Received 7 November 2022, accepted 25 November 2022, date of publication 12 December 2022,
date of current version 15 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3226780

RESEARCH ARTICLE

Data-Efficient Estimation of Remaining Useful Life for Machinery With a Limited Number of Run-to-Failure Training Sequences

GERMAN STERNHARZ, AYMAN ELHALWAGY, AND TATIANA KALGANOVA, (Member, IEEE)
CEDPS, Department of Electronic and Electrical Engineering, Brunel University London, Uxbridge, UB8 3PH Middlesex, U.K.

Corresponding author: Tatiana Kalganova (tatiana.kalganova@brunel.ac.uk)

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC).

ABSTRACT Prognostics and Health Monitoring (PHM) of machinery is a research area with great relevance to industrial applications as it can serve as a foundation for safer, more cost-efficient operation and maintenance. The prediction of Remaining Useful Life (RUL) plays an important part in this field and has seen significant advances from the introduction of machine learning methods. However, these methods typically require model training with a large number of run-to-failure sequences, which are often not feasible to obtain due to the required time and cost investments. The present study addresses this issue by introducing a novel methodology, which first quantifies the deviation from the machine's health and fault state and then calculates a machine Health Index (HI) prior to the prediction of RUL. In addition, the start of a degradation state is determined. Alternative implementations of the proposed methodology are compared utilising several methods, including Support Vector Regression (SVR), Long Short-Term Memory (LSTM) Neural Network (NN), Mahalanobis Distance (MD), and LSTM Autoencoder (AE) NN. The methodology is applied to the open turbofan degradation (C-MAPSS) and bearing vibration (FEMTO-ST PROGNOSTIA) datasets. When a reduced subset of training sequences is used, the prediction results demonstrate that the proposed methodology largely outperforms the baseline method without HI generation. For example, when comparing prediction errors of the C-MAPSS dataset at a reduction of the available number of training sequences to 5%, the proposed method shows an average prediction improvement by 6.5% - 19.2% relative to the baseline method. The presented approach is therefore suitable to improve model generalisation for cases with a limited number of training sequences. When the full training set is utilised, the most resource-saving variant of the proposed approach achieves an average training duration of 8.9% compared to the baseline method. Hence, an additional contribution of the presented data-efficient approach is the reduction of required computing resources, which has implications on training time, energy consumption, and environmental impact.

INDEX TERMS Computational efficiency, data fusion, data reduction, environmental impact, feature reduction, green AI, incomplete data, RUL prediction.

I. INTRODUCTION

The estimation of Remaining Useful Life (RUL) belongs to the domain of Prognostics and Health Monitoring (PHM) and aims to quantify the time or number of cycles until the monitored machinery or component reaches a predefined degraded state, such as complete failure or a safety-critical condition. As such, RUL estimation can form a major part of a predictive maintenance approach with the potential to benefit economical and safety considerations. Costly machine failure and downtime can be reduced, and maintenance programmes optimised through prognostics-based planning. Otherwise potentially undetected faults or degradation conditions can be spotted, which is especially relevant for safety-critical applications.

Algorithms in this field can be categorised based on the level of output information they provide (e.g., detection, localisation, and/or assessment of faults) [1]. Another

The associate editor coordinating the review of this manuscript and approving it for publication was Yu Wang.

common categorisation is based on the model foundation [2], which is either physics-based, data-driven, or a hybrid combining aspects of both approaches. However, further subcategories (such as the distinction of data-driven methods into statistical and AI-based) and differences in terminology can be encountered in literature, a detailed survey of which is given in [3]. Physics-based methods have the potential to operate on conditions previously never encountered during monitoring. However, physics-based approaches are often limited by their inflexibility to updating with on-line data and become impracticable for high degrees of system complexity and environment noise [2], [4], [5]. On the other hand, data-driven methods, which often utilise AI-based methods, typically achieve better statistical performance, are real-time capable and alleviate the need for manual feature engineering by algorithmic optimisation and transformations of the input space towards the prediction task [6], [7], [8], [9]. Data-based prognostics methods largely benefitted from and developed alongside the advancements in machine learning, computing power, and large-scale sensor data. A major limitation of these methods, however, is typically the large amount of required training data, which can be unfeasible to obtain due to time and cost restrictions. For RUL prediction tasks, the foundation of such training data is typically a multitude of machine units, which provide measurements across their full life cycles until failure.

Compared to the acquisition of these run-to-failure sequences, samples of solely healthy or faulty machinery data (i.e., without the full life cycle data of the machine) are much more readily available and producible. This observation is supported by the much smaller number of open run-to-failure vibration datasets [10], [11], [12] compared to the number of open vibration datasets covering various faults and healthy baseline conditions without intermediate life cycle data [13], [14], [15], [16], [17], [18], [19], [20], [21]. For the purpose of the present study, such data can be referred to as binary (i.e., healthy and faulty) condition data. Faulty condition measurements can be acquired, for example, from diagnostic field data recorded after occurrence and detection of a fault. Alternatively, faults can be intentionally induced into machinery components for experimental acquisition of fault conditions without the need for potentially resource-intensive run-to-failure measurements. Such controlled fault conditions minimise the risk of damaging other interacting components or provoking subsequent complete machine failure, such that the fault can be rectified cost-effectively. These observations are used to the advantage of the proposed methodology, as it employs binary condition data (i.e., data from healthy and faulty condition) to create models of machine condition deviation from the healthy and faulty state.

In addition to the labour, hardware and energy cost associated with comprehensive run-to-failure tests, the environmental impact should be considered. Adverse effects of energy consumption, such as greenhouse gas emissions represent an increasing reason for concern across industries, including the domains of AI and Machine Learning (ML) [22], [23], [24]. At the same time, only a minority of current publications in the AI space seem to address the training efficiency of the proposed methods [22], [24].

Some literature already attempts to use a reduced dataset to address the issue of incomplete data. However, many of these methods, such as in [25] use application-specific analytical approaches to model a specific behaviour, thus limiting the scope of their application. In other cases, such as in [26] the practical application of the methodology is limited by over-simplified approximations of the failure signal, which may not generalise to all cases. Other methods, whilst making use of more flexible AI-based approaches [27], fail to explore the generalisation ability of the approach.

These considerations provide the reasoning for the proposed methodology. Binary (i.e., health and fault) condition data is used to form models, which quantify the current machine deviation from the healthy and faulty state. As this data is easier to obtain compared to full run-to-failure sequences, these models can take advantage from additional training cases, if available. Alternatively, the health and fault conditions are extracted from the start and end section of available run-to-failure training sequences. This condition deviation data is then used to calculate an HI value, which in turn is processed to detect the occurrence of initial degradation. The available (often limited) number of run-to-failure training sequences is then processed to generate corresponding HI training functions, which are then used to train a RUL prediction model.

The resulting contributions of the present work are summarised below.

- A novel methodology for RUL prediction and degradation detection from a limited number of run-to-failure training sequences is introduced. It is the first approach taking advantage of distinct (healthy and faulty) condition data for training, in addition to (limited) run-to-failure training sequences.
- A novel HI calculation method is proposed, implemented using MD and AE, to model the current deviation from the healthy and faulty machine state, followed by the HI calculation from the ratio of these modelled condition deviations. In a broader sense, this represents a novel feature reduction and data fusion approach for machine condition data.
- Several variants of the proposed methodology (based on MD, AE, Least Squares, LSTM, and SVR) are implemented and compared in terms of prediction accuracy, variance, and training time requirements on a varying reduced number of training sequences.
- In contrast to previous publications dealing with RUL prediction from limited training data, the proposed methodology is applied to two public benchmark datasets (C-MAPSS and FEMTO), indicating its capability to improve model generalisation compared to baseline LSTM prediction and providing insights into reduction of training duration due to training data reduction.

The structure of the present paper is outlined in the following. The next Section II reviews existing research relevant to the present study. The subsequent Section III outlines the theoretical foundation of the main involved methods, which are utilised by the proposed methodology. An overview of this methodology is presented in the thereafter following Section IV with subsections devoted to details of its main processing steps. Section V covers two open datasets used for validation and comparison of the proposed methodology, including the presentation and discussion of obtained results. Finally, Section VI presents the main findings, giving conclusions, as well as current challenges and suggestions for future work in the domain of RUL prediction.

II. RELATED WORK

This section gives an overview of existing studies dealing with related issues of RUL prediction and training data reduction in this domain. Strengths and limitations of the reviewed publications are outlined to identify the distinguishing aspects and relevance of the present study.

A. NN-BASED RUL PREDICTION

Deep learning (DL) based approaches for RUL prediction have grown in popularity in recent years due to a vast increase in available computational power allowing for a higher research output, as well as the possibility of more complex NN models. A variety of RUL application domains have utilised DL approaches [28], [29], [30]. For instance, one study [29] proposes an LSTM NN combined with attention mechanism and Particle Swarm Optimisation (PSO) for lithium battery RUL prediction. Another study [28] investigates wind turbine gearbox RUL prediction using ML and concludes that NN based methods provide the best accuracy out of the compared methods. The authors had access to a large volume of wind turbine supervisory control and data acquisition (SCADA) data as well as vibration data and found that fusing the two types of data as opposed to using just SCADA data can extend the prediction period from a month to up to 6 months, which empirically demonstrates the value of using a variety of data as opposed to just large quantities. Multi-feature fusion is utilised in this paper to take advantage of these benefits found in literature.

Recurrent Neural Network (RNN) models are highly popular for time-series inference tasks due to their long-term dependency handling ability. For instance, Guo et al. [31] proposed an RNN HI, which utilises a novel feature selection process based on correlation and monotonicity. This work further proves the benefit of multivariate data for data-driven models as well as the effectiveness of RNN models for time-series modelling. However, due to the vanishing gradient problem commonly observed when training RNNs [32], [33], Long Short Term Memory (LSTM) [34] and Gated Recurrent Unit (GRU) NNs [35] are preferred to traditional RNN architectures. These RNN variants subsequently perform better in the RUL problem that this work covers, and hence, LSTM is also used as a foundation in the present study. For instance, Zhang [36] proposes a 2-layer LSTM Network which takes advantage of attention mechanism to prioritise relevant feature learning and thereby improve the model efficiency, with a 1-dimensional Convolutional Neural Network (CNN) for feature extraction preceding it and a Multi-Layer Perceptron (MLP) which outputs the RUL value following it. Various studies make use of CNN [37], [38], which are primarily used in visual recognition tasks [39] but have also shown promise with time-series problems [40]. One such study [37] proposes a Deep CNN (DCNN) for the prediction of RUL. The proposed model is a multi-layered CNN which does not take advantage of pooling, as the authors noted its unsuitability for time series-based tasks due to their low dimensionality. The authors input data from various sensors into the model with a sliding window and output a single classification of RUL as the output. The model was empirically validated on the C-MAPSS degradation dataset from NASA, where the authors set a maximum constant of 125 for the RUL value across all motors. However, this means that the proposed model cannot predict further than 125 cycles from failure, which can be required in some practical cases. In contrast to that, the methodology introduced in the present paper employs degradation detection, which considers the characteristics of individual monitored machine units instead of applying a global RUL limit, thus indicating and processing a beginning degradation trend on a unit-by-unit basis.

B. HI-BASED RUL PREDICTION

The calculation of an HI as a means of feature dimensionality reduction was utilised before in previous studies [30], [31], [41], [42]. A study by Yang et al. [43] compared directly modelling the RUL and using the HI as an intermediate variable to estimate the RUL, and found that the proposed method, which utilises the latter is able to consistently outperform the former. The authors further noted that in practice, RUL can fluctuate based on changes in operating condition and measurement noise. The authors employ various methods of smoothing the HI degradation curve to account for these issues. Another method proposed by Wei et al. [44] uses a dynamic conditional variational autoencoder to construct the HI of multi-sensor systems. The authors also provide further evidence that HI-based RUL approaches are more effective than direct RUL modelling. However, whilst the method makes use of operating condition information which, as Wei et al. noted, may not be available for all systems, the authors prove the effectiveness of their approach regardless of the availability of this data.

However, in contrast to the present study, the aforementioned studies utilise complete datasets and do not investigate the prediction performance under limited data availability. Furthermore, the proposed approach for HI calculation is based on the MD and AE LSTM NN (depending on the implementation) and can take advantage of more feasibly obtainable discrete healthy/faulty condition data, if available, in addition to run-to-failure sequences.

C. RUL PREDICTION FROM LIMITED TRAINING DATA

RUL prediction typically requires extensive data, covering a large amount of run-to-failure life cycle sequences, which has an adverse environmental footprint and may require substantial time and financial investments. This applies both for the acquisition of data itself as well as the subsequent data processing and model generation.

Several research papers recognised this issue. Certain methods in the field of transfer learning focus on domain adaptation and operate on incomplete RUL labels [8], [45], [47]. In these cases, model training is performed using labelled samples at certain machine conditions (in the source domain) in combination with unlabelled samples of other machine conditions (in the target domain). While these methods operate on fractionally labelled information, they still require complete sequences of input features, limiting the data reduction capability of these methods. The same limitation applies to the method presented in [48], which investigates gradual reduction of labelled training data. However, the full set of training data is used throughout in a semi-supervised approach utilising Restricted Boltzmann Machine (RBM) and LSTM.

In the domain of RUL prediction of cutting tools, a study [25] utilises incomplete data covering the first part of the life cycle sequence and cutting it short before the life cycle end is reached. While sensor data is used for the training of a NN, the method is based on classic analytical models for tool wear, like those introduced by Takeyama, Murata [49] and Usui [50]. Therefore, this approach can be classified as a hybrid method and is limited in its application to cutting tools. Another paper [51] investigates the crack growth of steel plates and bearing degradation to estimate the RUL using sparse and fragmented data samples. However, the missing samples (either individually or in fragments) are randomly selected from the full life-cycle sequence, approximating the distribution of the original data range. Another study [26] constructs a prediction model by fitting a Bernstein distribution to the lifetime numbers and using its parameters to define an assumed degradation function. A large reduction in data-usage is achieved, however the drawback of this approach is that the initially assumed degradation function may not be representative of the actual degradation trend. Moreover, the method operates on a single degradation feature and this feature reaches a constant end value for every run-to-failure sequence, limiting the method's potential for more complex cases with signals, which are either multivariate or have varying value ranges per run-to-failure sequence. Two papers deal with the RUL prediction of wind turbine bearings [27], [52] from an incomplete life cycle sequence. The earlier study [52] uses a state-space model constructed from an empirical equation for bearing wear based on the spalling area propagation. Online degradation data is then used to update the state and predict the RUL with a particle filter. As another hybrid approach, it is limited in its application to rolling element bearings. In the later study [27], an Elman NN is used to obtain a data-driven condition model instead. However, the main limitation of both studies is that just a single run-to-failure sequence is used for demonstration and validation, such that its generalisation ability remains questionable.

