Failure-Mode-Informed Development of Remaining Useful Life Prognostics

Kiavash Fathi[1], Mihaela Mitici[2], Tobias Kleinert[3], and Hans Wernher van de Venn[4]

[1,4] *Institute of Mechatronic Systems, Zurich University of Applied Sciences, 8400 Winterthur, Switzerland*
fath@zhaw.ch
vhns@zhaw.ch

[2] *Faculty of Science, Utrecht University, Heidelberglaan 8, 3584 CS, Utrecht, The Netherlands*
m.a.mitici@uu.nl

[1, 3] *Chair of Information and Automation Systems for Process and Material Technology, RWTH Aachen University, 52064 Aachen, Germany*
kiavash.fathi@rwth-aachen.de
kleinert@iat.rwth-aachen.de

ABSTRACT

Various environmental and operating conditions affect the degradation behavior of physical assets, leading to various degradation trajectories and ultimately to distinct failure modes. To obtain accurate Remaining Useful Life (RUL) prediction, it is important to distinguish between such degradation trajectories and their associated failure modes. In this paper, we develop a framework where we analyze the latent space of autoencoders using spectral clustering to evaluate the similarity in degradation trajectories and failure modes in training datasets. This failure-mode-informed training sets are then used to develop failure-specific regressors for RUL prediction. On one hand, this reduces the amount of data needed to effectively train prognostics models. In addition, the accuracy of the RUL predictions is further improved. We demonstrate this using the C-MAPSS dataset, which provides fleet-based run-to-failure sequences under varying operating conditions and failure modes. We argue that latent information about different degradation mechanisms can be inferred from sensor readings, enabling the construction of failure-mode-specific RUL regressors. Our results show that this failure-mode-informed data separation reduces the amount of training data needed to generate RUL prognostics by up to 55%, while simultaneously improving prognostics accuracy - the Root Mean Square Error (RMSE) is reduced by 3%.

1. INTRODUCTION

With the rapid advancement of digitalization across industries, an ever-increasing volume of data is being constantly generated (Fähndrich, 2023). This surge in data generation increases the demand on the IT infrastructure, escalates the costs and efforts associated with data storage, processing and also lifecycle management associated with AI-enhanced models (Legenvre, Autio, & Hameri, 2025). The sheer scale of data complicates the creation and maintenance of robust data pipelines and machine learning solutions (Yang & Desell, 2022; Faubel, Schmid, & Eichelberger, 2023).

Furthermore, the constant changes in industrial use cases lead to different types of data distribution shift (DDS) (Fathi & van de Venn, 2024). DDS detection and taking measures against it, ensures the performance of the deployed AI-enhanced solution and the accuracy of its predictions. In fact, unlike conventional software problems, issues with AI models normally demonstrate themselves as reduced efficiency by drop in their accuracy and/or reliability (Fathi, Ristin, Sadurski, Kleinert, & Van De Venn, 2024). In that regard, DDS detection further is exasperated by the large data volumes, making data and model maintenance in some scenarios close to impossible when not scaled properly.

In the related literature targeting DDS in the input distribution of AI models, one approach is to use domain adaptation techniques to overcome its hurdles (Li et al., 2021; Liu et al., 2020). These solutions use data from the target distribution to adapt the model trained on a source training distribution (Nejjar, Geissmann, Zhao, Taal, & Fink, 2024). The mere assumption here is that the target data distribution is dif-

ferent from the training distribution. This as a result puts focus only on out-of-distribution scenarios. These solutions do not consider the possibility of having different source components in the system generating the data (Zhou, Liu, Qiao, Xiang, & Loy, 2022).

In the domain of predictive maintenance, there are several sources of change that result in data generation from different working conditions of a given asset. These so-called source components can be generated by the measures taken directly by the asset operator (Quiñonero-Candela, Sugiyama, Schwaighofer, & Lawrence, 2022). As an example, a CNC machine can be programmed to produce different types of product units. The changes in the production procedure results in different signal sequences read from the asset, creating different source components for different types of product units (Fathi, Sadurski, Kleinert, & van de Venn, 2023).

While foundation models and autoencoder architectures have been extensively explored in recent years, there remains a notable gap in practical approaches that explicitly utilize the latent information available in the input data distribution to dynamically detect and separate different source components (such as specific failure modes) prior to model training (Yan et al., 2024; Wu, Wu, Tan, & Xu, 2024; Fathi, van de Venn, & Honegger, 2021; Hou, Xu, Zhou, Yang, & Fu, 2020). This latent information can be the impact of uncontrollable factors such as different environmental conditions, wear and tear in system, failure modes, which should be taken into account when training data-driven solutions. In fact, it is shown in this paper that by training separate AI-enhanced models per source component, it is possible to reach higher accuracy with significantly less data. In the current paper, we use the latent information indicating the failure mode of the studied system to separate the available data.

We illustrate our approach using the C-MAPSS dataset (Saxena & Goebel, 2008). This dataset includes multiple operating conditions and failure modes (in FD001-FD004 datasets), which impact the input data distributions. Despite the availability of this information, most studies on RUL prediction concentrate merely on enhancing prediction model accuracy (Rivas, Delipei, & Hou, 2022; Jiang & Kuo, 2017), detecting and fusing anomalies and alarm thresholds in the analysis (Aydemir & Acar, 2020; Wang & Zhao, 2024; Mitici, de Pater, Barros, & Zeng, 2023; De Pater, Reijns, & Mitici, 2022), sidelining dataset-specific characteristics and the impact of different failure modes of the system. Complementary to these studies, we aim to demonstrate how failure-aware regression for predicting the RUL of an asset can reduce the data volume required whilst increasing its prediction accuracy.

Our approach is in line with the new AI regulations from European Union (EU). Specifically, Article 10(4) of the EU AI act denotes that during model training for a high-risk AI system, the training data must reflect the specific context in which the system will be used (European Union, 2024). This includes functional purpose, operating environment, *etc.*, only to the extent necessary for the system's intended use. Therefore, for compliance with the said article, we partition the training dataset from the C-MAPPS dataset by failure mode to reflect the distinct functional and contextual settings associated with the degradation in the system. This enables us to develop failure-specific models which not only require less data per class but also enhance RUL predictive performance.

2. DATA DESCRIPTION

The C-MAPPS dataset contains run-to-failure data from simulated turbofan engines. This datasets is created by NASA using the Commercial Modular Aero-Propulsion System Simulation (C-MAPPS) (Saxena & Goebel, 2008). This dataset contains timeseries of 21 sensors, available in 4 subsets, namely, FD001, FD002, FD003 and FD004 as follows:

- **FD001**: one operating condition and one failure mode
- **FD002**: multiple operating conditions and one failure mode
- **FD003**: one operating condition and two failure modes
- **FD004**: multiple operating conditions and two failure modes

In this study, we focus on the FD003 subset. Subsets FD001 and FD002 contain only a single failure mode, which does not support our objective of discriminating between different faults. Conversely, FD004 involves both multiple failure modes and multiple operating conditions. Using FD004 would introduce confounding variables, making it difficult to isolate the specific influence of the failure modes from the influence of changing operating conditions using our current methodology.

