*Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023)*
*Edited by* Mário P. Brito, Terje Aven, Piero Baraldi, Marko Čepin and Enrico Zio
 *Published by* Research Publishing, Singapore.
doi: 10.3850/978-981-18-8071-1_P101-cd

# Constructing health indicators for systems with few failure instances using unsupervised learning

Ingeborg de Pater

*Faculty of Aerospace Engineering, Delft University of Technology, The Netherlands.*
*E-mail: i.i.depater@tudelft.nl*

Mihaela Mitici

*Faculty of Science, Utrecht University, The Netherlands.*
*E-mail: m.a.mitici@uu.nl*

Health indicators are crucial to assess the health of complex systems. In recent years, several studies have developed data-driven health indicators using supervised learning methods. However, due to preventive maintenance, there are often not enough failure instances to train a supervised learning model, i.e., the data is unlabelled with an unknown actual Remaining Useful Life (RUL). In this paper, we therefore propose an unsupervised learning model to construct a health indicator for an aircraft system. The considered system is operated under highly-varying operating conditions. We train a Convolutional Neural Network (CNN) to predict the sensor measurements from the operating conditions. We train this neural network solely with the sensor measurements of just-installed, non-degraded systems. The CNN therefore learns the normal range of the sensor measurements, given the operating conditions, for non-degraded systems only. For a degraded system, the predicted sensor measurements deviate from the actual sensor measurements. Based on the prediction errors, we construct a health indicator for the aircraft system. We apply this approach to develop a health indicator for the aircraft turbofan engines of dataset DS02 and DS06 of N-CMAPSS. The resulting health indicators have a high prognosability of 0.91 for DS02 and of 0.83 for DS06, a mean trendability of 0.86 for DS02 and of 0.87 for DS06, and a mean monotonicity of 0.31 for DS02 and of 0.33 for DS06, and can thus be used to make a reliable assessment of the system's health.

*Keywords*: Health indicator, few failure instances, unsupervised learning, varying operating conditions, high-frequency data, Convolutional Neural Network

## 1. Introduction

In the field of Prognostics and Health Management (PHM), health indicators represent the health of, or degradation in, a system. Health indicators are used in various PHM applications, such as fault detection, the identification of changes in the degradation trend and for predicting the Remaining Useful Life (RUL) of systems.

Most studies employ physics-based models to develop health indicators. For instance, the Paris law for crack propagation is often combined with Kalman/particle filters to model the health of airframe panels (Yiwei et al. (2017)), while features from the time-domain (such as root mean square) and frequency domain (such as kurtosis) are used to evaluate the health of rolling element bearings from vibration signals (Gupta and Pradhan (2017)). Unfortunately, pure physics-based models are not available for many complex systems (Chao et al. (2022)). In this paper, we therefore develop a data-driven method to construct a health indicator instead.

In recent years, the focus has thus shifted to the development of data-driven models to construct a health indicator. For instance, in Wang et al. (2008); Khelif et al. (2014), a health indicator is obtained by a linear regression of the health of a system on the sensor measurements. This approach requires many labeled data-monitoring samples, i.e., data samples for which the true RUL or the true health of the system is known. Most expensive or safety-critical systems, however, are preventively maintained or replaced (far) before failure. Moreover, assessing the true health of a system is usually expensive or even impossible (Fink et al. (2020)). Most data samples com-

ing from expensive or safety-critical systems are therefore unlabelled, i.e., the true RUL or the true health state is unknown. Also in this paper, we consider a problem where only a limited number of failure instances (and corresponding labelled data samples) are available.

An unsupervised or semi-supervised learning method is often used to create a health indicator when few failure instances are available. Specifically, a model is trained with the unlabelled data samples of non-degraded systems only to learn the normal range of sensor measurements. A health indicator is then constructed by detecting deviations from this normal range (Fink et al. (2020)). The unsupervised learning model is often an autoencoder, which only learns to reconstruct the sensor measurements of non-degraded systems. In Ye and Yu (2021); de Pater and Mitici (2023), the reconstruction errors of such an autoencoder are used to construct a health indicator, while in Zhai et al. (2021); Fu et al. (2021), the embedding of the autoencoder (i.e., the hidden state with the smallest dimension) is used to construct a health indicator instead.