A study [53] implemented a method based on Support Vector Machine (SVM), where a reduced training set was provided. Specifically, between 33% to 36% of bearing run-to-failure training sequences were shortened, while the remaining majority of training sequences remained complete. A limitation of this study is that the method outputs survival probability values instead of RUL values. Moreover, the prediction is limited to 9 discrete values, which is detrimental for an accurate prediction of the monitored machine condition. A similar SVM-based approach is followed in [54]. The potential of SVM-based RUL prediction, particularly at limited availability of training data, is also pointed out in a review study [3]. This is based on the high training data demand typically imposed by common NN-based methods in contrast to SVM and SVR, which motivates the use of SVR in the present study.

Whilst the topic of RUL is widely covered in literature, only a small proportion of these papers aims to address the topic of run-to-failure sequence availability. However, in a real-world environment, it is not always assured that full run-to-failure sequences will be available, which means that any approach which aims to have practical feasibility must account for a lack of data. To the best of the authors' knowledge, no currently existing methods for RUL prediction from limited data use open datasets, limiting reproducibility and comparability of existing publications. To address this issue, this paper utilises two open degradation datasets commonly used to validate RUL approaches, thus allowing for future studies to compare any new approaches to the present work and encourage research into more efficient RUL approaches that focus on data reduction techniques. The intention is to thereby increase the feasibility of practical application, where available training sequences might be limited, with an added environmental benefit of reduced power consumption of algorithms.

III. THEORETICAL FOUNDATION

This section provides the theoretical introduction to individual methods utilised in course of this paper, followed by an illustration of the proposed methodology for data-efficient RUL prediction.

A. LONG SHORT-TERM MEMORY (LSTM) NEURAL NETWORK

The LSTM NN is a variant of the RNN architecture proposed by Hochreiter [34] to mitigate the vanishing gradient problem [32], [33]. The vanishing gradient commonly occurs when training a deep RNN using the backpropagation through time algorithm; when unrolling an RNN to update the weights, due to the depth of the network, the calculated derivatives are prone to exponentially ‘‘vanishing’’ or ‘‘exploding’’. This

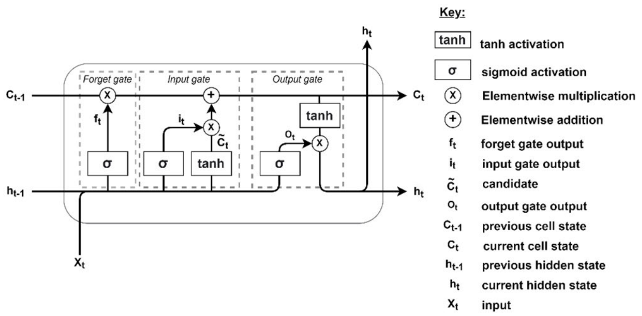

FIGURE 1. Schematic architecture of an LSTM cell (courtesy of [34]).

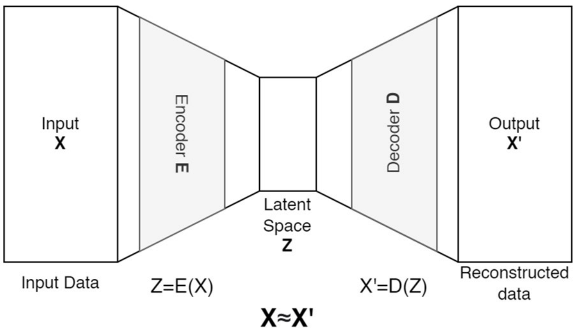

FIGURE 2. Schematic architecture of an Autoencoder model.

is worsened by the shared weights of the RNN across time, which results in a higher likelihood of the vanishing or exploding gradient issue occurring in comparison to a standard feedforward network.

The LSTM cell architecture, illustrated in Figure 1, addresses this issue using a specialised gate structure with three gates: forget gate, input gate and output gate. The forget gate, which outputs a vector f_t, determines which information is forgotten from the previous cell state based on the previous hidden cell state h_{t-1} and cell input x_t in the current cell state. This operation is mathematically represented in (1). The input gate, which outputs a vector i_t, controls which information from the candidate vector $\tilde{C}_t$ is stored in the cell state. The input gate equation is described in (2), and the equation to determine the candidate values is described in (3). The new cell state is determined by the elementwise product (also known as the Hadamard product and denoted by the operator $\bigcirc$) of the previous cell state with the forget gate output. This is followed by an elementwise addition with the Hadamard product of the input gate output and the candidate vector $\tilde{C}_t$. This is mathematically described in (4). The output gate output is determined using (5), then used in the calculation for the new hidden state, described in (6). Weight matrices and bias vectors are denoted by W and b, respectively. Subscripts f, i, and o refer to the parameters of the forget, input, and output gate, respectively.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{1}$$

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{2}$$

$$\tilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t * \tanh(C_t) \tag{6}$$

1) AUTOENCODER ARCHITECTURE

The Autoencoder (AE) architecture was introduced by Hinton [55] as a method of utilising unsupervised learning to acquire encodings of unlabelled data in latent space. The autoencoder operates by "encoding" the input values using dimensionality reduction into latent space, then "decoding" the latent space representation to reconstruct the original input values. This operation is illustrated in Figure 2.

B. LEAST SQUARES REGRESSION

Least Squares (LS) is an approach used in regression analysis to determine an approximated solution for overdetermined systems. This is accomplished by minimising the total residual of the sum of squares of the variables observed. There are two categories of LS approaches that can be used depending on the problem: linear or non-linear. Linear regression is often preferred due to simplicity and interpretability but can only be used when each coefficient is a constant or a product of a variable. Non-linear LS is more flexible modelling capability but sacrifices the ease of interoperability that linear LS models offer. The present study will employ the use of a subcategory of linear LS regression referred to as polynomial LS.

In polynomial LS, a function can be modelled by a polynomial in the form of (7) with polynomial order N_p and coefficients a_{i_p}.

$$y = a_0 + a_1 x + \ldots + a_{i_p} x^{i_p} + \ldots + a_{N_p} x^{N_p}, \tag{7}$$

which can also be represented for N samples of x and y in matrix form as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 & x_1 & \cdots & x_1^{N_p} \\ 1 & x_2 & \cdots & x_2^{N_p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & \cdots & x_N^{N_p} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N_p} \end{bmatrix}$$

$$y = Xa. \tag{8}$$

For overdetermined systems, i.e., where $N_p < N - 1$, an LS approximation of the polynomial coefficients can be obtained from

$$a = \left(X^{\mathrm{T}} X\right)^{-1} X^{\mathrm{T}} y. \tag{9}$$

C. MAHALANOBIS DISTANCE

The Mahalanobis Distance (MD) [56] represents the deviation from a set X of multidimensional samples (i.e., the input space) to a sample x of the same dimensionality. As a distance metric, the MD is always positive. In addition, the MD is a relative quantity in standard deviations, and therefore scale-invariant towards its input samples, not requiring feature scaling.

The MD is calculated with (10), which contains the inverse covariance matrix K^{-1} of X, the vector of input feature means μ, and the single multidimensional feature sample x.

$$d\left(x\right)=\sqrt{\left(x-\mu\right)^{\mathrm{T}}K_{XX}^{-1}\left(x-\mu\right)} \tag{10}$$

D. SUPPORT VECTOR REGRESSION

Support Vector Regression (SVR) [57] is a supervised learning algorithm from the Support Vector Machine (SVM) [58] family that is used to solve regression problems. In comparison to LS regression, which approximates a function to minimise the sum of squared errors, SVR allows flexibility to define the error limit acceptable in the model using the epsilon tube parameter ε, and finds a best fitting function within the defined error constrains that minimises the L2 norm of the vector coefficients. The SVR algorithm also accounts for outliers outside the error limit using a slack variable ξ. This value denotes the deviation from the error margin ε, as the error margin by itself may be unfeasible for more complex data trends in some cases. The slack variable is minimised using a regularisation parameter C in the cost function, where, as C approaches 0, the tolerance for the slack variable ξ decreases and as C approaches 1 the tolerance for slack variables increases.

In the case of multidimensional inputs, a hyperplane is used as the decision boundary instead of a two-dimensional line. To fit the hyperplane to the data, a kernel is used to map the training data to a higher dimension. Some examples of kernels that are commonly used include linear, non-linear, polynomial and Radial Basis Function (RBF), the latter of which is utilised in the present study.

This section outlined the theory of methods, which are part of the proposed methodology for data-efficient RUL prediction. Their specific application within the proposed framework is presented next.

IV. PROPOSED METHODOLOGY FOR DATA-EFFICIENT RUL PREDICTION

This section presents the proposed methodology by first providing an initial outline, which is then followed up by subsections covering details of the involved steps and methods.

The proposed methodology consists of three main steps, which are indicated in the overview graphic in Figure 3 with a reference to the corresponding subsection. First (Section IV-A), the health and the fault condition models are created from binary (i.e., healthy or faulty) condition data. A limited number of run-to-failure training sequences is then fed into the health and fault models, providing health and fault condition deviation sequences (d_h and d_f), respectively. In the second step (Section IV-B), both values are combined through the calculation of an HI value. The HI is then used in a degradation detection step, which provides an additional component of monitoring information and can be used to separate relevant degradation data from healthy condition data downstream. In the final step (Section IV-C), the degradation data is processed by a prediction model, which outputs the final RUL prediction.

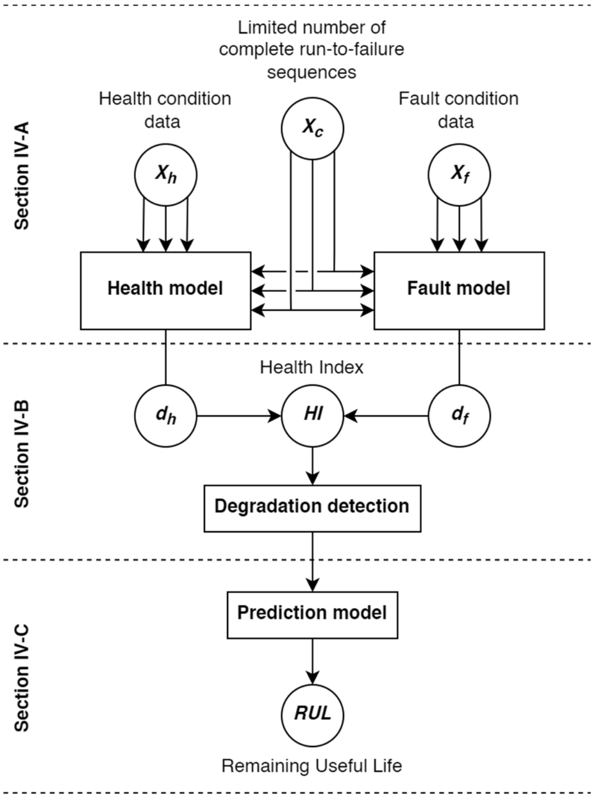

FIGURE 3. Overview graphic of the proposed RUL prediction method.

The health and the fault model (and thus the HI) are created based on two approaches, i.e., Mahalanobis Distance (MD) [56] and LSTM Autoencoder (AE) NN [59]. For the RUL prediction model, three approaches are used, namely quadratic polynomial LS extrapolation (QPoly), Support Vector Regression (SVR) [57], and Long Short-Term Memory (LSTM) NN [34]. The proposed approaches are presented, and their results are compared in course of this study.

In addition, a direct RUL prediction based on the dataset features (i.e., without HI calculation) is performed as the baseline for the comparison to evaluate the impact of the introduced condition features.

Table 1 lists all combinations of methods, which are implemented, validated, and compared in this study. Since different variants of the proposed methodology are implemented based on different methods, the combined methods, forming the foundation for each approach, are given in Table 1 as well.

A. MACHINE CONDITION DEVIATION

As the foundation for the HI calculation (addressed in Section IV-B), the deviation d_h of the current machine condition sample x from the completely healthy condition data X_h on one hand and the deviation d_f from the completely degraded condition data X_f on the other hand shall be determined in an unsupervised manner. With data-efficiency in

TABLE 1. Implemented and compared RUL prediction methods.

Abbrev.	Method for health and fault models	+	Method for RUL prediction
LSTM (baseline)	N/A	+	LSTM Neural Network
MD-QPoly (proposed)	Mahalanobis Distance	+	Quadratic Polynomial Least Squares Regression
MD-SVR (proposed)	Mahalanobis Distance	+	Support Vector Regression
AE-SVR (proposed)	LSTM Autoencoder Neural Network	+	Support Vector Regression
MD-LSTM (proposed)	Mahalanobis Distance	+	LSTM Neural Network

mind, data for both reference conditions are used for training, while full run-to-failure sequences are not required.

In the present study, two approaches are implemented and later compared for this task: LSTM AE NN, and MD. Both methods are utilised to construct two models each. The "health model" (providing the deviation d_h) is constructed from X_h while the "fault model" is conditioned from X_f (providing the deviation d_f).

Condition data X_h and X_f can be obtained from dedicated measurements or available run-to-failure sequences by assuming that the first and last n_{hf} samples represent the health and fault condition, respectively. Then, a run-to-failure sequence X with N samples, can be subdivided into X_h, X_f, and the intermediate degradation data X_{itm}, the latter of which is not used by the health and fault models. This is described by (11), where x_i represents the ith multivariate data sample of the sequence X.

$$X = \begin{bmatrix} X_h \\ X_{itm} \\ X_f \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_{n_{hf}} \end{bmatrix} \\ \begin{bmatrix} x_{n_{hf}+1} \\ \vdots \\ x_{N-n_{hf}-1} \end{bmatrix} \\ \begin{bmatrix} x_{N-n_{hf}} \\ \vdots \\ x_N \end{bmatrix} \end{bmatrix} \quad (11)$$

1) LSTM AUTOENCODER

After training of the AE NN with binary condition data X_h or X_f, the desired deviation value d_h or d_f is obtained as the difference between the current condition sample x and its corresponding AE reconstruction as described by (12).

$$\begin{aligned} d_h &= x - D_h(E_h(x)) \\ d_f &= x - D_f(E_f(x)) \end{aligned} \quad (12)$$

To use multiple features for the AE-based condition deviation models, the deviations of individual features are combined into a single function by standardisation and averaging. Progressions of a raw feature, its AE reconstruction and the reconstruction error are shown later in the results section for both the health and the fault models in Figure 14 and Figure 15, respectively.

2) MAHALANOBIS DISTANCE

The MD-based health and fault models are created from (10), where the health data X_h and fault data X_f are used to calculate the respective covariance matrix K_{XX} and mean feature vector μ for the respective health or fault model. The output of (10) then represents the health deviation d_h or fault deviation d_f, respectively.

3) FEATURE SUBSET OPTIMISATION BY THE GENETIC ALGORITHM (GA)

In frame of the proposed method, the MD is utilised to quantify the deviation d of current machine condition to a reference (healthy or faulty) condition. For this purpose, it is desirable to maximise the MD between opposing (i.e., healthy and faulty) condition data X_h and X_f. This is achieved by the selection of a data feature subset S, which is optimised towards the maximum MD separation d_{sep} defined in (13). The equation contains MD values $d_h(X_h)$ between individual healthy samples x_h and the healthy reference state X_h, as well as the opposing MD values $d_f(X_h)$ between individual healthy samples x_h and the faulty reference state data X_f. A Genetic Algorithm (GA) [60] is implemented and employed in the present study for this optimisation task.

