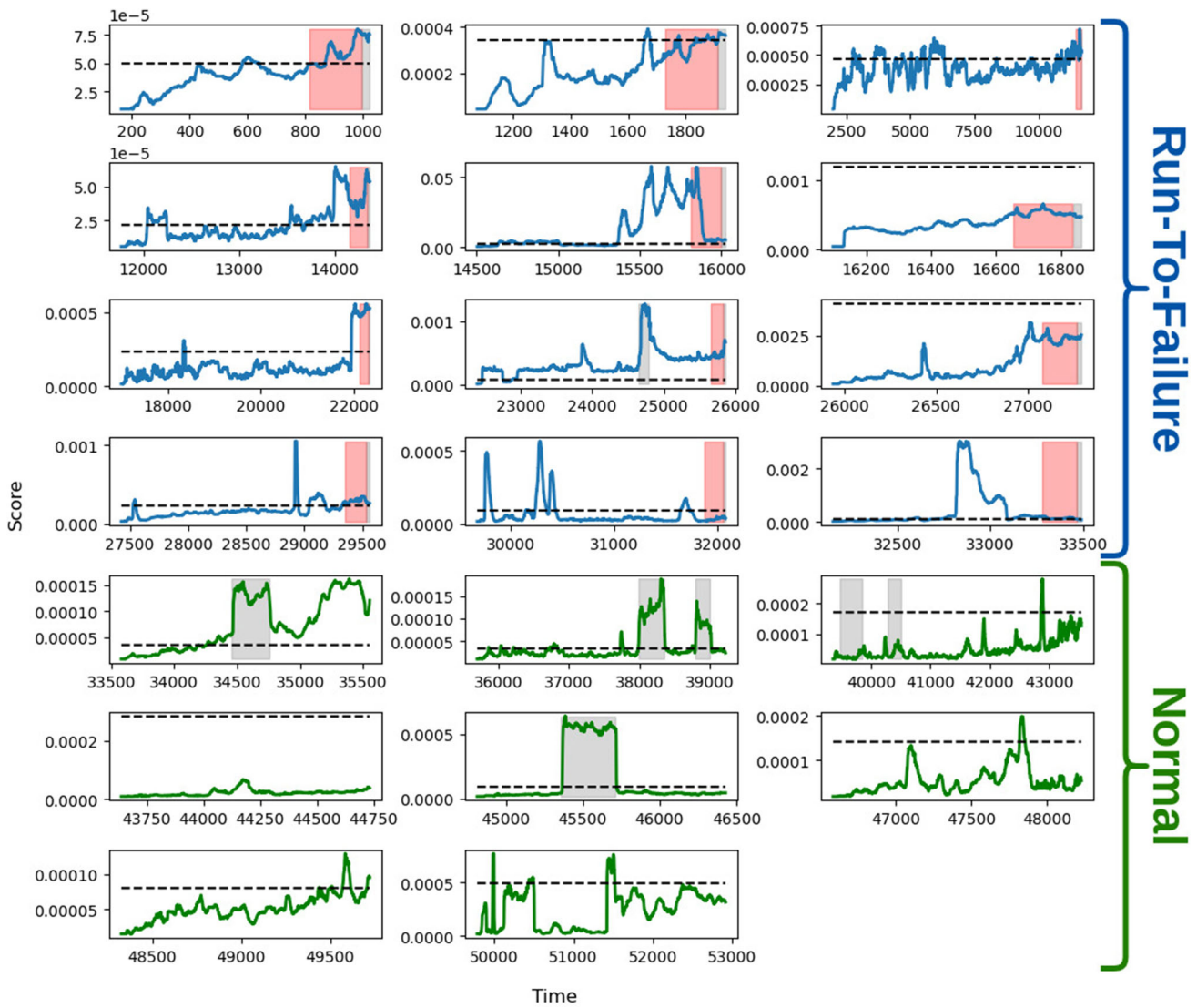


Fig. 14 Results using SBD distance metric with PB algorithm

learning, but does not rely on any labeled data. Similarly to PB, we use the same algorithm to calculate the profile, which is used to train the model. The second challenge encountered relates to the setting of a threshold on the produced loss of the DL model to detect anomalies. Using a small amount of data to train DL models results in an over-fitted model. So, if a naive threshold is used, it would probably produce many false positives. To solve the threshold issue, we choose the self-tuning technique, again as used in PB algorithm. This allows us to conduct an apples-to-apples comparison.