In this paper, we also train an unsupervised learning model with unlabelled data samples to learn the normal range of sensor measurements. However, we consider an aircraft system that is operated under highly varying operating conditions. The normal range of the sensor measurements depends on these operating conditions. In contrast to the other papers, we therefore do not use an autoencoder. Instead, we train a Convolutional Neural Network (CNN) to predict the sensor measurements at a certain time from the operating conditions at that time. We train this CNN solely with measurement samples coming from non- or only slightly degraded systems. We then use the prediction error of the CNN to detect deviations from the normal system behaviour due to increasing degradation, to select relevant sensors for detecting degradation and to construct a health indicator for the aircraft system.

We apply this methodology to create a health indicator for the aircraft turbofan engines of dataset DS02 and DS06 in the N-CMAPSS dataset (Arias Chao et al. (2021)). The resulting health indicators have a prognosability of 0.91 for DS02 and of 0.83 for DS06, a mean trendability of 0.86 for DS02 and of 0.87 for DS06, and a mean monotonicity of 0.31 for DS02 and of 0.33 for DS06. We thus obtain good results with a relatively simple and easily explainable method. A similar approach is used in Lövberg (2021) to preprocess the sensor data of N-CMAPSS before using it as input to a neural network that predicts the RUL. However, in this paper, it is merely used as a preprocessing technique: No sensors are selected and no health indicator is created.

In the rest of this paper, we first discuss the methodology in Section 2, and then present the results for the N-CMAPSS dataset in Section 3. We then provide conclusions and suggestions for further research in Section 4.

## 2. Methodology: health indicator construction for an aircraft system with limited failure instances

### 2.1. *CNN for predicting the sensor measurements*

Let $\mathbf{Y}^{e,f} = [\mathbf{Y}_1^{e,f}, \mathbf{Y}_2^{e,f}, \dots, \mathbf{Y}_{N^{e,f}}^{e,f}]$ denote all multi-sensor measurements for an aircraft system $e$ collected during a flight $f$, with $N^{e,f}$ the number of multi-sensor measurements collected during this flight. Here, $\mathbf{Y}_i^{e,f} = [y_i^{e,f,1}, y_i^{e,f,2}, \dots, y_i^{e,f,m}]$, with $y_i^{e,f,j}$ the measurement of sensor $j$ of time-step $i$ of flight $f$ and system $e$. Last, $m$ denotes the number of sensors.

Similar, let $\mathbf{X}^{e,f} = [\mathbf{X}_1^{e,f}, \mathbf{X}_2^{e,f}, \dots, \mathbf{X}_{N^{e,f}}^{e,f}]$ be the corresponding measurements of the operating conditions of flight $f$ of system $e$. Here, $\mathbf{X}_i^{e,f} = [x_i^{e,f,1}, x_i^{e,f,2}, \dots, x_i^{e,f,o}]$, with $x_i^{e,f,j}$ the measurement of operating condition $j$ of time-step $i$ of flight $f$ of system $e$. Here, $o$ denotes the number of measured operating conditions.

We input the data samples in a CNN. All data samples therefore must have equal dimensions. In contrast, the length $N^{e,f}$ varies per flight $f$ and per system $e$. For each flight $f$ of system $e$, we therefore create $N^{e,f} - n$ data samples of length $n$ from $\mathbf{Y}^{e,f}$. Let $\tilde{\mathbf{Y}}_i^{e,f} = [\mathbf{Y}_i^{e,f}, \mathbf{Y}_{i+1}^{e,f}, \dots, \mathbf{Y}_{i+n-1}^{e,f}]$ denote the $i^{\text{th}}$ data sample containing $n$ multi-sensor measurements from flight $f$ of system $e$, with $i \in$

