



TCRSCANet: Harnessing Temporal Convolutions and Recurrent Skip Component for Enhanced RUL Estimation in Mechanical Systems

Abdul Wahid¹ · John G. Breslin¹ · Muhammad Ali Intizar²

Received: 23 June 2023 / Accepted: 14 December 2023 / Published online: 29 January 2024
© The Author(s) 2024

Abstract

Estimating the remaining useful life (RUL) of critical industrial assets is of crucial importance for optimizing maintenance strategies, enabling proactive planning of repair tasks, enhanced reliability, and reduced downtime in prognostic health management (PHM). Deep learning-based data-driven approaches have made RUL prediction a lot better, but traditional methods often do not look at the similarities and differences in the data, which lowers the accuracy of the estimates. Previous attempts to use Long Short-Term Memory (LSTM) networks for RUL prediction have failed because they depend on learned features for regression at the very end of the time step. The single objective function for estimation also constrains the learned representations, which has an impact on RUL estimation. The goal of this study is to find out how to predict the RUL of mechanical systems using complex sensor data. To do this, we present a data-driven framework called temporal convolution, along with a recurrent skip component and an attention mechanism network called TCRSCANet. It uses a combination of temporal convolution, recurrent skip parts, and an attention mechanism to make RUL estimation more accurate. The recurrent skip component finds long-term patterns in time series data, while temporal convolution pulls out high-level features from longer sequences. Finding hidden representations and degradation-development interactions between features at each window position in the input matrix is what the attention layer does to focus on the most important information for RUL estimation. The proposed methodology is tested and validated against the well-established C-MAPSS dataset, which focuses on aircraft degradation. The TCRSCANet model is better at predicting RUL as compared to other state-of-the-art methods because it uses the root mean square error (RMSE) and a scoring function to measure performance. The results of this study demonstrate the importance of the recurrent skip component and attention mechanisms for determining how long an industrial asset will be valuable.

Keywords Industry 4.0 · Smart manufacturing · Recurrent skip component (RSC) · Attention layer (AL) · Remaining useful life (RUL) · Temporal convolution with recurrent-skip-component and attention mechanism network (TCRSCANet)

Abbreviations

PHM Prognostic health management
RUL Remaining useful life
ANN Artificial neural network
SVR Support vector regression

RF Random forest
LSTM Long short-term memory
RNN Recurrent neural network
CNN Convolutional neural network
RSC Recurrent skip component
SVRM Support vector regression machine
BPNN Back propagation neural network
TCRSA Temporal convolution recurrent skip component with attention

✉ Abdul Wahid
a.wahid2@universityofgalway.ie

John G. Breslin
john.breslin@universityofgalway.ie

Muhammad Ali Intizar
ali.intizar@dcu.ie

¹ Data Science Institute (DSI), University of Galway, IDA Business Park, Galway H91TK33, Ireland

² School of Electronic Engineering, Dublin City University, Glasnevin Campus, Dublin 9 D09NR58, Ireland

1 Introduction

In the industrial sector, maintenance plays a critical role. It helps to ensure that equipment and machinery are functioning at their optimal level. Proper maintenance can help

to prevent breakdowns and downtime, which can be costly for businesses [1]. By keeping machinery and equipment in good working order, maintenance can help to increase the reliability of products and services and therefore enhance the performance of the business [2]. In the modern age of the Industrial Internet of Things [3], it's quite usual to use a multitude of sensors to keep track of machines' operational behaviours. The associated wealth of data being generated has resulted in a significant amount of interest in using this sensor data for making decisions in areas like maintenance management [4], quality improvement [5] and machine health monitoring tasks such as anomaly/fault detection and prognostics, which involve calculating the remaining useful life (RUL) of machines in action [6].

The estimation of RUL spans several disciplines, forming a significant part of the wider domain of PHM [7]. This domain investigates the performance of a system throughout its lifespan, specifically from the moment the most recent maintenance was conducted until the point of system failure or when performance deterioration hits a predefined threshold [1]. Mechanical equipment has become increasingly difficult to monitor in recent years, and mechanical reliability has therefore become very important, so the methods of prognosis and health management have received more attention than before [8, 9]. Predicting RUL in PHM has emerged as a proactive maintenance method as it helps in making suitable maintenance decisions at the right time and avoids poor estimations of when maintenance should otherwise be carried out, which can result in unforeseen catastrophic failures [10]. In [11] researchers came up with different ways to predict RUL, which can be put into three groups: (1) physics-based approaches; (2) hybrid-model approaches, and (3) data-driven approaches.

Physics-based methods leverage our comprehension of a system's breakdown processes (such as the expansion of cracks) to generate a mathematical representation of the system's decline process and calculate the RUL. Based on physics or fundamental theories, these mathematical models numerically capture the behaviour of a system [12]. Determining the parameters of the model typically necessitates carefully structured experiments and a considerable amount of empirical data. To discover and update these parameters from condition data, prognostics often employ statistical methods, such as regression and Bayesian updating. This method requires comprehensive knowledge and vast physical experience from researchers [13]. Furthermore, the creation of dynamic mathematical and physical models to represent various failure modes within a system presents a significant challenge [6].

Data-driven strategies are based on the idea of forming a non-linear correlation between historical data and the respective normal operating conditions of a machine. Artificial Neural Networks (ANN) [14], support vector

regression (SVR) [15], and random forest (RF) [16] are examples of traditional data-driven models. These models typically involve two steps: extracting the features manually and learning the degrading behaviour. However, these methods rely on human input and do not fully understand how manual feature extraction and degrading behaviour learning are related, thus further reducing the estimation accuracy. Recent advances in deep learning [17] have made it possible for data-driven approaches to successfully decrease the need for human intervention by learning directly about degradation from unprocessed monitoring data. Data-driven techniques require one or more sets of data that extend to the point of failure to anticipate the RUL and depict the deterioration process [18]. When there are no a priori degradation sequences available, the indirect method for RUL estimation is the best option. Furthermore, a hybrid model approach integrates model-based and data-driven methods to increase RUL prediction accuracy and overcome the gaps in different methods [19]. However, due to complicated system degradation mechanisms, there have not been many studies on hybrid techniques [18]. Consequently, in the field of RUL estimation, data-driven methodologies are becoming more popular.