Regarding the DL architecture, we have chosen to adopt the TranAD model of Tuli et al. (2022), which corresponds to a state-of-the-art transformer DL model and is endowed with valuable properties, such as capability to operate with fewer training data than competitors and to reconstruct data after a small amount of epochs. In summary, we train the model using a small portion of data that refers (most probably) to a

Table 2 *PdM_TranAD* F2 score

| PH | AD_1 | AD_2 | AD_3 |
|----|--------|--------|--------|
| 8 | 0.31 | 0.28 | 0.24 |
| 12 | 0.42 | 0.36 | 0.29 |
| 24 | 0.60 | 0.45 | 0.31 |

normal operation of the monitored component. This dataset is extracted at the very beginning of the episode and requires no further domain expertise. After that, we compute the difference between the reconstructed data and the real one and use it as an anomaly score. We expect that the model will be capable of reconstructing normal data with significantly higher accuracy than faulty data. We refer to the TranAD model with the additional extensions as *PdM_TranAD*.

The results are shown in Table 2. At a first glance, we observe that we obtain higher F2 score than the Prophet,

Table 3 F2 score for different distance metric choices, and baseline prophet results

| PH | Distance | AD_1 | AD_2 | AD_3 |
|----|----------------|-------------|-------------|-------------|
| 8 | <i>Prophet</i> | 0.21 | 0.2 | 0.15 |
| | PB-DTW | 0.28 | 0.28 | 0.26 |
| | PB-RBF | 0.29 | 0.27 | 0.19 |
| | PB-SBD | 0.36 | 0.34 | 0.27 |
| | PB-EUC | 0.32 | 0.29 | 0.26 |
| 12 | <i>Prophet</i> | 0.35 | 0.3 | 0.17 |
| | PB-DTW | 0.38 | 0.36 | 0.32 |
| | PB-RBF | 0.45 | 0.38 | 0.24 |
| | PB-SBD | 0.47 | 0.42 | 0.31 |
| | PB-EUC | 0.44 | 0.39 | 0.32 |
| 24 | <i>Prophet</i> | 0.57 | 0.4 | 0.17 |
| | PB-DTW | 0.55 | 0.47 | 0.37 |
| | PB-RBF | 0.60 | 0.41 | 0.23 |
| | PB-SBD | 0.64 | 0.45 | 0.28 |
| | PB-EUC | 0.65 | 0.51 | 0.38 |

Highest values per column are in bold for different Predictive Horizon values

but the results are still relatively low; e.g., the results are significantly inferior to those presented later in Table 3. Note that, these results are highly influenced by the training of the model, since the training set is relatively small (profile size is equal to 100) and not necessarily related to normal data. This leads to the suggestion to use DL models with caution, especially regarding the training phase in dynamic cases as ours, where training data are difficult to collect.

Engineering a similarity-based PdM solution

Now we turn our attention to the similarity-based PB solution. To this end, we do not rely solely to the simple algorithmic logic of PB, but the aim is to engineer a suitable PdM solution.

Impact of distance metrics

As mentioned earlier, the PB algorithm calculates the anomaly score of new data as their distance from the profile. But the choice of the distance metric does make a big impact. In our scenario, we test the Euclidean (EUC, default), DTW, RBF and SBD. DTW is a distance metric that seeks the optimal match between two time series and deals with misalignment of signals. Although our signal consist of dimensions of fixed number, i.e., force over 500 angles, DTW can be useful in scenarios as speed change, which may shift the signal, i.e. the modules may hit harder in to the material as they contact it. Similarly, the SBD (Shape-based distance) distance metric measures the similarity between two time