In generation 0, an initial population of feature subsets is randomly selected. On one hand, the average $\bar{d}_h$ of MD values is calculated between the reference set X_h and each of its samples x_h. On the other hand, the average $\bar{d}_f$ of MD values is calculated between the reference set X_f and each of the opposing set's samples x_h. The objective function of averaged MD separation $\bar{d}_{sep}$ is represented by (13), which is then maximised by the GA. In each generation, the objective function of each population member (i.e., feature subset candidate S) is evaluated and 5 highest ranking members are selected for crossover to breed a descendent population (i.e., the next generation), assigning or dismissing individual features based on the parents' properties. A 30% mutation rate (chance of random feature inclusion into S) is introduced, and 5 randomly selected members are retained into the next generation to avoid convergence on a local optimum.

$$\max_S \bar{d}_{sep} = \bar{d}_h\left(X_h\left(S\right)\right) - \bar{d}_f\left(X_h\left(S\right)\right) \quad (13)$$

B. HEALTH INDEX (HI) CALCULATION

The HI aims to represent a measure of the relative health condition of the monitored machine, where a value of 1 typically indicates the condition of a new or unworn machine and a value of 0 indicates a condition of machine failure, a critical fault or a degraded condition requiring maintenance.

With data-efficiency in mind, the goal of the proposed HI generation is to generate HI values at any stage of the engine life cycle, utilising solely binary (i.e., healthy and faulty) conditions for the constructed model. As such, the proposed method for HI calculation is largely unsupervised.

The variance in condition signatures and degradation progressions of different monitored machines leads to varying ranges of the condition deviation output by the fault and health model. Despite that, a normalised HI within the interval [0, 1] is obtained by calculating a ratio involving the deviation outputs of the health model d_h and fault model d_f given in (14). This approach was utilised in a previous study of the authors for the quantification of harmonic and random vibration contributions [61]. To balance the contributions of both the fault and health model, min-max scaling is applied to the output of the health and fault model individually based on training data. It should be noted that full run-to-failure sequences are not required to determine the scaling parameters, as the minimum and maximum deviation output is obtained from available healthy and faulty condition data.

$$\mathrm{HI}\,(x) = \frac{d_f(x)}{d_h(x) + d_f(x)} \tag{14}$$

The deviation functions d_f and d_h based on MD are shown later in the results section in Figure 12 for an exemplary run-to-failure sequence from the C-MAPSS FD001 training dataset.

After application of (14), the obtained HI is smoothened by a moving average. The range of the smoothened HI obtained from training data X_h and X_f is then used to calculate min-max scaling parameters for the HI. A HI resulting from the deviation sequences of Figure 12 is shown in Figure 13.

1) DEGRADATION DETECTION

The determination of the HI can be considered as a feature reduction process to a single quantity. The occurrence of the degradation start is detected from the HI by assuming that a predefined number of initial HI samples of each sequence represents the healthy machine condition. A threshold is determined from 3 standard deviations of these initial healthy HI samples. Additionally, a tolerance of 5 samples exceeding this threshold is allowed to lower the sensitivity of the triggered degradation detection to outliers.

As seen from the sequence later in Figure 13, a typical progression of the HI has a consistently high value at the beginning of the run. This phase represents the healthy condition, as it does not indicate a decline of the machine health. Since a degradation trend has not yet developed, it is assumed that a machine-specific and meaningful RUL prediction cannot be obtained during this phase. A common approach to address this issue is to exclude all training samples or relabel those above a certain RUL value to a predefined constant value (in case of C-MAPSS, typically between 120 and 130) [37], [48], [62], [63], [64], [65]. However, this approach neglects variability between individual machine units.

In contrast to that, the two-step procedure of unit-based degradation detection followed by RUL prediction provides several advantages. On one hand, earlier RUL prediction is possible, provided that the specific machine shows an earlier degradation trend. On the other hand, non-meaningful RUL predictions (on healthy condition data prior to a degradation trend) can be avoided or indicated as such to the monitoring operator.

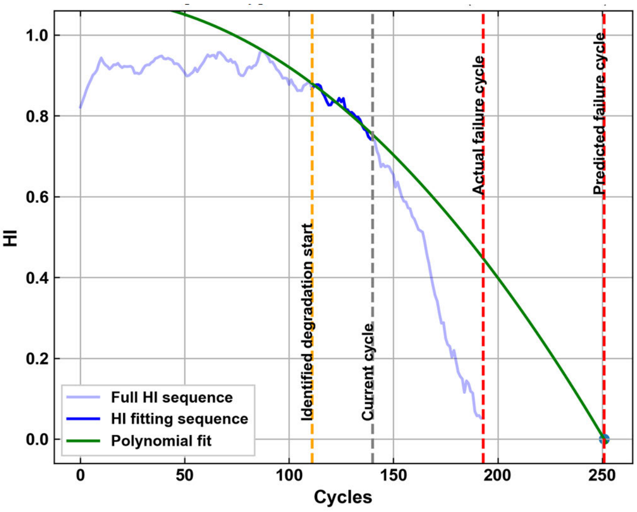

FIGURE 4. Quadratic polynomial fit for RUL prediction on an MD-based HI sequence calculated from a C-MAPSS FD001 training sequence at current cycle 140, leading to a predicted failure cycle overestimation by 58 cycles.

C. REMAINING USEFUL LIFE (RUL) PREDICTION

The RUL prediction step is based on a model, which relates the determined reduced features (condition deviation and HI) to the actual RUL value. Three different methods for this task are presented in the following subsections.

1) POLYNOMIAL LEAST SQUARES REGRESSION

After the degradation point is detected (see Section IV-B-1), thereafter following HI samples of each sequence are used to fit a polynomial function using LS. This avoids non-meaningful predictions on healthy conditions prior to a degradation trend and simplifies the function shape, leading to a closer fit using lower polynomial orders.

A second order (i.e., quadratic) polynomial is fitted in this study. Figure 4 and Figure 5 show the same exemplary full MD-based HI sequence (see Section IV-B) at two different times, i.e., cycles, of prediction. The polynomial is fitted to the data between the detected degradation start and the most recent HI observation at the current cycle t_c. Therefore, the current cycle has an impact on the resulting RUL prediction. Since more data becomes available in Figure 5 (t_c =160) compared to Figure 4 (t_c =140) for the polynomial fit during the progression through the operating cycles, a better fit is achieved in Figure 5, i.e., towards the end of the HI sequence, leading to a more accurate prediction.

The closest polynomial root at a future time value represents the predicted failure cycle $t_{f,pred}$ of the corresponding machine. The predicted RUL is then calculated as $RUL_{pred} = t_{f,pred} - t_c$.

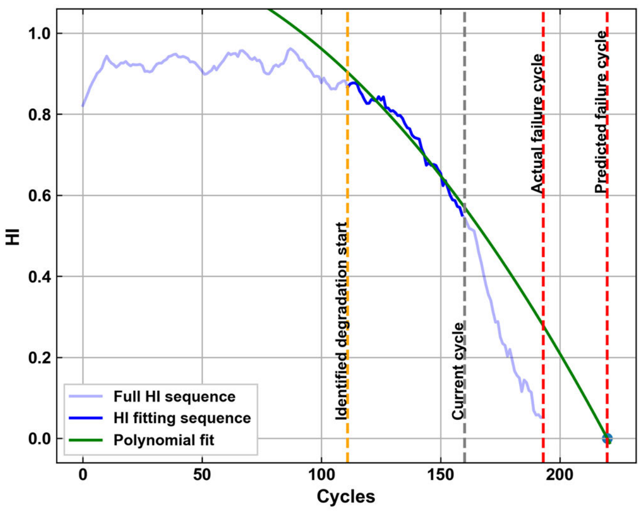

FIGURE 5. Quadratic polynomial fit for RUL prediction on an MD-based HI sequence calculated from a C-MAPSS FD001 training sequence at current cycle 160, reducing the predicted failure cycle overestimation to 27 cycles.

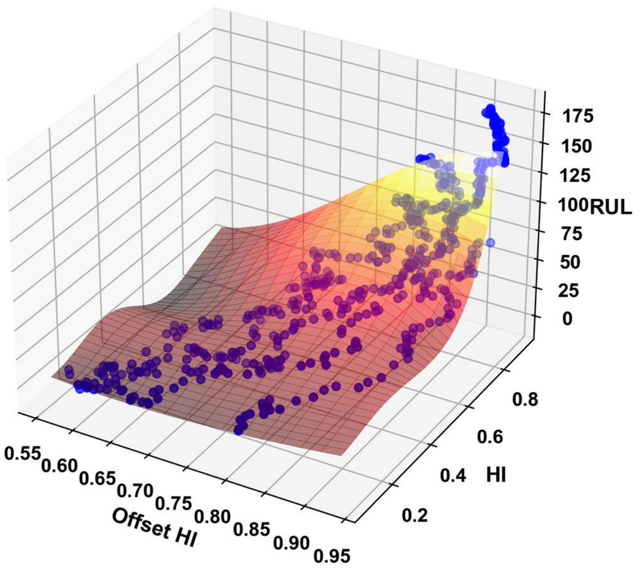

FIGURE 6. Data points of 5 complete C-MAPSS FD001 training sequences and resulting SVR plane for RUL prediction.

2) SUPPORT VECTOR REGRESSION

To consider time-dependency of the degradation in the SVR prediction, a lookback is introduced using an HI offset of -50 samples from the current sample as an additional input feature. Both the offset and current HI samples are fed into the SVR model to predict the RUL. As outlined in the theory Section III-D, the SVR produces a hyperplane, relating the input features to the predicted output RUL value. Data points from 5 run-to-failure training sequences from C-MAPSS FD001 are shown with the resulting 3D plane in Figure 6.

The Radial Basis Function (RBF) is used as the SVR kernel and the regularisation parameter C and epsilon-tube parameter ε are optimised as discussed later in Section V.

TABLE 2. Main properties of the C-MAPSS datasets.

Dataset	Operating conditions	Fault conditions	Training sequences	Test sequences
FD001	1	1	100	100
FD002	6	1	260	259
FD003	1	2	100	100
FD004	6	2	249	248

3) LSTM NEURAL NETWORK

An LSTM NN, as described in Section III-A, is a common choice for RUL prediction problems due to the LSTM's ability to incorporate time-dependency of input features into the mapping function to output the desired prediction variable. For the proposed method, three inputs are used: the condition deviations from the MD-based fault and health models (Section IV-A) with the resulting HI ratio (Section IV-B) are provided as inputs to the LSTM model for RUL prediction. The proposed method is referred to as MD-LSTM in the following.

For a baseline comparison with a more traditional approach, the raw original dataset features are used as inputs to the LSTM model, without condition deviation and HI functions. In accordance with the typical approach of existing literature [37], [48], [62], [63], [64], [65], the effective prediction range is limited to a maximum RUL of 125 by relabelling earlier training data to this RUL value. For a direct comparison of the used input features' impact, this training approach is used for both the proposed MD-LSTM method and baseline LSTM method.

The feature matrix of each engine sequence is then subdivided into smaller rolling sequences of a chosen window length. These overlapping partial sequences serve as samples fed into the model. The used window length is optimised together with other model hyperparameters, incl. the number of hidden LSTM layers, as specified later in Section V. On each LSTM layer, a dropout rate of 0.2 is applied to combat overfitting.

V. EXPERIMENTAL RESULTS AND DISCUSSION

Two open datasets are presented and used to compare and validate the proposed RUL prediction method. Each dataset section is structured into an introduction of the corresponding dataset, model training procedure for individual methods, and performance comparison of different methods as a function of the number of available run-to-failure training sequences.

A. SIMULATED TURBOFAN ENGINE DEGRADATION DATA

An open dataset of turbofan engine degradation [66] was developed by NASA through simulation with the software tool C-MAPSS (Commercial Modular Aero-Propulsion System Simulation). As summarised by Table 2, the full dataset package consists of 4 datasets, which are referred to as FD001 - FD004 and differ by the number of operating conditions and failing engine components, i.e., fault conditions.

TABLE 3. Features of the C-MAPSS dataset.

Feature ID	Description
0	Engine number
1	Time (cycles)
2	Operational setting 1
3	Operational setting 2
4	Operational setting 3
5	Sensor 1
6	Sensor 2
…	…
25	Sensor 21
26	RUL (target output variable)

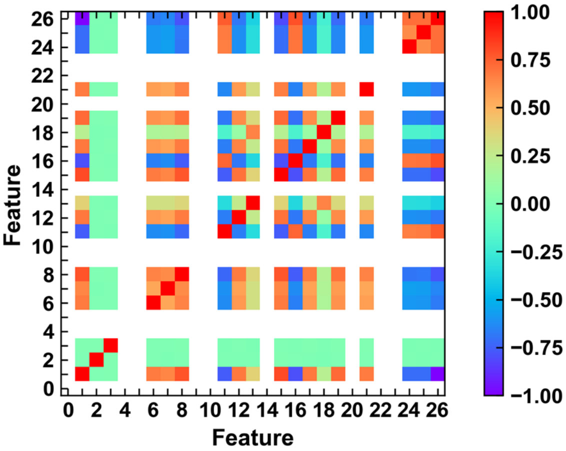

FIGURE 7. Correlation matrix of C-MAPSS FD001 training dataset features, constant features displayed in white.

Additionally, the datasets differ by the number of test sequences and the maximum available number of run-to-failure training sequences, each of which covers the life span of an aircraft engine starting from an unknown state of initial wear. The provided test sequences are cut short before the failure event at RUL values, which should be estimated by RUL prediction. Each dataset sample represents one flight-cycle and includes 26 multivariate features. As all training sequences reach RUL=0, the target RUL labels are calculated from the flight cycle vector t as RUL=abs(max(t)-t) and subsequently added to the available set of features. All dataset features are listed in Table 3.

A colour-coded correlation matrix of all features in C-MAPSS FD001 is shown in Figure 7. Some features consist of constant values without a meaningful representation of correlation values and are therefore omitted from Figure 7 as blank entries. Certain features show a clear positive or negative correlation with the target RUL variable (feature 26), which indicates their potential value in RUL prediction. However, the correlation matrix also shows that several features are highly correlated among each other, potentially providing redundant information.

1) MODEL TRAINING

In case of the healthy and faulty condition deviation models (HI generation stage), the first and last n_{fh} samples (representing health and fault condition data, respectively) are extracted from each sequence of the full training set and used for training. Prior to this, the following approach is used for datasets FD002 and FD004 to consider the impact of different operating conditions on degradation trend signatures: The samples of each training set are first subdivided into 6 subsets based on the 6 operating conditions (see Table 2). A separate condition deviation model is then calculated and used for each distinct operating condition.

