Developing health indicators and RUL prognostics for systems with few failure instances and varying operating conditions using a LSTM autoencoder

de Pater, Ingeborg; Mitici, Mihaela

DOI
10.1016/j.engappai.2022.105582

Publication date
2023

Document Version
Final published version

Published in
Engineering Applications of Artificial Intelligence

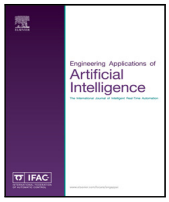
Citation (APA)
de Pater, I., & Mitici, M. (2023). Developing health indicators and RUL prognostics for systems with few failure instances and varying operating conditions using a LSTM autoencoder. *Engineering Applications of Artificial Intelligence*, *117*, Article 105582. https://doi.org/10.1016/j.engappai.2022.105582

Engineering Applications of Artificial Intelligence 117 (2023) 105582

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Developing health indicators and RUL prognostics for systems with few failure instances and varying operating conditions using a LSTM autoencoder

Ingeborg de Pater [a,*], Mihaela Mitici [b]

[a] Faculty of Aerospace Engineering, Delft University of Technology, HS 2926 Delft, The Netherlands
[b] Faculty of Science, Utrecht University, Heidelberglaan 8, 3584 CS Utrecht, The Netherlands

ARTICLE INFO

Keywords:
Remaining Useful Life prognostics
Health indicators
Unlabelled data samples
Autoencoder
Varying operating conditions
Attention

ABSTRACT

Most Remaining Useful Life (RUL) prognostics are obtained using supervised learning models trained with many labelled data samples (i.e., the true RUL is known). In aviation, however, aircraft systems are often preventively replaced before failure. There are thus very few labelled data samples available. We therefore propose a Long Short-Term Memory (LSTM) autoencoder with attention to develop health indicators for an aircraft system instead. This autoencoder is trained with unlabelled data samples (i.e., the true RUL is unknown). Since aircraft fly under various operating conditions (varying altitude, speed, etc.), these conditions are also integrated in the autoencoder. We show that the consideration of the operating conditions leads to robust health indicators and improves significantly the monotonicity, trendability and prognosability of these indicators. These health indicators are further used to predict the RUL of the aircraft system using a similarity-based matching approach. We illustrate our approach for turbofan engines. We show that the consideration of the operating conditions improves the monotonicity of the health indicators by 97%. Also, our approach leads to accurate RUL estimates with a Root Mean Square Error (RMSE) of 2.67 flights only. Moreover, a 19% reduction in the RMSE is obtained using our approach in comparison to existing supervised learning models.

1. Introduction

Complex technical systems are crucial for the safe and reliable operation of machines, vehicles, manufacturing processes, etc. The unexpected failures of such systems lead to costly unplanned downtime and potential safety risks. To limit the number of failures, systems are often replaced preventively (Ochella et al., 2022; Koutroulis et al., 2022). However, due to such preventive, early replacements, the actual failure time of the system is unobserved. This complicates the estimation of the Remaining Useful Life (RUL) for such systems.

Especially in aviation, preventive replacement of complex, safety–critical systems is common. Replacing these systems early is preferred over keeping them running for a long time and risking a failure. Consequently, the data-monitoring samples from such systems are often unlabelled, i.e., the corresponding true RUL is unknown. In the rare case when such a system does fail during operation, the failure time is observed and the data-monitoring samples coming from this system are labelled (i.e., the true RUL is known) (Berghout et al., 2020). This mix of very few labelled data samples, but many unlabelled data samples is often seen for complex aircraft systems.

Common RUL prognostics models are Convolutional Neural networks (de Pater et al., 2022; Shen et al., 2021; de Pater and Mitici, 2022) and Long Short-Term Memory (LSTM) Neural networks (Xiang et al., 2020), which directly predict the RUL. However, such supervised learning methods require the availability of many labelled data samples to train accurate prognostic models. This makes these supervised learning approaches unsuitable for complex, safety–critical aircraft systems.

Instead, accurate RUL prognostics can be obtained by first developing a health indicator using the unlabelled data samples and signal reconstruction, i.e., an autoencoder learns the normal system behaviour with the unlabelled data samples (Fink et al., 2020; Malhotra et al., 2016). This autoencoder is then used to detect deviations from the normal system behaviour that emerge due to increasing degradation (Fink et al., 2020). This approach has been considered in, for instance, Malhotra et al. (2016) and Ye and Yu (2021). In Malhotra et al. (2016), a health indicator is developed using the reconstruction error of a LSTM autoencoder and a linear regression model. Similarly, in Ye and Yu (2021), a health indicator is obtained based on the reconstruction errors of a LSTM autoencoder and a Gaussian distribution. In contrast, in Gugulothu et al. (2017), Yu et al. (2019), Fu et al. (2021) and in Zhai et al. (2021), the embeddings of a recurrent autoencoder and a conditional variational autoencoder, respectively, are used to develop a health indicator. In Liu et al. (2020), a health indicator for low-frequency time-series is developed using both the reconstruction error

* Corresponding author.
E-mail address: i.i.depater@tudelft.nl (I. de Pater).

https://doi.org/10.1016/j.engappai.2022.105582
Received 15 July 2022; Received in revised form 19 October 2022; Accepted 28 October 2022
Available online xxxx

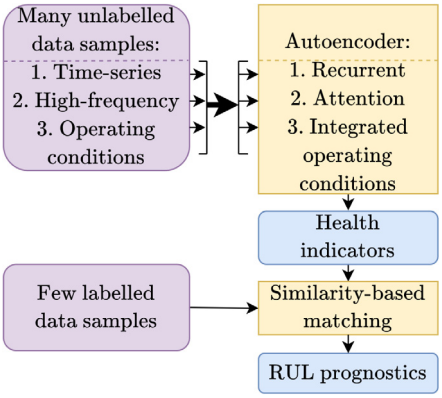


Fig. 1. Schematic overview of the consider approach.

and the embeddings of a LSTM autoencoder. With such health indicators, accurate RUL prognostics are obtained with just a few labelled data samples.

However, the autoencoders mentioned above cannot be directly applied to develop health indicators for complex aircraft systems. First, health-monitoring sensor measurements are usually recorded at a high frequency during flight. Moreover, the flight itself is several hours long. As such, the time-series of measurements of each flight is long. For the case study presented in this paper, the aircraft system performs many flights, each containing 60-303 time-steps. In contrast, existing studies consider autoencoders that use fixed-length data samples of 5 (Yu et al., 2019) to 40 (Liu et al., 2020) time-steps only.

Also, the conditions under which a system operates are expected to influence the degradation of the system (Wei et al., 2021). This is especially the case for aircraft, where the operating conditions vary due to weather conditions, flying routes, flying altitudes etc. (Wang et al., 2022). One of the major open challenges for Prognostics and Health Management (PHM), in aviation and in other application domains, is therefore to develop health indicators and RUL prognostics that are robust to varying operating conditions (Fink et al., 2020; Ochella et al., 2022; Koutroulis et al., 2022).

When proposing health indicators using an autoencoder, only one study accounts for the operating regime: a high-level cluster of similar operating conditions (Zhai et al., 2021). The operating conditions of aircraft, however, are highly-varying during each flight. For example, the altitude of an aircraft continuously changes during a flight. Considering only a few aggregated clusters as in Zhai et al. (2021) would thus lead to a loss of information (Fink et al., 2020).

To develop a health indicator for systems with high-frequency measurements and highly-varying operating conditions, we propose to use the reconstruction error of a LSTM autoencoder (LSTM-AE) with (i) attention and (ii) integrated operating conditions. LSTM neural networks are well suited to process varying-length time-series, while avoiding the vanishing gradient problem (Vasilev, 2019). However, a standard recurrent autoencoder cannot reconstruct long time-series of sensor data well: The final embedding of the autoencoder cannot contain all relevant features of a long input sample. Moreover, the final embedding contains more information about the last sensor measurements than about the first sensor measurements of the flight (Vasilev, 2019). In language-related application with long sentences, these problems have been successfully elevated by implementing attention, giving state-of-the-art results (Fink et al., 2020). Inspired by this, we also apply attention in the LSTM-AE.

To develop health indicators robust to the highly-varying operating conditions, we input the operating conditions in the autoencoder. These operating conditions are not encoded and then reconstructed, but merely used "informatively", i.e., the LSTM-AE is informed on the aircraft operating conditions solely to assist in encoding and reconstructing the sensor data samples. In contrast with Zhai et al. (2021), no information on the operating conditions is thus lost by clustering the operating conditions.

We apply our methodology to develop health indicators and RUL prognostics for the aircraft turbofan engines of the new N-CMAPSS dataset (Arias Chao et al., 2021). An overview of the considered approach is in Fig. 1. The obtained health indicators have a high monotonicity (0.38), trendability (0.95) and prognosability (0.94). We show that the monotonicity of the health indicators decreases with 97% when the operating conditions are not considered in the LSTM-AE. Also, the monotonicity decreases by 11% when no attention is applied in the LSTM-AE.

Having the health indicators, we divide the lifetime of each engine in a healthy and an unhealthy stage. Last, we estimate the RUL of the engines in the unhealthy stage with a similarity-based matching method, using the health indicators and the few available labelled data samples (Yu et al., 2020; Lyu et al., 2020; Malhotra et al., 2016). Due to the high monotonicity, trendability and prognosability of the health indicators, the overall RMSE of these RUL prognostics in the unhealthy stage is only 2.67 flights.

The main contributions of this study are:

1. We propose an unsupervised learning approach for health indicator construction and RUL prognostics for systems with very few labelled data samples, i.e., very few failures. Such systems are rarely operated until failure, but rather replaced preventively. We show that our approach outperforms existing supervised learning methods for the considered system. Specifically, the RMSE of the RUL estimates is 19% lower compared to existing supervised learning methods.
2. We develop health indicators by integrating the highly-varying operating conditions of the system in a LSTM autoencoder. This makes the health indicators robust to these highly-varying operating conditions and improves their monotonicity, trendability and prognosability significantly. Moreover, the obtained health indicators have a high trendability even when the operating conditions from the test set differ significantly from the operating conditions in the training set.
3. We use attention in the LSTM autoencoder to handle the high-frequency measurements gathered during the long flights of the considered system. We show that using attention improves the monotonicity of the health indicators by 11%. This is particularly relevant for novel technical systems whose health is monitored continuously and at a high-frequency.