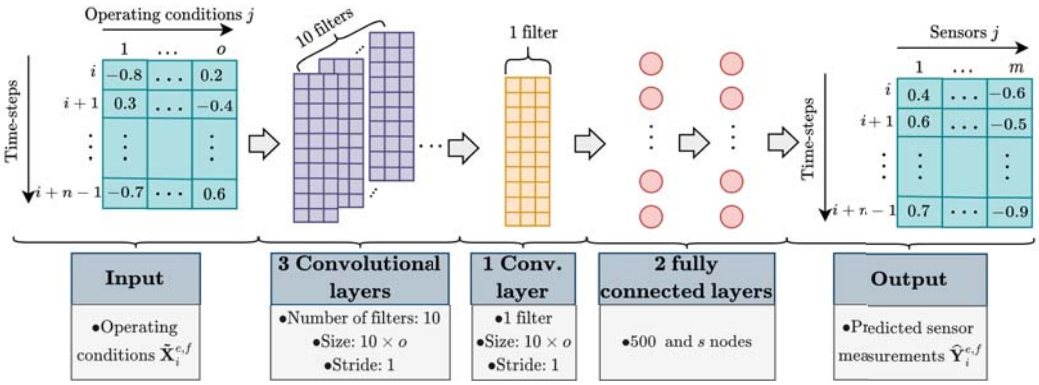



Fig. 1.: Schematic overview of the CNN.

$\{1, 2, \ldots, N^{e,f} - n + 1\}$. Similar, we create the corresponding $i^{\text{th}}$ sample with the operating conditions of flight $f$ of system $e$, denoted by $\tilde{\mathbf{X}}_i^{e,f} = [\mathbf{X}_i^{e,f}, \mathbf{X}_{i+1}^{e,f}, \ldots, \mathbf{X}_{i+n-1}^{e,f}]$. These data samples are overlapping: The multi-sensor measurements $\mathbf{Y}_g^{e,f}$ or operating conditions $\mathbf{X}_g^{e,f}$ of one time-step $g$ may be included in multiple samples.

The goal of the CNN is to predict the multi-sensor measurements $\tilde{\mathbf{Y}}_i^{e,f}$ using the operating conditions $\tilde{\mathbf{X}}_i^{e,f}$ as input. The considered CNN first consists of three convolutional layers. Each convolutional layer has ten filters of size $10 \times o$, and a stride of one. We then have a fourth convolutional layer with one filter of size $10 \times o$, that combines all ten feature maps into one single feature map. This single feature map is used as input to three fully connected layers, where the first two layers each have 500 and $s$ nodes respectively. The last layer has $n \cdot m$ nodes. This layer outputs the predicted sensor measurements $\hat{\mathbf{Y}}_i^{e,f}$. We apply after each layer, except the last layer, the ReLU activation function. For the last layer, we consider the linear activation function instead. Moreover, we initialize the weights of each layer with Kaiming normal initialization. A schematic overview of the considered CNN is in Figure 1.

The loss function of the CNN is the mean squared prediction error between the actual sensor measurements $\tilde{\mathbf{Y}}_i^{e,f}$ and the predicted sensor measurements $\hat{\mathbf{Y}}_i^{e,f}$. We regard this as an unsupervised learning approach, since we do not need any information on the RUL or health of the system to train the CNN.

### 2.2. *Sensor selection for the health indicator*

The degradation of a system influences the measurements of some sensors more than others. To construct the health indicator, we therefore only select the sensors for which the degradation is clearly present in the prediction errors over time. Specifically, let $\epsilon_i^{e,f,j}$ be the *squared* prediction error for sensor $j$, sample $i$, engine $e$ and flight $f$:

$$\epsilon_i^{e,f,j} = \sum_{g=i}^{i+n-1} \left( \hat{y}_{i,g}^{e,f,j} - y_{i,g}^{e,f,j} \right)^2, \qquad (1)$$

where $\hat{y}_{i,g}^{e,f,j}$ denotes the predicted measurement of sensor $j$ for flight $f$, system $e$ and time-step $g$, in sample $i$. Let $\mathcal{L}_f^{e,j}$ be the mean squared prediction error (MSE) for all data samples $\tilde{\mathbf{Y}}_i^{e,f}$ ($i \in \{1, 2, \ldots, N^{e,f} - n + 1\}$) for sensor $j$ and a system $e$ during a flight $f$:

$$\mathcal{L}_f^{e,j} = \frac{\sum_{i=1}^{N^{e,f}-n+1} \epsilon_i^{e,f,j}}{n \cdot (N^{e,f} - n + 1)}. \qquad (2)$$

Note that eq. (2) is more complicated than the "standard" expression for the MSE, since the data samples are overlapping. For each sensor $j$, we have the time-series of the mean error $\mathcal{L}^{e,j} = \left[\mathcal{L}_1^{e,j}, \mathcal{L}_2^{e,j}, \ldots, \mathcal{L}_{F^e}^{e,j}\right]$. Here, $F^e$ denotes the number of flights of a system $e$.