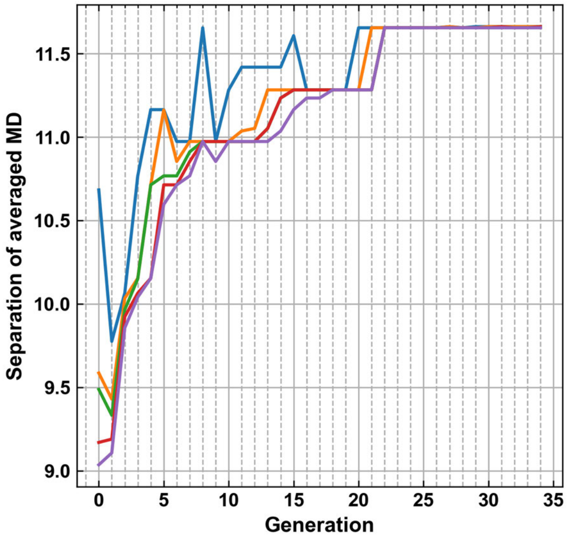

FIGURE 8. MD-based GA optimisation of 5 best scoring feature subset candidates per generation, C-MAPSS FD001 training data.

According to the procedure outlined in Section IV-A-3), a GA is executed on C-MAPSS FD001 for 100 generations. As seen in Figure 8, the GA converges to a maximum average MD separation after approximately 30 generations, providing a final list of highest scoring feature subsets.

As a result of the optimisation, the features 13, 15, 19 of the C-MAPSS dataset are chosen in this study for MD calculation. While separate optimisations for the FD002 - FD004 datasets and for individual operating conditions could lead to further performance improvements, the stated MD input features identified from FD001 are used throughout all C-MAPSS dataset for consistency and were found to provide satisfactory results.

Figure 9 shows the MD applied to 7 exemplary run-to-failure training sequences. The merit of the MD becomes visible compared to one of the relevant raw input features: in the two rightmost sequences, the raw feature 13 does not show a clear degradation trend towards the respective failure cycle, whereas the MD of the multivariate input from features 13, 15, 19 shows a monotonic increase on average with a maximum MD at the lifecycle end.

For the initial model training and search of optimised hyperparameters of the prediction model, a reduced set of $N_{tr,red}$ =20 (out of N_{tr} =100) complete run-to-failure sequences is assumed to be available.

A randomised grid search of hyperparameters is performed by subdividing the training data of the $N_{tr,red}$ =20

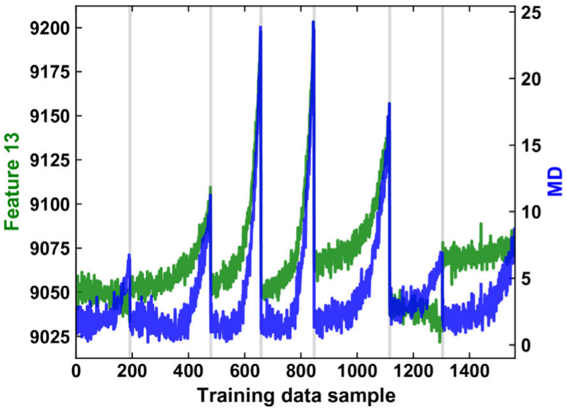

FIGURE 9. Progression of raw feature 13 and MD (combining features 13, 15, 19) for 7 complete C-MAPSS FD001 training sequences.

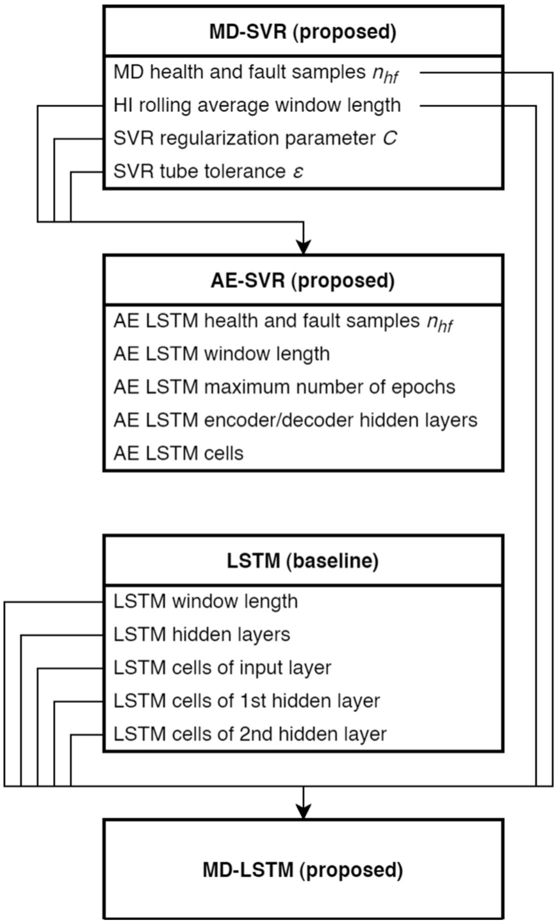
MD-SVR (proposed)
MD health and fault samples n_{hf}
HI rolling average window length
SVR regularization parameter C
SVR tube tolerance ε
AE-SVR (proposed)
AE LSTM health and fault samples n_{hf}
AE LSTM window length
AE LSTM maximum number of epochs
AE LSTM encoder/decoder hidden layers
AE LSTM cells
LSTM (baseline)
LSTM window length
LSTM hidden layers
LSTM cells of input layer
LSTM cells of 1st hidden layer
LSTM cells of 2nd hidden layer
MD-LSTM (proposed)

FIGURE 10. Usage of optimised hyperparameters between implemented methods.

complete sequences into 18 training sequences and 2 validation sequences for scoring of hyperparameter sets. Each evaluated set of hyperparameters is repeatedly trained and scored 10 times, whereafter the mean validation scores are used to determine the final hyperparameter values.

To reduce the dimensionality of the hyperparameter search space and computing time, a subset of hyperparameters is optimised for each of the implemented models (Table 1). That is, related hyperparameter optimisation results from a preceding method optimisation are retained for other methods. This is illustrated with Figure 10, where each method (represented by a block) lists its optimised hyperparameters. Arrows indicate which hyperparameter results are taken over from one optimisation to a different one. The specific search space per hyperparameter and final optimised values are given in Table 4. As the hyperparameters shown in the last column of Table 4 are used for the implemented models, these values simultaneously provide the definition of the used NN structures, which are also visualised in Figure 11.

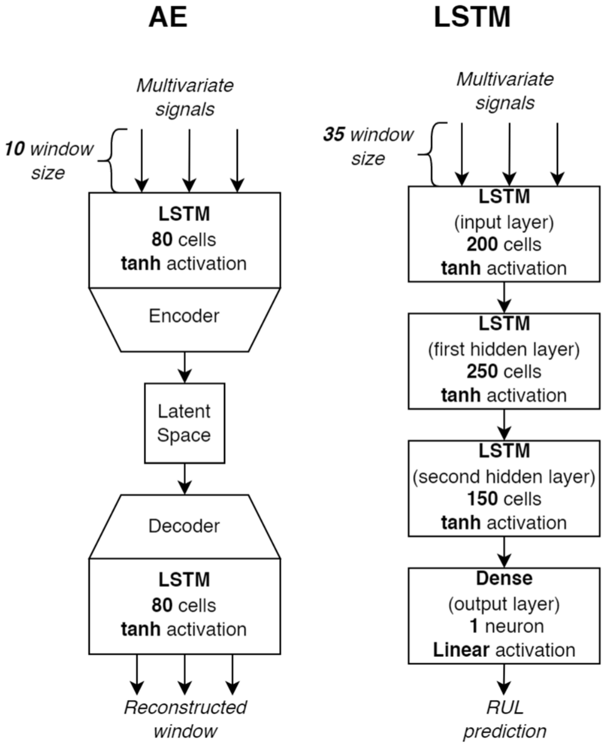

FIGURE 11. Used NN structures: AE condition deviation model (left); LSTM RUL prediction model (right).

As shown in Figure 10, the proposed MD-LSTM method combines optimised parameters for HI-generation from MD-SVR and the LSTM hyperparameters optimised for the baseline LSTM method. By this approach, the direct impact of the proposed feature set reduction on LSTM model performance with limited data can be observed. It should be noted that this procedure favours the baseline LSTM method, because a hyperparameter optimisation specifically for the proposed MD-LSTM method could further improve its performance. For the introduced HI-generation step (in methods MD-SVR, AE-SVR, MD-LSTM), n_{hf} first and last samples from each sequence of the full training set are extracted and used.

2) HEALTH INDEX (HI) CALCULATION

Corresponding to the procedure detailed in Section IV-A, HI progressions are created by MD on one hand and by

TABLE 4. C-MAPSS FD001 training hyperparameter search space and final optimisation values for implemented RUL prediction methods.

Parameter	Search space	Final (used) value
MD-SVR (proposed)		
MD health and fault samples n_{hf}	{5, 15, 25, 35, 45}	5
HI rolling average window length	{5, 10, 15, 20}	10
SVR regularization parameter C	{50, 150, 250, 350, 450, 550}	150
SVR tube tolerance ε	{10, 30, 50, 70}	10
AE-SVR (proposed)		
AE LSTM health and fault samples n_{hf}	{25, 50, 75}	25
AE LSTM window length	{5, 10, 15, 20}	10
AE LSTM maximum number of epochs	{10, 110, 210, 310}	10
AE LSTM encoder/decoder hidden layers	{0, 1, 2}	0
AE LSTM cells	{5, 20, 35, 50, 65, 80, 95, 110}	80
LSTM (baseline)		
LSTM window length	{5, 20, 35, 50, 65}	35
LSTM hidden layers	{0, 1, 2, 3, 4}	2
LSTM cells of input layer	{50, 100, 150, 200, 250, 300}	200
LSTM cells of 1st hidden layer	{50, 100, 150, 200, 250, 300}	250
LSTM cells of 2nd hidden layer	{50, 100, 150, 200, 250, 300}	150

LSTM AE on the other hand. Results from both methods are presented and discussed below.

a: MAHALANOBIS DISTANCE (MD)-BASED HI

Using (10), MD-based models for health condition deviation d_h and fault condition deviation d_f are conditioned. The output MD of both models for a complete run-to-failure validation sequence is plotted in Figure 12.

As per the procedure described in Section IV-B, the HI shown in Figure 13 is determined from the condition deviation functions shown in Figure 12. In addition, the flight cycle with a starting degradation is determined according to Section IV-B-1) and is also highlighted in Figure 13.

b: LSTM AUTOENCODER (AE)-BASED HI

Outputs of the AE-based condition deviation fault and health models are shown in Figure 14 and Figure 15, respectively. In addition to early stopping on model convergence, the number of epochs is used as an optimised hyperparameter (as shown under AE-SVR in Table 4). It was observed that limiting the number of epochs aids in preventing the adaptation of the model to unseen data, as the purpose of the AE is to provide an accurate reconstruction of the reference (healthy or faulty) condition only. Both figures show that the AE reconstruction aligns closely to the reference state, while producing a higher reconstruction error at the opposing state. As a result, the reconstruction error increases towards to faulty condition in Figure 14 (health model) and decreases towards the faulty condition in Figure 15 (fault model), as desired.

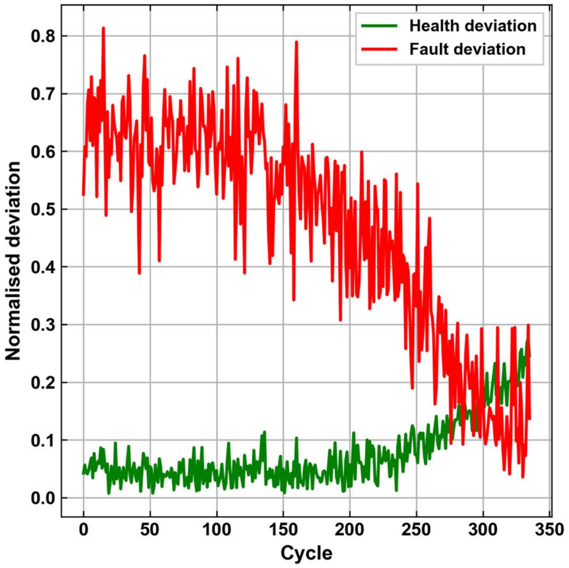

FIGURE 12. Overlaid MD-based health deviation d_h and fault deviation d_f for an exemplary run-to-failure sequence.

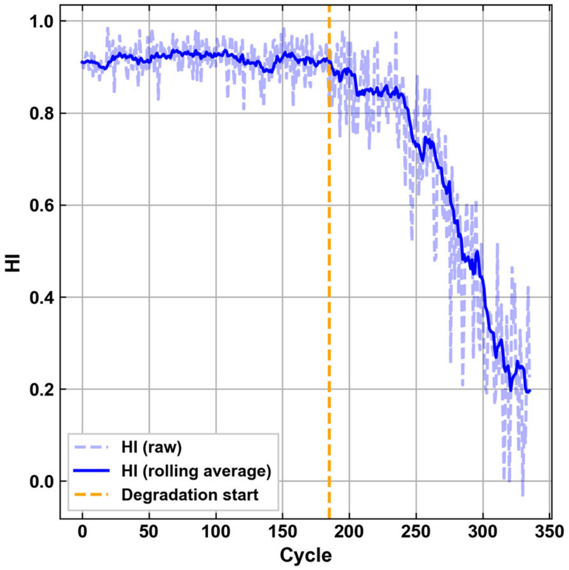

FIGURE 13. HI resulting from MD-based health and fault deviations in Figure 12 after scaling and rolling average.

The resulting AE-based health and fault deviation functions are overlaid in Figure 16 and the corresponding HI is shown in Figure 17. Compared to the MD-based deviation and HI functions, the AE-based alternatives show a clearer distinction between the health and fault conditions.

This is visible in the clearer crossing of the AE-based condition deviation functions (comparing Figure 12 and Figure 16) and a lower AE-based HI value at the final degradation cycle (comparing Figure 13 and Figure 17). However,

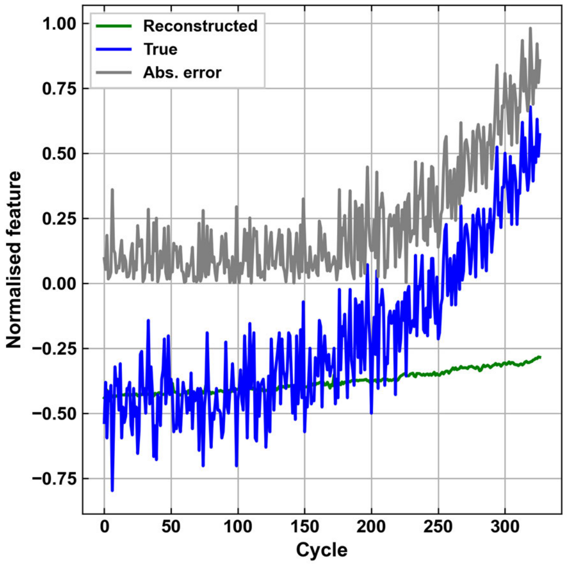

FIGURE 14. AE reconstruction for healthy condition data of feature 13.