The remainder of this paper is organized as follows. We first introduce the proposed methodology to construct the health indicators in Section 2. Then, we introduce the considered N-CMAPSS dataset in Section 3, and present the resulting health indicators in Section 4. Last, we introduce the similarity-based matching approach to develop RUL prognostics in Section 5, and analyse the RUL prognostics in Section 6. The conclusions are provided in Section 7.

2. Methodology — constructing health indicators with a LSTM autoencoder

In Section 2.1, we introduce the LSTM-AE with attention and integrated operating conditions. In Section 2.2, we use the reconstruction errors from this autoencoder to construct a health indicator.

2.1. LSTM-AE with local Luong attention and informative operating conditions

In this section, we introduce the Long Short-Term Memory autoencoder (LSTM-AE). Let $\mathbf{X}^{e,f} = \{\mathbf{X}_t^{e,f}, t \in \{1, 2, \dots, n^{e,f}\}\}$ be the multi-sensor measurements of an aircraft system e during a flight f,

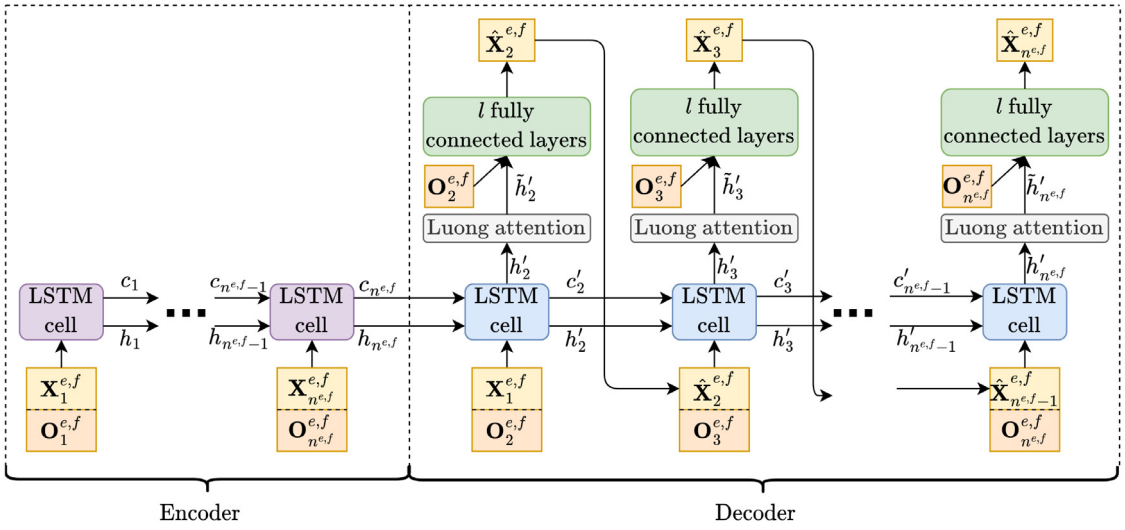


Fig. 2. A schematic overview of the considered LSTM-AE with informative operating conditions.

with $n^{e,f}$ the number of multi-sensor measurements of flight f. Here, $\mathbf{X}_t^{e,f} = [X_t^{e,f,1}, X_t^{e,f,2}, \dots, X_t^{e,f,m^s}]$ is the tth multi-sensor measurement of this flight f, with m^s the number of sensors considered. The LSTM-AE first consists of an encoder, that maps the multi-sensor measurements $\mathbf{X}^{e,f}$ to an embedding with a smaller dimension (i.e., encodes), and then a decoder, that reconstructs these measurements from this embedding. The objective of the LSTM-AE is to minimize the total absolute reconstruction error $\mathcal{L}$ of each flight:

$$\mathcal{L} = \sum_{t=2}^{n^{e,f}} \left| \hat{\mathbf{X}}_t^{e,f} - \mathbf{X}_t^{e,f} \right|, \tag{1}$$

with $\hat{\mathbf{X}}_t^{e,f}$ the reconstructed sensor measurements $\mathbf{X}_t^{e,f}$ at time-step t of system e during flight f, i.e., the output of the LSTM-AE. We train this LSTM-AE solely with the unlabelled sensor data samples from a just-installed aircraft system, i.e., when the system is still considered healthy.

Beside the multi-sensor measurements, also the operating conditions during each flight are available. Let $\mathbf{O}^{e,f} = \{\mathbf{O}_t^{e,f}, t \in \{1, 2, \dots, n^{e,f}\}\}$ be the operating conditions during flight f with system e. Here, $\mathbf{O}_t^{e,f} = [O_t^{e,f,1}, O_t^{e,f,2}, \dots, O_t^{e,f,m^o}]$ denotes the operating conditions at time-step t during this flight, and m^o is the number of operating conditions. The operating conditions are used as input for both the encoder and the decoder. But in contrast with the sensor measurements, the operating conditions are not encoded and then reconstructed, but merely used "informatively": they assist in encoding and decoding the sensor measurements. A schematic overview of the considered LSTM-AE is in Fig. 2.

2.1.1. Encoder

At each time step t during a flight f, the goal of the encoder is to encode the multi-sensor measurement $\mathbf{X}_t^{e,f}$ to the short-term state h_t, which has a smaller dimension. The encoder consists of $n^{e,f}$ LSTM-cells for this flight f (Hochreiter and Schmidhuber, 1997; Gers et al., 2000). At time step t, we consider as input to the LSTM-cell (i) the long-term state c_{t-1} and the short-term state h_{t-1} of previous time-step $t-1$, (ii) the multi-sensor measurement $\mathbf{X}_t^{e,f}$, and (iii) the operating conditions $\mathbf{O}_t^{e,f}$. Each LSTM-cell consists of 3 gates (see Fig. 3):

The forget gate determines which part of the long-term state is (partly) erased (Vasilev, 2019; Géron, 2018):

$$g_t = \sigma\left(W_g \mathbf{X}_t^{e,f} + V_g \mathbf{O}_t^{e,f} + U_g h_{t-1} + b_g\right). \tag{2}$$

Here, W_g, V_g and U_g are the weight matrices connecting $\mathbf{X}_t^{e,f}, \mathbf{O}_t^{e,f}$ and h_{t-1} to the output g_t respectively, and b_g is the bias of this layer. Also, $\sigma()$ denotes the logistic activation function.

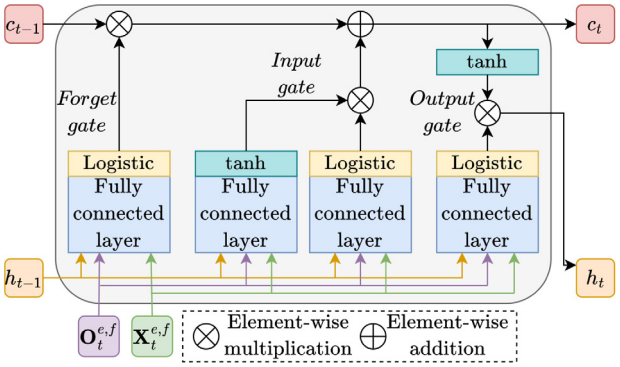


Fig. 3. A schematic overview of a LSTM-cell with informative operating conditions.

The input gate first proposes a new candidate long-term state c_t^{can} (Vasilev, 2019; Géron, 2018):

$$c_t^{\text{can}} = \tanh\left(W_c \mathbf{X}_t^{e,f} + V_c \mathbf{O}_t^{e,f} + U_c h_{t-1} + b_c\right), \tag{3}$$

where W_c, V_c and U_c are the weight matrices connecting $\mathbf{X}_t^{e,f}, \mathbf{O}_t^{e,f}$ and h_{t-1} to the output c_t^{can} respectively, and b_c is the bias of this layer. A tanh (hyperbolic tangent) activation function is considered. Next, the input gate determines which parts of the candidate long-term state are added to the new long-term state (Vasilev, 2019; Géron, 2018):

$$i_t = \sigma\left(W_i \mathbf{X}_t^{e,f} + V_i \mathbf{O}_t^{e,f} + U_i h_{t-1} + b_i\right). \tag{4}$$

Again, W_i, V_i and U_i are the weight matrices connecting $\mathbf{X}_t^{e,f}, \mathbf{O}_t^{e,f}$ and h_{t-1} to the output i_t respectively, and b_i is the bias of this layer.

Last, we update the long-term state with the output of the forget and input gate as follows (Vasilev, 2019; Géron, 2018):

$$c_t = \left(c_{t-1} \otimes g_t\right) \oplus \left(i_t \otimes c_t^{\text{can}}\right), \tag{5}$$

where $\oplus$ and $\otimes$ denote element-wise addition and element-wise multiplication, respectively.