Let $E$ be the set with all training systems for which sensor measurements are available from the

moment of installation until failure. For each sensor $j$, we calculate the mean trendability $\mathcal{T}^j$ using the Spearman correlation coefficient between the time-series $\mathcal{L}^{e,j}$ and the flights $\{1, 2, \ldots, F^e\}$ for all systems $e \in E$ (Lei et al. (2018)):

$$\mathcal{T}^j = \frac{1}{|E|} \sum_{e \in E} \mathcal{T}^{e,j}, \tag{3}$$

$$\mathcal{T}^{e,j} = \frac{F^e \sum_{f=1}^{F^e} r(\mathcal{L}_f^{e,j}) f - \sum_{f=1}^{F^e} r(\mathcal{L}_f^{e,j}) \sum_{f=1}^{F^e} f}{\sqrt{F^e \sum_{f=1}^{F^e} \left(r(\mathcal{L}_f^{e,j})\right)^2 - \left(\sum_{f=1}^{F^e} r(\mathcal{L}_f^{e,j})\right)^2} \cdot \sqrt{F^e \sum_{f=1}^{F^e} f^2 - \left(\sum_{f=1}^{F^e} f\right)^2}},$$

where $r(\mathcal{L}_f^{e,j}), f \in \{1, 2, \ldots, F^e\}$ is the rank sequence of the time-series $\mathcal{L}^{e,j}$. The trendability $\mathcal{T}^{e,j}$ is between minus one and one. The correlation between the operating time and the health indicator is stronger when the trendability is closer to one or minus one. We also calculate for each sensor $j$ the prognosability $\mathcal{P}^j$ (also called consistency) over all systems $e \in E$ (Lei et al. (2018)):

$$\mathcal{P}^j = exp\left[\frac{-\text{STD}(\mathcal{L}_{F^e}^{e,j}, e \in E)}{\frac{1}{|E|}\sum_{e \in E}\left|\mathcal{L}_1^{e,j} - \mathcal{L}_{F^e}^{e,j}\right|}\right], \tag{4}$$

where $\text{STD}(\mathcal{L}_{F^e}^{e,j}, e \in E)$ denotes the standard deviation of the last values $\mathcal{L}_{F^e}^{e,j}$ of the time-series $\mathcal{L}^{e,j}$ over all $e \in E$. The prognosability is between zero and one. The health indicators of the different engines are more similar when the prognosability is closer to one.

We only select the sensors $j$ for which the mean trendability $\mathcal{T}^j$ and the prognosability $\mathcal{P}^j$ are above 0.75. Let $J$ be the set with selected sensors. We then calculate the final health indicator $h^e = [h_1^e, h_2^e, \ldots, h_{F^e}^e]$ for a system $e$, with:

$$h_i^e = \sum_{j \in J} \mathcal{L}_i^{e,j}. \tag{5}$$

## 3. Results

### 3.1. *Description of the data*

We apply the methodology in Section 2 to the aircraft turbofan engines of dataset DS02 of the N-CMAPSS dataset (Arias Chao et al. (2021)).This simulated dataset has a training set of six engines (engines 2, 5, 10, 16, 18 and 20) and a test set of three engines (engine 11, 14 and 15). For each engine in both the training and test set, all sensor measurements and operating conditions from the first flight after installation until the failure are available.

The dataset contains $o$=4 operating conditions, namely the altitude of the aircraft, the flight Mach number, the throttle-resolver angle and the total temperature at the fan inlet. Moreover, we consider the measurements of only the $m = 14$ physical sensors around different parts of the engine ($x_s$).The sensor measurements of the training engines are highly correlated with the measurements of at least one operating condition. For instance, the total pressure at the fan inlet (P2) has a correlation of 0.99 with the total temperature at the fan inlet. The total temperature at the LPT outlet (T50) has the lowest absolute correlation with the operating conditions. But even for this sensor, the highest absolute correlation, with the total temperature at the fan inlet, is still 0.54. This high correlation shows that the sensor measurements can be predicted using the operating conditions.