Table 4 F2 score achieve by PB algorithm using features and PCA

| PH | Solution | AD_1 | AD_2 | AD_3 |
|----|-------------|-------------|-------------|-------------|
| 8 | PB(feats.) | 0.37 | 0.34 | 0.26 |
| | PBr(feats.) | 0.47 | 0.43 | 0.38 |
| | PB(pca) | 0.33 | 0.30 | 0.22 |
| 12 | PB(feats.) | 0.48 | 0.41 | 0.28 |
| | PBr(feats.) | 0.60 | 0.53 | 0.44 |
| | PB(pca) | 0.42 | 0.38 | 0.26 |
| 24 | PB(feats.) | 0.63 | 0.49 | 0.30 |
| | PBr(feats.) | 0.83 | 0.63 | 0.48 |
| | PB(pca) | 0.56 | 0.45 | 0.27 |

Highest values per column are in bold for different Predictive Horizon values

series based on coefficient-normalized cross-correlation as proposed in Paparrizos and Gravano (2016). Finally, the RBF kernel similarity measure is tested too (with $\sigma = 0.5$). In Table 3, we can see the best results for F2 score of PB using each one of the distance metrics. The same parameters are used for all different PHs, where the results are optimized for PH equal to 8 h. Although the differences are small, the best choice among distance metrics seems to be the SBD. For PH=8, the improvements of SBD over the default Euclidean is 17.2% and, over Prophet, it is 70% for the same profile size (equal to 100 samples).

In addition, we apply the profile reset R_p policy, incorporating the domain knowledge for all four distance metrics, where PB resets based on expert feedback (every time a module of the press is changed). We refer to this approach as *PBr*. In these distance metrics-oriented experiments, PBr could not outperform the PB solution. Although, the inherent of contextual information is useful, its benefits should not be taken for granted and assessed along with the choice of distance metric or other aspects (e.g., feature engineering).

Finally, compared with the results are shown in Table 2, it seems that using the right distance metric, PB outperforms PdM_TransAD. The main lesson learned is that using the right distance metric, PB outperforms PdM_TransAD. These findings are aligned with the statement in the discussion of the article Schmidl et al. (2022), indicating that in some cases, traditional machine learning methods and simpler approaches may outperform DL techniques in time series anomaly detection tasks, which bears similarities with our case.

Impact of feature engineering

Viewing each force signature as a time series, we examine the impact of extracting time domain features. Specifically, we use the median, peak-to-peak, variance, standard deviation, root mean square, skewness and kurtosis of the force