As data becomes more detailed, the identification and extraction of performance deterioration signals from multi-sensor data becomes a major technical problem for complex systems [18]. This has a substantial impact on prediction performance. Unfortunately, the usual method of manually constructing features for complex domains necessitates a lot of human effort, and the majority of information can be lost, so performance cannot be assured. Deep learning, as discussed in [17], has emerged as an effective and beneficial alternative to some aspects of human labour. For instance, in modern natural language processing (NLP) research, recurrent neural network (RNN) and convolutional neural network (CNN) models, as discussed in [20], have gained significant popularity. Due to their strong ability to learn features, these methods have also provided effective solutions for identifying faults. In [21] a simple machine-health monitoring system was proposed based on auto-encoder, CNN, and RNN. In [22] a deep-learning-based framework for prognostics and health management was proposed. A CNN utilising an auto-encoder for induction motor diagnosis was introduced in [23]. A hybrid model incorporating a Local Feature-based Gated Recurrent Unit (LFGRU) network was proposed in [24]. This method merges the creation of hand-crafted features with the process of automatically learning features for machine health monitoring. A vanilla Long-Short Term Memory network (LSTM) was employed to achieve high precision in forecasting the RUL in [25]. This approach effectively leverages LSTM capabilities, especially in complex scenarios involving intricate operations, working conditions, model degradation, and significant noise.

It's interesting to note that LSTMs aren't among the best-performing models for various operating conditions [26]. Even though they are widely used, combining CNNs and LSTMs seems to outperform stand-alone CNNs and LSTMs [27]. Even though deep learning offers numerous advantages for prognosis and health monitoring, it's essential to acknowledge its notable limitations, especially in the context of identifying patterns within a multivariate time series. A significant challenge arises from the dynamic, non-stationary, and spatiotemporal characteristics of time-series signals. Such complexity can hinder deep learning models from accurately identifying and representing the inherent patterns within the data [18]. The dynamic, non-stationary, and spatiotemporal characteristics of time-series signals present a significant challenge. This complexity can impede the ability of deep learning models to accurately capture and represent the inherent patterns within the data [20]. In this case, another problem with deep learning models is that they usually use a network with a single input unit (single-head) to extract features from all the signals in the multivariate dataset [28]. This method presumes that a single unit is sufficiently robust to process all time-series variables effectively, an assumption that often falls short. Such an architecture usually depends on sequential models to encode past inputs and generate future predictions. Yet, in numerous practical situations, using a single unit to process information from heterogeneous sensor networks can lead to a sub-optimal model that fails to explicitly accommodate the diversity in time-varying inputs.

Predicting RUL in PHM [7] presents a distinct set of challenges that sets it apart from other spatiotemporal forecasting tasks, like traffic prediction. In the context of RUL, the goal isn't just to capture the temporal progression of a system's state but also to accurately determine when the system will fail. Graph Convolutional Networks (GCN) [29] are designed to handle structured data represented as graphs. They capture both local and global information through convolutions on nodes and their neighbourhoods. The major challenge for using GCN in RUL prediction lies in defining an appropriate graph structure for the machine or system in question. Also, while GCN is good at showing how nodes are connected in space, it might need to be changed to better show how they change over time, which is important for predicting RUL [30]. CNNs are very efficient at identifying local patterns and features from data and can be leveraged to capture local temporal dependencies in the degradation data. LSTMs, on the other hand, can model long-term dependencies, making them suitable for tracking the gradual degradation of systems over time. The hybrid architecture of CNN and LSTM can capture both short-term (local) and long-term (global) temporal dependencies in the data [31]. This can be particularly useful in situations where the degradation pattern shows varied behaviour over short and long periods.

Recently, models based on the concept of 'attention' have been employed to enhance the predictive capabilities of deep learning models [26]. The attention mechanism was first used in tasks associated with machine translation [32]. Given its interpretable weight values, it was widely used for image [33] and recommendation systems [34] tasks. The attention mechanism involves dynamically weighing the contributions of different inputs to the output of the model. The attention mechanism enables the model to prioritise crucial features and patterns within the input data. This proves particularly beneficial when handling data sequences like time series or language sequences [35]. Attention mechanisms have demonstrated their ability to boost performance in various areas, including speech recognition [36], machine translation [32], and image captioning [37]. Researchers have gotten very good results in several areas by adding these structures to deep learning models. These areas include health monitoring and prognosis [26, 38, 39]. By leveraging the ability of these architectures to extract and focus on important features, deep learning models can more accurately and reliably predict outcomes, leading to potential benefits in diagnosis, treatment, and overall patient care.

A temporal convolution and recurrent skip component with an attention mechanism are proposed in this paper. Using a 1D convolutional layer, the proposed deep learning framework extracts important features that change over time from input variables that are spread out in multiple dimensions. The extracted temporal features are transformed into abstract features by passing them through a fully connected layer. Following this, a recurrent layer is employed to identify patterns and long-term dependencies in the data [40]. By allowing the model to circumvent specific layers, the skip components assist in mitigating the vanishing gradient issue. The attention mechanism is utilized to evaluate the significance of various segments of the input sequence dynamically. This functionality allows the model to focus on the critical attributes and trends relevant to the given task, consequently improving its precision and effectiveness. In addition to aiding in the management of the challenge posed by varying input lengths, the attention mechanism enables the model to dynamically focus on specific segments of the input sequence.