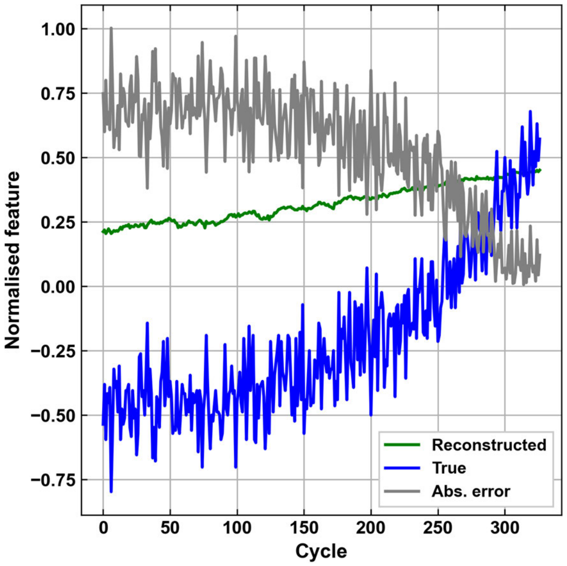

FIGURE 15. AE reconstruction for faulty condition data of feature 13.

it is important to note that the AE-based method requires substantially more computing resources compared to the MD-based counterpart.

3) RUL PREDICTION RESULTS

Based on condition deviations and HI presented in the previous section, RUL prediction is performed with various methods presented previously in Section IV-C. Implemented method configurations are individually listed in Table 1.

It is reasonable to focus on the pairing of the SVR methods (MD-SVR and AE-SVR) for one comparison, and the LSTM-based prediction methods (baseline LSTM and proposed MD-LSTM) for another comparison. The reason for this is that both SVR methods only differ in the used HI-generation approach (either AE or MD-based) and therefore allow a direct comparison of those HI-generation methods regarding their impact on RUL prediction performance. On the other hand, the LSTM RUL prediction methods give insight into the difference of the proposed features (condition deviation and HI) in comparison to a direct prediction from raw dataset features. The Least Squares prediction method (MD-QPoly) is treated separately, as it is the only presented method, which does not require any training sequences and is independent thereof.

Figure 18 and Figure 19 show RUL prediction results on 5 validation sequences from training with a reduced number

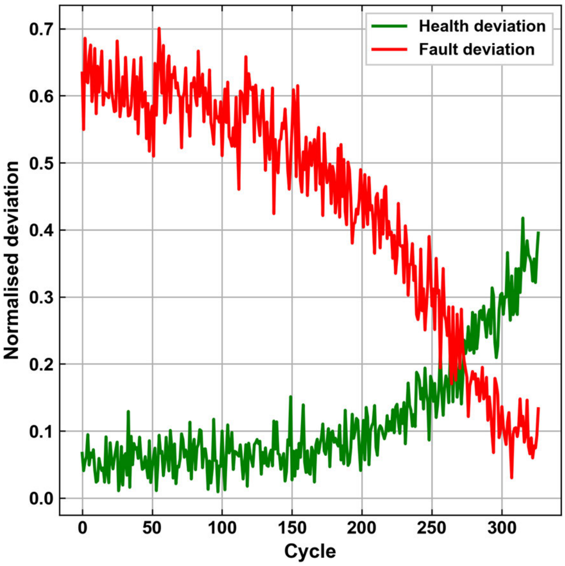

FIGURE 16. Overlaid AE-based health deviation d_h and fault deviation d_f for an exemplary run-to-failure sequence.

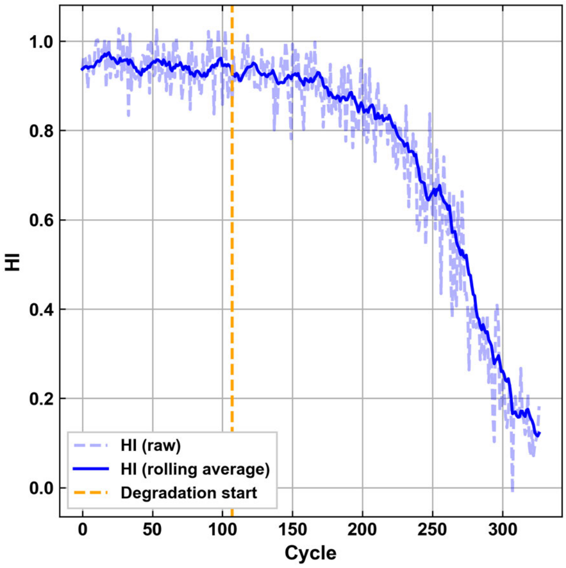

FIGURE 17. HI resulting from AE-based health and fault deviations in Figure 16 after scaling and rolling average.

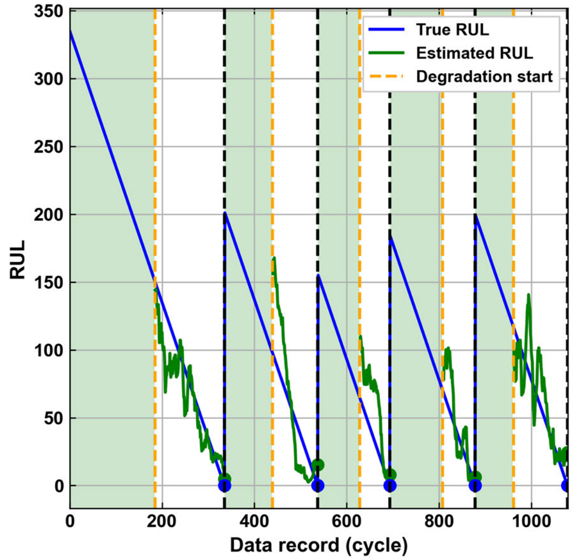

FIGURE 18. MD-SVR predictions of RUL based on 20 training sequences for 5 validation sequences (i.e., engines).

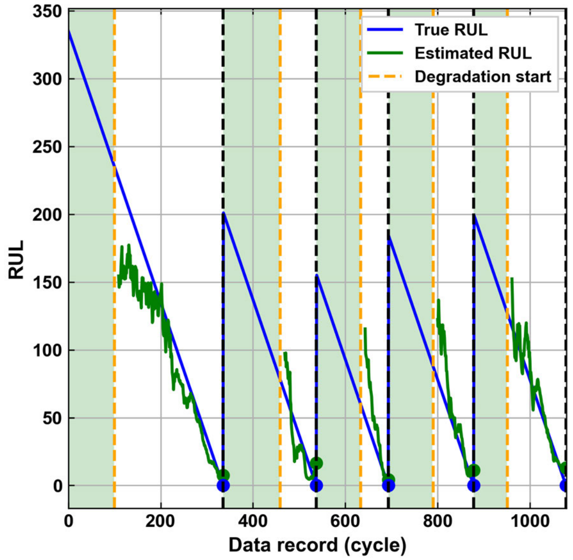

FIGURE 19. AE-SVR predictions of RUL based on 20 training sequences for 5 validation sequences (i.e., engines).

of $N_{tr,red}$ =20 sequences, for MD-SVR and AE-SVR, respectively. The merits of degradation detection become apparent as it provides an additional binary assessment of each machine's condition and indicates whether a meaningful RUL prediction can be expected. The determined pre-degradation (healthy) machine condition is highlighted with green in both graphs. Except for the second engine sequence in Figure 18 and Figure 19, the AE-SVR shows a similar or earlier degradation detection compared to MD-SVR. This indicates a clearer degradation trend in favour of the AE-based HI method, similar to the observations discussed in the previous section.

Figure 20 compares the RUL prediction performance on the C-MAPSS FD001 dataset for all methods listed in Table 1. To estimate the impact of a limited number of training sequences, comparisons are conducted for an increasing number of training sequences $N_{tr,red}$ plotted on the horizontal axis of Figure 20 as a percentage of the full training set with N_{tr} =100. The root mean square error (RMSE) is calculated according to (15) between the predicted final RUL values RUL_{pred} and true final RUL values RUL_{true} of the test dataset with N_{te} =100 sequences.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N_{te}} \left(\text{RUL}_{pred,i} - \text{RUL}_{true,i}\right)^2}{N_{te}}} \qquad (15)$$

For each box-and-whisker entry in Figure 20, the specified method (indicated by the legend) is trained 30 times with the specified number of training sequences (indicated by the horizontal axis) drawn randomly from the full set of N_{tr} =100 training sequences. Each of the trained models is evaluated on the full set of N_{te} =100 test sequences, providing one RMSE value. The repetition provides a more reliable performance assessment and leads to 30 RMSE test scores per test condition, which make up each individual entry in Figure 20. It should be also noted that increasing intervals of values on the horizontal axis are used, i.e., percentage point intervals of 5 between 0% - 20%; 15 between 20% - 50%; 25 between 50% - 100%. This is done for a more detailed comparison in the low data range, because the RMSE values show a higher gradient in this range (left part of Figure 20) compared to results from high data usage (right part of Figure 20).

P-values between both SVR methods (AE-SVR, MD-SVR) and between both LSTM methods (LSTM, MD-LSTM) are indicated in Figure 20. Low p-values below 0.05 are obtained on most comparisons, suggesting statistical significance. However, several result distributions (e.g., MD-SVR, AE-SVR, LSTM at $N_{tr,red}$ =15 training sequences) are skewed, as can be seen by the unsymmetrical inter-quartile ranges and whiskers. As such, the underlying sample distributions deviate from the normal distribution and p-values should be interpreted with care in those cases.

The mostly observed positive skewness (with a long tail towards higher RMSE values) is likely related to the influence of a few irregular outlier sequences in the full training set (with N_{tr} =100 sequences), which are randomly drawn into the reduced subset of training sequences (e.g., with $N_{tr,red}$ =15 sequences). Affected training runs result in a higher test RMSE. As the number of training sequences is increased (towards the right side of Figure 20), the proportion, and thus impact, of an individual outlier sequence is mitigated by remaining training sequences. Therefore, the resulting RMSE scores reach a rather symmetrical distribution starting from $N_{tr,red}$ =35 training sequences as shown by Figure 20.

A general trend in Figure 20 shows that the RUL prediction methods benefit from greater numbers of training sequences both in terms of increased average prediction accuracies as well as a reduced variance of the prediction results. This highlights the importance of the presented methodology, employing 30 repeated training/testing runs per experiment.

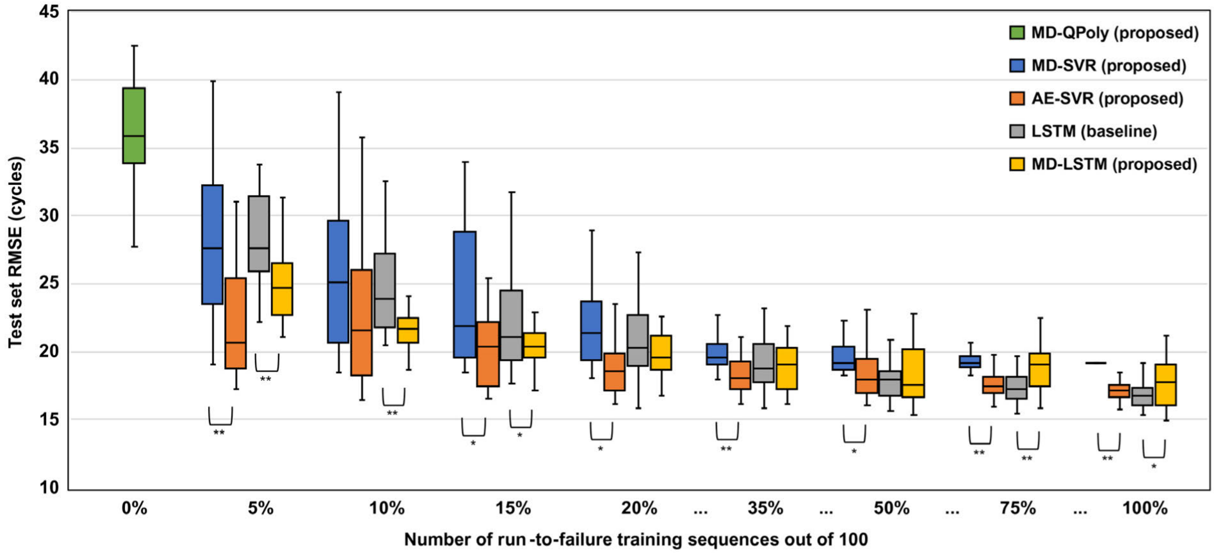

FIGURE 20. Prediction performance on the C-MAPSS FD001 test dataset over 30 runs per each box and whisker (RMSE, lower is better) of MD-QPoly, MD-SVR, AE-SVR, LSTM, MD-LSTM ($*p<0.05$, $**p<0.01$).

This is especially relevant at training cases with reduced numbers of used training sequences, which is the focus of this paper. However, Figure 20 also shows that even at full (i.e., 100%) training set utilisation, there is still a significant prediction variance on the NN-based compared methods. At the same time, most existing publications using the C-MAPSS dataset (e.g., [30], [62], [67]) report performance based on a single test evaluation only and thus do not account for the variance due to the stochastic nature of the presented NN-based methods. This omits relevant information on prediction consistency and likely leads to biased results since a best run might be reported, which is not representative of an average expected performance in real-word applications.

MD-Qpoly is the only method that functions independently from training sequences and only requires fault and health condition data for conditioning of the MD-based fault and health models. This feature allows MD-Qpoly to be utilised without any training data containing run-to-failure sequences but also leads to a wide dispersion of the RMSE range and highest overall RMSE values compared to other methods with available training sequences in Figure 20. This method can be further extended by expert knowledge or run-to-failure training data by employing the constrained LS method. For example, the range of estimated polynomial coefficients from run-to-failure training sequences can be stored to set coefficient boundaries for polynomial fits of HI sequences during the test or operational stage. The RUL is then predicted on unseen (i.e., test) data with the function constrains determined from training data.

At the cost of substantially higher computing demand, the previous observations in favour of AE-based HI are also reflected in the RUL prediction performance, as AE-SVR shows consistently lower overall RMSE values compared to MD-SVR in Figure 20. Nevertheless, MD-based condition and HI features used in the MD-LSTM method led to improved prediction performance of the proposed MD-LSTM method at limited numbers of training sequences below 20% compared to the baseline LSTM approach. This demonstrates that the proposed method improves the generalisation ability of the LSTM-based prediction when a vastly limited number of training sequences is available. In addition, Figure 20 shows that the proposed MD-LSTM method also results in the lowest RMSE variance in the low range of training sequences among the compared methods.

At a larger training set above 75%, the LSTM model can take advantage of the higher complexity of raw features, outperforming MD-LSTM by a small margin and performing similar to AE-SVR. These results suggest that LSTM-based RUL prediction would likely further benefit from AE-based HI generation. Such a method (i.e., AE-LSTM) should be considered for future work, as it has the potential to further outperform both MD-LSTM and the baseline LSTM method across the full range of reduced training sequences.