We select dataset DS02 for two reasons. First, dataset DS02 contains two distinct failure modes. The failure mode of training engines 2, 5 and 10 affects only the efficiency of the HPT (High Pressure Turbine), while the failure mode of the other training engines and the three test engines also affects the efficiency and flow of the LPT (Low Pressure Turbine). We use dataset DS02 to analyse the robustness of our approach to the presence of multiple failure modes in the training set. Second, all training engines and test engine 11 belong to flight class 3 (long flights), while test engine 14 belongs to flight class 1 (short flights) and test engine 15 belongs to flight class 2 (medium long flights). Since all training engines have the same flight class, we cannot incorporate this in the neural network. However, we use dataset DS02 to analyse the robustness of our approach to changing circumstances from the training to the test set.

#### 3.1.1. *Data preprocessing*

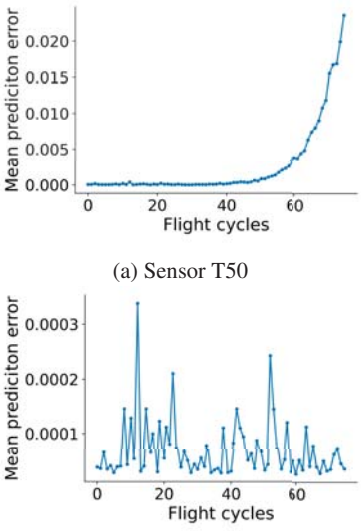
The operating conditions and sensor measurements are simulated at a high-frequency, with one measurement per sensor/operating condition per second. This gives 4311 to 18169 measurements per sensor per flight for the training engines. To reduce the training time of the CNN, we aggregate the measurements by taking the mean measurement per ten seconds.

The lifetime of each engine in the N-CMAPSS dataset is split in two parts. When the engine is just installed, the engine is in the normal degradation phase and the degradation is simulated using a linear model. After some time (16 to 17 flights for the training engines in DS02), the engine transitions to the abnormal degradation phase, and the degradation is modelled using an exponential model instead. However, we assume that it is unknown when this accelerated degradation phase starts for the training engines. Instead, we assume that the training engines are non- or only slightly degraded during the first ten flights. We thus use the first ten flights from each training engine to train the CNN. This mimics a real-life situation, in which it is challenging to determine, even in hindsight, when the degradation accelerates.

Last, we normalize the measurements for each sensor and each operating condition between minus one and one using min-max normalization. Here, the minimum and maximum measurements are calculated with the measurements of the first ten flights of all training engines.

### 3.2. *Training of the CNN*

We thus use the first ten flights of each training engine to train the CNN. Here, we consider a sample length of $n = 50$. For each training engine, we randomly select two flights (20%) of the first ten flights for validation. All samples from these validation flights are put in the validation set. The samples of the remaining flights are used for the training set. This gives 27743 training samples and 6685 validation samples. We train the CNN with the Adam optimizer with a learning rate of 0.001. We then perform a grid search to determine the number of epochs (50 or 100) and the number of nodes in the second fully connected layer $s$ (500 or 1000). The lowest validation loss is obtained with 100 epochs and $s = 500$, after 96 epochs. At this point, the mean squared prediction error $(\hat{y}_{i,g}^{e,f,j} - y_{i,g}^{e,f,j})^2$ for the validation set is only $6.0 \cdot 10^{-5}$ per single, normalized measurement of one sensor. We thus succeed very well in predicting the sensor measurements of non- or only slightly degraded engines from the operating conditions. In the rest of this study, we use the CNN with these weights of epoch 96.