3. METHODOLOGY

In this section we describe the steps of our approach to develop failure-mode-informed RUL prognostics by identifying and clustering engines with similar failure modes. Learning only on the degradation data (measurements) of a specific cluster, we are developing specialized, fault-specific regressors for RUL prediction. In this way, we are able to reduce significantly the amount of data needed to train the regressors while improving the accuracy of the RUL predictions. Below the steps of our proposed approach, see also Figure 1.

Step 1: Labeling training engines as Failure mode 1 or Failure mode 2

We note that the engines in CMAPPS, folder FD003, do not have a label indicating their failure mode. It is only known that these engines have either Failure mode 1 or Failure mode 2. In this step, we aim to label the engines in the training set as having Failure mode 1 or Failure mode 2 using an autoen-

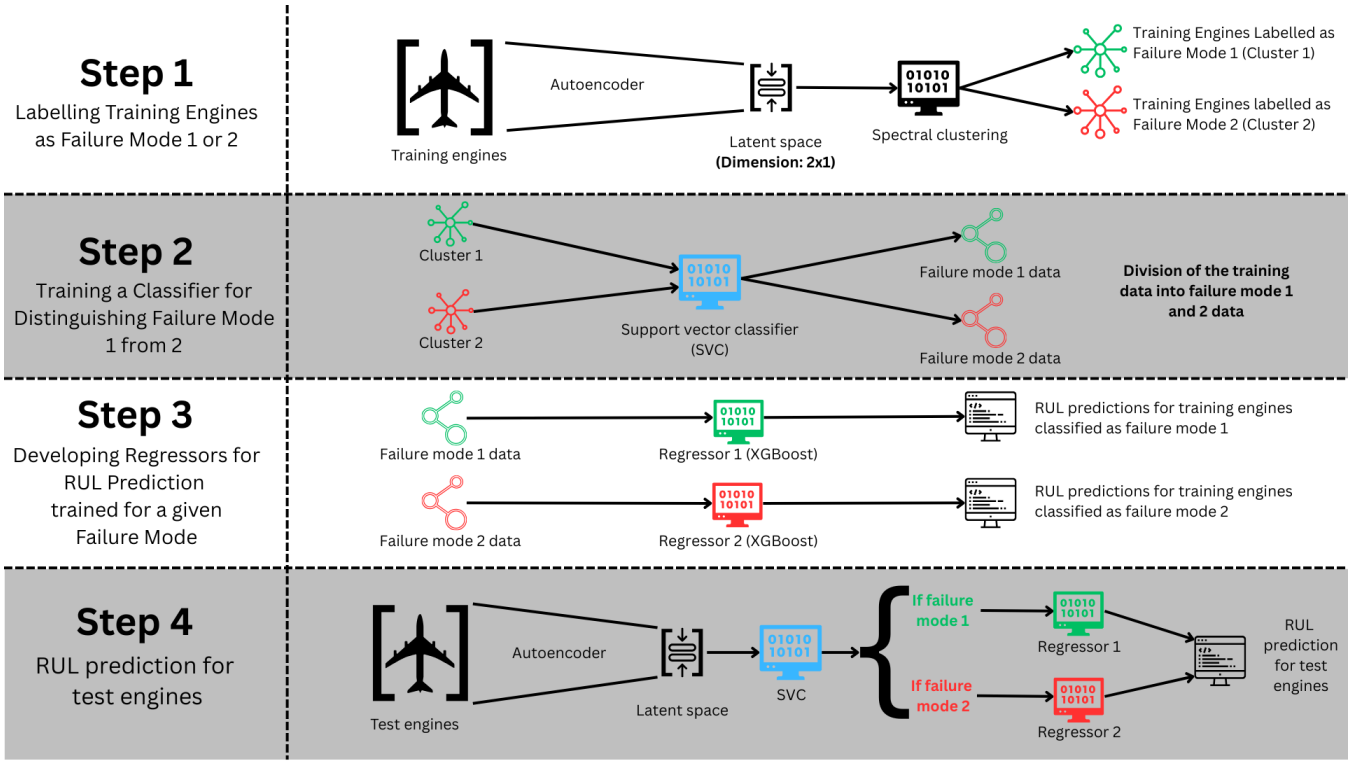


Figure 1. Overview of the proposed framework for failure-informed development of RUL prognostics.

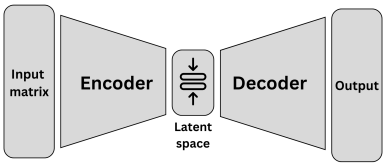


Figure 2. General overview of an autoencoder.

coder and spectral clustering, as follows.

First, the engine training data is used to train an autoencoder, see Figure 2. Table 1 shows the architecture of this autoencoder. The encoder is used to generate a latent space from the input data (training engines). The dimension of the latent space is set to 2×1 to easily visualize and interpret the possible clusters of the training engines in two dimensions. While higher dimensions could theoretically capture more complex fault representations, a two-dimensional space was explicitly chosen to provide an easily interpretable, visualizable representation of the data. To enhance generalizability to datasets with more than two failure modes, future work could expand the latent space dimensionality. The optimal number of dimensions could be established through expert evaluation of emerging clusters or by analyzing the mathematical distance between clusters to ensure distinct separation. This latent space is not further used for the decoder since we are interested only in analyzing the structure of the latent space generated by the encoder.

The obtained latent space is then analyzed using spectral clustering (Von Luxburg, 2007). Spectral clustering groups data points by interpreting them as a graph. The edges of this graph indicate the similarity between data points. This similarity is determined using a Gaussian kernel as follows:

$$A_{ij} = exp(-\frac{||x_i - x_j||^2}{2\sigma^2}), \quad (1)$$

where x_i and x_j are two arbitrary data points and σ controls the width of the neighborhood. Using the similarity scores, the data points are embedded using the eigenvectors of the graph Laplacian and clustered in this spectral space. As a result, we obtain 2 clusters of data points (see Figure 3). Each datapoint corresponds to 13 consecutive cycles of sensor measurements (window size of the autoencoder) for a specific training engine, with the associated true RUL corresponding to the last day of the window. The results show 2 well-separated clusters. Our hypothesis is that these 2 clusters represent the two failure modes available in FD003. We verify this hypothesis empirically in Section 4.

Overall, Step 1 provides 2 clusters of data points from the training engine data, which we label as data points of Failure mode 1 and data points of Failure mode 2.

