Unified Evaluation of Predictive Models for Failure Prediction

Apostolos Giannoulidis[1], Ioannis Iordanis[1] and Katerina Tzompanaki[1]

[1]ETIS, UMR 8051, CYU, ENSEA, CNRS, Paris, France

Abstract

Predicting machinery failures is central to predictive maintenance, with two main paradigms commonly used to estimate time to failure: Remaining Useful Life (RUL) prediction and survival analysis (SA). However, a fair comparison between them remains challenging, as existing studies typically focus on a single paradigm or a limited set of evaluation metrics. We propose a unified evaluation framework that enables systematic comparison by transforming model outputs between RUL and SA representations, allowing the computation of any metric reported in the literature. Using this framework, we evaluate 15 models on two time-series datasets for failure prediction. Our results show that RUL models generally achieve superior predictive performance than SA. Nevertheless, SA models provide more interpretable outputs through explicit modeling of uncertainty. This work provides a practical guidance for a fair model selection in predictive maintenance.

Keywords

Remaining Useful Lifetime, Survival Analysis, Predictive Maintenance

1. Introduction

Industry 4.0 integrates digital and physical systems, the Internet of Things (IoT), and Artificial Intelligence (AI) to create sensor-rich manufacturing environments. Cloud and edge platforms support automation, data collection, and real-time analytics, towards automated smart and safe industries [1, 2]. A central challenge in this context is predicting machine failures to enable Predictive Maintenance (PdM).

PdM has been approached using various methodologies, including unsupervised anomaly detection [3, 4, 5, 6, 7] and supervised classification [3]. However, when the objective is to estimate the exact time of failure [8, 9, 10, 11, 12], two main paradigms are commonly adopted: Remaining Useful Life (RUL) estimation and Survival Analysis (SA) [13]. Although both support PdM, direct comparison between them is difficult due to the different nature of their outputs, since RUL models produce a single number, while SA models derive the Individual Survival Distribution (ISD) [14]. This lack of a standardized benchmarking framework hinders progress, particularly in the context of Automated Machine Learning (AutoML), which relies on unified evaluations. The challenge is further compounded by the limited documentation of real-world model performance under comparable conditions.

Current research typically evaluates RUL and SA models in isolation [15, 16], limiting the scope of the insights obtained, while direct and fair comparisons across the two paradigms remain rare. Moreover, the widespread use of different evaluation metrics across studies further fragments the field, making it difficult to draw clear conclusions about relative model performance [14].

Motivated by the aforementioned challenges, we propose **UniFPE**, a Unified framework for Failure Prediction Evaluation for the systematic comparison of RUL and SA models. This framework enhances trustworthiness by enabling fair comparison between different model families—such as regression-based RUL models and probabilistic SA models—under the same conditions. Beyond identifying best-performing models, we also analyze the limitations of each approach in this work. Ultimately, our goal is to provide practical guidance for researchers in selecting predictive models for PdM applications.

Published in the Proceedings of the Workshops of the EDBT/ICDT 2026 Joint Conference (March 24-27, 2026), Tampere, Finland

apostolos.giannoulidis@cyu.fr (A. Giannoulidis); ioannis.iordanis@cyu.fr (I. Iordanis); aikaterini.tzompanaki@cyu.fr (K. Tzompanaki)

In this paper, we make three main contributions. We propose a unified post-processing method that enables the production of both RUL estimates and ISDs from standard regression and survival analysis models. We then perform a comprehensive evaluation of 15 representative models on two time-series PdM datasets, providing a systematic comparison for the failure prediction problem. Finally, we analyze the transformation between RULs and ISDs, comparing ISD-derived RULs with regression-based estimates and RUL-derived ISDs with those from SA models. This analysis reveals that RUL models generally provide more accurate and precise predictions approaching failure time compared to SA models.

2. Related work

Evaluating SA models alongside RUL models can be trivially done by applying RUL metrics after transforming the ISD into a single RUL estimate, usually by selecting a fixed probability threshold [13, 17]. In other work [18], the comparison between SA and RUL models was performed using only the C-index, which is applicable to both. Our methodology supports both C-index, but also enables a direct comparison of SA and RUL models in terms of any SA or RUL related metric. Thus, provides a more complete transformation of SA to RUL, without limiting the comparison between the two families to only ranking based metric C-index.

Remaining Useful Lifetime prediction, in the context of PdM, is typically formulated as a regression task, where, given historical failure and sensor data, the objective is to predict the time until failure. In data-driven PdM, researchers [19, 10, 11] employ established models such as XGBoost and Random Forests, as well as rule-based models [20]. Furthermore, Deep learning (DL) models designed for time-series data are also widely used [21]. In our evaluation we use and present both classical tabular and DL models.

Survival Analysis recently has seen increasing adoption in PdM applications [22], where researchers use SA models to estimate machines' ISD [14]. The Cox proportional hazards model [23] (CoxPH), despite being introduced over half a century ago, remains one of the most widely used survival analysis models [24, 18, 25, 26, 13], suggesting that recent developments have not outperformed classical approaches in this domain. The review in [27] provides a good overview of SA models. In our evaluation we include and discuss both classical SA methods but also more recent DL models.

3. Unified evaluation framework

Our goal is to propose a unified evaluation of *RUL* estimation methods and *SA* methods for the problem of failure prediction. As failures, we consider events that cause monitored entities (e.g., machines or vehicles) to malfunction, stop operating, or break down, requiring replacement or heavy maintenance. Both RUL and SA methods leverage sensors and log data arising from continuously monitored entities, aiming to predict the time until the next failure event. While both methodologies aim to solve the same problem, they rely on different formalizations and provide different types of output. In particular, at inference time, RUL methods output a single real-valued RUL estimate, whereas SA methods produce probability curves (ISDs). In the following, we introduce the necessary notation and formally define the problem of unifying the evaluation of RUL and SA methods. This is accomplished by proposing a transformation of their outputs into both probability curves and RUL estimates.