Figure 21 shows the training durations for individual methods over increasing percentages of used training sequences on the horizontal axis. On the vertical axis, average durations (over 30 runs) are provided in percentages relative to the traditional approach, which is represented by the baseline LSTM method utilising the full training set with 100% of training sequences. Model training is performed on an AMD Ryzen Threadripper 3990X 64-core CPU with 256 GB RAM. NN-based methods (AE-SVR, LSTM, MD-LSTM) are trained on an NVIDIA RTX A6000 GPU with a batch size of 512 until model convergence (i.e., until no improvement in validation loss over the past 10 epochs is gained).

A reduction of the number of training sequences is shown to have a clear impact, reducing training durations across all methods. NN training accounts for the majority of the

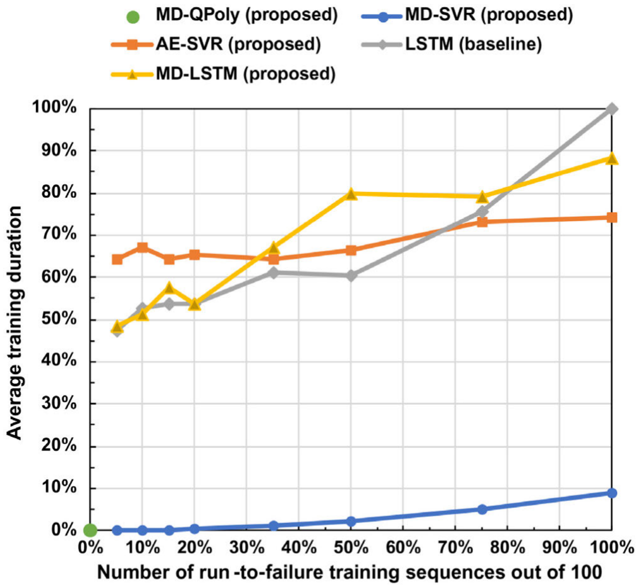

FIGURE 21. Average training durations over 30 runs per each data point of compared methods (C-MAPSS FD001), relative to LSTM (baseline) at full training set size of 100%.

observed time requirements, while the conditioning of MD and SVR models are substantially less demanding. As such, MD-SVR provides a 91% training time reduction for the full training set at a 14% increase in average RUL RMSE compared to the LSTM baseline. Despite a reduced number of NN inputs in MD-LSTM compared to LSTM, both methods show a similar progression of training time. The AE-SVR duration curve has a lower slope compared to LSTM and MD-LSTM due to the different NN architecture inherent to the AE NN model.

Concluding from the comparison on the C-MAPSS FD001 dataset, the proposed MD-LSTM method can be considered as the overall most beneficial for RUL prediction with a limited number of training sequences as it provides the lowest overall RMSE scores (along with AE-SVR) and lowest RMSE variance in the presented case. At the same time, the training time demand of MD-LSTM represents a compromise between NN-free methods (MD-QPoly and MD-SVR) and the AE-SVR method.

Based on this result, the proposed MD-LSTM method is compared further on the datasets C-MAPSS FD002 – FD004 in Figure 22 – Figure 24. Supplementary to the box and whisker plots of the RUL prediction results, the average and standard deviation values of prediction RMSE from all C-MAPSS datasets FD001 – FD004 are summarised in numerical form in Table 5.

It is visible that the overall performance of both the baseline LSTM and the proposed MD-LSTM methods is decreased when applied to C-MAPSS FD002 – FD004 (leading to higher RMSE values in Figure 22 – Figure 24) compared to the FD001 dataset (Figure 20). This is explained by a higher data complexity due to an increased number of

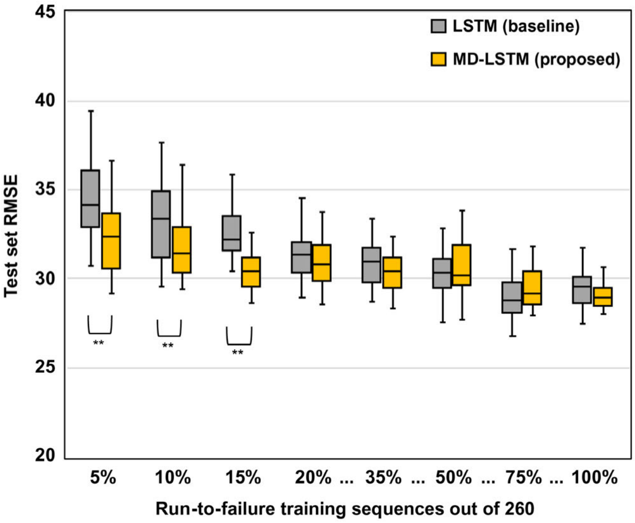

FIGURE 22. Prediction performance on the C-MAPSS FD002 test dataset over 30 runs per each box and whisker (RMSE, lower is better) of LSTM and MD-LSTM ($**p<0.01$).

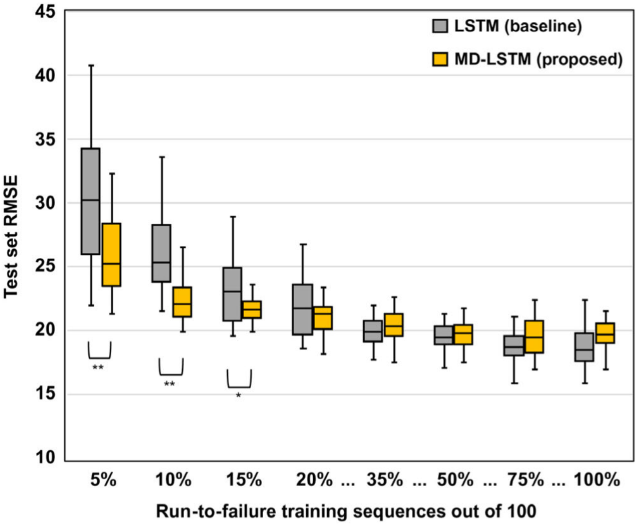

FIGURE 23. Prediction performance on the C-MAPSS FD003 test dataset over 30 runs per each box and whisker (RMSE, lower is better) of LSTM and MD-LSTM ($*p<0.05$, $**p<0.01$).

operating and/or failure conditions in the FD002 – FD004 datasets as shown in Table 2. Hence, FD004 shows the highest absolute RMSE values both for the baseline and proposed RUL prediction methods in comparison to datasets FD001 – FD003 (see Table 5). At the same time, FD004 shows the clearest separation between the performance scores of the baseline LSTM and the proposed MD-LSTM methods in Figure 24.

As in previous cases, the greatest difference in favour of the proposed MD-LSTM method is visible at highly reduced numbers of available training sequences in Figure 24. This is also reflected by the values in Table 5, where MD-LSTM shows an average RMSE of 32.8 at a training sequence reduction to 5%. This corresponds to a 19.2% prediction

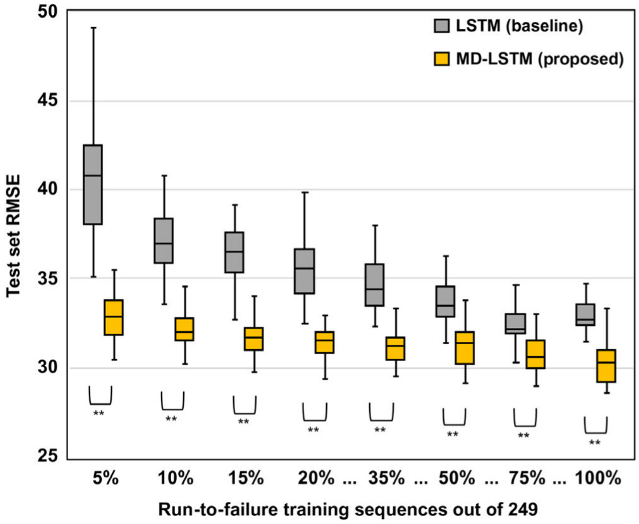

FIGURE 24. Prediction performance on the C-MAPSS FD004 test dataset over 30 runs per each box and whisker (RMSE, lower is better) of LSTM and MD-LSTM (**p<0.01).

improvement in relation to the baseline LSTM method with an average RMSE of 40.6. Additionally, MD-LSTM outperforms the baseline LSTM method at the whole range of reduced FD004 training sequences, including the full training set, i.e., at 100% of used training sequences).

This indicates that the presented approach significantly reduces data complexity of the raw FD004 dataset features while preserving relevant degradation information, which is represented by the generated condition deviation and HI functions. It should be also noted, that while the full FD004 training set (with N_{tr} =249) is larger than e.g., FD001 (with N_{tr} =100), it has the lowest ratio of training sequences per operating and fault conditions (i.e., 249/8 = 31.125) among all C-MAPSS datasets. Hence, the baseline LSTM method would likely benefit from a greater number of FD004 training sequences beyond 100%. This is also supported by the rather linear trend of decreasing RMSE values displayed by LSTM (baseline) in Figure 24, which seems further away from reaching convergence at 100% compared to other datasets in Figure 20, Figure 22, and Figure 23. At the same time, the issue of great training data demand is alleviated by the proposed MD-LSTM method, providing improved generalisability, which explains the clear performance increase especially on FD004 (Figure 24) in favour of the proposed MD-LSTM method.

B. VIBRATION DATA OF BEARING DEGRADATION

A bearing degradation dataset [10] was presented in frame of the IEEE PHM 2012 Data Challenge. The dataset was generated by the FEMTO-ST institute utilising a mechanical test rig "PROGNOSTIA". 3 load conditions were applied by the test rig, promoting an accelerated degradation of the test bearings. A total of 6 run-to-failure sequences are provided in the training set and 11 sequences in the test set. Degradation sequences of the test set are truncated to assess the RUL prediction performance of models based on the estimation error at the end of those sequences. The dataset contains accelerometer readings, with time series of acceleration measured in the vertical and horizontal axes of the test bearings. Additional temperature measurements are provided in the dataset as well, but only for 4 out of 6 training sequences and 7 out of 11 test sequences. For consistency, only the accelerometer channels are considered in this work. The accelerometer measurements are divided into segments of 1/10s duration, each obtained in intervals of 10s at a sampling frequency of 25.6kHz.

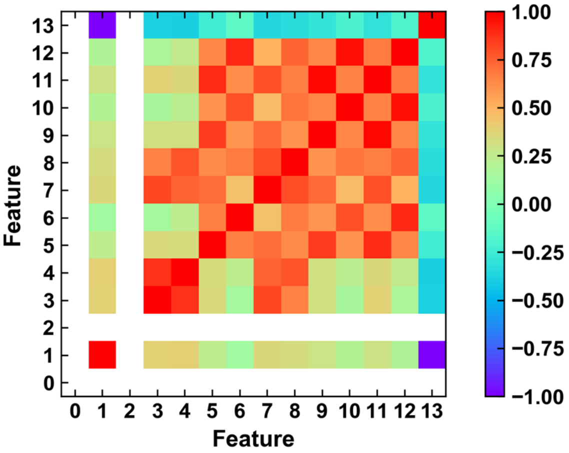

FIGURE 25. Correlation matrix of FEMTO training dataset features, constant features displayed in white.

Statistical features, commonly used for vibration condition monitoring [68], are calculated from both acceleration signals and arranged into a feature matrix. One segment of continuous 1/10s signals is thereby processed to a single statistical value sample. The resulting feature set is described in Table 6 and serves as the foundation of the following experiments. The dataset contains run-to-failure sequences, each obtained at one of 3 operating conditions. The operating condition is denoted by a number 1 (4000 N, 1800 rpm), 2 (4200 N, 1650 rpm), or 3 (5000 N, 1500 rpm), which is included as feature 2 as shown in Table 6.

Figure 25 shows the correlation matrix of all features of the training set. Feature 2 (the condition number) shows a blank field, due to its constant value per run-to-failure sequence. Most features show a high positive correlation with each other. As expected, the RUL value (feature 13) is negatively correlated to the sequence time (feature 1).

Analogous to the procedure in the C-MAPSS dataset, features of the FEMTO dataset are processed by GA optimisation (Section IV-A-3) to determine a feature subset providing maximum MD separation. Convergence of the 5 best scoring subsets per generation is shown in Figure 26. The features 4, 7, 8, 9, 12 are determined by this procedure and chosen as inputs for the MD calculation.

Figure 27 shows a plot of both the acceleration RMS in vertical direction (feature 4) and an overlay of the MD-based

TABLE 5. Prediction performance on C-MAPSS test datasets (RMSE, lower is better).

	Relative number of run-to-failure training sequences															
	5%		10%		15%		20%		35%		50%		75%		100%	
Method	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.
C-MAPSS FD001																
MD-QPoly (proposed)	36.1*	3.9*														
MD-SVR (proposed)	28.3	6.2	26.5	7.6	26.5	10.2	22.0	3.3	20.9	4.5	20.0	3.1	19.3	**0.6**	19.2	**<0.1**
AE-SVR (proposed)	**22.2**	4.3	23.8	7.5	22.2	7.2	**19.6**	4.8	**18.3**	**1.6**	18.5	1.9	17.7	1.3	17.2	0.6
LSTM (baseline)	28.1	3.5	24.6	3.1	22.0	3.3	20.8	2.8	19.0	1.8	**17.8**	**1.3**	**17.4**	1.1	**16.8**	1.0
MD-LSTM (proposed)	24.9	**2.7**	**21.7**	**2.1**	**20.5**	**1.7**	19.7	**1.5**	19.1	1.9	18.2	2.1	18.8	1.6	17.7	1.6
C-MAPSS FD002																
LSTM (baseline)	34.4	2.1	33.2	2.1	32.7	1.6	31.4	**1.3**	31.0	1.5	**30.3**	**1.3**	**29.1**	**1.3**	29.5	1.1
MD-LSTM (proposed)	**32.2**	**1.9**	**31.8**	**1.9**	**30.6**	**1.4**	**30.9**	1.6	**30.4**	**1.2**	31.0	2.8	29.6	1.4	**29.2**	**1.0**
C-MAPSS FD003																
LSTM (baseline)	30.1	4.9	26.0	3.1	23.3	2.9	21.6	2.1	**19.9**	**1.1**	**19.5**	1.2	**18.7**	**1.2**	**18.6**	1.7
MD-LSTM (proposed)	**25.7**	**2.9**	**22.7**	**2.3**	**21.8**	**1.2**	**21.2**	**1.6**	20.3	1.3	19.7	**1.1**	19.5	1.5	19.6	**1.2**
C-MAPSS FD004																
LSTM (baseline)	40.6	3.1	37.1	1.6	36.4	1.7	35.7	2.1	34.9	1.8	33.7	**1.3**	32.5	**0.9**	33.0	**1.1**
MD-LSTM (proposed)	**32.8**	**1.3**	**32.1**	**1.2**	**31.7**	**0.9**	**31.5**	**1.2**	**31.2**	**0.9**	**31.5**	1.8	**30.8**	1.3	**30.3**	1.2

Avg.: average; Std.: standard deviation; *Results independent of the number of used training sequences

TABLE 6. Features of the FEMTO dataset.