The key contributions of this work can be summarised as follows:

- A deep learning framework is proposed that efficiently extracts significant features exhibiting local or short-term time-dependency patterns within multi-dimensional input time series. This model perceives all data encapsulated within a designated time window as an indivisible entity, disregarding the length of the time series or the number of features embedded within the windowed data. The main contribution of this work is how it interprets a

whole-time window as a statement that expresses all the data for the remaining time in its usable lifetime through its temporal relationship.

- The introduction of a recurrent skip component featuring an attention mechanism. Learning the weighted combination of hidden representations at each position of the time window in the input matrix is the purpose of the Recurrent-Skip-Component (RSC). At various time periods, the attention mechanism extracts the critical temporal patterns and feature relationships from a time series. Additionally, it acquires knowledge of the various degradation patterns that span the complete duration of the input time series.
- The recurrent skip component extends the period of the information flow using recurrent connections and therefore eases the optimization process.
- We evaluated the proposed approach on the C-MAPSS dataset. The findings suggest that our approach exhibits superior performance in comparison to numerous other data-driven strategies for RUL prediction.

Structure of the Paper: The remainder of this paper is structured as follows. Section 2 presents some state-of-the-art data-driven approaches. Section 3 outlines the problem formulation. Section 4 presents the framework and explains the structural elements of the proposed model. Section 5 explains the dataset and pre-processing techniques used. It also presents the evaluation metrics used to verify the proposed approach on the CMAPSS dataset. Section 6 elaborates on the network design and offers a comparative analysis with other prominent approaches. Section 7 presents the ablation study. In conclusion, Sect. 8 finalizes the paper and sheds light on potential avenues for future research.

2 Literature Review

In the past few years, the fields of machine learning and deep learning have seen significant advancements. They have been extensively employed to address issues in manufacturing and industrial systems, delivering outstanding results in RUL prediction. An overview of various RUL approaches was provided in [41]. Over the years, new ways to figure out the RUL of mechanical systems have been developed. These include support vector regression [15], recurrent neural networks [11], LSTMs [25], deep convolutional neural networks [42], and hybrid deep learning networks [31]. An accurate RUL prediction enables a mechanical system to seamlessly plan for future maintenance and repair. For estimating the remaining life of complex mechanical systems, deep learning architectures such as RNN and CNN have been extensively implemented.

2.1 RNN and LSTM-Based RUL Estimation

Unsupervised representation learning for sequences using RNNs has been successful in a variety of domains, including text, video, audio, and time series data (e.g., sensor data). In literature, RNNs have been utilised to analyse time-series data and forecast degradation patterns [11]. However, it suffers from the problem of vanishing gradients, [43] which can make it difficult for RNNs to learn long-term dependencies in timeseries data. An LSTM that captured the long-term dependencies between features in the time-series data was proposed in [44]. A vanilla LSTM that could be used to improve prediction accuracy in complex operations, working conditions, model deterioration, and high noise was proposed in [25]. In [45], LSTM was leveraged to learn the non-linear relationships between the RUL and degradation features. It utilised three evaluation indicators to pinpoint the most representative degradation features for RUL prediction. This method led to improved results in comparison with conventional machine learning algorithms such as back-propagation neural networks (BPNN) and support vector regression machines (SVRM). In another study, a bidirectional LSTM for RUL prediction was introduced [46]. They first established a health index (HI) and then tracked the HI fluctuations to predict the RUL. In a different study, a transfer learning-based bidirectional LSTM was introduced [47]. This LSTM could be trained on various relevant datasets before fine-tuning it on the target dataset. This bidirectional LSTM encapsulates temporal information from both past and future scenarios by processing input sequences in both forward and backward directions.

2.2 Convolutional Neural Network-Based RUL Estimation

In recent times, CNNs and LSTM networks have emerged as effective methodologies for pattern recognition across various domains, including computer vision [48, 49]. Because the RUL estimation problem and pattern recognition are so closely related, identical strategies can be used to address the RUL estimation problem [50]. Because they can extract features from both one-dimensional and two-dimensional data, CNN has been able to accurately predict the RUL of complex systems [42]. In [51], CNNs were proposed to predict RUL for aero-engines. The authors used a time wrapping (TW) technique to pre-process the raw data samples and were able to extract more information about the degradation state of the system. However, the use of TW results in an increase in the dimensionality of the input data, making it more difficult to create an effective neural network (NN) model for the task. A 1D CNN for predicting the RUL was proposed in [28]. In this paper, the authors used a 1D CNN with a complete convolutional layer to extract the spatial and

temporal features from a given time series. 1D CNNs are well-suited to handle the challenges posed by highly nonlinear and multidimensional sensor streaming data. They were successfully adopted to capture the salient features of the sensor signals [42]. A simplified 1D CNN was proposed in [52]. It bypassed a large number of parameters by using maximum pooling layers instead of fully connected layers. Additionally, 1D CNN was combined with bi-directional LSTM [53] to extract abstract features and encode temporal knowledge. Recently, CNN produced promising results in RUL prediction, which was used to collect spatial information without taking into account the time-series correlation to the data. Meanwhile, 2D convolutions are computationally expensive compared to 1D convolutions. It has been observed that, under equivalent conditions, a 1D CNN has a lower computational complexity [54], thus making it well-suited for real-time applications.

2.3 Hybrid Methods for RUL Estimation

The adoption of a hybrid method and the implementation of a parallel multi-model approach have emerged as effective strategies for integrating the strengths of various models. These models collect different types of data at distinct time intervals, leading to more precise predictions. This is a method of improving accuracy that had not been explored previously [55]. A hybrid deep learning model was proposed in [31] which implemented 1D CNN with the LSTM for RUL prediction. A methodology utilizing 2D convolution for RUL predictions, founded on sliding windows, was put forward in [42, 46] and it supported the use of LSTM networks for these RUL estimations. While LSTM networks can establish long-term time dependencies, their ability to extract features is marginally less effective compared to that of CNNs [56]. Both CNN and LSTM networks possess the distinctive skill to extract features from data. The latest architectures for estimating RUL tend to rely on a singular approach, whether that be CNN, DNN, or LSTM as evidenced in [57]. In [58], a composite network was introduced. This method leveraged the time window technique and gradient boosting regression for scrutinizing degradation trends within sensor sequences. Concurrently, the LSTM networks were deployed to uncover patterns and trends concealed within lengthy sensor data sequences.