The output gate constructs the short-term state h_t. The output gate first determines which parts of the long-term state c_t are transferred to the short-term state (Vasilev, 2019; Géron, 2018):

$$p_t = \sigma\left(W_p \mathbf{X}_t^{e,f} + V_p \mathbf{O}_t^{e,f} + U_p h_{t-1} + b_p\right). \tag{6}$$

Here, W_p, V_p and U_p are the weight matrices connecting $\mathbf{X}_t^{e,f}, \mathbf{O}_t^{e,f}$ and h_{t-1} to the output p_t respectively, and b_p is the bias of this layer. The

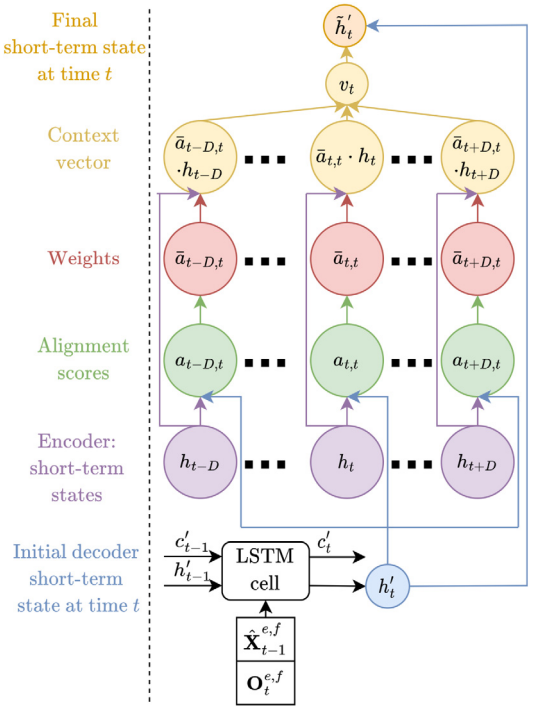


Fig. 4. Schematic overview of the local Luong attention mechanism.

new short-term state h_t is now constructed as (Vasilev, 2019; Géron, 2018):

$$h_t = p_t \otimes \tanh(c_t). \tag{7}$$

2.1.2. Decoder

The decoder reconstructs the sensor measurements $\mathbf{X}^{e,f}_t$, $t \in \{2, \dots, n^{e,f}\}$ using the short-term states from the encoder. We first obtain at each time-step t of flight f the short-term state h'_t of the decoder with a LSTM-cell. Next, local Luong attention is used to update the short-term state h'_t to the augmented short-term state $\tilde{h}'_t$. Last, we input the augmented short-term state together with the operating conditions at time-step t to a fully connected neural network. This network outputs the reconstructed sensor measurements $\hat{\mathbf{X}}^{e,f}_t$ (see Fig. 2).

Recurrent layer. The first layer of the decoder consists of $n^{e,f} - 1$ LSTM-cells. At time-step t of flight f, we consider the decoder short-term state h'_{t-1} and the decoder long-term state c'_{t-1} of time-step $t-1$ as input to the LSTM-cell. If $t = 2$, we consider the last short-term state $h_{n^{e,f}}$ and long-term state $c_{n^{e,f}}$ of the encoder as input instead. Moreover, we input the previous reconstructed sensor measurements $\hat{\mathbf{X}}^{e,f}_{t-1}$ during the testing phase. During the training phase, we use teacher forcing instead (Vasilev, 2019), i.e., we input the true sensor measurements $\mathbf{X}^{e,f}_{t-1}$. If $t = 2$, we always input the true sensor measurements from time-step $t = 1$. Last, we use the operating conditions $\mathbf{O}^{e,f}_t$ of time-step t as input, to assist in decoding the sensor measurements. The output of the LSTM-cell is the decoder short-term state h'_t and the decoder long-term state c'_t.

Local luong attention. The dimension of the last encoder short-term state $h_{n^{e,f}}$ is too small to contain all relevant features of a long flight. Moreover, the last encoder hidden state contains more information about the last sensor measurements than about the first sensor measurements of the flight (Vasilev, 2019). We therefore use local Luong attention (Luong et al., 2015) to update the decoder short-term states with all the encoder short-term states $h_t, t \in \{1, 2, \dots, n^{e,f}\}$. A schematic overview of the local Luong attention mechanism is in Fig. 4.

First, we compute how well the initial short-term state h'_t of the decoder aligns with the encoder short-term states $h_j, j \in \{t-D, t-D+1, \dots, t+D\}$. Here, D is the window size of the local attention mechanism. The alignment score $a_{j,t}$ between the decoder short-term state h'_t and the encoder short-term state h_j is determined as (Luong et al., 2015):

$$a_{j,t} = h'^T_t W_a h_j,\ j \in \{t-D, \dots, t+D\}. \tag{8}$$

Here, W_a is the weight matrix belonging to the attention mechanism, and T denotes the transpose. With these alignment scores, the weights $\bar{a}_{j,t}$ are derived with the softmax function (Luong et al., 2015):

$$\bar{a}_{j,t} = \frac{e^{a_{j,t}}}{\sum_{k=t-D}^{t+D} e^{a_{k,t}}},\ j \in \{t-D, \dots, t+D\}. \tag{9}$$

Next, the weights are used to derive the context vector v_t (Luong et al., 2015):

$$v_t = \sum_{j=t-D}^{t+D} \bar{a}_{j,t} h_j \tag{10}$$

With this context vector, the decoder short-term state is updated with one fully connected layer (Luong et al., 2015):

$$\tilde{h}'_t = \tanh\left(W_h [v_t, h'_t]\right), \tag{11}$$

with W_h a weight matrix belonging to this fully connected layer.

Fully connected layers. Last, we reconstruct the sensor measurements at time-step t with l fully connected layers. As input, we use both the augmented short-term state $\tilde{h}'_t$ and the operating conditions $\mathbf{O}^{e,f}_t$. By adding the operating conditions as input, we ensure that it is not useful for the augmented short-term states $\tilde{h}'_t$ to contain any information on the current operating conditions. We thus truly aim to encode and decode the sensor measurements only. Here, the first $l-1$ layers have the tanh activation function. The last layer has a linear activation function, and contains m^s nodes.

2.2. Constructing a health indicator with the reconstruction errors of the LSTM-AE

We train the LSTM-AE only with the unlabelled sensor data samples of just-installed aircraft systems, i.e., from systems that are considered healthy. We therefore expect that the reconstruction errors increase when a system degrades over time (Malhotra et al., 2016). The reconstruction errors of the trained LSTM-AE are thus used to derive a health indicator.

Let $\mathcal{L}^{e,s}_f$ be the mean reconstruction loss of a sensor s during flight f performed by system e:

$$\mathcal{L}^{e,s}_f = \frac{1}{n^{e,f} - 1} \sum_{t=2}^{n^{e,f}} \left| \hat{X}^{e,f,s}_t - X^{e,f,s}_t \right|, \tag{12}$$

with $\hat{X}^{e,f,s}_t$ the tth reconstructed measurement of sensor s during flight f performed by system e. Let $\mathcal{L}^{e,s} = \{\mathcal{L}^{e,s}_f, f \in \{1, 2, \dots, F^e\}\}$ be the time-series of the reconstruction loss for a sensor s that monitors system e. Then, we define $\lambda^e = \{\lambda^e_f, f \in \{1, 2, \dots, F^e\}\}$ as the health indicator of an engine e, with

$$\lambda^e_f = \sum_{s=1}^{m^s} \mathcal{L}^{e,s}_f. \tag{13}$$

3. Case study — aircraft engines

In this section, we first describe the considered data set in Section 3.1. Then, we describe the preprocessing of the data in Section 3.2, and illustrate the dataset in Section 3.3. Last, we introduce the metrics to evaluate the health indicators in Section 3.4.

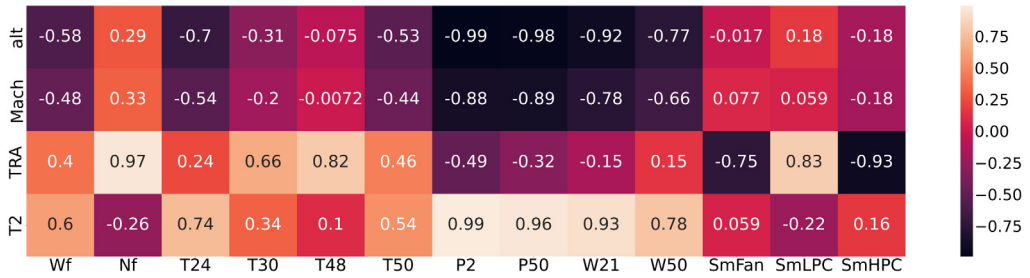


Fig. 5. Heatmap of the correlation between the sensor measurements and operating conditions — training engines of DS02, N-CMAPSS.

Table 1
Sensors selected based on the correlation. LPC — Low Pressure Combustor. HPC — High Pressure Combustor. HPT — High Pressure Turbine. LPT — Low Pressure Turbine.

Symbol	Description	Unit
Wf	Fuel flow	pps
Nf	Physical fan speed	rpm
T24	Total temperature at LPC outlet	°R
T30	Total temperature at HPC outlet	°R
T48	Total temperature at HPT outlet	°R
T50	Total temperature at LPT outlet	°R
P2	Total pressure at fan inlet	psia
P50	Total pressure at LPT outlet	psia
W21	Fan flow	pps
W50	Flow out of LPT	lbm/s
SmFan	Fan stall margin	–
SmLPC	LPC stall margin	–
SmHPC	HPC stall margin	–

3.1. Aircraft engines in the N-CMAPSS data set

We consider dataset DS02 of the *new* N-CMAPSS data set (Arias Chao et al., 2021). Here, the degradation of aircraft turbofan engines is simulated with the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) model of NASA (Arias Chao et al., 2021). There are some key differences between this new data set compared to the previous C-MAPSS data set (see Saxena and Goebel (2008)).

First, dataset DS02 of N-CMAPSS contains a limited number of engines: the training set contains only 6 engines. The test set contains 3 engines, namely engine 11, 14 and 15. For each engine e in the training and the test set of DS02, sensor measurements are available during each flight f from engine installation until engine failure (i.e., run-to-failure instances). Let F^e denote the number of flights performed by engine e. Besides the sensor measurements, N-CMAPSS also contains the operating conditions of the flights of the aircraft. A total of $m^{\text{o}} = 4$ operating conditions are available: the altitude of the aircraft (alt), the flight Mach number (Mach), the throttle-resolver angle (TRA) and the total temperature at the fan inlet (T2). Last, N-CMAPSS contains high-frequency sensor measurements, with one measurement per sensor/operating condition per second.

In the beginning of the engine's lifetime, the N-CMAPSS simulator generates sensor measurements using a linear, slow degradation model. Afterwards, an exponential, accelerated degradation model is used to simulate the sensor measurements instead (Arias Chao et al., 2021). The degradation of the engines when the linear, slow degradation model is used is still small, so we consider these sensor measurements as coming from "healthy", just-installed engines. We thus train our LSTM-AE only with the sensor measurements obtained with this slow degradation model. Let f_{a}^e be the last flight for which the sensor measurements are generated with the slow degradation model.