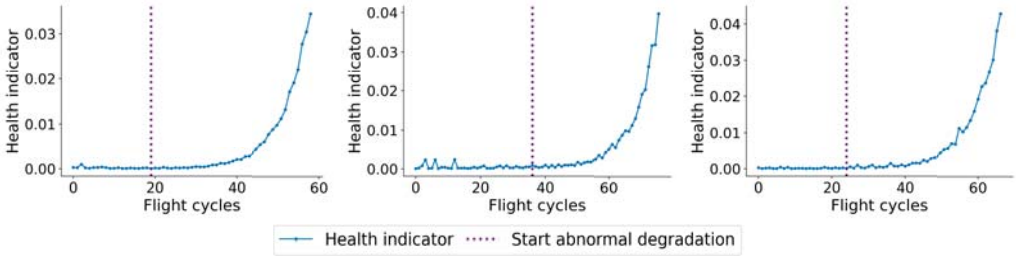

(a) Sensor T50

(b) Sensor P2

Fig. 2.: The mean prediction error $\mathcal{L}_f^{e,j}$ per flight $f$ for engine $e = 2$, sensors $j =$ T50 and $j =$ P2.

Figure 2 shows the mean prediction error $\mathcal{L}^{e,j}$ over time for training engine 2 and for sensor $j =$ P2 and $j =$ T50. For sensor T50, the mean prediction error increases over time for engine 2. This is reflected by the high Spearman trendability of 0.85. In contrast, the mean prediction error of sensor P2 only shows random fluctuations, and no trend over time. The trendability with this sensor and engine is indeed only -0.06. Including the prediction errors of sensor P2 in our final health indicator thus only introduces additional noise. This shows the importance of performing a sensor selection prior to making the health indicator.

Table 1 shows the two out of 14 sensors that are selected to make the final health indicator, i.e.,




(a) Test engine 11, DS02 (b) Test engine 14, DS02 (c) Test engine 15, DS02

Fig. 3.: Health indicator for the three test engines of DS02.

Table 1.: Table with all selected sensors $j \in J$ for the health indicator construction for DS02.

| Sensor $j$ | Description | $\mathcal{T}^j$ | $\mathcal{P}^j$ |
|---|---|---|---|
| T48 | Total temperature at HPT outlet | 0.86 | 0.91 |
| T50 | Total temperature at LPT outlet | 0.92 | 0.80 |

Table 2.: The trendability $\mathcal{T}^e$, monotonicity $\mathcal{M}^e$ and prognosability $\mathcal{P}$ for the test engines, DS02.

| Engine $e$ | Trend. $\mathcal{T}^e$ | Mono. $\mathcal{M}^e$ | Prog. $\mathcal{P}$ |
|---|---|---|---|
| 11 | 0.83 | 0.31 | - |
| 14 | 0.83 | 0.31 | - |
| 15 | 0.93 | 0.30 | - |
| Mean/all | 0.86 | 0.31 | 0.91 |

all sensors for which the mean trendability $\mathcal{T}^j$ and the prognosability $\mathcal{P}^j$ exceed the threshold of 0.75 for the training engines. The two selected sensors measure the temperature at the HPT and LPT outlet. This is in line with the fault mode of the engines.

### 3.3. *Health indicators*

Figure 3 shows the health indicator for all three test engines. The health indicators are roughly flat in the normal degradation phase, with a value between 0.000 and 0.002. After this phase, the health indicator exponentially increases, until the engines fail when the health indicator is between 0.034 (engine 11) and 0.042 (engine 15).

To quantitatively evaluate the health indicators, we calculate for each test engine $e$ the trendability $\mathcal{T}^e$ (see eq. 3) and the monotonicity $\mathcal{M}^e$ (de Pater and Mitici (2023)):