2.4 Attention-Based RUL Estimation

The attention mechanism has been widely used in NLP and has shown exceptional results [36]. It was introduced as an innovative approach for handling sequential data. The transformer encoder module, which includes a self-attention module, allows for processing data inputs based on their timestamps [59]. This self-attention mechanism

assigns varying weights to each timestamp, thereby understanding the importance of each input in the sequence [59]. In this network, the attention and linear layers are arranged alternately, with shortcut components used to establish connections between them. The authors of [60] introduced a framework for RUL prediction that uses an attention-based methodology. They used an attention mechanism to simultaneously weigh time steps and features in the 2D sequence data output by the LSTM in the time dimension. The attention and linear layers are arranged alternately in this network, with shortcut components used to establish connections between them. A new way to predict RUL was suggested in the research paper called Dual Attention-based Temporal Convolutional Network for RUL Prediction [61]. This approach is called the distributed attention-based temporal convolutional network (TCN). It uses an attention mechanism that applies a SoftMax function to sensor input data to determine the importance, or ‘weight’, of each feature at a specific time. The weighted data is then fed into the TCN, which extracts critical features. The outputs of the network are subjected to another attention mechanism, which weights the outputs in the time domain. Finally, the weighted outputs are averaged across all temporal intervals within the specified time frame. This approach has shown promise in accurately forecasting the RUL of machinery [61].

It is critical to extract important and relevant features. Attention mechanisms allow the model to prioritize different aspects of input data based on their importance [39]. This improves the model’s ability to learn complex patterns and correlations in the data [60]. A multi-dimensional self-attention network was proposed in [26]. Numerous attention networks process the input after it has undergone a linear transformation and then analyse the transformed input from their distinct perspectives. This permits the model to concentrate on different segments of the input distinctively. The multi-head attention mechanism can improve the model’s ability to extract important features from data. Another type of attention mechanism, called self-attention, specifically looks at the relationships between features within a single input sequence [38]. This feature allows the model to identify the intricate dependencies between features, which are challenging to capture with conventional attention mechanisms. Our model is based on a self-attention mechanism designed to learn the correlations and degradation patterns present within multi-sensor data. This approach considers all time steps and features at the same time, which helps the system learn how different sensor readings interact over time. By doing so, the approach identifies the important patterns and relationships in the data that might be missed by other modelling methods. This is an effective way to understand how these interactions contribute to the degradation of the system.

3 Problem Formulation

Estimating RUL can be interpreted as a regression task. We utilize data gathered from sensors installed on machinery (specifically engines in our scenario) as inputs, while the designated RUL labels serve as outputs. Consider a system with N components (turbojet engines). The goal is to predict when the current operational component for which the multi-variate time-series data is available will fail. Let K be the number of sensors installed on each engine component. Therefore, an i^{th} component throughout its useful time produces a multivariate time series which is as follows:

$$X_i = x_1^i, x_2^i, x_3^i, \dots, x_{T_i}^i$$

where $X_i \in \mathbb{R}^{T_i \times K}$ represents the complete training trajectory, $x_t^i \in \mathbb{R}^K$ represents the K -dimensional vector corresponding to the k sensor reading at time-stamp t , T_i represents the total number of time-stamps of n engines throughout their lifetime, and K is the number of sensors installed on each engine. Finally, we denote the RUL corresponding to the time-series X_i^t as RUL_i^t . In summary, the goal is to predict when failure will occur from a given time series for the training instances X_i . For the failed instances n , the length of training trajectory T_i corresponds to the entire lifetime from start to end, and for the in-process instance, the length T_i corresponds to the remaining operational life until the last remaining sensor reading.

4 TCRSCANet Framework

This section provides a detailed overview of TCRSCANet, which utilizes temporal convolution, a recurrent skip component, and an attention mechanism. The proposed approach extracts hidden representations of sensor data and identifies degradation trends. The overall flowchart of the framework is shown in Fig. 1. It includes three main stages (1) Data preprocessing; (2) Model construction; and (3) RUL estimation.

4.1 Training TCRSCANet

This section explains the data-driven prognostic framework that captures and identifies the degrading features from the absolute values of the monitoring data to predict the RUL. Since some sensor data is continuous and contains no relevant information, data visualization is employed to select meaningful sensor data. The normalization method explained in Sect. 5.3 is used for data preprocessing. The overall picture of the entire process is presented in the flowchart shown in Fig. 1. It mainly involves three main stages. The first stage is preparing to monitor data and assigning RUL labels to train and test data for RUL estimation. In

the second stage, the training data is fed through a 1D CNN that extracts temporal features from monitoring data. The extracted features are fed through a fully connected layer, creating abstract features. The final stage is the recurrent skip component with an attention mechanism. The recurrent skip component increases the temporal range of the information flow and facilitates optimization. The attention module calculates a weighted sum derived from enhanced features. The outputs are combined through a fully connected layer to predict RUL.

4.2 1D Temporal Convolutional Layer

CNN is highly effective for extracting insights from large image datasets due to the hierarchical nature of vision [21]. CNN has been successful in computer vision and has been applied to process time sequences by treating time as a spatial dimension similar to that of a two-dimensional image. However, predicting RUL involves using features from different sensors, resulting in less correlation between spatially neighbouring features. To extract significant features and map internal characteristics, we utilized a 1D convolutional neural network. Table 1 outlines the details of the layers and hyperparameters used to train the proposed architecture.