3.2. Data preprocessing

The training dataset consists of 6 engines, which together perform 446 flights. With 28 sensors and $4311 \leq n^{e,f} \leq 18\,169$ measurements per sensor per flight, we have a total of 147 million data points in the training data set. However, most of the measurements of the 28 sensors are highly correlated. For example, the flow out of the low pressure turbine and the flow out of the high pressure turbine have a correlation of 1.00. To reduce the computational load when training the LSTM-AE, without comprising the information contained in the sensor measurements, we select only one of two or more sensors that have a correlation of 0.99 or higher. This results in the selection of $m^{\text{s}} = 13$ sensors from the available 28 sensors (see Table 1). With this, the number of sensor measurements considered in the training dataset is reduced to 68 million.

To further reduce the computational load for training the LSTM-AE, we aggregate the sensor measurements and operating conditions per minute. In other words, we consider the mean measurement and operating condition per minute. This reduces the number of sensor measurements in the training set to 1.143 million.

Moreover, the sensor measurements and the operating conditions are normalized using min–max normalization:

$$\frac{2 \cdot (X_t^{e,f,s} - X_{\min}^s)}{X_{\max}^s - X_{\min}^s} - 1, \tag{14}$$

$$\frac{2 \cdot (O_t^{e,f,o} - O_{\min}^o)}{O_{\max}^o - O_{\min}^o} - 1, \tag{15}$$

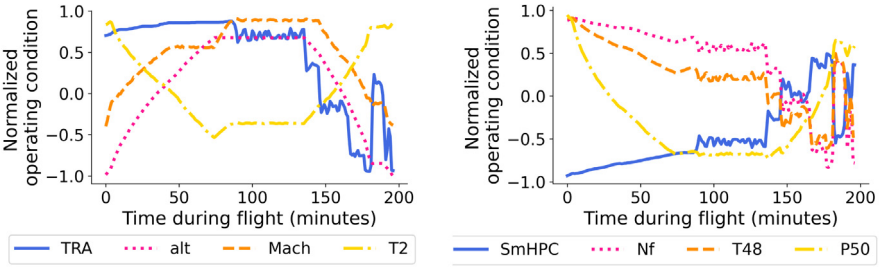
where $X_{\min}^s/O_{\min}^o$ and $X_{\max}^s/O_{\max}^o$ are the minimum and maximum measurement of sensor s/operating condition o in the training set respectively.

Last, there are only 101 flights in the training set where the sensor measurements are generated with the linear, slow degradation model. We therefore use data augmentation to increase the number of data samples for training the LSTM-AE (Chao et al., 2022). For each flight $f \leq f_{\text{a}}^e$ performed by engine e, we consider time-windows with a size of $60, 70, 80, \ldots, n^{e,f} - 10, n^{e,f}$ time-steps. These time-windows are rolled over flight f of engine e with a step size (i.e., stride) of 5 min. In this way, we subtract for each time-window with a size of $60, 70, 80, \ldots, n^{e,f} - 10, n^{e,f}$ time-steps, several time-series of multi-sensor measurements (i.e., data samples) from flight f of engine e. With this approach, 25433 data samples are obtained to train the LSTM-AE.

3.3. Illustration of N-CMAPSS data set

Fig. 6(a) shows the normalized operating conditions during the first flight of engine 2 from the training dataset. Fig. 6(b) shows the normalized sensor measurements of sensors SmHPC, Nf, T48 and P50. These figures and the correlation heatmap in Fig. 5 show that the sensor measurements are highly correlated with the operating conditions. For example, the correlation between the total pressure at the LPT outlet (P50) and the altitude of the aircraft (alt) is −0.98 (see Fig. 5).

Fig. 7 shows the mean normalized sensor measurement per flight, for all flights performed by engine 2 and for sensors SmHPC, Nf, T48 and P50. These mean sensor measurements do not exhibit a clear trend towards failure. A more extensive analysis is thus necessary to obtain a health indicator.



(a) Normalized operating conditions (b) Normalized sensor measurements

Fig. 6. Normalized operating conditions and normalized sensor measurements — flight 1, training engine 2 of DS02, N-CMAPSS.

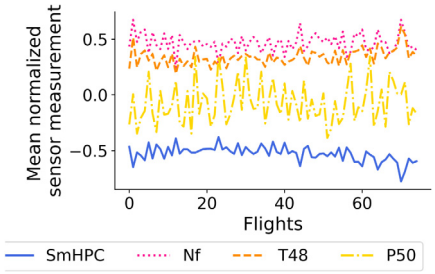


Fig. 7. Mean normalized sensor measurement per flight — training engine 2 of DS02, N-CMAPSS.

3.4. Metrics to evaluate the health indicators

We evaluate the health indicators with the monotonicity ($\mathcal{M}$), trendability ($\mathcal{T}$) and prognosability ($\mathcal{P}$) metrics as follows:

Monotonicity. We measure the monotonicity $\mathcal{M}$ of the health indicator λ^e of an engine e as follows (Liu et al., 2020):

$$\mathcal{M} = \frac{1}{F^e - 1}\left|\sum_{f=1}^{F^e-1} I(\lambda^e_{f+1} - \lambda^e_f) - I(\lambda^e_f - \lambda^e_{f+1})\right|,$$

$$I(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0. \end{cases}$$

Trendability. We consider the Spearman correlation coefficient between the health indicator λ^e and the flights $\{1, 2, \dots, F^e\}$ to measure the trendability $\mathcal{T}$ for an engine e (Lei et al., 2018; Koutroulis et al., 2022):

$$\mathcal{T} = \frac{F^e \sum_{f=1}^{F^e} r^{\lambda^e}_f f - \left(\sum_{f=1}^{F^e} r^{\lambda^e}_f\right)\left(\sum_{f=1}^{F^e} f\right)}{\sqrt{\left(F^e \sum_{f=1}^{F^e} (r^{\lambda^e}_f)^2\right) - \left(\sum_{f=1}^{F^e} r^{\lambda^e}_f\right)^2} \cdot \sqrt{\left(F^e \sum_{f=1}^{F^e} f^2\right) - \left(\sum_{f=1}^{F^e} f\right)^2}}, \tag{16}$$

where $r^{\lambda^e}_f, f \in \{1, 2, \dots, F^e\}$ is the rank sequence of the health indicator λ^e.

Prognosability. We consider the following prognosability metric $\mathcal{P}$ (also called consistency) (Lei et al., 2018; Liu et al., 2020):

$$\mathcal{P} = exp\left[\frac{-\text{STD}(\lambda^e_{F^e}, e \in E^{\text{test}})}{\frac{1}{|E^{\text{test}}|}\sum_{e \in E^{\text{test}}}\left|\lambda^e_1 - \lambda^e_{F^e}\right|}\right], \tag{17}$$

where $\text{STD}(\lambda^e_{F^e}, e \in E^{\text{test}})$ is the standard deviation of the last health indicator values $\lambda^e_{F^e}, e \in E^{\text{test}}$ (with F^e the last flight of engine e), and E^{test} is the set with the test engines of DS02.

Table 2
Considered hyperparameters of the LSTM-AE.

Hyperparameter	Value
Hyperparameters - architecture	
Hidden size h_t, c_t, h'_t and c'_t	4
Window-size D	5
Number of fully connected layers l	3
Number of nodes first $l-1$ fully connected layers	128
Hyperparameters - optimization	
Optimizer	Adam (Kingma and Ba, 2014)
Number of epochs	100
Training–Validation split	90%–10%
Initial learning rate	0.01
Decrease learning rate when no improvement in validation loss for ... epochs in a row	10
Decrease learning rate by	$\frac{1}{10}$

4. Results — health indicator for aircraft engines

In this section, we present the health indicators developed for the engines in DS02, N-CMAPSS. We first describe the hyperparameters of the LSTM-AE in Section 4.1, and then the sensor selection in Section 4.2. Next, we present the health indicators from the LSTM-AE in Section 4.3. Last, we compare our approach with other methods in Section 4.4.

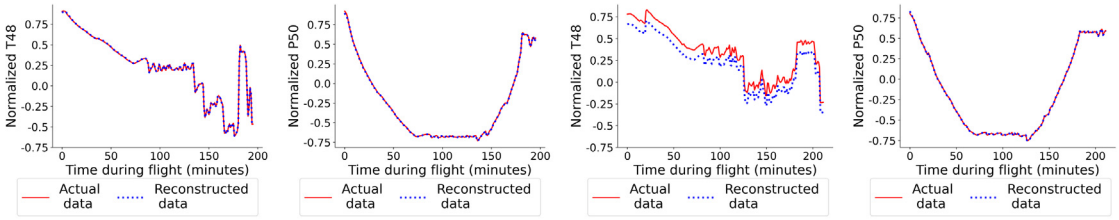
4.1. Hyperparameters of LSTM-AE

Table 2 shows the considered hyperparameters for the LSTM-AE. The architecture is derived using a grid search. After training, we select the weights belonging to the lowest validation loss. Using a computer with an Intel Core i7 processor (4 CPU cores) and 8Gb RAM, it took on average 4.04 min to train the LSTM-AE for one epoch.

4.2. Sensor selection for constructing a health indicator

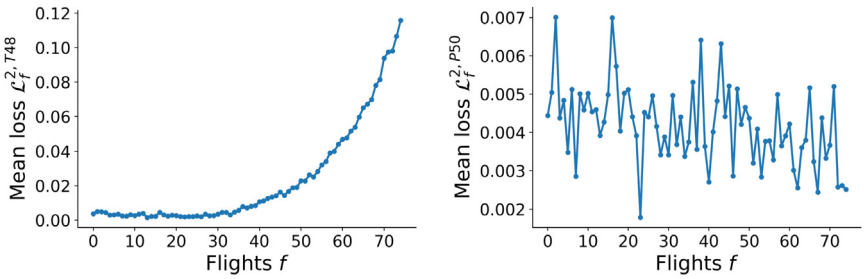
Fig. 8 shows the reconstructed measurements of sensors T48 and P50 for the first and the last flight of training engine 2. During the first flight, the reconstructed measurements are very close to the actual measurements for both sensors. In contrast, the reconstructed measurements of sensor T48 deviate considerably from the actual measurements during the last flight of engine 2 (Fig. 8(c)). This is expected, since the engine is severely degraded just before failure, while we train the LSTM-AE with sensor measurements from slightly-degraded engines only. This does not hold, however, for all sensors: the reconstructed measurements of sensor P50 are still very close to the actual sensor measurements (Fig. 8(d)).