3.1. Background

For each entity $e \in \mathbf{E}$, where $\mathbf{E}$ denotes the set of entities, we observe multivariate time-series data $\mathbf{X}^e \in \mathbb{R}^{m^e \times n^e}$, where n^e is the number of covariates and m^e is the number of available timestamps for the particular entity e. At timestamp j, we observe $\mathbf{x}_j^e \in \mathbf{X}^e$, which represents the sensor readings produced by entity e at time j, with $\mathbf{x}_j^e \in \mathbb{R}^n$. Note, that depending on the underlying predictive

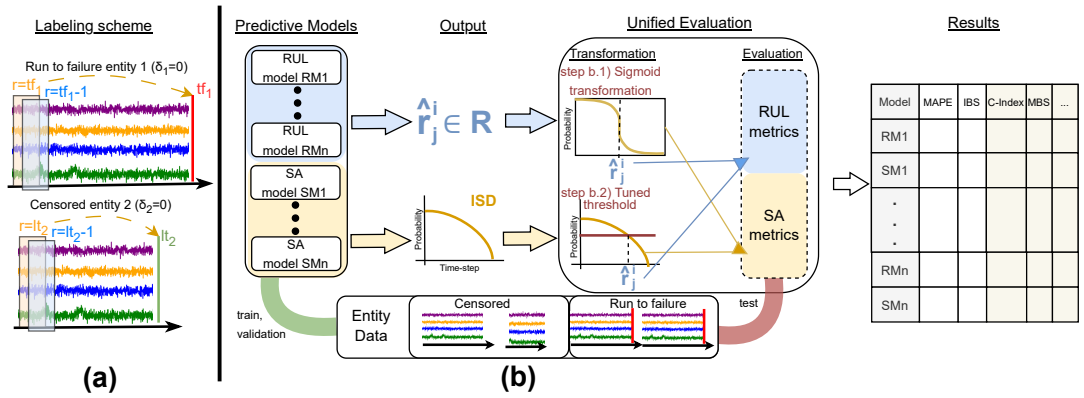

Figure 1: Overview of (a) labeling scheme in PdM dataset, and (b) UniFPE for SA and RUL models.

model, instead of x_j^e, we may consider a sequence (or a window) of the h most resent values of entity $x_{j-h,j}^e = \{x_{j-h}^e, x_{j-h+1}^e, ..., x_j^e\}$.

We classify the entities into two categories, *run-to-failure* (RTF) marked with and *censored* entities. We refer to RTF entities as those that experience a failure event, which marks the end of their operational life. Respectively, We refer to *censored* entities as those for which data collection stops before a failure event occurs. We use the notation δ_e to indicate in which category each entity belongs. Specifically, $\delta_e = 1$ indicates that the entity e is an RTF entity and $\delta_e = 0$ indicates that the entity e is censored. For RTF entity e, tf^e refers to the timestamp of the failure event, while for a different *censored* entity e, lt^e corresponds to the last observed timestamp.

Then, we use the notation r_j^e to denote the remaining time until the event (either failure or censoring) for entity e at timestamp j. Specifically, r_j^e is defined as the time elapsed from timestamp j until the failure time tf^e if $\delta_e = 1$ ($r_j^e = tf^e - j$), or until the censoring time lt^e if $\delta_e = 0$ ($r_j^e = lt^e - j$). Finally we formalise the dataset used by predictive models as $\mathcal{D} = \left\{ \left(\{r_j^e, \mathbf{x}_j^e\}_{j=1}^{n_e}, \delta_e \right) \right\}_{e \in \mathbf{E}}$, where n_e denotes the number of available samples for entity e.

RUL prediction The first family - RUL prediction - uses regression models to predict the RUL r_j^e of an entity e at a timestamp j. Here, we impose the constraint of considering only RTF entities in the training data. Specifically, we consider the subset of the dataset $\mathcal{D}^{RUL} = \left\{ \{r_j^e, \mathbf{x}_j^e\}_{j=1}^{n_e}, \delta_e \right\}_{e \in \mathbf{E}}$, with $\delta_e = 1$. Then, given D^{RUL}, regression models are trained to compute the function $\mathcal{F}(\mathbf{x}_j^e; \theta) : \mathbb{R}^{m^e} \rightarrow \mathbb{R} \sim r_j^e$, where θ represents the model's parameters, utilizing the actual RUL labels r_j^e. These models, in inference time, provide estimations of the actual (unknown) RUL $\hat{r}_j^e$.

Survival Analysis aims to provide the survival probability of failing entities in several future timestamps. Typically, methods in this family derive the ISD [14] of the entity, represented as a curve showing the survival probability of the entity as time progresses. In classical SA, a unique ISD will be derived for each different patient based on their state. Inherently in PdM, we produce a different ISD for every observation x_j^e, which represents the state of the entity at each timestamp j. Formally, given $\mathcal{D}$, the ISDs the SA models aim to learn is: $\widehat{S}(t \mid \mathbf{x}_j^e) = \mathbb{P}(r_j^e > t \mid \mathbf{x}_j^e), \qquad t \geq 0$, that characterizes the time-to-event distribution for entity e at time j with covariates $\mathbf{x}_j^e$ (or $x_{j-h,j}^e$). The estimation of ISDs is typically performed by maximizing a (partial or full) likelihood associated with a parametric, semi- or non-parametric survival model.

3.2. Unified Comparison

Next, we discuss evaluation metrics used to asses RUL and SA methods and present UniFPE. Specifically, we present how SA-generated ISDs can be transformed into RUL predictions, and on the other side, how RUL predictions for different timestamps can be converted into ISDs. Figure 1(a) illustrates how labels are derived from entities, as described in the previous section. Then Figure 1(b) provides a graphical

Table 1
Evaluation metrics

Name	Input	Description
BS(t)	ISD, r, δ	Measures both calibration and discrimination.
IBS	ISD, r, δ	Integral of BS(t) over time.
MBS	ISD, r, δ	Maximum value of BS(t), shows the highest uncertainty.
C-index	Risk score, r, δ	Measures the discrimination capability of the model.
AUCROC	Risk score, r, δ	Measures the discrimination between failed and non-failed entities by a given time.
MAPE	Predicted RUL, r, ($\delta = 1$)	Mean absolute percentage error between predicted and true RUL.
MdAPE	Predicted RUL, r, ($\delta = 1$)	Median absolute percentage error.
MSE	Predicted RUL, r, ($\delta = 1$)	Mean squared error between predicted and true RUL.
R^2	Predicted RUL, r, ($\delta = 1$)	Measure of goodness of fit [28].

overview of UniFPE. The figure illustrates deployed predictive models, including both RUL and SA models, all trained and validated on the same data (note that RUL models ignore censored data). It further shows how RUL predictions are converted into ISDs and how ISDs produced by SA models are translated into RUL predictions, enabling a unified evaluation of all predictive models on the test set. Finally, Table 1 summarizes the evaluation metrics used by UniFPE.