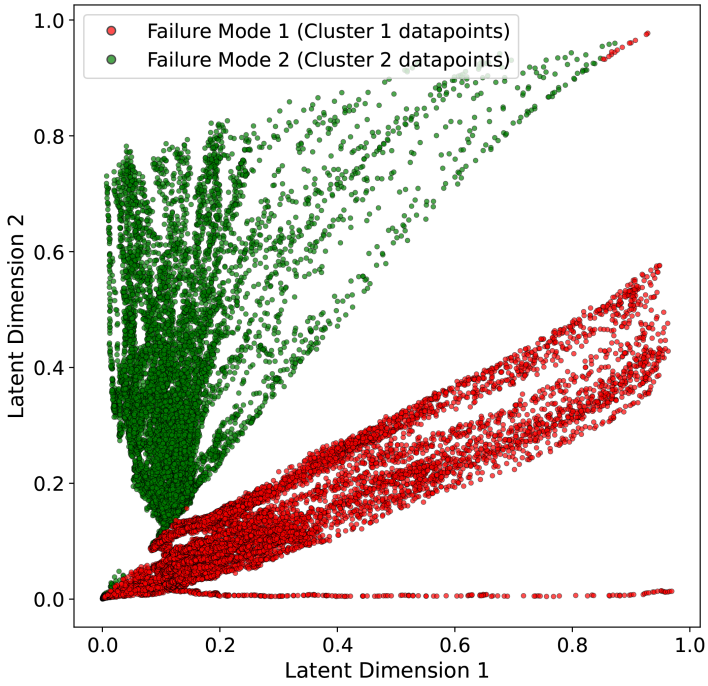


Figure 3. Failure mode 1 and Failure mode 2 clusters as the output of spectral clustering in the latent space (Step 1).

Step 2 - Training a Support Vector Classifier (SVC) to distinguish Failure mode 1 from Failure mode 2

Having obtained the 2 clusters of failure modes for the training engine data in Step 1, in Step 2 we now train a SVC to obtain a classification decision boundary separating the data points labeled as Failure mode 1 from the data points labeled as Failure mode 2. This decision boundary facilitates a fast failure mode classification of new data points. Table 2 shows the hyperparameters of the SVC.

Figure 4a shows the decision boundary (solid yellow line). The darker the color of the data points, both green and red, the higher the true RUL value associated with these data points, i.e., the engine is in the initial phase of the monitoring. The results show that the SVC is confident in classifying data points of low RUL (close to failure), with SVC decision score close to ± 240 corresponding to top-left and bottom-right corners. SVC is less confident in classifying data points of high RUL (far from failure), with the SVC decision score close to 0 corresponding to the bottom-left corner.

Following the results obtained with SVC, we refer to the engine training data points classified as red in Figure 4a as Failure mode 1 data C_1 with $||C_1|| = 10611$ training data points, and the ones classified as green data points as Failure mode 2 data C_2 with $||C_2|| = 12909$ training data points.

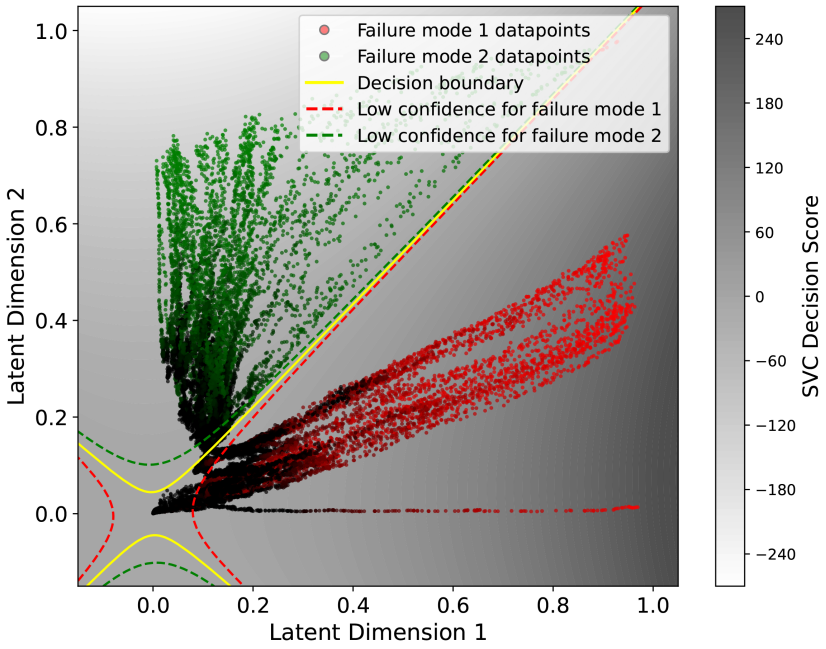
Step 3 - Developing failure-specific regressors for RUL prediction trained using either data points in C_1 or C_2

Here we aim to develop failure-specific regressors R_1 and R_2 for RUL prediction, by training them either with data from Failure mode 1 (C_1) or Failure mode 2 (C_2), respectively. We consider XGBoost (Chen & Guestrin, 2016) to train these regressors. For each regressor the hyper parameters of the XGBoost are attained using a grid search.

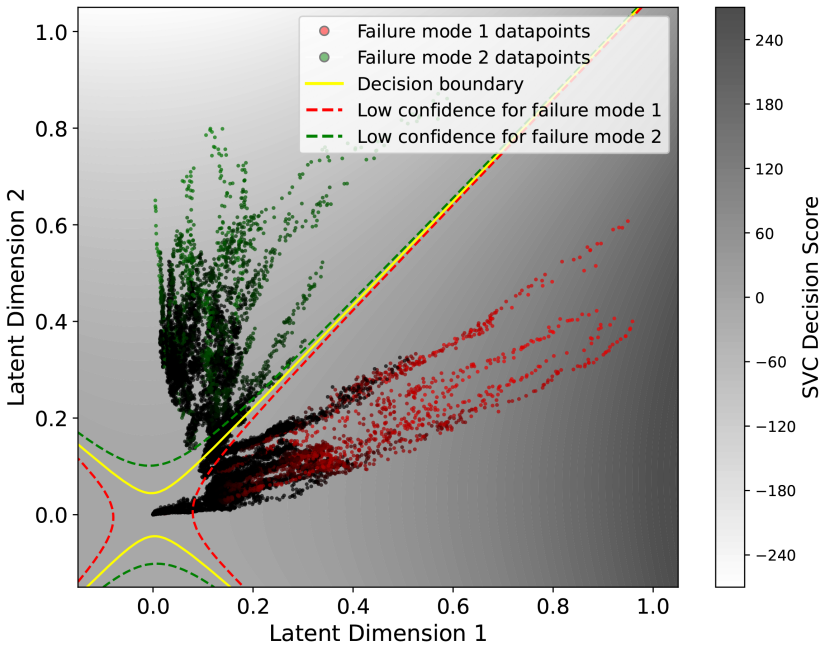
For comparison reasons, we also consider a third regressor R for RUL prediction, which is trained on all data available ($C = C_1 \cup C_2$), we do not consider failure modes.

Step 4 - RUL prediction for test engines using failure-specific regressors

In this step, we estimate the RUL of the test engines in FD003 as follows. First, we classify the engine as Failure mode 1 or Failure mode 2 using the trained SVC (Step 2). Once the test engine has been classified as Failure mode 1 (Failure mode 2), we select the fault-specific regressor R_1 (R_2) to estimate the RUL of this engine. We mention that RUL estimation takes into account the last 13 sensor measurement available for this test engine. By using a fault-specific regressor, we are able to use less data that is that is more relevant (same fault mode) for the considered test engine, and at the same time improve



(a) FD003 training data



(b) FD003 test data

Figure 4. **FD003** Failure mode classification in latent space via SVC. Darker points represent higher RUL values.