The 1D CNN extracts short-term features and local dependencies from input time sequences with multiple sensor measurements. Let $h^{(j-1)}$ denote the input sequence and h^j denote the output of the j th layer. The output from the $(j-1)$ th layer serves as input to the j th layer. Since each layer has multiple feature maps, we denote the l th feature map of the layer j as h_l^j , which is computed as follows:

$$h_l^j = \varphi \left(\sum_i h_i^{j-1} * \overline{w_{i,l}^j} + b_l^j \right) \quad (1)$$

where $*$ denotes the convolutional operation and φ represents the linear activation function. $\overline{w_{i,l}^j}$ is the weight kernel of the 1D convolutional layer, and b_l^j is the bias of l th feature map of the j th layer. The CNN extracts many features, and using all the features strongly affects the prediction process. To address this challenge, each feature map is down-sampled for important features using pooling. We used max pooling in this work. Serving as a progressive mechanism, it reduces the spatial size of feature representations, enhancing computational efficiency and mitigating neuron overfitting. In CNN, max-pooling is the primary sub-sampling process, as it can operate independently of the convolutional operation. 1D-max-pooling is defined as

$$h_{l_i}^{j+1} = \text{down}(h_{l_{nbk}}^j) \quad (2)$$

where $(*)$ represents the down-sampling function corresponding to max-pooling, $h_{l_i}^{j+1}$ is the i th element of the fea-

Fig. 1 The overall flowchart of the proposed TCRSCANet

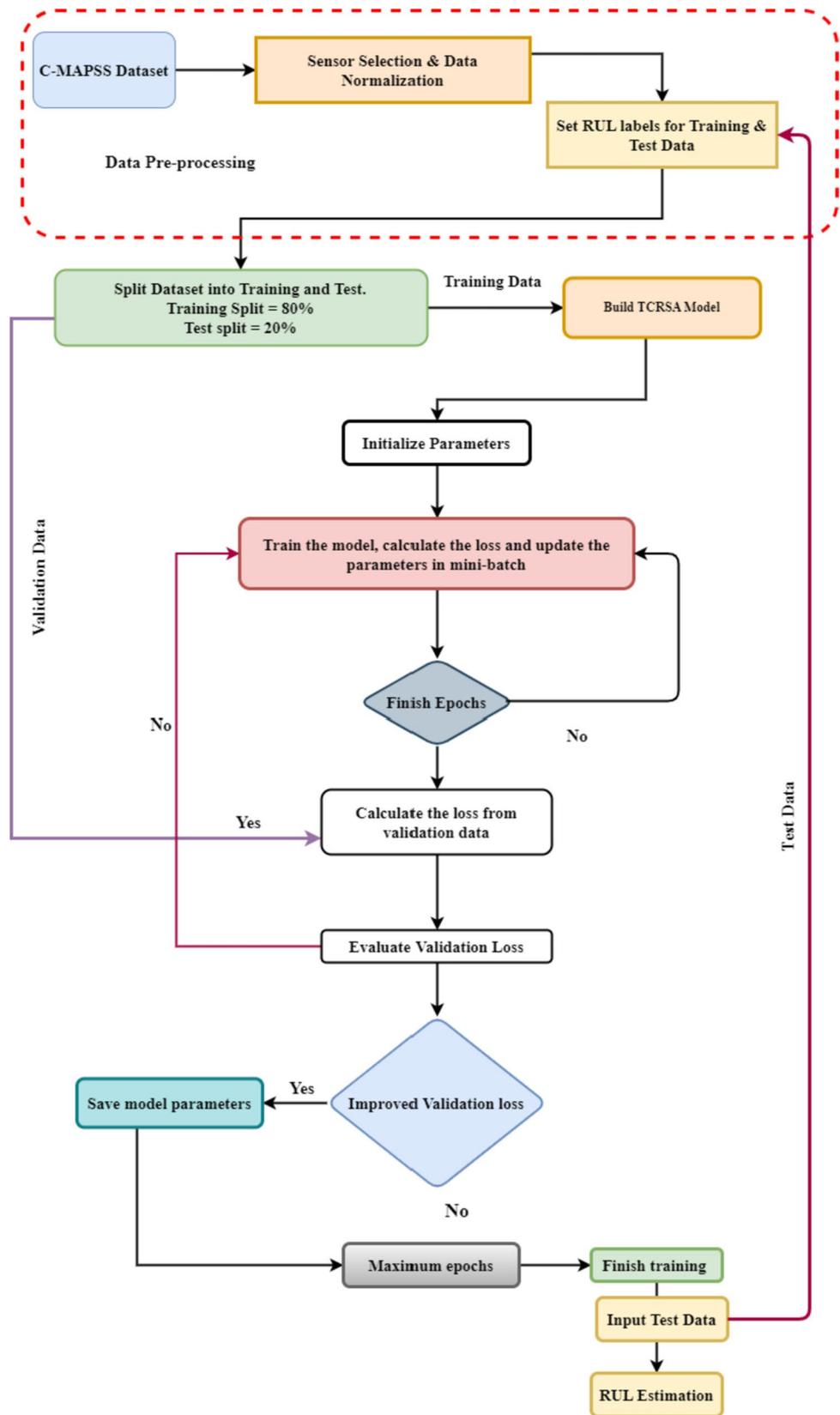


Table 1 Details of the network architecture

Layer Type	Description	Number of parameters
Layer 1	Conv 1D 1D max-pooling	Filters = 32, f size = 3, stride = 1, padding = same, activation = ReLU Pool_size = 2, stride = 2, padding = same
Layer 2	Conv 1 D 1D max-pooling	Filters = 64, f size = 3, stride = 1, padding = same, activation = ReLU Pool_size = 2, stride = 2, padding = same
Layer 3	Conv 1 D 1D max-pooling	Filters = 128, f size = 3, stride = 1, padding = same, activation = ReLU Pool_size = 2, stride = 2, padding = same
Layer 4	Fully connected (FC) Dropout	Size-seq_len \times channels, activation = ReLU d = 0.2
Layer 5	Recurrent layer	Hidden units = 100, activation = Tanh
Layer 6	Recurrent skip component (RSC)	Hidden units = 100
Layer 7	Attention layer attention weights	Head = 8, window size = 5, activation = Tanh SoftMax
Layer 8	Fully connected (FC)	Layer size = 8
Batch size	32	
Epochs	500	
Learning rate	0.0001	
Optimizer	Adam	
Window size	30	