3.2.1. RUL evaluation

For evaluating RUL predictions, numerous metrics have been proposed [29]. A typical choice involves metrics commonly used in regression problems, which measure the distance between a prediction and the actual RUL (e.g., squared error, absolute error). However, because prediction errors become more critical as the entity approaches failure, it is often preferable to use the Mean Absolute Percentage Error (MAPE), defined as MAPE $= \frac{1}{N}\sum_{j=1}^{N}\left|\frac{\hat{r}_j - r_j}{r_j}\right|$, where N is the number of samples. For example, predicting 210 days instead of 200 days yields the same absolute and squared error as predicting 12 days instead of 2 days. Yet, the latter case is far more harmful in a PdM scenario, since errors near failure carry higher risk. Instead of mean of the absolute presentage errors, someone can choose the median (MdAPE), towards robustness on outlier errors. Although we use MAPE and MdAPE as the main metrics for evaluating RUL predictions, we also compute a variety of additional metrics (refer to Table 1), all of which are provided along with our implementation.

3.2.2. SA evaluation

In survival analysis, two widely used evaluation metrics are the Brier score and the Concordance index (C-index) [14]. The C-index evaluates whether a model assigns higher risk scores to entities that fail earlier. When models output an ISD, it is common to compute the C-index across time by using the ISD probability at each step as the score, yielding a C-index curve. Since the C-index is not specific to SA models, it can also be applied to RUL predictions by converting them to risk scores (e.g., using $-\hat{r}$). While the C-index measures discriminative ability (capturing only the ranking of predicted risks and ignoring their calibration), the Brier score [30] evaluates both calibration and discrimination, and is defined as: BS$(t) = \frac{1}{N}\sum_{j=1}^{N}\left(I(r_j > t) - \hat{S}_j(t)\right)^2$, where $S_j(t)$ is the predicted survival probability at time j, $I(r_j > t)$ indicates whether the observed time-to-failure r_j exceeds t, and N is the number of samples. The Brier score can also be computed across all ISD time steps, forming a curve, from which the Integrated Brier Score (IBS) summarizes performance: IBS $= \frac{1}{\tau}\int_0^{\tau}$ BS$(t)\,dt$.

We compute all aforementioned metrics (Brier, IBS, and C-index), and also include the best C-index across time steps and the worst Brier score (Maximum Brier Score, MBS), with primarily focus on IBS and MBS for comparing predictive models. Note that MBS is particularly useful, as IBS can be dominated by the very small (and thus good) Brier scores at early and very late time steps. These regions often

behave like noise, since making reliable predictions at those stages is trivial (e.g., predicting near-certain survival at initial time steps and near-certain failure at very large time steps).

3.2.3. Producing RUL estimations from ISDs.

Instead of adopting a fixed threshold (e.g., 0.5) as done in [13, 17], we use a tuned threshold on the ISD that minimizes MAPE on a sampled validation set after training the SA model. A graphical representation of extracting $\hat{r}_j^e$ from ISD can been seen in the step b.2) of Figure 1. An SA model achieving a perfect IBS implies that it produces ISDs with probability 1 for all times prior to the event (i.e failure) and probability 0 from the event time onward. Consequently, a model with perfect IBS (i.e., $IBS = 0$) will also achieve perfect MAPE, regardless of the use of calibrated threshold. This is because the predicted probability will always cross the decision threshold exactly at the true failure time, yielding an RUL estimate equal to the actual.

3.2.4. Producing ISDs from RUL estimations

Although C-index can be directly used to evaluate RUL models, applying IBS requires first transforming RUL predictions into survival curves. A straightforward way is to rely entirely on the predicted RUL $\hat{r}$ and define a hard-mapped survival function $S(t) = HM(t, \hat{r}) = \{1 \text{ if } t < \hat{r},\ 0 \text{ otherwise}\}$. HM transformation provides a simple interpretation of RUL as an ISD. In the case of a perfect RUL model (with $MAPE = 0$), HM will result in Brier score and IBS equal to zero as well. But, for imperfect models, the Brier score evaluates each sample at time t as either 0 or 1, meaning that the HM transformation favors samples with $t < \min(r_i, \hat{r}_i)$ or $t > \max(r_i, \hat{r}_i)$ and penalizes those where $\min(r_i, \hat{r}_i) < t < \max(r_i, \hat{r}_i)$. As a result, even small deviations between predicted and true RUL lead to the maximum penalty, reflecting the absence of uncertainty in the curve.

To address this limitation, we introduce a smoother transformation using the sigmoid function, inspired by its widespread use in neural networks for converting hard decisions into class probabilities [31]. A graphical demonstration of such transformation is depicted in the step b.1) step of Figure 1. The sigmoid yields an ISD corresponding to a logistic failure-time distribution centered at $\hat{r}$. In our evaluation, we report results using the sigmoid transformation, while providing the HM-based results in our repository [32].

4. Evaluation

In this section, we describe the evaluation setup used to tune and assess the performance of SA and RUL models. Subsequently, we present the PdM datasets used in our analysis, and finally, we report and discuss the results of our study. Result and implementation is available in [32].

4.1. Evaluation setup

4.1.1. Models

In this study we include the following SA models. First, the Cox Proportional Hazards model (CoxPH) [23], a semi-parametric approach that estimates the effect of covariates on event times through hazard ratios. Second, the Random Survival Forests (RSF) model [33], which extends random forests to handle censored data and generate non-parametric survival functions. Third, the Recurrent Deep Survival Model (RDSM) [34], an LSTM-based architecture specifically designed to model time-dependent covariates and capture nonlinear temporal dynamics in survival predictions. Finally, we also include DeepHit [35] a non-parametric model, which leverages NN to learn the distribution of survival times directly.