Table 1. Architecture of the Autoencoder Network.

Layer Type	Output	Parameter & Details
Input	$(13, 21)$	**Input layer**
BatchNormalization	$(13, 21)$	Normalization
Flatten	13×21	—
Dense	256	Fully connected layer
LeakyReLU	256	$\alpha = 0.3$
Dense	128	—
LeakyReLU	128	$\alpha = 0.3$
Dense (Latent)	2	sigmoid activation L1 ($\lambda = 0.01$)
LeakyReLU	2	$\alpha = 0.3$
Dense	128	—
LeakyReLU	128	$\alpha = 0.3$
Dense	256	—
LeakyReLU	256	$\alpha = 0.3$
Dense	13×21	Reconstruction layer
LeakyReLU	13×21	$\alpha = 0.3$
Reshape	$(13, 21)$	**Output layer**

Table 2. SVC hyperparameters.

Parameter	Value
Kernel	Polynomial
Degree	2
Gamma	$1/(input\ size \times input\ variance)$
Tolerance (`tol`)	1×10^{-5}

RUL prediction accuracy.

For comparison reasons, we also estimate the RUL of the test engines in FD003 using the regressor R which has been trained on the entire training dataset C. In Section 5, we compare the accuracy of our proposed method (regressors R_1 and R_2) with the baseline regressor R.

4. EMPIRICAL VERIFICATION OF CLUSTERS REPRESENTING FAILURE MODES

Our proposed solution focuses mainly on separating different failure modes and then training separate RUL prediction models per each failure mode. However, there is no information available from the NASA C-MAPSS dataset, indicating the failure mode of the run-to-failure sequences in folder FD003. Therefore, we conduct the following experiments to ensure that the mentioned failure mode 1 and failure mode 2 data actually represent the failure types implicitly contained in the FD003 dataset.

We claim that, if the failure mode prediction from the SVC is consistent for each available run-to-failure sequence, then the classification can indicate the failure mode of the said sequence. To examine the stability of the SVC predictions, we quantized each run-to-failure sequence into 5 intervals as follows. **Interval 1** (true RUL values range from 125 to 100 cycles), **Interval 2** (true RUL values range from 100 to 75 cycles), **Interval 3** (true RUL values range from 75 to 50 cycles), **Interval 4** (true RUL values range from 50 to 25 cycles), **Interval 5** (true RUL values range from 25 to 0 cycles).

Thereafter, for each interval in the training and test engine data, we count the changes in the output of the classification of the SVC model. Table 3, indicates the changes in each RUL interval.

Table 3. Consistency in failure mode classification across training and test data for 100 engine data (units).

RUL Interval in Cycles	Training: failure mode changes	Test: failure mode changes
[125, 100)	4 / 100	14 / 100
[100, 75)	0 / 100	0 / 100
[75, 50)	0 / 100	0 / 100
[50, 25)	0 / 100	2 / 100
[25, 0]	2 / 100	0 / 100

As shown in this table, the majority of failure mode classification inconsistency is in the interval of 125 to 100 days, which contains the highest RUL values possible. In fact, in the healthy state of an aircraft engine, which is equivalent to data from interval 5, it is not feasible to identify the failure which will occur in the up coming cycles. Thus, the higher inconsistency in the SVC predictions for this interval is negligible and we can assume that the outputs of the SVC actually represent the failure modes in the dataset.

For a better inspection of the data which suffer from SVC prediction inconsistency, as also plotted the run-to-failure data with the each cycle colored and marked with its corresponding failure mode in Figures 5 and 6.

As the last step for verifying our assumptions, we use the developed model on the rest of the available data from the C-MAPSS dataset. Figures 7, 8 and 9 show the results of using our solution on dataset which either do not have 2 failure mode (either 1 or more than 2 failure modes) or have different operating conditions as opposed to the prerequisites of failure exactly 2 failure modes and one operating condition.

5. RESULTS - RUL PREDICTION

In this section we quantify the performance of our proposed approached (steps 1-4) where we consider failure-specific regressor to estimate the RUL of the test engines. We compare this performance against the case where a generic regressor is used for RUL prediction where no information about the failure modes is considered.

Let T_1 and T_2 denote the set of test engines in FD003 labeled as Fault mode 1 or Fault mode 2 respectively, where $||T_1|| = 55$ and $||T_2|| = 45$. Let $T = T_1 \cup T_2$ denote the set containing all test engines. Let RUL_i^{True} denote the true