The different trends towards the time of failure of sensors T48 and P50 are also shown in Fig. 9. The reconstruction loss of sensor



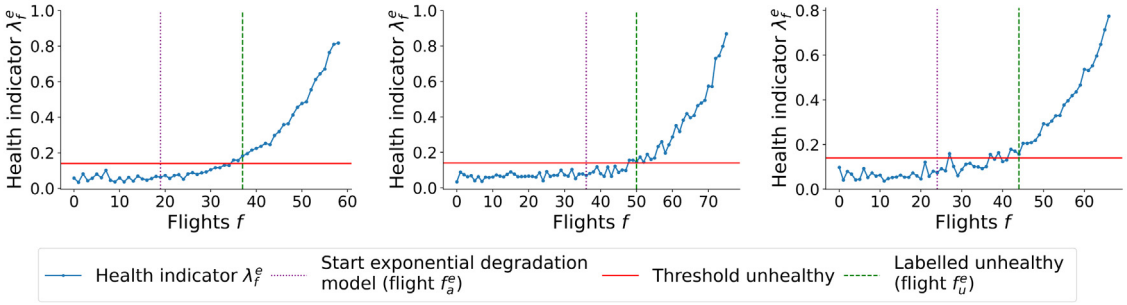
(a) First flight - sensor T48 (b) First flight - sensor P50 (c) Last flight - sensor T48 (d) Last flight - sensor P50

Fig. 8. Actual and reconstructed sensor measurements with the LSTM-AE — first and last flight, training engine 2 of DS02, N-CMAPSS.



(a) Sensor T48 ($\mathcal{C}^{\mathrm{T48}}$ - 0.95) (b) Sensor P50 ($\mathcal{C}^{\mathrm{P50}} = 0.01$)

Fig. 9. Mean loss $\mathcal{L}_f^{e,s}$ per flight with the LSTM-AE — training engine 2 of DS02, N-CMAPSS.



(a) Engine 11 (b) Engine 14 (c) Engine 15

Fig. 10. Health indicator with the LSTM-AE — test engines 11, 14 and 15 of DS02, N-CMAPSS.

T48 monotonically increases towards failure, while the reconstruction loss of sensor P50 resembles random noise. Let $C^{e,s}$ be the Spearman correlation coefficient (see Eq. (16)) between the reconstruction loss $\mathcal{L}^{e,s}$ of sensor s monitoring engine e and the flights f, i.e., the operating time. Let C^s be the mean over the Spearman correlation coefficients $C^{e,s}$ for sensor s, where the mean is taken over all training engines e. This mean correlation is close to 1 for sensors for which the loss clearly increases towards failure. In contrast, it is close to 0 for sensors for which the loss shows no trend towards failure. To construct a health indicator, we include in Eq. (13) only those sensors for which the mean Spearman correlation C^s between the reconstruction loss and the flights is 0.5 or larger. In this way, we do not construct the health indicator with sensors that are very weakly correlated with the time to failure, such as sensor P50: These sensors add little to no information on the degradation to the health indicator.

4.3. Health indicators of the test engines

Fig. 10 shows the obtained health indicators, and Table 3 shows the sensors selected to construct the health indicators and the health indicator metrics. The three test engines fail when the health indicator equals roughly 0.8/0.9, which is reflected by the high prognosability of 0.94. The increasing trend of the health indicators towards failure is reflected by the mean trendability of 0.95. Some small noise is visible in the health indicators, which is reflected by the mean monotonicity is 0.38.

The operating conditions for test engine 11 are similar to the operating conditions of the engines in the training set, while the operating conditions of test engines 14 and 15 are different than in the training set (Arias Chao et al., 2021). The monotonicity for engine 14 and 15 is indeed lower than for engine 11. However, the trendability is still high for all three test engines. Our approach thus achieves a high trendability and prognosability even when the operating conditions are different than in the training set.

4.4. Comparison with other autoencoders

We compare the health indicators from the proposed approach with the health indicators from several other methodologies: With other recurrent autoencoders, with the LSTM-AE without attention or operating conditions, and with standard, non-recurrent autoencoders. The results for these other methodologies are in Table 3 as well.

Table 3
Evaluation of the health indicators for various autoencoders — test engines 11, 14 and 15 of DS02, N-CMAPSS. $\mathcal{M}$ — Monotonicity. $\mathcal{T}$ — trendability. $\mathcal{P}$ — prognosability. The best results are denoted in bold.

		Engine 11		Engine 14		Engine 15		Mean		
	Selected sensors	$\mathcal{M}$	$\mathcal{T}$	$\mathcal{M}$	$\mathcal{T}$	$\mathcal{M}$	$\mathcal{T}$	$\mathcal{M}$	$\mathcal{T}$	$\mathcal{P}$
Proposed method										
LSTM-AE	W50, SmFan, SmLPC, SmHPC, Wf, T24, T30, T48, T50	0.48	**0.97**	0.31	**0.91**	**0.36**	**0.97**	**0.38**	**0.95**	0.94
Proposed method with other recurrent autoencoders										
GRU-AE	W50, SmFan, SmLPC, SmHPC, Wf, T24, T30, T48, T50	0.21	0.94	0.23	0.79	0.12	0.93	0.18	0.89	0.94
BiGRU-AE	W50, SmLPC, SmHPC, T48, T50	0.24	0.88	0.23	0.84	0.03	0.79	0.17	0.84	**0.95**
BiLSTM-AE	W50, SmFan, SmLPC, SmHPC, Wf, T24, T30, T48, T50	**0.52**	**0.97**	**0.33**	0.90	0.30	**0.97**	**0.38**	0.94	0.94
Proposed method without operating conditions (no o.c.) or without attention (no att.)										
LSTM-AE-no o.c.	SmLPC	0.00	0.65	0.01	0.53	0.03	0.39	0.01	0.52	0.67
LSTM-AE-no att.	W50, SmFan, SmLPC, SmHPC, Wf, T24, T30, T48, T50	0.48	**0.97**	0.23	0.88	0.30	**0.97**	0.34	0.94	0.94
Other non-recurrent autoencoders										
1D-CAE	SmHPC	0.10	0.55	0.01	0.54	0.09	0.28	0.07	0.45	0.77
FAE	W21, W50, SmFan, SmLPC, SmHPC, Nf, T24, T30, T48, T50, P50	0.38	0.77	0.12	0.52	0.18	0.82	0.23	0.70	0.89

Other recurrent autoencoders. We compare the obtained health indicators with the health indicators from four other recurrent autoencoders: the Gated Recurrent Unit autoencoder (GRU-AE), the bidirectional GRU-AE (BiGRU-AE) and the bidirectional LSTM-AE (BiLSTM-AE). For each recurrent autoencoder, we also implement local Luong attention and integrate the operating conditions.

For the BiLSTM-AE, we consider a bidirectional LSTM encoder (Vasilev, 2019, Chapter 8). However, in the decoder, the reconstructed sensor measurements of time-step t are used as input in the LSTM-cell at time-step $t+1$. We therefore cannot consider a bidirectional decoder as well. For the GRU-AE, we replace each LSTM-cell in the autoencoder by a GRU-cell (Vasilev, 2019, Chapter 7). For the BiGRU-AE, we replace each LSTM-cell in the BiLSTM-AE with a GRU-cell.

Table 3 shows the monotonicity, trendability and prognosability for the health indicators of the different recurrent autoencoders. With the (Bi)GRU-AE, the monotonicity and the trendability are considerably lower than with the LSTM-AE. The prognosability with the BiGRU-AE is 0.95, slightly higher than the prognosability of 0.94 with the LSTM-AE.

The monotonicity and the prognosability are the same with the LSTM-AE and the BiLSTM-AE. With the LSTM-AE, however, the mean trendability of 0.95 is slightly higher than the mean trendability of 0.94 for the BiLSTM-AE.

The same subset of sensors is selected to create the health indicators for the GRU-AE, the LSTM-AE and the BiLSTM-AE. For the BiGRU-AE, however, less sensors are selected to create the health indicators: Here, sensor SmFan, Wf, T24, and T30 are not selected.

No operating conditions or no attention. We also analyse the health indicators when the operating conditions are *not* incorporated in the LSTM-AE, i.e., when the operating conditions $\mathbf{O}^{e,f}$ are completely removed from the autoencoder. When not considering the operating conditions, no sensor had a Spearman trendability for the training engines of 0.5 or higher. Instead, we only include sensor SmLPC, with the highest trendability of all sensors. Without incorporating the operating conditions, the monotonicity, trendability and prognosability of the health indicators decrease considerably (see Table 3).

Table 3 also shows the metrics when we do incorporate the operating conditions, but when no attention is used. The prognosability is the same with and without attention. The monotonicity, however, is lower when not using attention (0.34 instead of 0.38). Also the trendability is slightly lower (0.94 instead of 0.95).

Other non-recurrent autoencoders. Last, we compare our method with two standard, non-recurrent autoencoders, namely a one-dimensional Convolutional autoencoder (1D-CAE) and a fully connected autoencoder (FAE). Both the sensor measurements and the operating conditions are selected as input, though only the sensor measurements are reconstructed. To construct fixed-length input samples, we consider a sliding time-window with a fixed size of 16 time-steps and a stride of 1. To create the health indicator value for a flight f of an engine e, we use the mean reconstruction loss $\mathcal{L}_f^{e,s}$ over all time-windows of size 16 and stride 1 of this flight f.