For RUL estimation, we incorporate commonly used regression models for RUL prediction as well as state-of-the-art time-series regression models that have demonstrated superior performance in recent benchmarks [36]. Classical machine learning methods comprise of: i) XGBoost [37], a gradient-boosted ensemble of decision trees, ii) CatBoost [38], which incorporates ordered boosting and target statistics

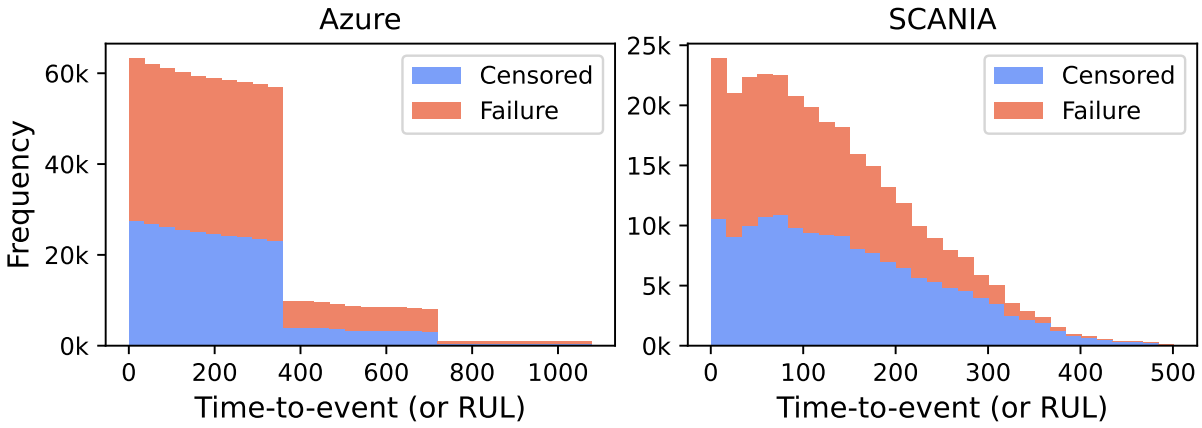

Figure 2: Distribution of labels in PdM datasets.

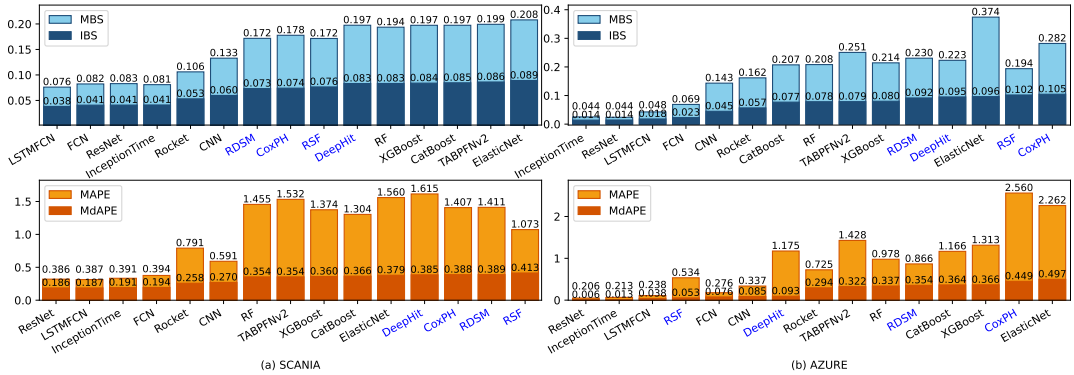

Figure 3: Performance of RUL and SA models in both SA (upper section) and RUL (lower section) metrics on the Scania (a) and AZURE (b) datasets. SA methods are highlighted with blue color.

to effectively handle categorical variables, iii) TabPFN [39], a tabular foundation model, iv) Elastic Net regression [40], and v) Random Forests (RF) [41]. In addition, we evaluate several deep learning architectures tailored for sequence modeling, including ResNet [42], a residual convolutional network adapted for regression; InceptionTime [43], which uses multi-scale inception modules for time-series forecasting; standard CNNs [44] adapted for regression; Fully Convolutional Networks (FCN) [45], which omit pooling layers to preserve temporal resolution; LSTM-FCN [46], combining recurrent and convolutional feature extraction; and ROCKET [47], a linear model built on features generated by random convolutional kernels.

4.1.2. PdM Datasets

This study focuses on PdM, consequently, we have included two industrial datasets consisting of dynamic, multivariate time-series data, with ground-truth derived from failure and maintenance logs. Figure 2 depicts the distribution of the time to event labels r_j^i, for both RTF ($\delta_i = 1$) and censored ($\delta_i = 0$) entities. In our evaluation, we maintain a fixed ratio of censored to RTF samples of 0.5 for both datasets.

SCANIA dataset [48] comes from the IDA 2024 Industrial Challenge and consists of telemetry and specification data from several SCANIA trucks. It includes 113 operational features, both numerical and categorical, like engine type and wheel configuration.

AZURE dataset [49] comprises hourly measurements of voltage, rotational speed, pressure, and vibration signals collected from 100 simulated machines under realistic industrial setting, provided by Microsoft Azure. In addition to the sensor readings, the dataset includes records of maintenance operations for individual machine components, failure events, corresponding to machine breakdowns,

and categorical data of error states.

For both dataset we split the data into train, validation and test set. The spliting was done such that 60% of the RTF entities belong to training set, 20% of RTF entities belong to validation set and the final 20% of RTF entities belong to test set. In the training set we also include *Censored* entities which are only leverage from SA methods, which inherently have the capability of considering censored data to improve the predictions. Finally categorical features are transformed using one-hot-encoding for both datasets.

4.1.3. Hyperparameter Tuning and Final Evaluation

For each method, we optimize the hyperparameters (or the architecture, in the case of deep learning models) using the Mango Bayesian optimization framework [50] on the training and validation sets, while final model performance is evaluated on the test set. We perform 20 optimization steps and retain the best configuration. For RUL models, the optimization metric is MAPE, whereas for SA models it is IBS. For a fair comparison across models, the test set includes only RTF entities (i.e., entities that ultimately fail) since computing regression-based metrics requires ground-truth RUL values.

4.2. Evaluation of SA and RUL models

In this section, we present the results of our evaluation study. We begin by discussing the best-performing predictive models overall and assessing the impact of the tuned thresholding strategy. We then focus specifically on the ability of the predictive models to estimate RUL and to generate valid ISDs.

4.2.1. Best predictive models overall

Figure 3 reports the performance of RUL and SA models on the SCANIA (a) and Azure (b) datasets, evaluated using IBS and MBS (SA metrics), and MAPE and MdAPE (RUL metrics). At first glance, it is evident that regression models specifically designed for time-series data e.g., ResNet, LSTMFCN, Inception Time, FCN, consistently achieve the best performance across both metrics and datasets.