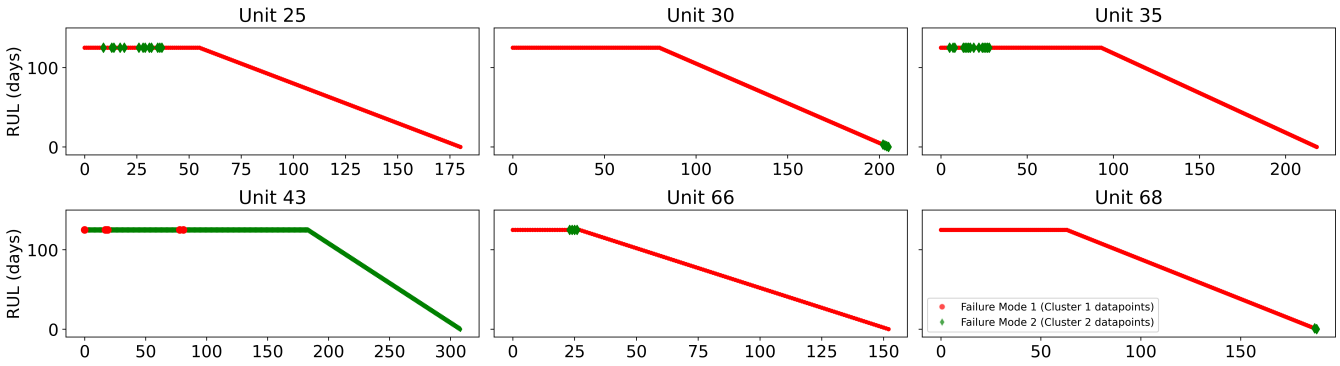


Figure 5. Training engine data with inconsistent failure mode classification

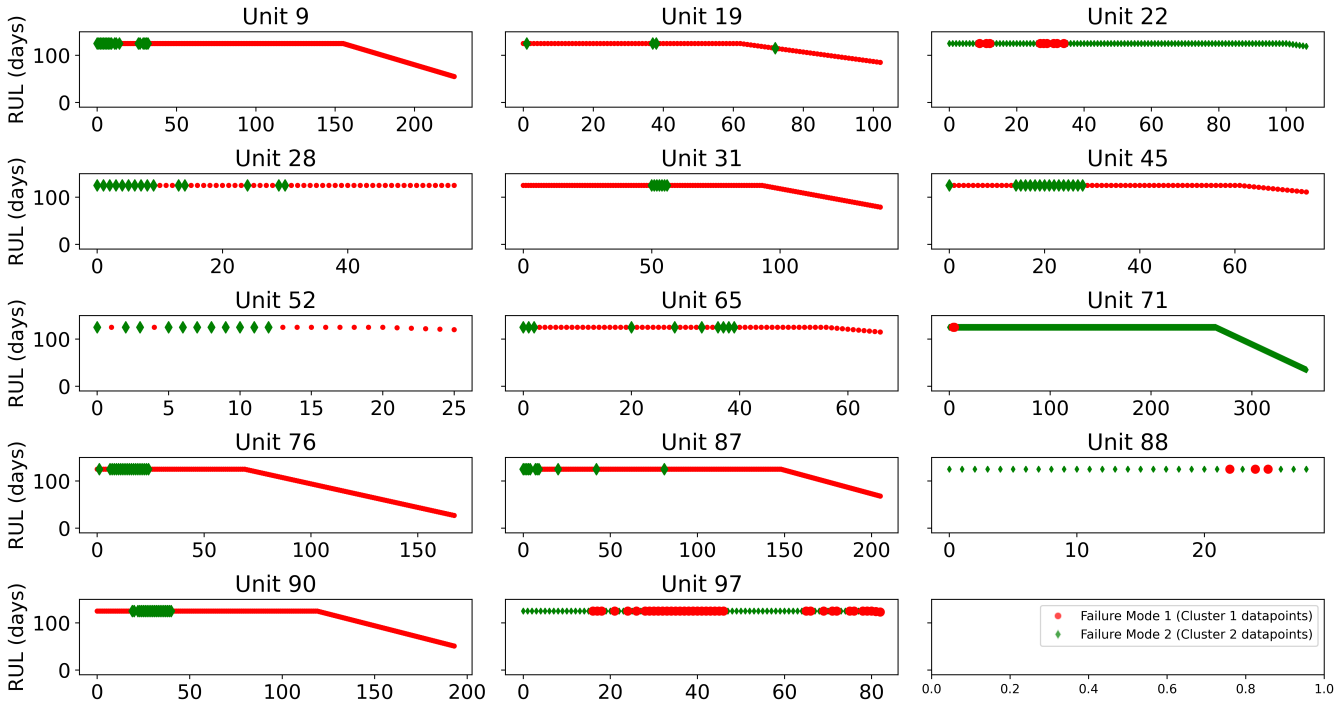


Figure 6. Test engine data with inconsistent failure mode classification

RUL of the i^{th} engine, $i \in \{T_1, T_2, T\}$. Let $\widehat{RUL}_i^{Regressor}$, where $Regressor \in \{R_1, R_2, R\}$, $i \in \{T_1, T_2, T\}$.

For each test engine, we consider the data available from the last 13 cycles to estimate their RUL using our methodology in Section 3. Below are the performance metrics used to evaluate the RUL predictions.

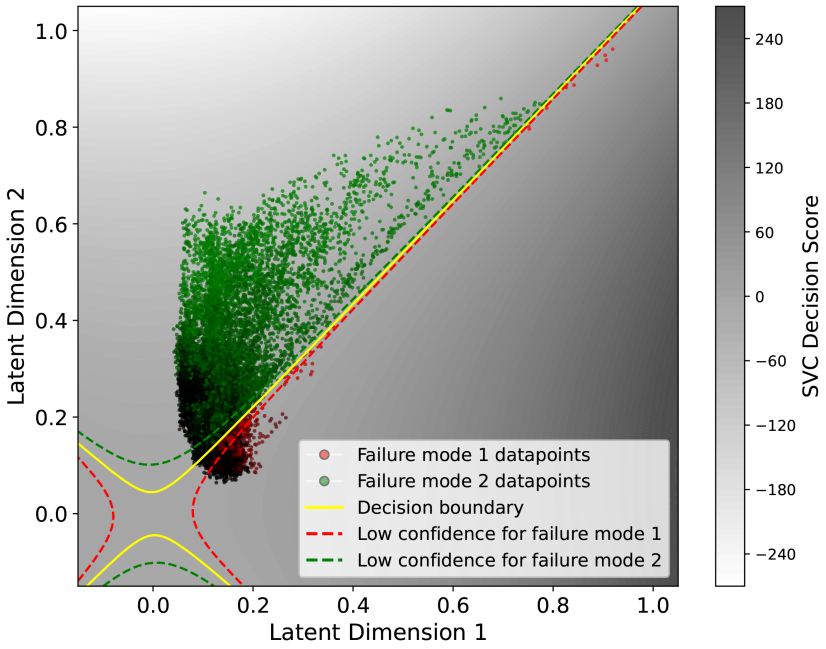
1. RMSE between the RUL predictions of the failure-specific regressor R_1 for the test engines in T_1 and the corresponding true RUL:

$$RMSE_{T_1,R_1} = \frac{\sqrt{\sum_i (RUL_i^{True} - \widehat{RUL}_i^{R_1})^2}}{||T_1||}, i \in T_1 \quad (2)$$

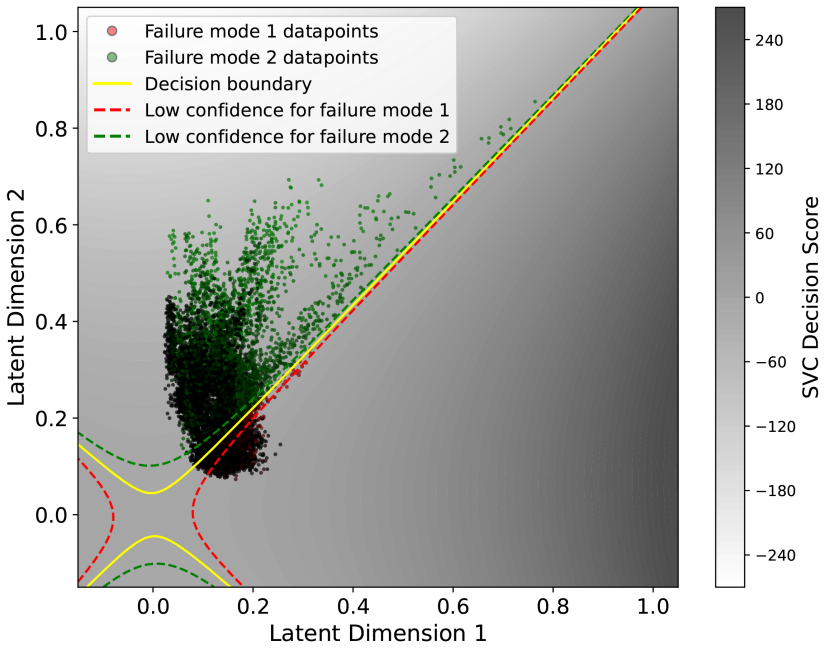
2. $RMSE_{T_1,R}$: The root mean squared error between the predictions of the failure-specific regressor R_1 for the test engines in T_1 and the corresponding ground truth values calculated as

$$\frac{\sqrt{\sum_i (RUL_i^{True} - \widehat{RUL}_i^{R})^2}}{||T_1||}, i \in T_1 \quad (3)$$