Feat. ID	Description	Equation
0	Run number	N/A
1	Time (minutes)	N/A
2	Operating condition number	N/A
3	RMS (H)	$\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$, where x is the data sample and N is the number of samples in the processed window
4	RMS (V)	
5	Kurtosis (H)	$\sqrt{\frac{\sum_{i=1}^{N}(x_i-\bar{x})^4}{(N-1)\sigma^4}}$, where $\bar{x}$ is the mean and σ is the standard deviation
6	Kurtosis (V)	
7	Absolute maximum (H)	$\max(\lvert x\rvert)$
8	Absolute maximum (V)	
9	Crest factor (H)	$\frac{\max(\lvert x\rvert)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}}$
10	Crest factor (V)	
11	Impulse factor (H)	$\frac{\max(\lvert x\rvert)}{\frac{1}{N}\sum_{i=1}^{N}\lvert x_i\rvert}$
12	Impulse factor (V)	
13	RUL (target output variable)	$\lvert t-\max(t)\rvert$, where t is the time vector

H: Horizontal direction accelerometer; V: Vertical direction accelerometer

health condition deviation. The full training set of 6 run-to-failure sequences is covered by Figure 27. While the resemblance of feature 4 is visible in the MD, the MD shows a clearer slope between the start and end of each training sequence, which is a desired outcome. Figure 27 also illustrates challenging characteristics of the dataset: the shown sequences have a wide range of inconsistent durations, feature values and noise levels.

TABLE 7. FEMTO training hyperparameter search space and final optimisation values for LSTM.

Parameter	Search space	Final value
LSTM window length	{5, 35, 65, 95, 125}	95
LSTM hidden layers	{0, 1, 2, 3, 4}	2
LSTM cells of input layer	{50, 100, 150, 200, 250, 300}	250
LSTM cells of 1st hidden layer	{50, 100, 150, 200, 250, 300}	200
LSTM cells of 2nd hidden layer	{50, 100, 150, 200, 250, 300}	250

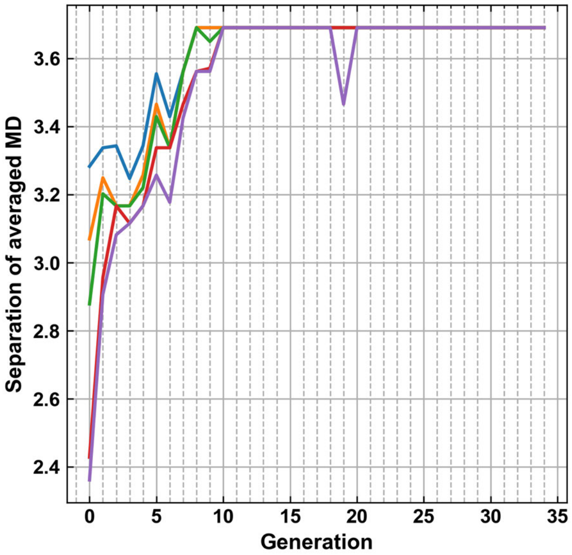

FIGURE 26. MD-based GA optimisation of 5 best scoring feature subset candidates per generation, FEMTO training data.

From the observations on the C-MAPSS results, it was found that the MD-LSTM method combines satisfying

TABLE 8. Prediction performance on the FEMTO test dataset (RMSE, lower is better).

	Relative number of run-to-failure training sequences											
	16.7%		33.3%		50.0%		66.7%		83.3%		100.0%	
Method	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.
LSTM (baseline)	**70.4**	**5.9**	**66.1**	**6.6**	**63.2**	**7.6**	63.7	**7.0**	58.7	**5.5**	60.5	5.2
MD-LSTM (proposed)	73.7	10.2	69.6	8.8	63.4	10.8	**60.5**	9.4	**58.0**	6.9	**54.6**	5.2

Avg.: average; Std.: standard deviation

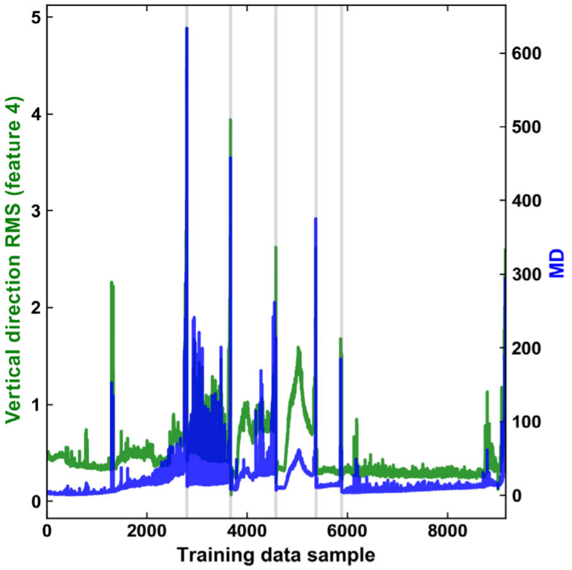

FIGURE 27. Progression of raw feature 4 and MD (combining features 4, 7, 8, 9, 12) for 6 complete training sequences.

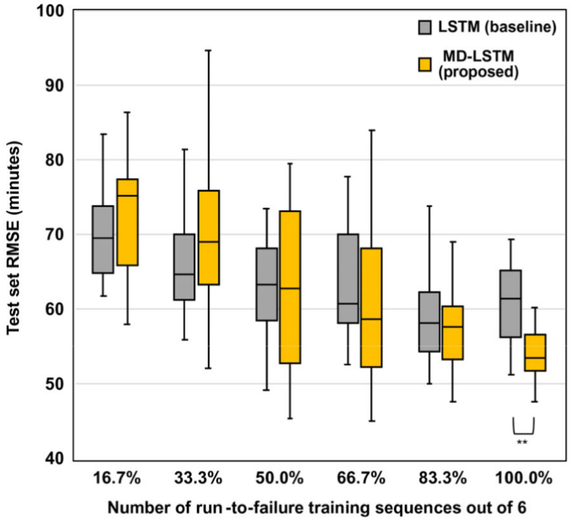

FIGURE 28. Prediction performance on the FEMTO test dataset over 30 runs per each box and whisker (RMSE, lower is better) of LSTM, MD-LSTM (**p<0.01).

characteristics in terms of prediction variance, and prediction accuracy at reduced numbers of training sequences. The MD-LSTM method is therefore applied to the FEMTO dataset and compared to the baseline LSTM method in the following.

A similar procedure to the one in Section V-A is used for the hyperparameter search. Table 7 shows the hyperparameter search space, which is optimised by a random grid search. A split of 4 training and 2 validation sequences is used for the hyperparameter optimisation. The final hyperparameter values are determined from the best validation score of the baseline LSTM method.

Afterwards, the final model architecture is applied to both the baseline LSTM and the proposed MD-LSTM methods. Model training is performed on both methods with an increasing number of training sequences (between 1 and 6) and each time evaluated on the full test set with 30 repetitions per model training and evaluation run.

Similar to the C-MAPSS results (Figure 20), an overall trend of decreasing RUL prediction RMSE is visible over an increasing number of used training sequences in Figure 28. However, the differences between the compared the baseline LSTM and the proposed MD-LSTM methods are less clear here. The results show similar performance with a wider RUL dispersion of MD-LSTM up to 4 out of 6 (i.e., 66.7%) training sequences. At 6 (i.e., 100%) training sequences, however, MD-LSTM outperforms the baseline LSTM method as shown in Figure 28.

Several factors likely contribute to a less conclusive method comparison by the FEMTO dataset compared to C-MAPSS. In comparison to the C-MAPSS dataset, a smaller range of possible training sequences (6 in contrast to 100) is available in the FEMTO dataset for evaluation of an increasing training sequence count. The number of used features in the FEMTO dataset (Table 6) is smaller than the number of features in the C-MAPSS dataset (Table 3), so a greater ratio of feature reduction is achieved by HI generation on the C-MAPSS dataset. Moreover, the C-MAPSS dataset is based on 21 sensors, likely encapsulating more relevant information for RUL estimation compared to FEMTO features, which stem from only two physical quantities (vibration readings in two directions). This is also supported by Figure 25, which shows that most derived features of the FEMTO dataset are mutually correlated. Introduction of additional features and tuning of pre-processing parameters (such as a window size beyond 1/10s) could provide a more informative dataset for RUL prediction.

VI. CONCLUSION AND FUTURE WORK

This paper introduced a methodology for RUL prediction utilising a limited number of run-to-failure sequences, as well as binary health and fault condition data. The approach can be considered a feature reduction and data fusion method, aimed at improving generalisation where few training sequences or limited computing resources are available.

Several configurations of the method were implemented and compared to the baseline LSTM method. Using the open C-MAPSS turbofan degradation dataset, it was shown that AE-based HI-generation provides a higher RUL prediction accuracy compared to MD-based HI generation, albeit at a substantially higher computing effort. Conversely, it was found that the MD-based approach is especially suited for conditions where quick model training is required, or computing power is limited. Using the full training set, a substantial training time reduction to 8.9% of the baseline method was achieved by MD-SVR (Figure 21). When the number of training sequences is reduced to 5%, the training duration of both the baseline LSTM method and the proposed MD-LSTM is 47% - 48%, while the MD-SVR training duration reduced further to 0.04%. This work therefore has implications on considerations for green AI [24], contributing to the development of more efficient and environmentally friendly algorithms.

The choice between the application of the proposed method through either the AE or MD-based approach therefore depends on the prioritisation between prediction accuracy or computing time requirements. Nevertheless, the introduced MD-based LSTM prediction (MD-LSTM) was able to consistently outperform the baseline LSTM approach for training cases using under 20% of training sequences of the full training set. For example, when comparing the prediction RMSE at a reduction of the available number of training sequences to 5%, the greatest impact was seen on the C-MAPSS FD004 dataset, where the proposed method showed an average prediction improvement by 19.2% relative to the baseline method. The proposed method therefore demonstrated an improvement of generalisation from a limited number of training sequences.

A further RUL prediction approach based on Least Squares regression was presented. This method is applicable when no training sequences are available and has the lowest computing demand. On the flipside, it assumes a polynomial (e.g., quadratic) progression of the HI and showed the highest prediction error among the compared methods.

An additional comparison of baseline LSTM and MD-LSTM on the FEMTO dataset of bearing degradation was less conclusive. The results showed largely similar prediction performance with greater variance in the MD-LSTM results but also partially improved prediction accuracy in favour of MD-LSTM. The reasons are likely related to characteristics of the dataset, such as its initially limited size and a high level of inconsistency between training sequences.

From the obtained findings and literature, the following areas are highlighted as current challenges and future work in the research of prognostics and health monitoring for rotating machinery.

- Due to the demonstrated potential of the MD-based HI generation, it is suggested to further develop this method to approach the performance of AE-based HI generation while retaining the computational efficiency of the MD-based method. This can be approached by an optimisation of MD features towards a consistent HI range in addition to a maximisation of MD separation.
- Conversely, AE-based HI generation offers the potential to encapsulate complex nonlinear relationships of the processed condition data and may be effective for data fusion of complex signal patterns. The reconstruction ability of AE for various signal types and signal properties (such as periodic and statistical characteristics) should be therefore further investigated. At the same time, techniques to maximise the reconstruction error for untrained data should be developed.
- It was shown across all methods that the reduction of training sequences leads to a wider spread of prediction performance. In other words, after the random sampling of limited training sequences, part of the training runs resulted in a competitive model performance, whereas models from different samples of training sequences underperformed. Methods for advance assessment of training data (regarding data distribution and quality) should be therefore researched. The desired consequence is optimised training efficiency, leading to minimised data and energy consumption.
- A segmentation of machine degradation into different sections has the potential to improve the prediction performance. In case of the polynomial regression method, this can be incorporated by fitting a suitable polynomial (in terms of polynomial order and constraints) to the identified degradation phase. A common but inflexible approach in NN-based prediction, is to limit the training range of RUL values. Expanding on that, separate models for different prediction ranges (e.g., for long-term, mid-term and short-term predictions) should be considered. The presented HI-generation approach in combination with clustering can contribute to automatically identify relevant degradation phases for further development in this area.
- Future research should consider contextual data to distinguish between actual fault information of the monitored component and independent external factors, such as potential changes in environmental conditions. Operational Modal Analysis can be used to provide supplementary context information as it is suited to assess large-scale structural changes, which are subject to operating and environmental influence, and global degradation itself, in turn influencing the response of localised components. Consideration of both global and local machine condition could increase the reliability

and accuracy of the implemented monitoring/prediction system.

DATA ACCESS STATEMENT

This research is supported by open datasets [10], [66] cited in the ''References'' section of this paper.

REFERENCES

[1] A. Rytter, ''Vibrational based inspection of civil engineering structures,'' Ph.D. thesis, Aalborg Univ., Aalborg, Denmark, 1993, vol. R9314.

[2] T. Brotherton, G. Jahns, J. Jacobs, and D. Wroblewski, ''Prognosis of faults in gas turbine engines,'' in *Proc. IEEE Aerosp. Conf.*, Mar. 2000, pp. 163–172, doi: 10.1109/AERO.2000.877892.

[3] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, ''Machinery health prognostics: A systematic review from data acquisition to RUL prediction,'' *Mech. Syst. Signal Process.*, vol. 104, pp. 799–834, May 2018, doi: 10.1016/j.ymssp.2017.11.016.

[4] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, ''Deep learning and its applications to machine health monitoring,'' *Mech. Syst. Signal Process.*, vol. 115, pp. 213–237, Jan. 2019, doi: 10.1016/j.ymssp.2018.05.050.

[5] N. B. Jones and Y.-H. Li, ''A review of condition monitoring and fault diagnosis for diesel engines,'' *Lubrication Sci.*, vol. 6, no. 3, pp. 267–291, 2000, doi: 10.1002/tt.3020060305.

[6] Y. Wang, Y. Zhao, and S. Addepalli, ''Remaining useful life prediction using deep learning approaches: A review,'' *Proc. Manuf.*, vol. 49, pp. 81–88, Jan. 2020, doi: 10.1016/j.promfg.2020.06.015.

[7] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, ''Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks,'' *J. Sound Vib.*, vol. 388, pp. 154–170, Feb. 2017, doi: 10.1016/j.jsv.2016.10.043.

[8] B. C. Wen, M. Q. Xiao, X. Q. Wang, X. Zhao, J. F. Li, and X. Chen, ''Data-driven remaining useful life prediction based on domain adaptation,'' *PeerJ Comput. Sci.*, vol. 7, pp. 1–25, Sep. 2021, doi: 10.7717/peerj-cs.690.

[9] C. Li, R.-V. Sánchez, G. Zurita, M. Cerrada, and D. Cabrera, ''Fault diagnosis for rotating machinery using vibration measurement deep statistical feature learning,'' *Sensors*, vol. 16, no. 6, p. 895, Jun. 2016, doi: 10.3390/s16060895.