On the SCANIA dataset, considering SA metrics, SA models outperform classical tabular regression methods in terms of IBS, but are outperformed by time-series deep learning RUL models. With respect to RUL metrics, SA models exhibit performance comparable to tabular RUL models, for both MdAPE and MAPE. In the Azure dataset, time-series RUL models achieve the best results in terms of both IBS and MBS, while SA models do not surpass tabular RUL approaches. Regarding MAPE and MdAPE, time-series–based RUL models again achieve the best performance. In contrast, SA models don't exhibit consistent behavior. Instead, performance varies across individual methods: RSF and DeepHit achieve low MdAPE values, whereas RDSM and CoxPH show performance comparable to that of tabular RUL models. Although the ranking of methods differs between SA and RUL metrics in both datasets, the overall performance trends remain consistent. An exception is observed for RSF and DeepHit in terms of MdAPE, where they are ranked higher than when evaluated using IBS. The reason for this discrepancy lies in the use of calibrated thresholds for deriving RUL estimates, as discussed in the following section.

4.2.2. Evaluation of calibrated threshold

Although RSF and DeepHit on the Azure dataset do not produce more calibrated probabilities than other methods, as reflected by their IBS scores, they assign consistent probability values at the true time of failure. Figure 4 illustrates RSF predictions for a representative entity in Azure dataset, where red markers indicate the probability assigned by each survival curve at the true failure time (i.e., the actual RUL). These values cluster around 0.77 rather than approaching zero, indicating a systematic shift in probability scale rather than random miscalibration. Because this behavior is consistent across samples, applying a calibrated threshold enables a more accurate estimation of RUL, thereby reducing the error between predicted and true RUL.

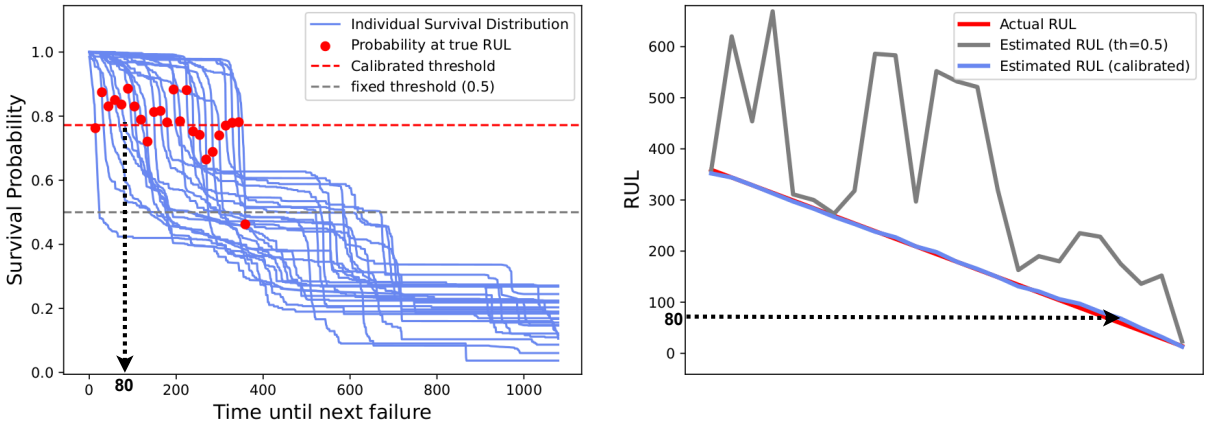

Figure 4: Left: example of the produced ISDs using RSF. Right: the RUL derived from thresholding on ISDs.

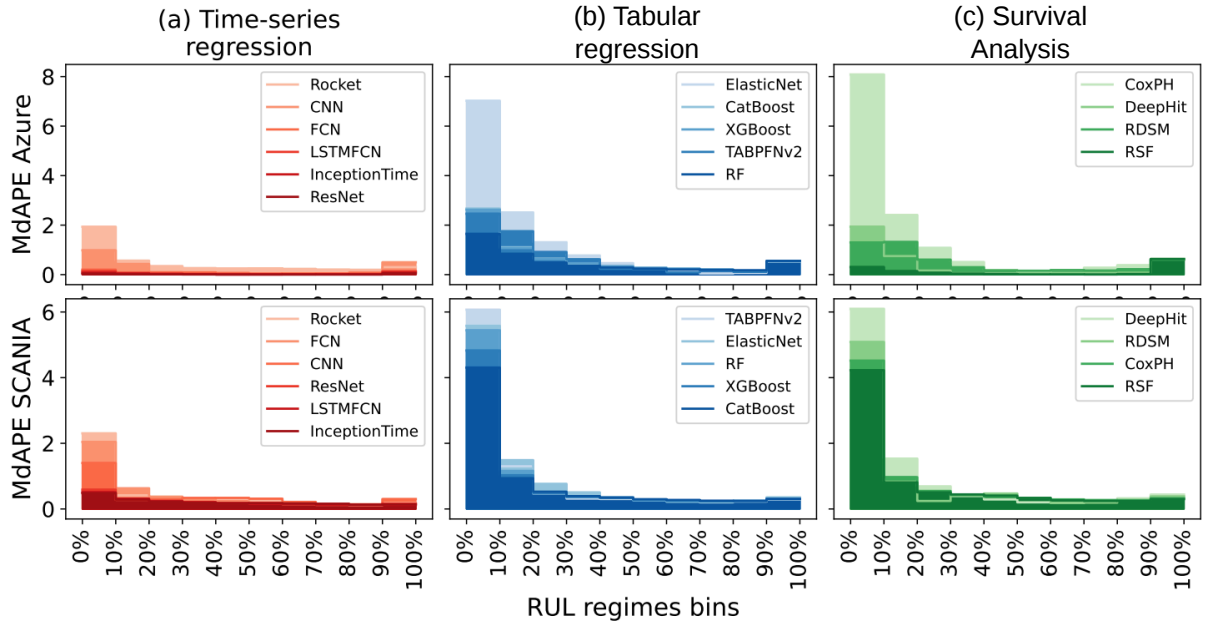

Figure 5: MdAPE distribution on RUL regims, in both datasets, for (a) time-series RUL, (b) SA, (c) tabular RUL models.