ture map h_l^j and $h_{l_{nbk}}^i$ denotes the set of values in 1D pooling vicinity of h_l^{i+1} . After the temporal features are extracted a fully connected (FC) layer is used to generate the abstract features K_j . The abstract features are expressed as;

$$K_j = [K_1, K_2, \dots, K_i]$$

where k_i represents the abstract features with $1 \times n$ dimensions.

4.2.1 Attention Mechanism

The attention mechanism [35] has succeeded in natural language processing (NLP). Classical machine learning models like linear and logistic regression do not possess the ability to reason sequentially or utilize perception and memory through attention [62]. However, newer models like RNNs and Transformers have incorporated these features, enabling them to perform sequential reasoning. In this study, we propose an attention mechanism that focuses on one part of the sequence and guides the entire process of reasoning. The attention mechanisms with the RCS enable the model to focus on the most important parts of the sequential input data, leading to improved performance in identifying the degradation patterns in the input sequence. It allows the model to dynamically adapt to changing dependencies between time steps during the health states. This helps in identifying the critical features by focusing on what is relevant at a given time. These mechanisms improve the understanding of different

time-varying degradation patterns between time stamps. To obtain the output a dual-stage attention mechanism is implemented. The first stage consists of an encoder with input features that extract the relevant features in our case the degradation patterns. The second stage consists of temporal attention which selects the most relevant time stamps of the model. So we obtain the attention weights from the abstract features $K_j = [k_1, k_2, \dots, k_i]$ where k_i serves as the input sequence, T is the length of the time interval and n the number of input dimensions. The input sequence K_j at time step i with k input features is replaced by u_i as it adaptively selects the relevant input features which are expressed as

$$h_i = f(h_{i-1}, u_i) \quad (3)$$

where $h_{i-1} \in \mathbb{R}^m$ represents the hidden states, i the time interval, m the size of the hidden states, and f represents one of the RNN variants. Similarly u_i can be expressed as

$$u_i = (\alpha_i^n h_i + b) \quad (4)$$

Here the α_i^n denotes the attention weights and the bias of the k input feature maps at any given time. The attention weights are calculated by implementing a feedforward network which is a similarity cosine function. It scores the correlation between u_i and the randomly initialized vector u_v . The correlation score between features and degradation is used to determine the degradation attention weight. The higher the score, the stronger the correlation, and the higher the degradation attention weight assigned to the

feature. Finally, a soft-max function is employed to calculate the degradation weight λ_w^k as;

$$\lambda_w^k = \frac{\exp(\text{score}(u_i^k u_v))}{\sum_i \exp(\text{score}(u_i^k u_v))} \quad (5)$$

The next stage includes a decoder that reveals key degradation patterns. The decoder comprises a SoftMax layer, and its input is a fusion of the output and the context vector. The context vector, a weighted sum of the hidden states from the input sequence, is influenced by the alignment scores. It can be formulated as follows:

$$c_j = \sum_i \lambda_w h_i \quad (6)$$

The similarity and the soft-max functions are used to calculate the λ_w of each state h_i . Finally, c_j is fed through a fully connected layer to make RUL predictions at a j th time stamp. The parameters of TCRSCANet are summarized in Table 1.

4.3 Recurrent-Skip-Component (RSC)

We propose RSC [40]. RSC is a form of deep neural network architecture that amalgamates the advantages of recurrent neural networks (RNNs) and residual components. The idea behind RSC is to allow information from earlier time steps to bypass multiple hidden layers and directly connect to the output layer. This helps in mitigating the issue of vanishing gradients, where the gradients of the back-propagation algorithm become too small to effectively train the network. The RSC makes optimisation easier by increasing the temporal range of the information flow. The skip-links are created between the present hidden cell and hidden cells in adjacent periods that are in the same phase. Together, the attention mechanism can help enhance the ability of the RSC network to selectively focus on important parts of the input data and improve its performance on sequential prediction tasks. The merger of these two strategies, RSC and the attention mechanism, enhances performance compared to utilising either one individually, contingent upon the specific task and data set. Using these methods together, however, needs careful tweaking and real-world research to make sure that the model can correctly find relevant connections in the data. We combined RSC with the attention mechanism for the hidden state computation of the RSC network as follows:

This assists in mitigating the issue of gradient disappearance

- We calculate the hidden state in the RSC network as usual, accounting for the skip connection between the current and previous hidden states.
- To assign attention weights to every point in the input sequence, we utilize a fully connected layer with Soft-

Max activation. This process generates a probability distribution across the sequence, indicating the relative importance of each data point. This not only assures the values are non-negative but also ensures their sum equals one, enabling a more insightful understanding of the sequence's structure.

- The weighted sum, representing the significance of each step-in time, is processed through a fully connected layer coupled with a non-linear activation function. This step creates a refined hidden state for every moment in the sequence, effectively capturing the intricate correlations and dependencies within the data. Thus, it provides a comprehensive representation of the input sequence at each step, offering an enhanced understanding of the underlying patterns.
- Use this hidden state as input to the output later for making predictions.