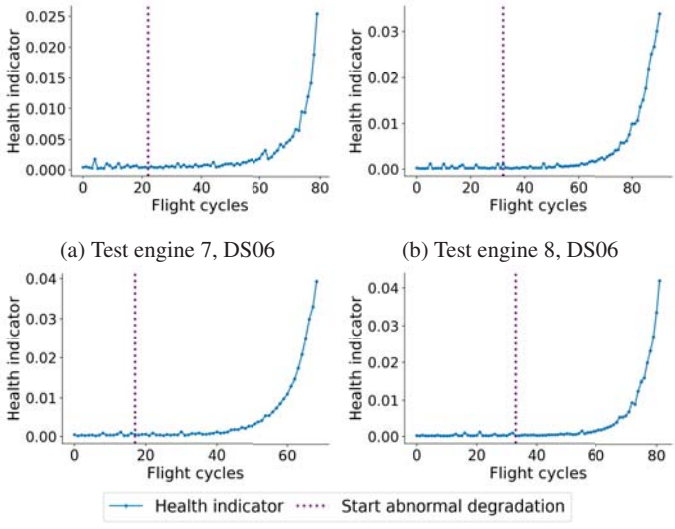
$$\mathcal{M}^e = \frac{\left|\sum_{f=1}^{F^e-1} I\left(h^e_{f+1} - h_f\right) - I\left(h^e_f - h^e_{f+1}\right)\right|}{F^e - 1},$$
$$I(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}. \quad (6)$$

Last, we calculate the prognosability $\mathcal{P}$ over all three test engines (see eq. 4).

Table 2 shows these three metrics for the three test engines. All engines have a high trendability between 0.83 and 0.93. The correlation between the operating time and health indicator is thus large, i.e., the health indicator increases as the engine degrades over time. Also the prognosability of 0.91 is high. The engines thus have a roughly equal initial and final health indicator value. Last, the monotonicity of the engines is 0.30 or 0.31, due to small fluctuations in the health indicators. The high trendability and prognosability enable the employment of these health indicators in RUL prognostic models or for fault detection.

### 3.4. *Robustness of methodology - dataset DS06 of N-CMAPSS*

To verify the robustness of our approach, we also use the proposed methodology on dataset DS06 of N-CMAPSS. This dataset consists of six training engines (labelled as 1, 2, 3, 4, 5 and 6) and 4 test engines (labelled as 7, 8, 9 and 10). These engines have the same names as some engines in DS02, but they are different engines with different sensor data. All engines in DS06 degrade according to the same failure mode, that affects the efficiency and flow of the Low Pressure Compressor (LPC)




(a) Test engine 7, DS06 (b) Test engine 8, DS06

(c) Test engine 9, DS06 (d) Test engine 10, DS06

Fig. 4.: Health indicator for the four test engines of DS06.

and the High Pressure Compressor (HPC). We select this dataset to verify if our approach also works with another failure mode. Moreover, we want to analyse if we select other sensors when the engines degrade according to another failure mode, that affects other parts of the engine.

Table 3.: Table with all selected sensors $j \in J$ for the health indicator construction for DS06.

| Sensor $j$ | Description | $\mathcal{T}^j$ | $\mathcal{P}^j$ |
|---|---|---|---|
| T30 | Total temperature at HPC outlet | 0.88 | 0.89 |
| T48 | Total temperature at HPT outlet | 0.89 | 0.87 |
| T50 | Total temperature at LPT outlet | 0.92 | 0.88 |

In contrast with dataset DS02, which has only one flight class in the training set, in dataset DS06 all three flight classes are present in both the training and test set. We therefore add the flight class of an engine as the input to the neural network. We do this by one-hot encoding the flight classes, i.e., we add three binary variables to the input, one for each flight class. Each binary variable is one if the engine belongs to that flight class, and zero otherwise. Beside including the flight classes, we do not change the data preprocessing, the considered CNN or the training process in any other way. The lowest validation loss is now obtained after 90 epochs, with a mean squared prediction error of $1.1 \cdot 10^{-4}$ per single, normalized measurement of one sensor in the validation set.

Table 4.: The trendability $\mathcal{T}^e$, monotonicity $\mathcal{M}^e$ and prognosability $\mathcal{P}$ for the test engines, DS06.

| Engine $e$ | Trend. $\mathcal{T}^e$ | Mono. $\mathcal{M}^e$ | Prog. $\mathcal{P}$ |
|---|---|---|---|
| 7 | 0.84 | 0.27 | - |
| 8 | 0.85 | 0.22 | - |
| 9 | 0.80 | 0.44 | - |
| 10 | 0.90 | 0.38 | - |
| Mean/all | 0.87 | 0.33 | 0.83 |

Table 3 shows the sensors that are selected to make the final health indicator for dataset DS06. Beside sensor T48 and T50, we now also select sensor T30. This sensor is located at the HPC of the engine, which is in line with the failure mode.