We further evaluate the impact of tuned threshold by comparing the performance of all SA models on both datasets using a calibrated threshold versus a fixed threshold of 0.5, which is used in prior works [13, 17]. Overall, tuned threshold reduce MAPE and MdAPE by approximately 28% compared to the fixed-threshold setting, indicating that *interpreting survival probabilities through dataset-specific thresholds yields more reliable RUL estimates in practice.*

4.2.3. Evaluation of RUL estimations

Taking a closer look at the model results, we observe that the main difference between high-performing models (time-series regression models), SA models, and classical regression models lies in their ability to accurately predict near-failure instances. Figure 5 presents the MdAPE achieved by each method across different RUL bins. To obtain this visualization, we partition the RUL labels into 10 uniformly distributed bins, such that each bin contains one-tenth of the samples. From Figure 5, we observe that all models achieve relatively good performance in the mid-range RUL regions. However, classical regression models and SA models exhibit substantial deviations from the true RUL values in the initial bins, which correspond to near-failure instances. While this behavior is also present for some time-series regression

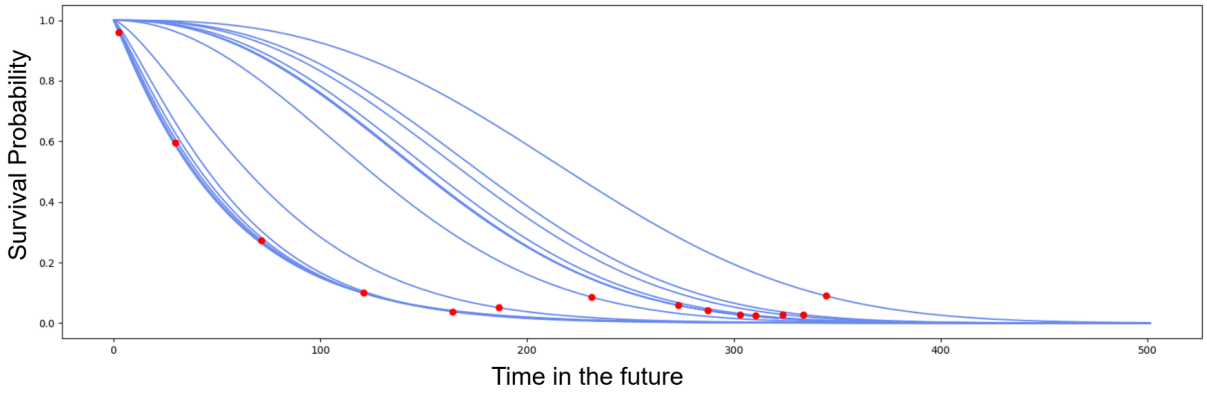

Figure 6: ISDs produced by the RDSM model for one particular SCANIA vehicle, red dots point the actual RUL.

models, their relative error remains considerably lower. Overall, classical regression and SA models appear to capture the median survival behavior of entities, producing accurate predictions for middle RUL regions, but fail to adequately capture near-failure behavior.

Focusing specifically on SA models, accurately modeling near-failure instances requires producing steep survival curves, such that the probability drops sharply at the true failure time. This limitation is illustrated in Figure 6, which shows ISDs generated by the RDSM model for a vehicle in the SCANIA dataset. In the figure, the red markers denote the true RUL values for each observation. For instances far from failure, the survival probability assigned at the true RUL is near zero, which is the desired behavior. However, for instances close to failure, although the corresponding ISDs shift their mass toward earlier times, indicating higher risk, the curves are not sufficiently steep. As a result, the survival probability is not close to zero at the actual RUL, leading to inaccurate RUL estimations when a threshold-based conversion is applied. *This illustrates a limitation of SA models in capturing sharp transitions near failure in the examined long time-series PdM settings.*

4.2.4. Evaluation of generated ISDs

For regression models we transform RUL estimations to ISDs leveraging the sigmoid function instead of using step function HM, as explained in Section 3.2.4. This choice is validated by our experiments on both datasets for all 10 regression models; namely, we observe that the sigmoid transformation resulted on statistically better IBS and MBS than the HM transformation (statistical significance is measured using Wilcoxon test [51]). In particular, *the sigmoid transformation achieved lower IBS and MBS in all cases.*

4.3. Interpretability of SA and RUL models

The models examined in this study differ in how they convey failure risk over time. RUL models remain point-estimate predictors; however, our proposed sigmoid-based transformation allows their scalar outputs to be expressed as smooth ISDs, yielding a consistent representation of temporal risk centered at the predicted RUL. Because the sigmoid employs a fixed smoothness parameter, this representation does not vary across operating conditions (i.e., the same sigmoid transformation is applied regardless of the observed covariates). Therefore, it reflects a post-hoc uncertainty rather than instance-specific aleatoric uncertainty inferred from the data.

In contrast, SA models estimate ISDs directly as part of their training objective, providing a view of how predicted risk evolves over time. In the examined PdM cases, particularly in near-failure regimes, these models tend to produce relatively smooth survival curves, which limits the precision of RUL extraction, reflected in higher MdAPE in the lowest RUL bins (see Figure 5). Calibrated thresholds partially mitigate this effect by aligning probability levels with dataset-specific error objectives, improving RUL accuracy without altering the underlying uncertainty structure. Overall, the results expose a practical trade-

off: time-series RUL models yield sharper near-failure estimates but rely on post-hoc mechanisms to express uncertainty, whereas SA models provide a more direct temporal risk representation but struggle to provide low survival probability in critical regimes. From an interpretability perspective, these differences affect how decisively and transparently models communicate failure risk to downstream decision-makers.

5. Conclusion and Future Directions

In this work, we propose UniFPE for comparing RUL and SA models for failure prediction and apply it to two predictive maintenance datasets. Our results show that time-series regression models consistently outperform both SA models, including RDSM, which is specifically designed for time-dependent data, and classical regression approaches. Moreover we show that, accurate RUL predictions can yield probability curves that are more calibrated and more discriminative than ISDs produced directly by SA models. A possible explanation is that regression models have a simpler learning objective, predicting a single continuous value, in contrast to SA models which aim to estimate full survival distributions, introducing additional complexity that may limit their ability.