The updating procedure can be written as

$$h_t = \sigma(X_i W_{xh} + h_{t-q} W_{hh} + b_h) \quad (7)$$

In this scenario, q is the number of skipped hidden cells. At each window position of the input matrix, the attention mechanism, as proposed in [32] effectively discerns a weighted mixture of specific hidden states. This process allows the system to understand and emphasize the importance of certain elements over others in the data. The attention weights $\alpha_t \in \mathbb{R}^p$ at the current timestamp t are calculated as

$$\alpha_n = AM(H_t^R, h_{t-1}^R) \quad (8)$$

where $H_t^R = [h_{t-p}^R, \dots, h_{t-1}^R]$ represents the matrix stacking the hidden representations of RNN column-wise, and AM is the similarity function, which operates as a cosine or dot product. The RSC and LSTM outputs are combined using the dense layer and the final output is the concatenation of the weighted context vector $c_t = H_t \alpha_t$ and the hidden representation of the last window h_{t-1}^R .

$$h_t^D = W[c_{t,ht-1}] + b \quad (9)$$

where h_t^D represents the output from the 1D convolutional layer.

4.4 Time-Window Size

Segmenting sensor streaming data into sequences of moving windows is a common preprocessing step used in machine learning and artificial intelligence applications. This approach helps to break down the continuous stream of data into smaller, manageable chunks that can be used as inputs to a model [63]. By using moving windows, the

model can consider a sliding window of historical data when making predictions or decisions, which can help it better capture the temporal dynamics of the data. The choice of window size and step size can have a significant impact on the performance of the model, so these parameters should be carefully selected based on the specific requirements of the task at hand. In the RUL estimation, the sliding window approach is crucial. Since in prognostics, information degrades over time, the sliding window approach helps to contain this information. Moreover, the window size should be wide enough so that it can accommodate the maximum amount of data. The dimensions of the time window play a crucial role in influencing the process of estimating the RUL. Previously, in [27] window sizes of 15 and 30 were compared; however, a window size of 30 outperformed the window size of 15. Moreover, the sequences in the test data have a minimum running cycle length of 31 cycles; hence, the window length should be fewer than 31 cycles. Therefore, we adhered to the same strategy of a window size of 30 throughout our experiments. An example of a sliding time window with length T_{tw} is shown in Figs. 2 and 3. Figure 3 illustrates that for every run-to-failure time series, there is a time window that sweeps from the start point to the endpoint of the series. All the data from the previous time window is compiled into a high-dimensional feature vector and fed as an input to the network at each time step. The windows are processed one at a time and the results are combined

to produce a complete picture of the data being monitored. This approach allows for a more detailed analysis and helps to identify trends or patterns over time. The given input time series is formulated according to [51]. The input sequence can be expressed as:

$$W(X_i) = \{(X_i^{t-w_{tw}+1}, X_i^{t-w_{tw}}, \dots, X_i^t)\}_{t=w_{tw}}^{T_i}$$

$W(X_i)$ represents the segmented data sequences that are used as input to the TCRSCA model and T_{tw} represents the window size.

5 Dataset and Pre-processing

5.1 C-MAPSS

The Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset [64], developed by the NASA Glenn Research Centre, is a benchmark dataset that has been widely utilized in various studies [1, 27, 51]. As displayed in Table 2, C-MAPSS consists of four distinct sub-datasets, each offering a wealth of information concerning the progressive deterioration of turbofan engines. These engines are classified into four different groups based on their operational conditions and fault modes. Each group's data is split into a training set, which includes all data up to the end of

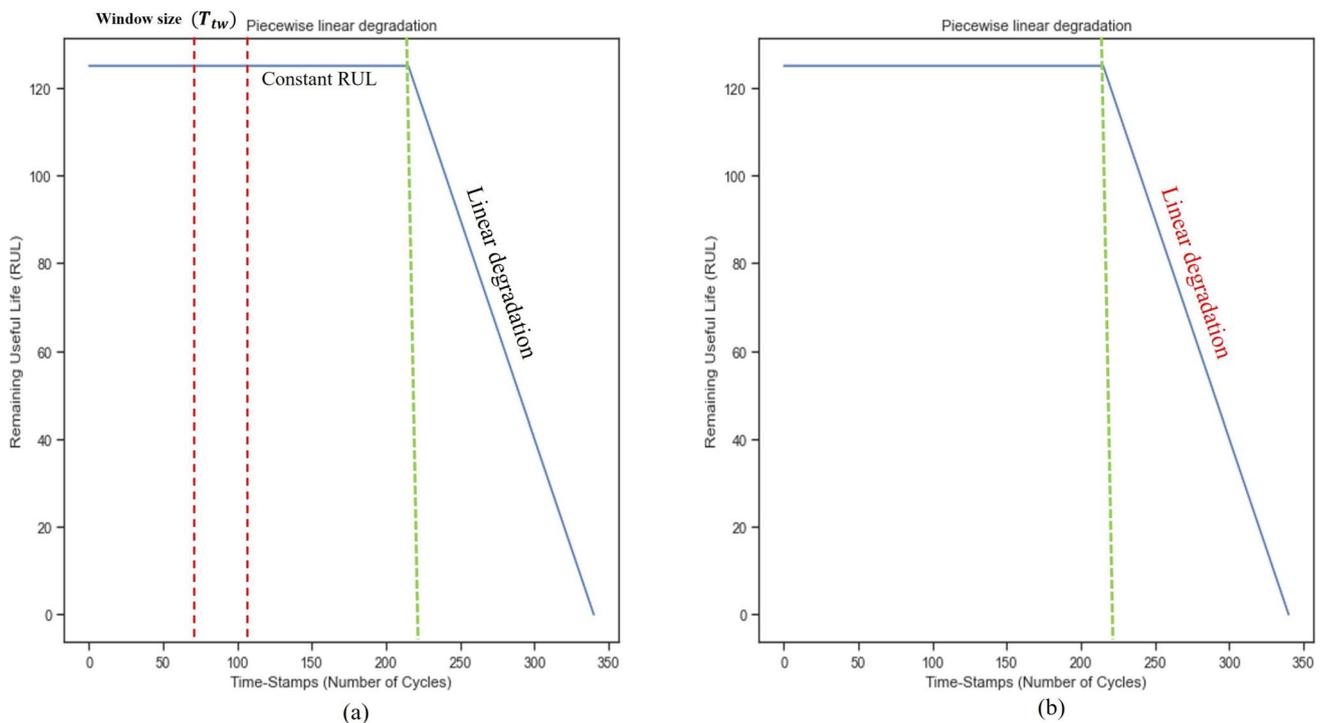


Fig. 2 Example of data segmentation using time window approach

Fig. 3 Time Window approach to split the data. A window length of 30 has been shown to work best for the FD001 dataset