Figure 4 shows the obtained health indicators of the test engines of DS06, and Table 4 shows the corresponding metrics. Just as with dataset DS02, the health indicators have a high trendability and

prognosability, and a slightly lower monotonicity. This shows the robustness of our approach to other failure modes.

## 4. Conclusions

In this paper, we construct a health indicator for aircraft engines. We first develop a CNN that predicts the sensor measurements from the operating conditions. This CNN is trained to predict the sensor measurement from non- or slightly degraded engines only. The sensor measurements of a degraded engine deviate from the sensor measurements of a non-degraded engine (given the operating conditions). The prediction error of the CNN quantifies this deviation, and is used to create a health indicator. We apply this approach to the aircraft turbofan engines in dataset DS02 and DS06 the N-CMAPSS dataset. The resulting health indicators have a mean trendability of 0.86 for DS02 and of 0.87 for DS06, a prognosability of 0.91 for DS02 and of 0.83 for DS06, and a mean monotonicity of 0.31 for DS02 and of 0.33 for DS03.

In this paper, we implicitly assume that all different ranges of possible operating conditions and all possible fault modes are present in the training set. Future studies could therefore analyze how well our approach generalizes when the operating conditions and fault modes differ between the training and the test set. Second, the current approach requires a few failure instances to select relevant sensors for the health indicator construction. Future studies could investigate if it is possible to select relevant sensors using less, or even no, failure instances. Third, future studies may perform an extensive grid search on the hyperparameters of our approach, in order to find the best health indicators. Last, future studies could analyze if the prediction errors of the different sensors can be used to identify the failure mode of (upcoming) engine failures.

## References

Arias Chao, M., C. Kulkarni, K. Goebel, and O. Fink (2021). Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data 6*(1), 5.

Chao, M. A., C. Kulkarni, K. Goebel, and O. Fink (2022). Fusing physics-based and deep learning models for prognostics. *Reliability Engineering & System Safety 217*, 107961.

de Pater, I. and M. Mitici (2023). Developing health indicators and rul prognostics for systems with few failure instances and varying operating conditions using a lstm autoencoder. *Engineering Applications of Artificial Intelligence 117*, 105582.

Fink, O., Q. Wang, M. Svensen, P. Dersin, W.-J. Lee, and M. Ducoffe (2020). Potential, challenges and future directions for deep learning in prognostics and health management applications. *Engineering Applications of Artificial Intelligence 92*, 103678.

Fu, S., S. Zhong, L. Lin, and M. Zhao (2021). A novel time-series memory auto-encoder with sequentially updated reconstructions for remaining useful life prediction. *IEEE Transactions on Neural Networks and Learning Systems 33*(12), 7114–7125.

Gupta, P. and M. Pradhan (2017). Fault detection analysis in rolling element bearing: A review. *Materials Today: Proceedings 4*(2), 2085–2094.

Khelif, R., S. Malinowski, B. Chebel-Morello, and N. Zerhouni (2014). Rul prediction based on a new similarity-instance based approach. In *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*, pp. 2463–2468. IEEE.

Lei, Y., N. Li, L. Guo, N. Li, T. Yan, and J. Lin (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical Systems and Signal Processing 104*, 799–834.

Lövberg, A. (2021). Remaining useful life prediction of aircraft engines with variable length input sequences. In *Annual Conference of the PHM Society*, Volume 13.

Wang, T., J. Yu, D. Siegel, and J. Lee (2008). A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In *2008 international conference on prognostics and health management*, pp. 1–6. IEEE.

Ye, Z. and J. Yu (2021). Health condition monitoring of machines based on long short-term memory convolutional autoencoder. *Applied Soft Computing 107*, 107379.

Yiwei, W., G. Christian, N. Binaud, B. Christian, R. T. Haftka, et al. (2017). A cost driven predictive maintenance policy for structural airframe maintenance. *Chinese Journal of Aeronautics 30*(3), 1242–1257.

Zhai, S., B. Gehring, and G. Reinhart (2021). Enabling predictive maintenance integrated production scheduling by operation-specific health prognostics with generative deep learning. *Journal of Manufacturing Systems 61*, 830–855.