On the other hand, SA models inherently provide uncertainty estimation, leading to more interpretable results than RUL models. Based on our findings, two main future research directions emerge: (i) investigating how SA models can benefit from the strengths of RUL models (e.g., whether pretrained RUL models can be adapted for survival analysis tasks), and (ii) developing improved methods for transforming RUL predictions into ISDs that explicitly incorporate uncertainty, rather than relying on fixed transformation.

6. Declaration on Generative AI

During the preparation of this work, spelling and grammar checking were supported by Overleaf and Grammarly. Typing assistant software was used to improve the spelling and grammar of the existing text. The authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Acknowledgments

This work has received funding from the Horizon Europe Framework Programme under Grant agreement No 101135775 (PANDORA) and ANR under the Grant agreement 24-CE23-6509 (AIDA).

References

[1] A. Giannoulidis, A. Gounaris, A. Naskos, N. Nikolaidis, D. Caljouw, Engineering and evaluating an unsupervised predictive maintenance solution: a cold-forming press case-study, Journal of Intelligent Manufacturing 36 (2025) 2121–2139. URL: https://doi.org/10.1007/s10845-024-02352-z. doi:10.1007/s10845-024-02352-z.

[2] T. Rögnvaldsson, S. Nowaczyk, S. Byttner, R. Prytz, M. Svensson, Self-monitoring for maintenance of vehicle fleets, Data Mining and Knowledge Discovery 32 (2018) 344–384. URL: https://doi.org/10.1007/s10618-017-0538-6. doi:10.1007/s10618-017-0538-6.

[3] A. Giannoulidis, A. Gounaris, A. Naskos, N. Nikolaidis, D. Caljouw, Leveraging feedback and causality-enriched multimodal context for predictive maintenance, IEEE Access (2025).

[4] S. Byttner, T. Rögnvaldsson, M. Svensson, Consensus self-organized models for fault detection (cosmo), Engineering Applications of Artificial Intelligence 24 (2011) 833–839. URL: https://www.sciencedirect.com/science/article/pii/S0952197611000467. doi:https://doi.org/10.1016/j.engappai.2011.03.002.

[5] A. Giannoulidis, A. Gounaris, A context-aware unsupervised predictive maintenance solution for fleet management, Journal of Intelligent Information Systems 60 (2023) 521–547.
[6] A. Giannoulidis, A. Gounaris, I. Constantinou, Exploring unsupervised anomaly detection for vehicle predictive maintenance with partial information., in: EDBT, 2024, pp. 753–761.
[7] A. Giannoulidis, A. Gounaris, N. Nikolaidis, A. Naskos, D. Caljouw, Investigating thresholding techniques in a real predictive maintenance scenario, ACM SIGKDD Explorations Newsletter 24 (2022) 86–95.
[8] A. Bennis, Neural networks for survival analysis and predictive maintenance, Ph.D. thesis, Université Paul Sabatier-Toulouse III, 2022.
[9] T. Tornede, A. Tornede, M. Wever, F. Mohr, E. Hüllermeier, 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, Springer, 2020, pp. 106–118.
[10] B. Einabadi, A. Baboli, M. Ebrahimi, Dynamic predictive maintenance in industry 4.0 based on real time information: Case study in automotive industries, IFAC-PapersOnLine 52 (2019) 1069–1074. URL: https://www.sciencedirect.com/science/article/pii/S2405896319313151. doi:https://doi.org/10.1016/j.ifacol.2019.11.337, 9th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2019.
[11] R. Prytz, S. Nowaczyk, T. Rögnvaldsson, S. Byttner, Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data, Engineering Applications of Artificial Intelligence 41 (2015) 139–150. URL: https://www.sciencedirect.com/science/article/pii/S0952197615000391. doi:https://doi.org/10.1016/j.engappai.2015.02.009.
[12] X. Li, Q. Ding, J.-Q. Sun, Remaining useful life estimation in prognostics using deep convolution neural networks, Reliability Engineering & System Safety 172 (2018) 1–11. URL: https://www.sciencedirect.com/science/article/pii/S0951832017307779. doi:https://doi.org/10.1016/j.ress.2017.11.021.
[13] M. Rahat, Z. Kharazian, P. S. Mashhadi, T. Rögnvaldsson, S. Choudhury, Bridging the gap: A comparative analysis of regressive remaining useful life prediction and survival analysis methods for predictive maintenance, in: Phm society asia-pacific conference, volume 4, 2023.
[14] H. Haider, B. Hoehn, S. Davis, R. Greiner, Effective ways to build and evaluate individual survival distributions, J. Mach. Learn. Res. 21 (2020).
[15] Y. Liu, J. Wen, G. Wang, A comprehensive overview of remaining useful life prediction: From traditional literature review to scientometric analysis, Machine Learning with Applications 21 (2025) 100704. URL: https://www.sciencedirect.com/science/article/pii/S2666827025000878. doi:https://doi.org/10.1016/j.mlwa.2025.100704.
[16] Y. Wang, M. Wu, X. Li, L. Xie, Z. Chen, A survey on graph neural networks for remaining useful life prediction: Methodologies, evaluation and future trends, 2024. URL: https://arxiv.org/abs/2409.19629. arXiv:2409.19629.
[17] C. Lillelund, F. Pannullo, M. Jakobsen, M. Morante, C. Pedersen, A probabilistic estimation of remaining useful life from censored time-to-event data (2024). doi:10.2139/ssrn.4814236.
[18] J. Xue, L. Wei, D. Jiang, F. Sheng, R. Greiner, J. Zhang, Survival analysis with machine learning for predicting li-ion battery remaining useful life, arXiv preprint arXiv:2503.13558 (2025).
[19] Z. Kharazian, T. Lindgren, S. Magnusson, H. Boström, Copal: Conformal prediction in active learning an algorithm for enhancing remaining useful life estimation in predictive maintenance 230 (2024) 195–217. URL: https://proceedings.mlr.press/v230/kharazian24a.html.
[20] M. Razgon, A. Mousavi, Relaxed rule-based learning for automated predictive maintenance: Proof of concept, Algorithms 13 (2020). URL: https://www.mdpi.com/1999-4893/13/9/219. doi:10.3390/a13090219.
[21] Y. Wang, H. Wu, J. Dong, Y. Liu, M. Long, J. Wang, Deep time series models: A comprehensive survey and benchmark, 2024. doi:10.48550/arXiv.2407.13278.
[22] A. Unknown, Prognostic modeling of predictive maintenance with survival analysis for mobile work equipment, Scientific Reports 12 (2022) 8529. doi:10.1038/s41598-022-12572-z.
[23] D. R. Cox, Regression models and life-tables, Journal of the Royal Statistical Society. Series B