The one-dimensional convolutional encoder consists of two blocks, each with two convolutional layers and one max pooling layer with a pooling size of 2. The filters of the convolutional layer have size 4 and a stride of 1. The first three convolutional layers have 8 channels, while the last convolutional layer has only 1 channel. The convolutional decoder consists of the same structure, but instead of pooling layers we use interpolating layers. We consider zero-padding for all convolutional layers. Moreover, all layers use the ReLU activation function, except the last layer of the decoder, which uses the linear activation function.

The encoder of the FAE consists of two fully connected layers. The number of neurons is halved for each subsequent fully connected layer. The decoder consists of the same structure, only here the number of neurons is doubled in each fully connected layer. Each layer applies the ReLU activation function, except the last layer of the decoder, which uses the linear activation function.

Table 3 shows the results for the 1D-CAE and the FAE. For the 1D-CAE, no sensor had a Spearman trendability for the training engines of 0.5 or higher. Instead, we only include sensor SmHPC, which has the highest trendability of all sensors. The trendability and prognosability are considerably higher when considering a recurrent autoencoder instead of the 1D-CAE or the FAE. This shows the added value of processing the time-series of sensor measurements with a recurrent autoencoder.

5. Methodology - Online RUL prognostics using similarity-based matching

In this section, we show how the health indicators developed in Section 4 are used for health state division (Section 5.1) and to obtain RUL prognostics (Section 5.2).

5.1. Health state division using Chebyshev's inequality

Before we predict the RUL, we first diagnose an engine as healthy or unhealthy. This is called health state division or diagnostics (Lei et al., 2018). An engine is diagnosed as unhealthy once its health indicator crosses a threshold η times in a row. This threshold is determined using Chebyshev's inequality (Singh et al., 2020; Kong and Yang, 2019). For our application, this inequality states that:

$$P(|\lambda_f^e - \mu| \geq k\sigma) \leq \frac{1}{k^2}, \tag{18}$$

where $k > 0$, $P(\cdot)$ denotes the probability, and μ is the mean and σ is the standard deviation of the health indicator values λ_f^e of the

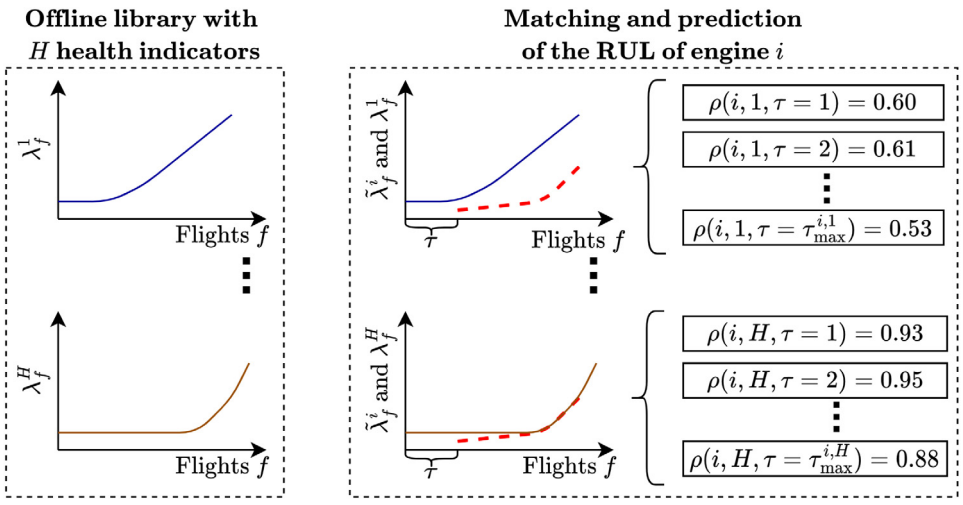
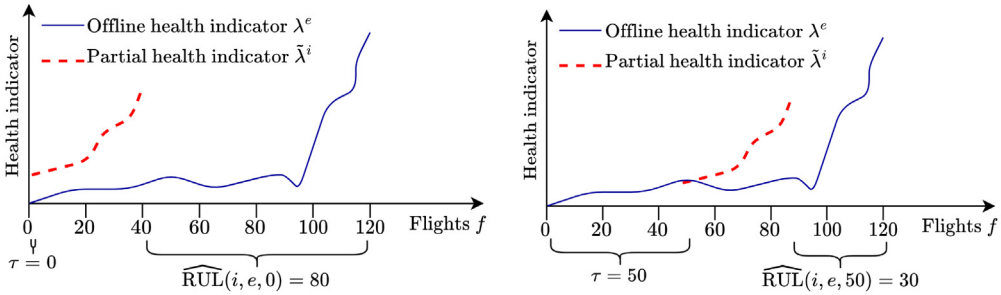


Fig. 11. Illustration of the similarity-based matching method to predict the RUL of engine i, with H health-indicators from the training set in the library.



(a) Matching procedure with *no* time-lag τ.

(b) Matching procedure with a time-lag of $\tau = 50$.

Fig. 12. The iterative process of matching the partial health indicator $\tilde{\lambda}^i$ with the offline health indicator λ^e, with different values for the time-lag τ.

training engines e and flights f for which the sensor measurements are simulated using the slow, linear degradation model (i.e., $f \leq f^e_a$). Thus, an engine is diagnosed as unhealthy as soon as the threshold $\mu + k\sigma$ is exceeded by the health indicator λ^e_f η times in a row. The probability that this occurs while the sensor measurements are generated using the slow, linear degradation model is less than $(\frac{1}{k^2})^\eta$. Let f^e_u be the flight during which an engine e is diagnosed as unhealthy. We start with predicting the RUL from flight f^e_u onwards.

5.2. Similarity-based matching method for RUL prognostics

Once an engine is diagnosed as unhealthy, we estimate its RUL after each flight using a similarity-based health indicator matching approach (Yu et al., 2020; Malhotra et al., 2016). These are online RUL prognostics, since the RUL prognostics are updated every time more sensor measurements become available.

Let $\tilde{\lambda}^i = \{\lambda^i_f, f \in \{\tilde{f} - M, \dots, \tilde{f}\}\}$ denote a partial health indicator available for engine i, using the sensor measurements available up to a flight $\tilde{f} \geq f^e_u$. Here, M is the fixed length of the partial health indicators. Our aim is to predict the RUL of engine i at flight $\tilde{f}$. For this, we consider a library with for each training engine e the offline health indicator λ^e. To predict the RUL, we match the partial health indicator $\tilde{\lambda}^i$ with all the offline health indicators λ^e in the library. A schematic overview of the matching procedure is in Fig. 11.

When matching, we determine the similarity between the partial health indicator $\tilde{\lambda}^i$ and the offline indicators λ^e in the library as the average Euclidean distance between these indices. To maximize the similarity between $\tilde{\lambda}^i$ and λ^e, i.e., to identify the best matches between $\tilde{\lambda}^i$ and λ^e, $\tilde{\lambda}^i$ is shifted along λ^e in the positive time-direction for τ flights.

Fig. 12 shows an example of a matching between the partial health indicator $\tilde{\lambda}^i$ and a health indicator λ^e from the library. In this example, the partial health indicator is $M = 40$ flights long, while the offline health indicator λ^e is 120 flights long. When $\tau = 0$, the Euclidean distance is based on the first 40 values of both health indicators, so between $\tilde{\lambda}^i$ and $\{\lambda^e_f, f \in \{1, 2, \dots, 40\}\}$. However, the similarity between these two health indicators is very small (see Fig. 12(a)). In Fig. 12(b), we therefore shift $\tilde{\lambda}^i$ forward with $\tau = 50$ flights. Now, the Euclidean distance between $\tilde{\lambda}^i$ and $\{\lambda^e_f, f \in \{51, \dots, 90\}\}$ decreases. Let $\tau^{i,e}_{\max}$, $\tau^{i,e}_{\max} = |\lambda^e| - M$ flights, denote the maximum number of flights $\tilde{\lambda}^i$ can be shifted forward when matching with λ^e, with $|\lambda^e|$ the length of the offline health indicator.

Given a time-lag τ, the average Euclidean distance $d(\tilde{\lambda}^i, \lambda^e, \tau)$ between λ^e and $\tilde{\lambda}^i$ is:

$$d(\tilde{\lambda}^i, \lambda^e, \tau) = \frac{1}{M}\sqrt{\sum_{f=1}^{M}\left(\lambda^e_{f+\tau} - \tilde{\lambda}^i_f\right)^2}, \tag{19}$$

with a corresponding preliminary RUL prognostic of (see also Fig. 12):

$$\widehat{\text{RUL}}(i, e, \tau) = |\lambda^e| - M - \tau, \tag{20}$$

and a similarity score of (Yu et al., 2020; Malhotra et al., 2016):

$$\rho(i, e, \tau) = exp(-d(\tilde{\lambda}^i, \lambda^e, \tau)/\gamma), \tag{21}$$

where $\gamma > 0$ is a parameter that influences the scaling of the score with respect to the Euclidean distance. The score $\rho(i, e, \tau)$ is higher when $\tilde{\lambda}^i$ and λ^e are more similar, given time-lag τ.

Let $\tilde{\rho}$ denote the highest similarity score of engine i, obtained across all training engines e in the library and all time-lags τ (see also Fig. 11),

Table 4
Health state division: Flight f_a^e, after which the sensor measurements are generated using the exponential degradation model, and flight f_u^e, during which the engine is diagnosed as unhealthy — test engines 11, 14 and 15 of DS02, N-CMAPSS. The best results are denoted in bold.