3. $RMSE_{T_2,R_2}$: The root mean squared error between the

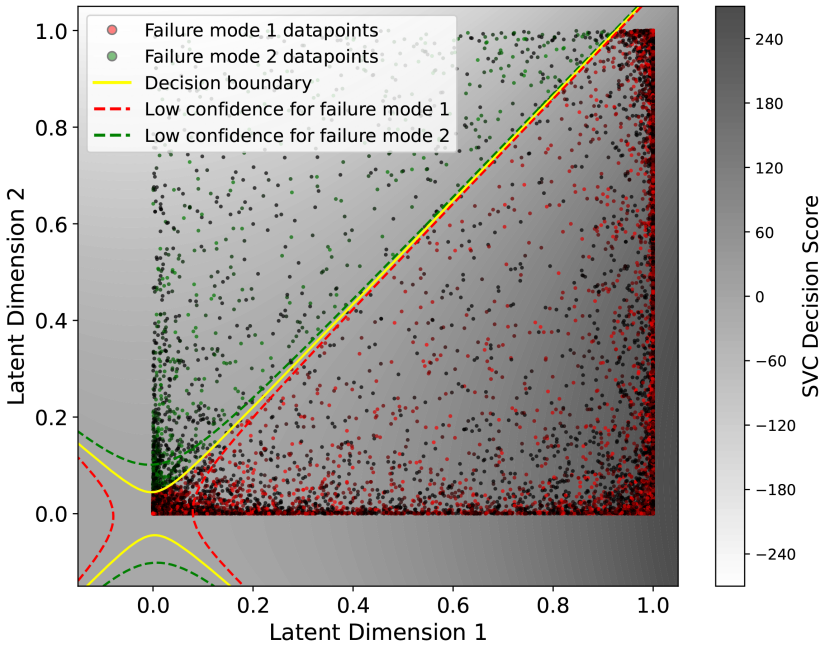


(a) FD001 training data

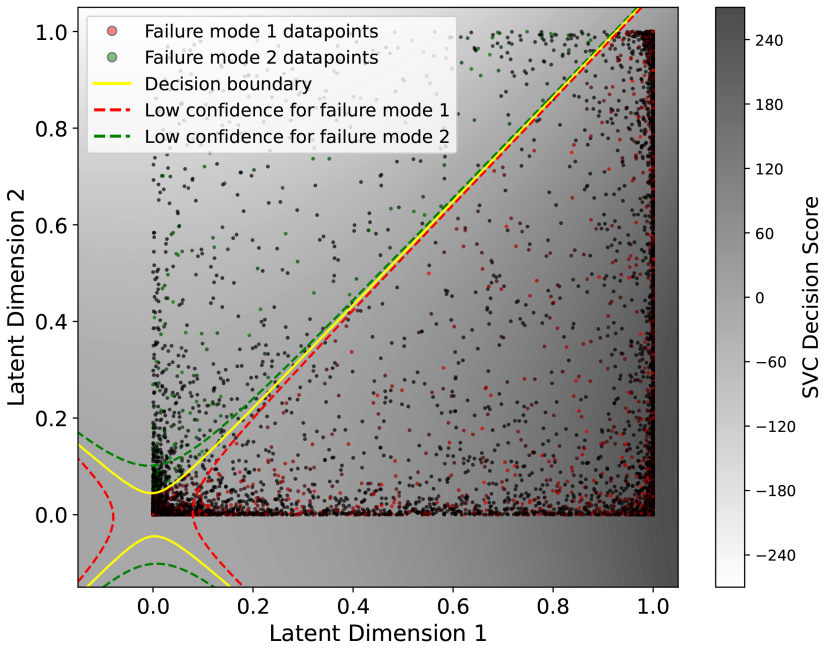


(b) FD001 test data

Figure 7. **FD001:** Run-to-failure data classification in latent space with **one operating condition** and **one failure mode** - Darker points represent higher RUL span

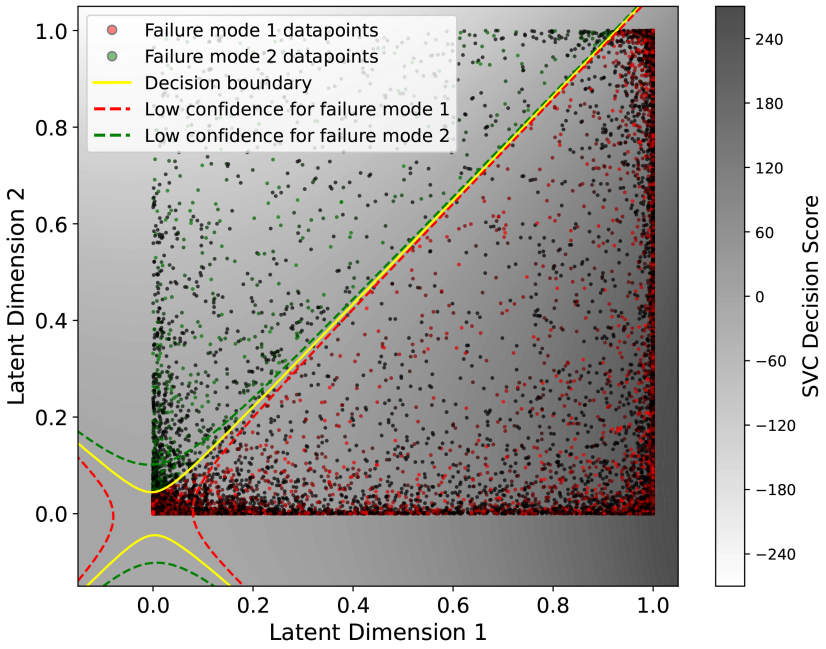


(a) FD002 training data

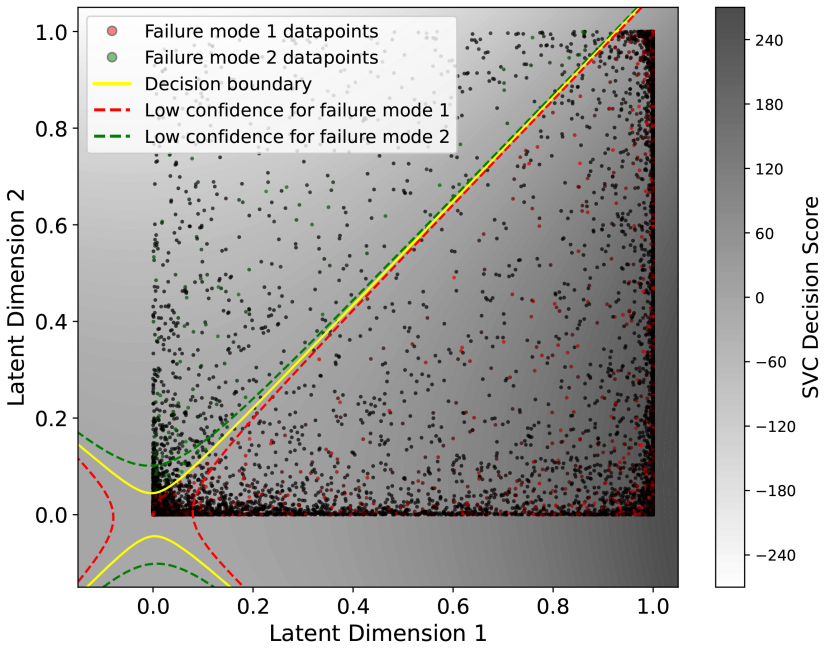


(b) FD002 test data

Figure 8. **FD002:** Run-to-failure data classification in latent space with **multiple operating conditions** and **one failure mode** - Darker points represent higher RUL span