[10] P. Nectoux. (2012). *FEMTO Bearing Data Set*. FEMTO-ST Institute. Accessed: Feb. 20, 2022. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#femto

[11] A. Agogino and K. Goebel. (2007). Milling Data Set. BEST Lab, UC Berkeley. Accessed: Feb. 21, 2022. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#milling

[12] J. Lee, H. Qiu, G. Yu, and J. Lin. (2007). Bearing data set. Center for Intelligent Maintenance Systems (IMS), University of Cincinnati. Accessed: Feb. 21, 2022. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#bearing

[13] Case Western Reserve University. *Bearing Data Center—Seeded Fault Test Data*. Accessed: Feb. 19, 2022. [Online]. Available: https://engineering.case.edu/bearingdatacenter

[14] The Prognostics and Health Management Society. (2009). *2009 PHM Challenge Competition Data Set*. Accessed: Feb. 20, 2022. [Online]. Available: https://phmsociety.org/public-data-sets/

[15] E. Bechhoefer. *Condition Based Maintenance Fault Database for Testing of Diagnostic and Prognostics Algorithms*. MFPT. Accessed: Feb. 20, 2022. [Online]. Available: https://www.mfpt.org/fault-data-sets/

[16] H. Huang and N. Baddour. (2019). Bearing vibration data under time-varying rotational speed conditions. University of Ottawa. Accessed: Apr. 20, 2022. [Online]. Available: https://data.mendeley.com/datasets/v43hmbwxpm/2, doi: 10.17632/v43hmbwxpm.2.

[17] Y. Liang. (2019). The motor fault diagnosis experiment dataset. Zenodo. Accessed: Apr. 17, 2022. [Online]. Available: https://zenodo.org/record/3553755, doi: 10.5281/zenodo.3553755.

[18] A. E. Treml, R. A. Flauzino, M. Suetake, and A. N. R. Maciejewski. (2020). Experimental database for detecting and diagnosing rotor broken bar in a three-phase induction motor. IEEE DataPort. Accessed: Apr. 17, 2022. [Online]. Available: https://ieee-dataport.org/open-access/experimental-database-detecting-and-diagnosing-rotor-broken-bar-three-phase-induction, doi: 10.21227/fmnm-bn95.

[19] O. Mey, W. Neudeck, A. Schneider, and O. Enge-Rosenblatt. (2020). *Unbalance Detection of a Rotating Shaft Using Vibration Data*. Kaggle. Accessed: Apr. 20, 2022. [Online]. Available: https://www.kaggle.com/datasets/jishnukoliyadan/vibration-analysis-on-rotating-shaft

[20] R. Viitala, J. Miettinen, T. Tiainen, and R. Viitala. (2020). Rotor & bearing vibration dataset. Aalto University. Accessed: Apr. 20, 2022. [Online]. Available: https://data.mendeley.com/datasets/pdrxyfprfk/1, doi: 10.17632/pdrxyfprfk.1.

[21] Universidade Federal do Rio de Janeiro COPPE/Poli/UFRJ. (2021). *MAFAULDA Machinery Fault Database*. Accessed: Feb. 19, 2022. [Online]. Available: http://www02.smt.ufrj.br/~offshore/mfs/page_01.html

[22] E. Strubell, A. Ganesh, and A. McCallum, ''Energy and policy considerations for modern deep learning research,'' in *Proc. 34th AAAI Conf. Artif. Intell. (AAAI)*, 2020, pp. 1393–13696, doi: 10.1609/aaai.v34i09.7123.

[23] P. Dhar, ''The carbon impact of artificial intelligence,'' *Nature Mach. Intell.*, vol. 2, no. 8, pp. 423–425, Aug. 2020, doi: 10.1038/s42256-020-0219-9.

[24] R. Schwartz, J. Dodge, N. Smith, and O. Etzioni, ''Green AI,'' *Commun. ACM*, vol. 63, no. 12, pp. 54–63, Dec. 2020, doi: 10.1145/3381831.

[25] H. Li, W. Wang, Z. Li, L. Dong, and Q. Li, ''A novel approach for predicting tool remaining useful life using limited data,'' *Mech. Syst. Signal Process.*, vol. 143, Sep. 2020, Art. no. 106832, doi: 10.1016/j.ymssp.2020.106832.

[26] N. Gebraeel, A. Elwany, and J. Pan, ''Residual life predictions in the absence of prior degradation knowledge,'' *IEEE Trans. Rel.*, vol. 58, no. 1, pp. 106–117, Mar. 2009, doi: 10.1109/TR.2008.2011659.

[27] B. Merainani, S. Laddada, E. Bechhoefer, M. A. A. Chikh, and D. Benazzouz, ''An integrated methodology for estimating the remaining useful life of high-speed wind turbine shaft bearings with limited samples,'' *Renew. Energy*, vol. 182, pp. 1141–1151, Jan. 2022, doi: 10.1016/j.renene.2021.10.062.

[28] J. Carroll, S. Koukoura, A. McDonald, A. Charalambous, S. Weiss, and S. McArthur, ''Wind turbine gearbox failure and remaining useful life prediction using machine learning techniques,'' *Wind Energy*, vol. 22, no. 3, pp. 360–375, Mar. 2019, doi: 10.1002/we.2290.

[29] J. Qu, F. Liu, Y. Ma, and J. Fan, ''A neural-network-based method for RUL prediction and SOH monitoring of lithium-ion battery,'' *IEEE Access*, vol. 7, pp. 87178–87191, 2019, doi: 10.1109/ACCESS.2019.2925468.

[30] Z. Kang, C. Catal, and B. Tekinerdogan, ''Remaining useful life (RUL) prediction of equipment in production lines using artificial neural networks,'' *Sensors*, vol. 21, no. 3, pp. 1–20, 2021, doi: 10.3390/s21030932.

[31] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, ''A recurrent neural network based health indicator for remaining useful life prediction of bearings,'' *Neurocomputing*, vol. 240, pp. 98–109, May 2017, doi: 10.1016/j.neucom.2017.02.045.

[32] S. Hochreiter, ''Recurrent neural net learning and vanishing gradient,'' *Int. J. Uncertainity, Fuzziness Knowl.-Based Syst.*, vol. 6, no. 2, pp. 107–116, 1998.

[33] R. Pascanu, T. Mikolov, and Y. Bengio, ''On the difficulty of training recurrent neural networks,'' in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28, 2013, pp. 1310–1318, doi: 10.1007/978-3-319-93145-6_3.

[34] S. Hochreiter and J. Schmidhuber, ''Long short-term memory,'' *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[35] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, ''On the properties of neural machine translation: Encoder–decoder approaches,'' in *Proc. 8th Workshop Syntax, Semantics Struct. Stat. Transl.*, 2014, pp. 103–111, doi: 10.3115/v1/W14-4012.

[36] H. Zhang, Q. Zhang, S. Shao, T. Niu, and X. Yang, ''Attention-based LSTM network for rotatory machine remaining useful life prediction,'' *IEEE Access*, vol. 8, pp. 132188–132199, 2020, doi: 10.1109/ACCESS.2020.3010066.

[37] X. Li, Q. Ding, and J.-Q. Sun, ''Remaining useful life estimation in prognostics using deep convolution neural networks,'' *Rel. Eng. Syst. Saf.*, vol. 172, pp. 1–11, Apr. 2018, doi: 10.1016/j.ress.2017.11.021.

[38] B. Yang, R. Liu, and E. Zio, ''Remaining useful life prediction based on a double-convolutional neural network architecture,'' *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9521–9530, Dec. 2019, doi: 10.1109/TIE.2019.2924605.

[39] S. Ren, K. He, R. Girshick, and J. Sun, ''Faster R-CNN: Towards real-time object detection with region proposal networks,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1–9.

[40] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *J. Syst. Eng. Electron.*, vol. 28, no. 1, pp. 162–169, Feb. 2017, doi: 10.21629/JSEE.2017.01.18.

[41] Y. C. Liu, X. F. Hu, and W. J. Zhang, "Remaining useful life prediction based on health index similarity," *Rel. Eng. Syst. Saf.*, vol. 185, pp. 502–510, May 2019, doi: 10.1016/j.ress.2019.02.002.

[42] M. M. Manjurul Islam, A. E. Prosvirin, and J.-M. Kim, "Data-driven prognostic scheme for rolling-element bearings using a new health index and variants of least-square support vector machines," *Mech. Syst. Signal Process.*, vol. 160, Nov. 2021, Art. no. 107853, doi: 10.1016/j.ymssp.2021.107853.

[43] F. Yang, M. S. Habibullah, T. Zhang, Z. Xu, P. Lim, and S. Nadarajan, "Health index-based prognostics for remaining useful life predictions in electrical machines," *IEEE Trans. Ind. Electron.*, vol. 63, no. 4, pp. 2633–2644, Apr. 2016, doi: 10.1109/TIE.2016.2515054.

[44] Y. Wei, D. Wu, and J. Terpenny, "Learning the health index of complex systems using dynamic conditional variational autoencoders," *Rel. Eng. Syst. Saf.*, vol. 216, Dec. 2021, Art. no. 108004, doi: 10.1016/j.ress.2021.108004.

[45] Y. Fan, S. Nowaczyk, and T. Rögnvaldsson, "Transfer learning for remaining useful life prediction based on consensus self-organizing models," *Rel. Eng. Syst. Saf.*, vol. 203, Nov. 2020, Art. no. 107098, doi: 10.1016/j.ress.2020.107098.

[46] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, "Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2416–2425, Apr. 2019, doi: 10.1109/TII.2018.2881543.

[47] J. Zhu, N. Chen, and C. Shen, "A new data-driven transferable remaining useful life prediction approach for bearing under different working conditions," *Mech. Syst. Signal Process.*, vol. 139, May 2020, Art. no. 106602, doi: 10.1016/j.ymssp.2019.106602.

[48] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliab. Eng. Syst. Saf.*, vol. 183, pp. 240–251, Jun. 2018, doi: 10.1016/j.ress.2018.11.027.

[49] H. Takeyama and R. Murata, "Basic investigation of tool wear," *J. Eng. Ind.*, vol. 85, no. 1, pp. 33–37, Feb. 1963, doi: 10.1115/1.3667575.

[50] E. Usui, T. Shirakashi, and T. Kitagawa, "Analytical prediction of cutting tool wear," *Wear*, vol. 100, nos. 1–3, pp. 129–151, Dec. 1984, doi: 10.1016/0043-1648(84)90010-3.

[51] X. Fang, R. Zhou, and N. Gebraeel, "An adaptive functional regression-based prognostic model for applications with missing data," *Rel. Eng. Syst. Saf.*, vol. 133, pp. 266–274, Jan. 2015, doi: 10.1016/j.ress.2014.08.013.

[52] J. Wang, Y. Liang, Y. Zheng, R. X. Gao, and F. Zhang, "An integrated fault diagnosis and prognosis approach for predictive maintenance of wind turbine bearing with limited samples," *Renew. Energ.*, vol. 145, pp. 642–650, Jan. 2020, doi: 10.1016/j.renene.2019.06.103.

[53] A. Widodo and B.-S. Yang, "Machine health prognostics using survival probability and support vector machine," *Expert Syst. Appl.*, vol. 38, no. 7, pp. 8430–8437, 2011, doi: 10.1016/j.eswa.2011.01.038.

[54] Van T. Tran and B.-S. Yang, "An intelligent condition-based maintenance platform for rotating machinery," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 2977–2988, 2012, doi: 10.1016/j.eswa.2011.08.159.

[55] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006, doi: 10.1126/science.1127647.

[56] P. C. Mahalanobis, "On generalized distance in statistics," in *Proc. Nat. Inst. Sci. India*, vol. 2, no. 1, pp. 49–55, 1936.

[57] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 1, 1997, pp. 155–161.

[58] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory (COLT)*, 1992, pp. 144–152.

[59] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using LSTMs," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, vol. 1, 2015, pp. 843–852.

[60] A. Fraser and D. Burnell, *Computer Models in Genetics*. New York, NY, USA: McGraw-Hill, 1970.

[61] G. Sternharz, T. Kalganova, C. Mares, and M. Meyeringh, "Comparative performance assessment of methods for operational modal analysis during transient order excitation," *Mech. Syst. Signal Process.*, vol. 169, Apr. 2022, Art. no. 108719, doi: 10.1016/j.ymssp.2021.108719.

[62] N. Costa and L. Sánchez, "Variational encoding approach for interpretable assessment of remaining useful life estimation," *Rel. Eng. Syst. Saf.*, vol. 222, Jun. 2022, Art. no. 108353, doi: 10.1016/j.ress.2022.108353.

[63] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Proc. Int. Conf. Prognostics Health Manage.*, Oct. 2008, pp. 1–6, doi: 10.1109/PHM.2008.4711422.

[64] J. Zhang, P. Wang, R. Yan, and R. X. Gao, "Long short-term memory for machine remaining life prediction," *J. Manuf. Syst.*, vol. 48, pp. 78–86, Jul. 2018, doi: 10.1016/j.jmsy.2018.05.011.

[65] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Jun. 2017, pp. 88–95, doi: 10.1109/ICPHM.2017.7998311.

[66] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Proc. Int. Conf. Prognostics Health Manage.*, Oct. 2008, pp. 1–9, doi: 10.1109/PHM.2008.4711414.

[67] S. Zhao, Y. Zhang, S. Wang, B. Zhou, and C. Cheng, "A recurrent neural network approach for remaining useful life prediction utilizing a novel trend features construction method," *Measurement*, vol. 146, pp. 279–288, Nov. 2019, doi: 10.1016/j.measurement.2019.06.004.

[68] W. Caesarendra and T. Tjahjowidodo, "A review of feature extraction methods in vibration-based condition monitoring and its application for degradation trend estimation of low-speed slew bearing," *Machines*, vol. 5, no. 4, p. 21, Sep. 2017, doi: 10.3390/machines5040021.

GERMAN STERNHARZ received the B.Sc. and M.Sc. degrees in aerospace engineering from Technische Universität Berlin, Germany. He is currently a Postdoctoral Researcher and a Research Assistant at the Department of Electronic and Electrical Engineering, Brunel University London, U.K. His research interests include applied machine learning, operational modal analysis, structural dynamics, and machinery condition monitoring.

AYMAN ELHALWAGY received the B.Eng. degree (Hons.) in electronic and computer engineering from Brunel University London, Uxbridge, in 2021, where he is currently pursuing the Ph.D. degree in electronic and electrical engineering.

His research interests include applied machine learning, fault detection and classification, and intelligent systems.

TATIANA KALGANOVA (Member, IEEE) received the B.Sc. (Hons.) and Ph.D. degrees. She is currently a Reader in intelligent systems and the Electronic and Computer Engineering Postgraduate Research Director of Brunel University London, Uxbridge, U.K. She has over 20 years of experience in design and implementation of applied intelligent systems.