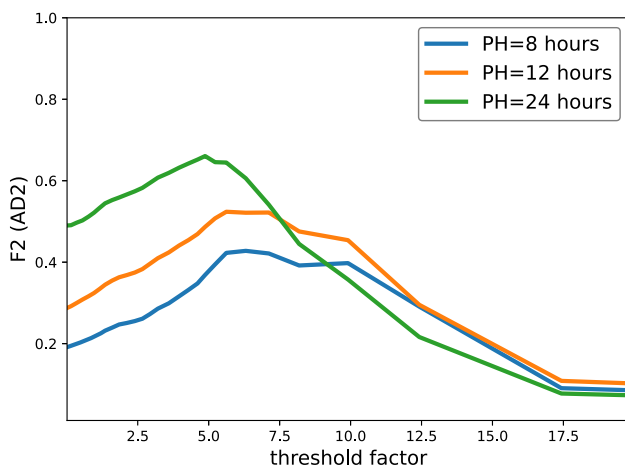


Fig. 15 Precision-Recall curve for PBr using features

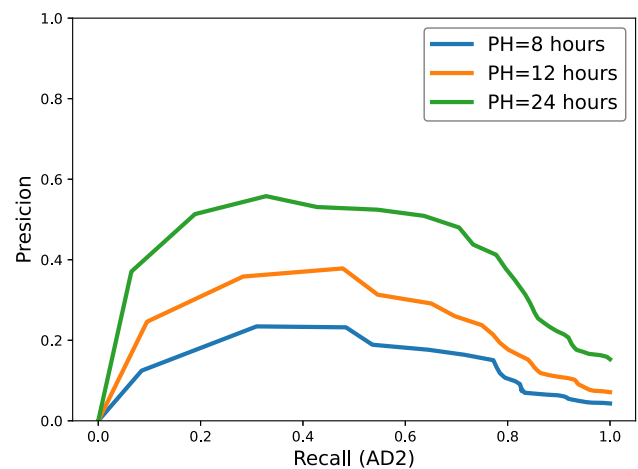


Fig. 16 F2 score against threshold values plot for PBr using features

signal, which are selected from Table 3 in Zhang et al. (2019). So, we first transform each punch to a 7d-vector and then, we apply the methodology of PB algorithm. Moreover, we test the application of Principal component analysis (PCA), to reduce the dimensionality of the 500 size vector to a 10d vector. To do this, we fit PCA using historic values and then we apply the transformation to upcoming punches. As shown from the results in Table 4, PB using features could lead to better results regarding AD_2 for the two shorter predictive horizons. Interestingly, the PBr approach (with profile size equal to 60 samples) managed to outperform both PB with features and PB with SBD distance. For PH = 8, the increase in F2 compared to Prophet now reaches 115%. Note that conceptually, it makes no sense to apply SBD along with the feature engineering suggested (and the corresponding F2 metrics do not improve).

We provide a trade-off analysis regarding the best engineering choice (*PBr* with features) by creating the precision-recall digram and a diagram which plots the F2 score against the different threshold factors (using AD_2) for all three PHs tested in Figs. 15 and 16, respectively. An interesting observation is that our solution is not extremely sensitive to fine-tuning (Fig. 15), which is also implied by Fig. 16, where, especially for the longer PH, precision is rather stable for a wide range of recall values.

The aforementioned suggest that time series transformations of data could lead to improvement of results. Another benefit is that, becomes easier for the end user to understand the changes in data looking at features rather than the whole signal. Finally, contrary to the distance metric experimentation, contextual information using R_p combined with feature engineering is shown to boost the results.

Table 5 *PBr*ref F2 score

| PH | AD_1 | AD_2 | AD_3 |
|----|--------|--------|--------|
| 8 | 0.36 | 0.26 | 0.10 |
| 12 | 0.44 | 0.30 | 0.11 |
| 24 | 0.50 | 0.29 | 0.09 |

Supervision

In this section, similarly to the experiment in Sect. 1, we test the usage of PB employing more supervision (which in practice transforms our methodology to a semi-supervised one). In detail, we investigate to leverage historic information to decide abnormal behavior of new data. To do so, we apply PB using further domain expert knowledge. Looking at historic data, experts selected punches that represent a healthy state for each different module and these were used as the profile for PB (in our case, this number of punches was 30). We use the notation of *PBr*ref for that implementation, which is a semi-supervised solution. Again we test all distance metrics and feature engineering to find the best results for this method, which are for SBD distance (see Table 5).

Although for AD_1 , the results are similar to PB using SBD and raw data, the low AD_2 and AD_3 indicate that, in most cases, this solution did not produce continuous alerts before the failure. Furthermore, the best score achieved is still lower than PBr and PB using features. Interestingly, *PBr*ref using features performed poorly (F2 score below 0.2 for AD_1). Specifically, in many episodes we observe the error starting high and ending low. This is another evidence that the properties of the signals change from time to time, and justifies our choice for an unsupervised solution.

Conclusion and future work

This work aims to answer practical aspects when engineering and deploying PdM solutions in real industrial cases, where no supervised learning solution can be efficiently applied. In summary, we first investigate the most appropriate KPIs to assess PdM solutions. To this end, we still employ metrics using true/false positives/negatives, but we propose a novel combination tailored to the PdM settings. In addition, we present a novel pipeline for PdM, which is based on building and resetting profile data; this profile data are used as reference datasets to reason about data anomalies. Then, given the KPIs, we tested the impact of different choices in distance metrics and feature engineering to conclude that the most suitable solution for our problem is to use SBD over time series, or even better, to perform feature engineering over these time series and convert them to vectors. Moreover, we have shown how the domain expertise could be leveraged in the rationale of the Profile Based solution. Finally, we propose an unsupervised framework for DL models dealing with Run-to-failure data.

The next step for an end-end PdM calls for a Decision Support System (DSS). Some interesting questions to be answered in near future could be how the KPIs could be used to gain knowledge from historic data to change configurations of existing solutions (e.g. threshold) on the fly? How the information about continuous or sparse alarms could be used in scheduling maintenance? Additionally, can the different approaches (e.g PdM_TransAD and PB) be fused to get better results?