(a) FD004 training data



(b) FD004 test data

Figure 9. **FD004:** Run-to-failure data classification in latent space with **multiple operating conditions** and **two failure mode** - Darker points represent higher RUL span

predictions of the regressor R_2 for the test data in T_2 and the corresponding ground truth values calculated as

$$\frac{\sqrt{\sum_i (RUL_i^{True} - \widehat{RUL}_i^{R_2})^2}}{||T_2||}, i \in T_2 \tag{4}$$

4. $RMSE_{T_2,R}$: The root mean squared error between the predictions of the regressor R for the test data in T_2 and the corresponding ground truth values calculated as

$$\frac{\sqrt{\sum_i (RUL_i^{True} - \widehat{RUL}_i^{R})^2}}{||T_2||}, i \in T_2 \tag{5}$$

5. $RMSE_{T,R_1\&R_2}$: The RMSE between the predictions of the failure-specific regressors R_1 and R_2 for the test engines in T_1 and T_2 respectively, and the corresponding true RUL:

$$\frac{RMSE_{T_1,R_1} \times ||T_1|| + RMSE_{T_2,R_2} \times ||T_2||}{||T||} \tag{6}$$

6. $RMSE_{T,R}$: The root mean squared error between the predictions of the regressor R for the test data in T and the corresponding ground truth values calculated as

$$\frac{\sqrt{\sum_i (RUL_i^{True} - \widehat{RUL}_i^{R})^2}}{||T||}, i \in T \tag{7}$$

We also evaluate whether our regressors overestimate or underestimate the RUL using the following asymmetric performance metrics.

$$Asym_{D,Regressor} = \sum_i score(e_i), i \in D, \tag{8}$$

where

$$\text{score}(e_i) = \begin{cases} \exp\left(-\frac{e_i}{13}\right) - 1, & \text{if } e_i < 0 \\ \exp\left(\frac{e_i}{10}\right) - 1, & \text{if } e_i \geq 0, \end{cases} \tag{9}$$

with $e_i = RUL_i^{True} - \widehat{RUL}_i^{Regressor}$.

Here, $D \in \{T_1, T_2, T\}$ and $Regressor \in \{R_1, R_2, R\}$.

We also consider the overall asymmetric performance metric: $Asym_{T,R_1\&R_2}$ = $Asym_{T,R_1} + Asym_{T_2,R_2}$

RUL prediction - performance evaluation

Table 4 shows, that when using Fault mode 1 specific regressor R_1 trained C_1, RMSE obtained for the test engines in T_1 is 16.20. Similarly, when using Fault mode 2 specific regressor R_2 trained on C_2, the RMSE obtained for the test engines in T_2 is 19.22. Most importantly, when considering both these results together as an aggregate, we obtain the $RMSE_{T,R_1\&R_2} = 17.62$ which is lower than $RMSE_{T,R} = 18.18$ obtained using a generic regressor that is trained on the entire training dataset $C = C_1 \cup C_2$, without using any information about the failure modes of the engines. In other words, the results show that we are able to increase the accuracy of RUL predictions by classifying the test engines as belonging to Failure mode 1 or Failure mode 2 and by using failure-specific regressors.

Table 5 shows, that when using Fault mode 1 specific regressor R_1 trained C_1, Asymmetric performance metric ($Asym$), obtained for the test engines in T_1 is 319.62. Similarly, when using Fault mode 2 specific regressor R_2 trained on C_2, the $Asym$ obtained for the test engines in T_2 is 557.58. Considering the aggregate of these two results $Asym_{T,R_1\&R_2} = 877.20$ which is lower than $Asym_{T,R} = 879.35$ obtained using a generic regressor. Also, these results shows again that is beneficial to use fault-specific regressors rather than generic regressors for RUL prediction even when we evaluate the tendency of the regressors to over or underestimate the RUL.

The provided tables, also demonstrate the accuracy of the trained models across different data slices representing failure mode 1 and failure mode 2.

6. CONCLUSION

In this paper, we have proposed a framework to develop RUL prognostics by using dedicated, failure-mode-informed training datasets. This approach analyzes the latent space obtained by processing the information available in the sensor measurements using autoencoders, and clusters degradation trajectories and failure modes in the training datasets. We have demonstrated how this approach can facilitate model training by significantly reducing the data volume required to train a RUL prognostics regressor (up to 55%), while simultaneously increasing its accuracy (reducing the RMSE by 3%). Although the absolute increase in RUL prediction accuracy may seem modest, the drastic reduction in required training data yields significant computational benefits. It inherently lowers the infrastructure costs associated with data storage, pipeline maintenance, and model retraining cycles, factors that are highly advantageous for scaling up AI solutions in real-world industrial applications. As future work, we aim to cluster and classify the working condition of the available data as well as to further limit the data regime used during model training for increased RUL prognostics accuracy.

ACKNOWLEDGMENT

We extend our thanks to Utrecht University for hosting one of the co-authors (Kiavash Fathi) as a visiting PhD student. The inspiring academic atmosphere, generous support, and open collaboration provided by both the faculty and staff greatly enriched the research and made this period especially meaningful. Their commitment to fostering cross-institutional research has played a vital role in making this collaboration possible.

	R_1	R_2	
Test engines classified as fault 1 ($\|\|T_1\|\|$ = 55)	$RMSE_{T_1,R_1}$=16.20	N/A	
Test engines classified as fault 2 ($\|\|T_2\|\|$ = 45)	N/A	$RMSE_{T_2,R_2}$=19.22	
All test engines ($T = 100$)	$RMSE_{T,R_1\&R_2}$=17.62		$RMSE_{T,R}$=18.18

Table 4. RMSE values per fault class and regression model

	R_1	R_2	
Test engines classified as fault 1 ($\|\|T_1\|\|$ = 55)	$Asym_{T_1,R_1} = 319.62$	N/A	
Test engines classified as fault 2 ($\|\|T_2\|\|$ = 45)	N/A	$Asym_{T_2,R_2} = 557.58$	
All test engines ($T = 100$)	$Asym_{T,R_1\&R_2} = 877.20$		$Asym_{T,R} = 879.35$

Table 5. Asym values per fault class and regression model

REFERENCES

Aydemir, G., & Acar, B. (2020). Anomaly monitoring improves remaining useful life estimation of industrial machinery. *Journal of Manufacturing Systems*, *56*, 463–469.