	Engine		
Method	11	14	15
Flight f_a^e	18	35	23
Proposed method			
LSTM-AE f_u^e	**30**	**36**	**32**
Proposed method with other recurrent autoencoders			
GRU-AE f_u^e	44	53	48
BiGRU-AE f_u^e	46	66	54
BiLSTM-AE f_u^e	**30**	**36**	**32**
Proposed method without attention (no att.)			
LSTM-AE- no att. f_u^e	**30**	**36**	37

i.e.,:

$$\tilde{\rho} = \max_{\substack{e \in E^{\text{train}}, \\ \tau \in \{0,1,\dots,\tau_{\max}^{i,e}\}}} \{\rho(i,e,\tau)\}, \tag{22}$$

where E^{train} denotes the set with all training engines. To estimate the RUL of engine i, we include all preliminary RUL prognostics $\widehat{\text{RUL}}(i,e,\tau)$ for which the score $\rho(i,e,\tau)$ is high enough, i.e., when (Malhotra et al., 2016):

$$\rho(i,e,\tau) \geq \alpha \cdot \tilde{\rho}, \tag{23}$$

with $\alpha \in [0,1]$. Let Π^i be the set of all combinations (e,τ) of training engines e and time lags τ, such that $\rho(i,e,\tau) \geq \alpha \cdot \tilde{\rho}$. Then the weight of RUL prediction $\widehat{\text{RUL}}(i,e,\tau)$, $(e,\tau) \in \Pi^i$, is:

$$p(i,e,\tau) = \frac{\rho(i,e,\tau)}{\sum_{(\epsilon,T)\in\Pi^i} \rho(i,\epsilon,T)}. \tag{24}$$

Finally, the predicted RUL RUL^i of engine i is (Yu et al., 2020):

$$\text{RUL}^i = \sum_{(e,\tau)\in\Pi^i} p(i,e,\tau) \cdot \widehat{\text{RUL}}(i,e,\tau). \tag{25}$$

6. Results - Online RUL prognostics for aircraft engines

In this section, we present the RUL prognostics for the test engines of dataset DS02. First, we analyse the health state division and the RUL prognostics in Section 6.1. Then, we compare our results with the results of the other autoencoders in Section 6.2. Moreover, we compare our RUL prognostics with the RUL prognostics of common supervised learning models that directly predict the RUL in Section 6.3. Last, we analyse the RUL prognostics for a decreasing number of offline health indicators in the library in Section 6.4.

6.1. Health state division and RUL prognostics

Table 4 shows the flight f_u^e during which the test engines of DS02 are diagnosed as unhealthy. Each test engine is labelled as unhealthy between 29 to 40 flights before failure (see also Fig. 10). This is 1 to 12 flights after the last flight f_a^e during which the sensor measurements are generated using the linear degradation model.

Fig. 13 shows the RUL predictions for the test engines of DS02. These RUL predictions are generated as soon as the engine is diagnosed as unhealthy, and then updated after each flight. The RUL of engine 11 is slightly overestimated when this engine is diagnosed as unhealthy, with a prediction error of −6 flights. In contrast, the RUL for engine 14 is slightly underestimated (a prediction error of 6 flights) after it is diagnosed as unhealthy. However, the RUL predictions of all test engines quickly converge to the true RUL as the engines approach their failure time. Table 5 shows the Root Mean Square Error (RMSE) and the

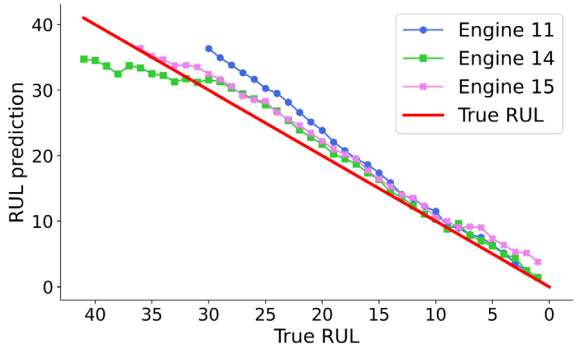


Fig. 13. RUL predictions with the LSTM-AE — test engines 11, 14 and 15 of DS02, N-CMAPPS. The first RUL prediction is made when the engine is declared unhealthy, and is updated after every flight.

Table 5
Evaluation of the RUL predictions for our proposed approach (LSTM-AE) versus other approaches — test engines 11, 14 and 15 of DS02, N-CMAPSS.

	Engine 11		Engine 14		Engine 15		All	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Proposed method								
LSTM-AE	2.89	3.50	1.89	2.41	1.95	2.12	2.18	2.67
Proposed method with other recurrent autoencoders								
GRU-AE	1.06	1.22	3.76	4.27	0.98	1.24	2.12	3.87
BiGRU-AE	1.24	1.72	3.02	3.45	1.73	2.80	1.91	2.36
BiLSTM-AE	2.72	3.41	2.42	3.02	2.77	2.96	2.62	3.12
Proposed method without attention (no att.)								
LSTM-AE-no att.	2.76	3.50	2.49	3.31	2.09	2.28	2.45	3.10

Mean Absolute Error (MAE) with these RUL predictions. The results show that RUL is well estimated for all three engines, with a RMSE between 2.12 and 3.50 flights only.

The hyperparameters of the health state division and the similarity-based matching, used to obtain these RUL results, are derived using a grid search with leave-one-out cross validation in the training set (Bishop, 2006). Here, the goal is to minimize the RMSE for the training engines. For the health state division, we obtain $\eta = 3$ and $k = 5$ (see Section 5.1). For the similarity-based matching method, we obtain $M = 10$, $\lambda = 0.01$ and $\alpha = 0.7$ (see Section 5.2).

6.2. Comparison with the RUL prognostics of other autoencoders

We also analyse the health state division and the RUL predictions with the health indicators of the best autoencoders of Section 4.4: Specifically, we consider the other recurrent autoencoders, and the LSTM-AE without attention.

The test engines are diagnosed as unhealthy during the same flights for the BiLSTM-AE and the LSTM-AE (see Table 4). Moreover, test engine 11 and 14 are diagnosed as unhealthy during the same flights for the LSTM-AE with and without attention. However, test engine 15 is diagnosed as unhealthy 5 flights later when no attention is used. For the BiGRU-AE and the GRU-AE, the engines are diagnosed as unhealthy after a later flight: For the GRU-AE, the engines are labelled as unhealthy 15 to 23 flights before failure, while for the BiGRU-AE, the engines are labelled as unhealthy only 10 to 13 flights before failure. This late diagnosis as unhealthy is expected, given the relatively low monotonicity and trendability of the (Bi)GRU-AE. It is, however, preferable if an engine is diagnosed as unhealthy far before failure, provided that the degradation in the engines is large enough to make accurate RUL predictions.

Table 5 also shows the MAE and the RMSE of the RUL prognostics for the other considered autoencoders. For all four considered autoencoders, we find that $\eta = 3$, $k = 5$ and $\lambda = 0.01$. The fixed length equals

Table 6
RMSE for the test engines 11,14 and 15 of DS02, N-CMAPSS, with various methodologies. The best results are denoted in bold.

	Engine			
	11	14	15	All
Proposed methodology				
LSTM-AE	**3.50**	**2.41**	**2.12**	**2.67**
Supervised learning neural networks				
1D-CNN	4.09	5.07	2.84	4.16
LSTM-NN	4.24	3.32	2.22	3.31

$M = 10$ for the GRU-AE, and $M = 5$ for the other autoencoders. The parameter α is 0.1 for the BiGRU-AE, 0.2 for the LSTM-AE without attention, 0.4 for the BiLSTM-AE and 0.5 for the GRU-AE.

The RUL predictions with the LSTM-AE have a lower overall RMSE and MAE then the RUL predictions with the BiLSTM-AE. This is as expected, since the health indicators of the LSTM-AE have a slightly higher trendability.

We cannot directly compare the RUL predictions of the (Bi)GRU-AE and the LSTM-AE: the engines are diagnosed as unhealthy much closer to failure when considering the (Bi)GRU-AE. In general, we expect that the RUL predictions improve when an engine degrades over time. We thus expect that the RMSE is lower when an engine is diagnosed as unhealthy later. Nevertheless, the overall RMSE of the LSTM-AE (2.67 flights) is better than the overall RMSE of the GRU-AE (3.87 flights) and only slightly worse than the overall RMSE of the BiGRU-AE (2.36 flights).

For each test engine, the RMSE with the LSTM-AE without attention is larger than or equal to the RMSE with the LSTM-AE with attention. This also holds for test engine 15, even though engine 15 is diagnosed as unhealthy 5 flights later when not using attention. Moreover, the overall RMSE equals 3.10 flights without attention, while it only equals 2.67 flights with attention. This shows the benefits of incorporating attention in the autoencoder.

6.3. Comparison with other, supervised learning methods

Last, we compare our results with the results of neural networks that directly output a RUL prediction, i.e., supervised learning methods. Here, we train two benchmark neural networks to directly predict the RUL: the one-dimensional convolutional neural network (1D-CNN) and the LSTM neural network (LSTM-NN). These two neural networks are also used as benchmark in Chao et al. (2022), and we thus use the same architecture and hyperparameters as in Chao et al. (2022). However, to allow for a fair comparison, we use the same sensors that are used as input to our approach (see Table 1) as input to the benchmark neural networks.

There is no straightforward method for health state division when using a supervised learning method. However, RUL predictions usually improve when the true RUL becomes smaller. The comparison of the RUL prognostics would thus not be fair if we do not use any health state division for the supervised learning methods. Instead, we apply the health state division of the proposed methodology also to the benchmark neural networks. For example, engine 11 is diagnosed as unhealthy at flight 30 with the proposed approach. We thus also predict the RUL of engine 11 with the benchmark neural networks from flight 30 onward.

Table 6 shows the RMSE of the RUL predictions with the various methodologies. The RMSE of the RUL prognostics is lowest for all test engines when considering our proposed approach. This shows that our approach works well for the considered data set with limited failure instances, compared to a supervised learning method.

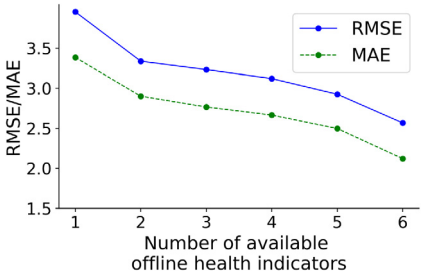


Fig. 14. Learning curve of the RMSE and MAE for the LSTM-AE — test engines of DS02, N-CMAPSS.

Table 7
Overview of the considered libraries. Here, E^{train} is the set with all training engines.