Author Contributions All authors actively contributed to the research and the writing. The implementation was conducted by the 1st author.

Funding Open access funding provided by HEAL-Link Greece.

Data Availability <https://github.com/EngineeringUnsupervisedPdM/EngUnsPdM>

Code Availability <https://github.com/EngineeringUnsupervisedPdM/EngUnsPdM>

Declarations

Conflict of interest The authors declare no Conflict of interest.

Ethics approval Not Applicable

Consent to participate Not Applicable

Consent for publication Not Applicable

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indi-

cate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Audibert, J., Michiardi, P., Guyard, F., Marti, S., & Zuluaga, M.A. (2020). Usad: Unsupervised anomaly detection on multivariate time series. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. KDD '20, pp. 3395–3404. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/3394486.3403392>.
- Bahri, M., Salutari, F., Putina, A., & Sozio, M. (2022). Automl: state of the art with a focus on anomaly detection, challenges, and research directions. *International Journal of Data Science and Analytics*, 14(2), 113–126. <https://doi.org/10.1007/s41060-022-00309-0>
- Carrasco, J., López, D., Aguilera-Martos, I., García-Gil, D., Markova, I., García-Barzana, M., Arias-Rodil, M., Luengo, J., & Herrera, F. (2021). Anomaly detection in predictive maintenance: A new evaluation framework for temporal unsupervised anomaly detection algorithms. *Neurocomputing*, 462, 440–452. <https://doi.org/10.1016/j.neucom.2021.07.095>
- Chakroun, A., Hani, Y., Elmhamedi, A., & Masmoudi, F. (2024). A predictive maintenance model for health assessment of an assembly robot based on machine learning in the context of smart plant. *Journal of Intelligent Manufacturing*. <https://doi.org/10.1007/s10845-023-02281-3>
- Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '16, pp. 785–794. Association for computing machinery, New York, NY, USA. <https://doi.org/10.1145/2939672.2939785>
- Dalzochio, J., Kunst, R., Pignaton, E., Binotto, A., Sanyal, S., Favilla, J., & Barbosa, J. (2020). Machine learning and reasoning for predictive maintenance in industry 4.0: Current status and challenges. *Computers in Industry*, 123, 103298. <https://doi.org/10.1016/j.compind.2020.103298>
- Davis, J., & Goadrich, M. (2006). The relationship between precision-recall and roc curves. In: Proceedings of the 23rd International Conference on Machine Learning. ICML '06, pp. 233–240. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/1143844.1143874>
- Diez, A., Khoa, N. L. D., Makki Alamdari, M., Wang, Y., Chen, F., & Runcie, P. (2016). A clustering approach for structural health monitoring on bridges. *Journal of Civil Structural Health Monitoring*, 6(3), 429–445. <https://doi.org/10.1007/s13349-016-0160-0>
- Diez-Oliván, A., Pagan, J. A., Khoa, N. L. D., Sanz, R., & Sierra, B. (2018). Kernel-based support vector machines for automated health status assessment in monitoring sensor data. *The International Journal of Advanced Manufacturing Technology*, 95(1), 327–340. <https://doi.org/10.1007/s00170-017-1204-2>
- Giannoulidis, A., Gounaris, A., Nikolaidis, N., Naskos, A., & Caljouw, D. (2022). Investigating thresholding techniques in a real predictive maintenance scenario. *SIGKDD Explorations Newsletter*, 24(2), 86–95. <https://doi.org/10.1145/3575637.3575651>
- Hundman, K., Constantinou, V., Laporte, C., Colwell, I., & Soderstrom, T. (2018). Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &

- Data Mining. KDD '18, pp. 387–395. Association for computing machinery, New York, NY, USA. <https://doi.org/10.1145/3219819.3219845>.
- Jacob, V., Song, F., Stiegler, A., Rad, B., Diao, Y., & Tatbul, N. (2021). Exathlon: a benchmark for explainable anomaly detection over time series. *Proc. VLDB Endow.*, 14(11), 2613–2626. <https://doi.org/10.14778/3476249.3476307>
- Jiang, Y., Yin, S. (2017). Recent results on key performance indicator oriented fault detection using the db-kit toolbox. In IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, pp. 7103–7108. <https://doi.org/10.1109/IECON.2017.8217242>
- Jiang, Y., & Yin, S. (2019). Recent advances in key-performance-indicator oriented prognosis and diagnosis with a matlab toolbox: Db-kit. *IEEE Transactions on Industrial Informatics*, 15(5), 2849–2858. <https://doi.org/10.1109/TII.2018.2875067>
- Korvesis, P., Besseau, S., & Vazirgiannis, M.: Predictive maintenance in aviation: Failure prediction from post-flight reports. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE), pp. 1414–1422 (2018). <https://doi.org/10.1109/ICDE.2018.00160>
- Li, X., Ding, Q., & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172, 1–11. <https://doi.org/10.1016/j.res.2017.11.021>
- Li, Z., Liu, K., Wang, X., Yuan, X., Xie, H., & Wang, Y. (2023). A signal-to-image fault classification method based on multi-sensor data for robotic grinding monitoring. *Journal of Intelligent Manufacturing*. <https://doi.org/10.1007/s10845-023-02259-1>
- Makridakis, G., Kyriazis, D., & Plitsos, S. (2020). Predictive maintenance leveraging machine learning for time-series forecasting in the maritime industry. In 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), pp. 1–8. <https://doi.org/10.1109/ITSC45102.2020.9294450>
- Munir, M., Siddiqui, S. A., Dengel, A., & Ahmed, S. (2019). Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7, 1991–2005. <https://doi.org/10.1109/ACCESS.2018.2886457>
- Paparrizos, J., Boniol, P., Palpanas, T., Tsay, R. S., Elmore, A., & Franklin, M. J. (2022). Volume under the surface: a new accuracy evaluation measure for time-series anomaly detection. *Proc. VLDB Endow.*, 15(11), 2774–2787. <https://doi.org/10.14778/3551793.3551830>
- Paparrizos, J., Kang, Y., Boniol, P., Tsay, R. S., Palpanas, T., & Franklin, M. J. (2022). Tsb-uad: an end-to-end benchmark suite for univariate time-series anomaly detection. *Proc. VLDB Endow.*, 15(8), 1697–1711. <https://doi.org/10.14778/3529337.3529354>
- Paparrizos, J., Liu, C., Elmore, A.J., Franklin, M.J. (2020). Debunking four long-standing misconceptions of time-series distance measures. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. SIGMOD '20, pp. 1887–1905. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/3318464.3389760>
- Paparrizos, J., & Gravano, L. (2016). k-shape: Efficient and accurate clustering of time series. *SIGMOD Rec.*, 45(1), 69–76. <https://doi.org/10.1145/2949741.2949758>
- Rögnvaldsson, T. S., Nowaczyk, S., Bytner, S., Prytz, R., & Svensson, M. (2018). Self-monitoring for maintenance of vehicle fleets. *Data Min. Knowl. Discov.*, 32(2), 344–384. <https://doi.org/10.1007/s10618-017-0538-6>
- Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLoS one*, 10(3), 0118432. <https://doi.org/10.1371/journal.pone.0118432>
- Salfner, F., Lenk, M., & Malek, M. (2010). A survey of online failure prediction methods. *ACM Comput. Surv. doi*, 10(1145/1670679), 1670680.
- Schmidl, S., Wenig, P., & Papenbrock, T. (2022). Anomaly detection in time series: A comprehensive evaluation. *Proc. VLDB Endow.*, 15(9), 1779–1797. <https://doi.org/10.14778/3538598.3538602>
- Serradilla, O., Zugasti, E., Rodriguez, J., & Zurutuza, U. (2022). Deep learning models for predictive maintenance: a survey, comparison, challenges and prospects. *Applied Intelligence*, 52(10), 10934–10964. <https://doi.org/10.1007/s10489-021-03004-y>
- Shi, Y., & Zhang, L. (2023). Modelling long- and short-term multi-dimensional patterns in predictive maintenance with accumulative attention. *Reliability Engineering & System Safety*, 237, 109306. <https://doi.org/10.1016/j.res.2023.109306>
- Silva Arantes, J., Silva Arantes, M., Fröhlich, H. B., Siret, L., & Bonnard, R. (2021). A novel unsupervised method for anomaly detection in time series based on statistical features for industrial predictive maintenance. *International Journal of Data Science and Analytics*, 12(4), 383–404. <https://doi.org/10.1007/s41060-021-00283-z>
- Tatbul, N., Lee, T.J., Zdonik, S., Alam, M., & Gottschlich, J. (2018). Precision and recall for time series. In Proceedings of the 32nd International Conference on Neural Information Processing Systems. NIPS'18, pp. 1924–1934. Curran Associates Inc., Red Hook, NY, USA
- Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician*, 72(1), 37–45. <https://doi.org/10.1080/00031305.2017.1380080>
- Theissler, A., Pérez-Velázquez, J., Kettelgerdes, M., & Elger, G. (2021). Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. *Reliab. Eng. Syst. Saf.*, 215, 107864. <https://doi.org/10.1016/j.res.2021.107864>
- Thiyagarajan, K., Kodagoda, S., Ulapane, N., Prasad, M.: A temporal forecasting driven approach using facebook's prophet method for anomaly detection in sewer air temperature sensor system. In: 2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA), pp. 25–30 (2020). <https://doi.org/10.1109/ICIEA48937.2020.9248142>
- Tornede, T., Tornede, A., Wever, M., Mohr, F., & Hüllermeier, E. (2020). Automl for predictive maintenance: One tool to rul them all. In *IoT Streams for Data-Driven Predictive Maintenance and IoT, Edge, and Mobile for Embedded Machine Learning*, pp. 106–118. Springer, Cham. https://doi.org/10.1007/978-3-030-66770-2_8
- Tuli, S., Casale, G., & Jennings, N. R. (2022). Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *Proc. VLDB Endow.*, 15(6), 1201–1214. <https://doi.org/10.14778/3514061.3514067>
- Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., Chen, J., Wang, Z., & Qiao, H. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In Proceedings of the 2018 World Wide Web Conference. WWW '18, pp. 187–196. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE. <https://doi.org/10.1145/3178876.3185996>
- Zha, D., Lai, K., Wan, M., & Hu, X. (2020). Meta-aad: Active anomaly detection with deep reinforcement learning. In 2020 IEEE International Conference on Data Mining (ICDM), pp. 771–780. IEEE Computer Society, Los Alamitos, CA, USA. <https://doi.org/10.1109/ICDM50108.2020.00086>
- Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., Chawla, N.V. (2019). A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In Proceedings of the thirty-third AAAI Conference on Artificial Intelligence and thirty-first innovative applications of Artificial Intelligence Conference and ninth AAAI Symposium on educational advances in artificial intelligence. AAAI'19/IAAI'19/EAAI'19. <https://doi.org/10.1609/aaai.v33i01.33011409>