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* (pp. 785–794).

De Pater, I., Reijns, A., & Mitici, M. (2022). Alarm-based predictive maintenance scheduling for aircraft engines with imperfect remaining useful life prognostics. *Reliability Engineering & System Safety*, *221*, 108341.

European Union. (2024). *Regulation (eu) 2024/1689, article 10: Data and data governance.* Official text of the Artificial Intelligence Act. Retrieved from https://artificialintelligenceact.eu/article/10/ (Accessed via ArtificialIntelligenceAct.eu on July 15, 2025)

Fähndrich, J. (2023). A literature review on the impact of digitalisation on management control. *Journal of management control*, *34*(1), 9–65.

Fathi, K., Ristin, M., Sadurski, M., Kleinert, T., & Van De Venn, H. W. (2024). Detection of novel asset failures in predictive maintenance using classifier certainty. In *2024 32nd mediterranean conference on control and automation (med)* (pp. 50–56).

Fathi, K., Sadurski, M., Kleinert, T., & van de Venn, H. W. (2023). Source component shift detection & classification for improved remaining useful life estimation in alarm-based predictive maintenance. In *2023 23rd international conference on control, automation and systems (iccas)* (pp. 975–980).

Fathi, K., & van de Venn, H. W. (2024). Data, models, and performance: A comprehensive guide to predictive maintenance in industrial settings. In *Recent topics in maintenance management.* IntechOpen.

Fathi, K., van de Venn, H. W., & Honegger, M. (2021). Predictive maintenance: an autoencoder anomaly-based approach for a 3 dof delta robot. *Sensors*, *21*(21), 6979.

Faubel, L., Schmid, K., & Eichelberger, H. (2023). Mlops challenges in industry 4.0. *SN Computer Science*, *4*(6), 828.

Hou, G., Xu, S., Zhou, N., Yang, L., & Fu, Q. (2020). Remaining useful life estimation using deep convolutional generative adversarial networks based on an autoencoder scheme. *Computational Intelligence and Neuroscience*, *2020*, 1–14. doi: 10.1155/2020/9601389

Jiang, J.-R., & Kuo, C.-K. (2017). Enhancing convolutional neural network deep learning for remaining useful life estimation in smart factory applications. In *2017 international conference on information, communication and engineering (icice)* (pp. 120–123).

Legenvre, H., Autio, E., & Hameri, A.-P. (2025). Creating value by combining ai and other open technologies: Cloud infrastructure as a pivotal asset. In *Contemporary issues in industry 5.0: Towards an ai integrated society* (pp. 137–161). Springer Nature Switzerland Cham.

Li, B., Wang, Y., Zhang, S., Li, D., Keutzer, K., Darrell, T., & Zhao, H. (2021). Learning invariant representations and risks for semi-supervised domain adaptation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 1104–1113).

Liu, Z., Miao, Z., Pan, X., Zhan, X., Lin, D., Yu, S. X., & Gong, B. (2020). Open compound domain adaptation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 12406–12415).

Mitici, M., de Pater, I., Barros, A., & Zeng, Z. (2023). Dynamic predictive maintenance for multiple components using data-driven probabilistic rul prognostics: The case of turbofan engines. *Reliability Engineering & System Safety*, *234*, 109199.

Nejjar, I., Geissmann, F., Zhao, M., Taal, C., & Fink, O. (2024). Domain adaptation via alignment of operation profile for remaining useful lifetime prediction. *Reliability Engineering & System Safety*, *242*, 109718.

Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. (2022). *Dataset shift in machine learning*. Mit Press.

Rivas, A., Delipei, G. K., & Hou, J. (2022). Predictions

of component remaining useful lifetime using bayesian neural network. *Progress in Nuclear Energy*, *146*, 104143.

Saxena, A., & Goebel, K. (2008). Turbofan engine degradation simulation data set. *NASA ames prognostics data repository*, *18*, 878–887.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, *17*, 395–416.

Wang, Y., & Zhao, X. (2024). A lightweight sensing data integrity detection method for the industrial internet of things. *IEEE Sensors Journal*, *24*(15), 25030–25040.

Wu, F., Wu, Q., Tan, Y., & Xu, X. (2024). Remaining useful life prediction based on deep learning: A survey. *Sensors*, *24*, 3454. doi: 10.3390/s24113454

Yan, P., Abdulkadir, A., Luley, P.-P., Rosenthal, M., Schatte, G. A., Grewe, B. F., & Stadelmann, T. (2024). A comprehensive survey of deep transfer learning for anomaly detection in industrial time series: Methods, applications, and directions. *IEEE Access*, *12*, 3768–3789. doi: 10.1109/access.2023.3349132

Yang, H., & Desell, T. (2022). A large-scale annotated multivariate time series aviation maintenance dataset from the ngafid. *arXiv preprint arXiv:2210.07317*.

Zhou, K., Liu, Z., Qiao, Y., Xiang, T., & Loy, C. C. (2022). Domain generalization: A survey. *IEEE transactions on pattern analysis and machine intelligence*, *45*(4), 4396–4415.

BIOGRAPHIES

Kiavash Fathi is a PhD researcher at Chair of Information and Automation Systems for Process and Material Technology at RWTH Aachen University in collaboration with the Institute of Mechatronic Systems at Zurich University of Applied Sciences (ZHAW). His research focuses on predictive maintenance, Industry 4.0, and human-centric AI, drawing on his background in mechatronics, automation, and control engineering. He has been involved in several industrial innovation projects, funded by Innosuisse[1], targeting data-driven maintenance and digital twin–based process optimization. His work emphasizes efficiency, minimal sensor reliance, and robust model lifecycle management.

[1] https://www.innosuisse.admin.ch/en

Mihaela Mitici is Assistant professor at Utrecht University, the Netherlands. Mihaela's expertise is in Operations Research with a focus on stochastic processes, decision-making under uncertainty, machine learning. Her application domains are predictive maintenance (RUL prognostics, diagnostics, maintenance planning) and mobility (operations and charging scheduling for electric vehicles).

Tobias Kleinert Prof. Dr.-Ing. Tobias Kleinert graduated in Mechanical Engineering at RWTH Aachen University and obtained PhD at Ruhr-Universität Bochum. He worked at BASF SE in various fields of industrial automation and digitalisation. Since 2020, he is head of Chair of Information and Automation Systems for Process and Material Technology at RWTH Aachen University. Focus of teaching and research are industrial data systems and flexible industrial automation and control.

Hans Wernher van de Venn Hans Wernher van de Venn received the Dr.-Ing. degree in Mechanical Engineering from RWTH Aachen University, Germany. He was Director of the Institute of Mechatronic Systems at the Zurich University of Applied Sciences (ZHAW) from 2006 to 2025. Under his leadership, the institute delivered numerous national and international research projects in robotics, automation, and mechatronics. He is the Founding President and Scientific Director of EIRAS (European Institute for Robotics, Automation & Resilient Systems). His research focuses on AI-driven industrial automation, mechatronic system design, human–robot collaboration, and applied robotics.