Size of library	# of libraries	Set of offline libraries $(\cdot)$
1	6	$\{(e_1) : e_1 \in E^{\text{train}}\}$
2	15	$\{(e_1, e_2) : e_1 \in E^{\text{train}}, e_2 \in E^{\text{train}} \setminus \{e_1\}\}$
3	20	$\{(e_1, e_2, e_3) : e_1 \in E^{\text{train}}, e_2 \in E^{\text{train}} \setminus \{e_1\}, e_3 \in E^{\text{train}} \setminus \{e_1, e_2\}\}$
4	15	$\{(e_1, e_2, e_3, e_4) : e_1 \in E^{\text{train}}, e_i \in E^{\text{train}} \setminus \{e_j, j = 1, 2, \dots, i-1\}, i = 2, 3, 4\}$
5	6	$\{(e_1, e_2, e_3, e_4, e_5) : e_1 \in E^{\text{train}}, e_i \in E^{\text{train}} \setminus \{e_j, j = 1, 2, \dots, i-1\}, i = 2, 3, 4, 5\}$
6	1	E^{train}

6.4. Impact of the number of available labelled data samples on the RUL prognostics

Due to prevent maintenance, most aircraft systems are replaced before their failure. There are therefore only limited labelled data samples available. In this section, we thus study the impact of the size of the library in the matching approach, i.e., the number of offline health indicators in the library, on the accuracy of the RUL prognostics. The health indicators are constructed using unlabelled data samples from the beginning of an engine's lifetime only. In real life, there are enough unlabelled data samples to train an autoencoder. We thus use the same online and offline health indicators as in Section 4.

Fig. 14 shows the RMSE and the MAE of the RUL prognostics for an increasing number of available offline health indicators in the library. This is also called the learning curve. We consider for each number of available offline health indicators, all possible combinations of historical health indicators that give a library of this size. For example, if two health indicators are available, we consider the following 15 libraries:

$$\{(1,2),(1,3),(1,4),(1,5),(1,6),(2,3),(2,4),(2,5),$$
$$(2,6),(3,4),(3,5),(3,6),(4,5),(4,6),(5,6)\}.$$

Here, $1, 2, \dots, 6$ denotes the offline health indicator from the first, second, $\dots$, sixth training engine respectively. An overview of all considered libraries is in Table 7. With each library, we predict the RUL of the test engines.

As expected, the RUL predictions improve when the size of the library increases. The decrease in the RMSE and MAE is highest when we consider two offline health indicators in the library, instead of just one. However, even when a library consists of just one offline health indicator, the RUL is well estimated with a RMSE of only 3.95 flights, and a MAE of only 3.38 flights. This shows that our approach works well when only very few labelled data samples are available.

7. Conclusion

In aviation, safety–critical aircraft systems usually undergo preventive maintenance. Consequently, only very few labelled sensor data

samples, with as label the true RUL, are available. Many labelled data samples, however, are required to train supervised learning models that directly predict the RUL. In this paper, we therefore instead propose to construct a health indicator by training a LSTM autoencoder (LSTM-AE) with unlabelled data samples (i.e., the corresponding true RUL is unknown). The reconstruction error with the LSTM-AE increases as the degradation in a system increases, and is therefore used as health indicator. The sensor measurements of aircraft systems are generated at a high frequency during flights of several hours. Each data sample thus consists of a long time-series of multi-sensor measurements. We apply attention in the LSTM-AE to handle these long time-series. Moreover, aircraft are operated under highly-varying operating conditions. To create robust health indicators, we thus integrate the operating conditions in the LSTM-AE.

Next, we divide the lifetime of each engine in a healthy and an unhealthy stage using Chebyshev's inequality on the health indicators. Then, we use the health indicators and the few available labelled data samples in a similarity-based matching approach to predict the RUL of the engines in the unhealthy stage.

We apply this approach to the aircraft engines in the new N-CMAPSS dataset (Arias Chao et al., 2021). The obtained health indicators have a high monotonicity (0.38), prognosability (0.94) and trendability (0.95). Moreover, the health indicators are indeed robust to the varying operating conditions: the trendability is also high for engines with operating conditions deviating from the operating conditions in the training set. Also the obtained RUL prognostics are accurate, with a RMSE of only 2.67 flights. Moreover, our approach outperforms supervised learning methods, that directly predict the RUL, with a decrease in the RMSE of 19%.

Our proposed methodology is illustrated for aircraft engines. However, the described methodology is also suitable for applications in other fields, such as wind turbine gearboxes, bearings in industrial applications or batteries. For future research, we therefore plan to apply this methodology for other components and systems in other industries. Moreover, for future research, we plan to analyse if the failure mode of an engine can be determined based on the reconstruction loss of different sensors.

CRediT authorship contribution statement

Ingeborg de Pater: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing, Visualization. **Mihaela Mitici:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data is publically available at the NASA open data portal.

References

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

Berghout, T., Mouss, L.-H., Kadri, O., Saïdi, L., Benbouzid, M., 2020. Aircraft engines remaining useful life prediction with an adaptive denoising online sequential extreme learning machine. Eng. Appl. Artif. Intell. 96, 103936.

Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.

Chao, M.A., Kulkarni, C., Goebel, K., Fink, O., 2022. Fusing physics-based and deep learning models for prognostics. Reliab. Eng. Syst. Saf. 217, 107961.

de Pater, I., Mitici, M., 2022. Novel metrics to evaluate probabilistic remaining useful life prognostics with applications to turbofan engines. In: PHM Society European Conference, Vol. 7. pp. 96–109.

de Pater, I., Reijns, A., Mitici, M., 2022. Alarm-based predictive maintenance scheduling for aircraft engines with imperfect Remaining Useful Life prognostics. Reliab. Eng. Syst. Saf. 221, 108341.

Fink, O., Wang, Q., Svensen, M., Dersin, P., Lee, W.-J., Ducoffe, M., 2020. Potential, challenges and future directions for deep learning in prognostics and health management applications. Eng. Appl. Artif. Intell. 92, 103678.

Fu, S., Zhong, S., Lin, L., Zhao, M., 2021. A novel time-series memory auto-encoder with sequentially updated reconstructions for remaining useful life prediction. IEEE Trans. Neural Netw. Learn. Syst..

Géron, A., 2018. Neural Networks and Deep Learning. O'Reilly.

Gers, F.A., Schmidhuber, J., Cummins, F., 2000. Learning to forget: Continual prediction with LSTM. Neural Comput. 12 (10), 2451–2471.

Gugulothu, N., Tv, V., Malhotra, P., Vig, L., Agarwal, P., Shroff, G., 2017. Predicting remaining useful life using time series embeddings based on recurrent neural networks. arXiv preprint arXiv:1709.01073.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kong, X., Yang, J., 2019. Remaining useful life prediction of rolling bearings based on RMS-MAVE and dynamic exponential regression model. IEEE Access 7, 169705–169714.

Koutroulis, G., Mutlu, B., Kern, R., 2022. Constructing robust health indicators from complex engineered systems via anticausal learning. Eng. Appl. Artif. Intell. 113, 104926.

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., Lin, J., 2018. Machinery health prognostics: A systematic review from data acquisition to RUL prediction. Mech. Syst. Signal Process. 104, 799–834.

Liu, C., Sun, J., Liu, H., Lei, S., Hu, X., 2020. Complex engineered system health indexes extraction using low frequency raw time-series data based on deep learning methods. Measurement 161, 107890.

Luong, M.-T., Pham, H., Manning, C.D., 2015. Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025.

Lyu, J., Ying, R., Lu, N., Zhang, B., 2020. Remaining useful life estimation with multiple local similarities. Eng. Appl. Artif. Intell. 95, 103849.

Malhotra, P., TV, V., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G., 2016. Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder. arXiv preprint arXiv:1608.06154.

Ochella, S., Shafiee, M., Dinmohammadi, F., 2022. Artificial intelligence in prognostics and health management of engineering systems. Eng. Appl. Artif. Intell. 108, 104552.

Saxena, A., Goebel, K., 2008. Turbofan engine degradation simulation data set. NASA Ames Progn. Data Repos. 1551–3203.

Shen, S., Lu, H., Sadoughi, M., Hu, C., Nemani, V., Thelen, A., Webster, K., Darr, M., Sidon, J., Kenny, S., 2021. A physics-informed deep learning approach for bearing fault detection. Eng. Appl. Artif. Intell. 103, 104295.

Singh, J., Darpe, A., Singh, S.P., 2020. Bearing remaining useful life estimation using an adaptive data-driven model based on health state change point identification and K-means clustering. Meas. Sci. Technol. 31 (8), 085601.

Vasilev, I., 2019. Advanced Deep Learning with Python: Design and Implement Advanced Next-Generation AI Solutions using TensorFlow and PyTorch. Packt Publishing Ltd.

Wang, J., Zeng, Z., Zhang, H., Barros, A., Miao, Q., 2022. An hybrid domain adaptation diagnostic network guided by curriculum pseudo labels for electro-mechanical actuator. Reliab. Eng. Syst. Saf. 228, 108770.

Wei, Y., Wu, D., Terpenny, J., 2021. Learning the health index of complex systems using dynamic conditional variational autoencoders. Reliab. Eng. Syst. Saf. 216, 108004.

Xiang, S., Qin, Y., Zhu, C., Wang, Y., Chen, H., 2020. Long short-term memory neural network with weight amplification and its application into gear remaining useful life prediction. Eng. Appl. Artif. Intell. 91, 103587.

Ye, Z., Yu, J., 2021. Health condition monitoring of machines based on long short-term memory convolutional autoencoder. Appl. Soft Comput. 107, 107379.

Yu, W., Kim, I.Y., Mechefske, C., 2019. Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. Mech. Syst. Signal Process. 129, 764–780.

Yu, W., Kim, I.Y., Mechefske, C., 2020. An improved similarity-based prognostic algorithm for RUL estimation using an RNN autoencoder scheme. Reliab. Eng. Syst. Saf. 106926.

Zhai, S., Gehring, B., Reinhart, G., 2021. Enabling predictive maintenance integrated production scheduling by operation-specific health prognostics with generative deep learning. J. Manuf. Syst. 61, 830–855.