- Zhang, W., Wang, L., Zhao, X., & Liu, Y. (2021). Robustprophet: Time series anomaly detection method. In 2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI), pp. 157–161. <https://doi.org/10.1109/CEI52496.2021.9574464>
- Zhang, W., Yang, D., & Wang, H. (2019). Data-driven methods for predictive maintenance of industrial equipment: A survey. *IEEE Systems Journal*, 13(3), 2213–2227. <https://doi.org/10.1109/JSYST.2019.2905565>
- Zhao, Y., Rossi, R. A., & Akoglu, L. (2021). *Automating Outlier Detection via Meta-Learning*. <https://doi.org/10.48550/arXiv.2009.10606>
- Zhu, Q., Xiong, Q., Yang, Z., & Yu, Y. (2023). A novel feature-fusion-based end-to-end approach for remaining useful life prediction. *Journal of Intelligent Manufacturing*, 34(8), 3495–3505. <https://doi.org/10.1007/s10845-022-02015-x>
- Zonta, T., Costa, C. A., Zeiser, F. A., Oliveira Ramos, G., Kunst, R., & Rosa Righi, R. (2022). A predictive maintenance model for optimizing production schedule using deep neural networks. *Journal of Manufacturing Systems*, 62, 450–462. <https://doi.org/10.1016/j.jmsy.2021.12.013>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.