(Methodological) 34 (1972) 187–220. URL: http://www.jstor.org/stable/2985181.
[24] C. Fernandez, C. S. Chen, C. P. Gaillard, A. Silva, Experimental comparison of ensemble methods and time-to-event analysis models through integrated brier score and concordance index (2024). URL: https://arxiv.org/abs/2403.07460. arXiv:2403.07460.
[25] C. Fernandez, C. S. Chen, P. Gaillard, A. Silva, Aggregation methods and comparative study in time-to-event analysis models, International Journal of Data Science and Analytics 20 (2025) 2767–2783.
[26] B. Coutinho, M. Moreira, E. Pereira, G. Gonçalves, Survival analysis-based system for predictive maintenance optimization, SN Computer Science 6 (2025). doi:10.1007/s42979-025-04291-9.
[27] P. Wang, Y. Li, C. K. Reddy, Machine learning for survival analysis: A survey, ACM Comput. Surv. 51 (2019). URL: https://doi.org/10.1145/3214306. doi:10.1145/3214306.
[28] N. Draper, Applied regression analysis, McGraw-Hill. Inc, 1998.
[29] A. Saxena, J. Celaya, E. Balaban, K. Goebel, B. Saha, S. Saha, M. Schwabacher, Metrics for evaluating performance of prognostic techniques, in: 2008 International Conference on Prognostics and Health Management, 2008, pp. 1–17. doi:10.1109/PHM.2008.4711436.
[30] G. W. Brier, R. A. Allen, Verification of weather forecasts, in: T. F. Malone (Ed.), Compendium of Meteorology: Prepared under the Direction of the Committee on the Compendium of Meteorology, American Meteorological Society, Boston, MA, 1951, pp. 841–848. URL: https://doi.org/10.1007/978-1-940033-70-9_68. doi:10.1007/978-1-940033-70-9_68.
[31] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016. URL: https://www.deeplearningbook.org/, chapter 6.2.2, Output Units.
[32] Implementation for unified evaluation of predictive models for failure prediction., https://anonymous.4open.science/r/UnifiedEvaluationForFP-3D46, 2026.
[33] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, M. S. Lauer, Random survival forests, The Annals of Applied Statistics 2 (2008) 841 – 860. URL: https://doi.org/10.1214/08-AOAS169. doi:10.1214/08-AOAS169.
[34] C. Nagpal, X. Li, A. Dubrawski, Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks, IEEE J. Biomed. Health Informatics 25 (2021) 3163–3175. URL: https://doi.org/10.1109/JBHI.2021.3052441. doi:10.1109/JBHI.2021.3052441.
[35] C. Lee, W. Zame, J. Yoon, M. Schaar, Deephit: A deep learning approach to survival analysis with competing risks, volume 32, 2018. doi:10.1609/aaai.v32i1.11842.
[36] Y. Zhang, L. Fang, Z. Qi, H. Deng, A review of remaining useful life prediction approaches for mechanical equipment, IEEE Sensors Journal 23 (2023) 29991–30006. doi:10.1109/JSEN.2023.3326487.
[37] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 785–794. URL: https://doi.org/10.1145/2939672.2939785. doi:10.1145/2939672.2939785.
[38] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, A. Gulin, Catboost: unbiased boosting with categorical features, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, Curran Associates Inc., Red Hook, NY, USA, 2018, p. 6639–6649.
[39] N. Hollmann, S. Müller, L. Purucker, A. Krishnakumar, M. Körfer, S. B. Hoo, R. T. Schirrmeister, F. Hutter, Accurate predictions on small data with a tabular foundation model, Nature (2025). URL: https://www.nature.com/articles/s41586-024-08328-6. doi:10.1038/s41586-024-08328-6.
[40] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, Journal of the Royal Statistical Society. Series B (Statistical Methodology) 67 (2005) 301–320. URL: http://www.jstor.org/stable/3647580.
[41] L. Breiman, Random forests, Machine Learning 45 (2001) 5–32. doi:10.1023/A:1010950718922.
[42] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. doi:10.1109/

CVPR.2016.90.

[43] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, F. Petitjean, Inceptiontime: Finding alexnet for time series classification, Data Mining and Knowledge Discovery 34 (2020) 1936–1962.

[44] B. Zhao, H. Lu, S. Chen, J. Liu, D. Wu, Convolutional neural networks for time series classification, Journal of Systems Engineering and Electronics 28 (2017) 162–169. doi:10.21629/JSEE.2017.01.18.

[45] Z. Wang, W. Yan, T. Oates, Time series classification from scratch with deep neural networks: A strong baseline, in: 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 1578–1585. doi:10.1109/IJCNN.2017.7966039.

[46] F. Karim, S. Majumdar, H. Darabi, S. Harford, Multivariate lstm-fcns for time series classification, Neural Networks 116 (2019) 237–245. URL: https://www.sciencedirect.com/science/article/pii/S0893608019301200. doi:https://doi.org/10.1016/j.neunet.2019.04.014.

[47] A. Dempster, F. Petitjean, G. I. Webb, Rocket: exceptionally fast and accurate time series classification using random convolutional kernels, Data Mining and Knowledge Discovery 34 (2019) 1454 – 1495. URL: https://api.semanticscholar.org/CorpusID:204949593.

[48] Z. Kharazian, T. Lindgren, S. Magnússon, O. Steinert, O. A. Reyna, Scania component x dataset: A real-world multivariate time series dataset for predictive maintenance, 2025. URL: https://arxiv.org/abs/2401.15199. arXiv:2401.15199.

[49] A. Biswas, Microsoft azure predictive maintenance, Kaggle Dataset, ???? URL: https://www.kaggle.com/datasets/arnabbiswas1/microsoft-azure-predictive-maintenance, accessed: 2025-06-01.

[50] S. Sandha, M. Aggarwal, S. S. Saha, M. Srivastava, Enabling hyperparameter tuning of machine learning classifiers in production, 2021. doi:10.1109/CogMI52975.2021.00041.

[51] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (1945) 80–83. URL: http://www.jstor.org/stable/3001968.