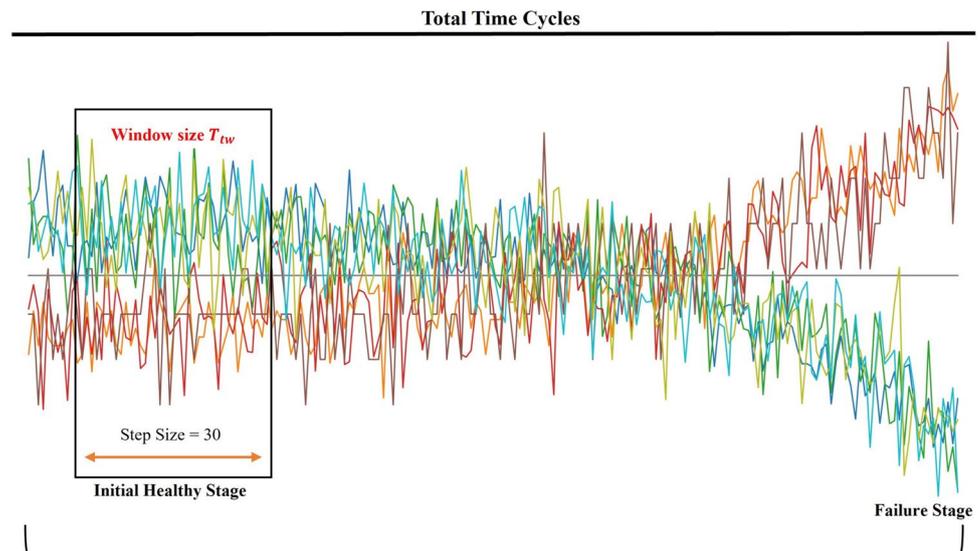


Table 2 C-MAPSS dataset details

Dataset	Number of engines in training set	Number of engines in test set	Operating conditions	Failure modes	Training samples	Test samples
FD001	100	100	1	1	17,731	9211
FD002	260	259	6	1	48,819	23,999
FD003	100	100	1	2	21,820	12,697
FD004	249	248	6	2	57,763	31,725

the engine's lifespan, and a test set that concludes before the engine's failure. The aim here is to leverage the sensor data in the test set to predict each machine's RUL. The dataset features a wide range of variables, including the engine number, the number of operational cycles, three operational settings, and 21 sensor measurements, along with each engine's RUL for validation purposes.

- Training set consists of the engine's run-to-failure data.
- Test set consists of the engine's functioning data recorded without the failure events.
- RUL labels are the information regarding the remaining cycles left for each engine in the test data.

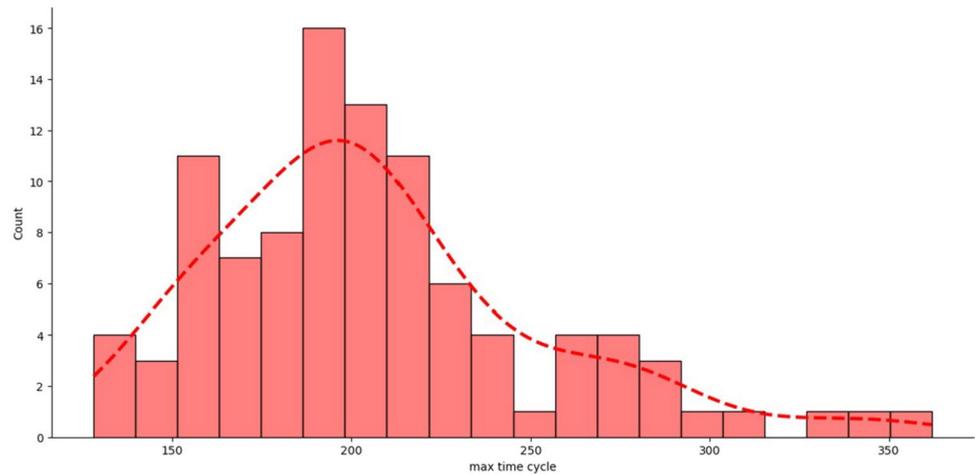
Each sub-dataset includes multiple sets of turbofan engine degradation data, encompassing three setting parameters that greatly affect engine performance and 21 sensor signals. These signals, sourced from different turbofan engine components, encapsulate degradation details like temperature, pressure, and speed. The dataset operates under the assumption that the turbofan engines are in prime operational condition at the onset. However, as the run cycle prolongs, the engines experience degradation until they are unable to

execute regular functions, a condition termed 'failure'. The maximum time cycle distribution is shown in Fig. 4. We observed that the maximum time cycles for an engine fall somewhere between 90 and 210 before any failure occurs.

The C-MAPSS dataset includes only the final cycle's RUL for each engine in the test data. To investigate the correlation between sensor data and RUL, the training data is augmented with RUL labels. The degradation of turbofan engines is a slow process that is difficult to detect in its early stages. However, when a fault occurs, the engine's performance declines, leading to its eventual failure when the RUL reaches.

The intricacy of the sub-datasets escalates proportionally with the rise in the count of operating and failure scenarios, rendering FD002 and FD004 notably complex datasets. The training and test data from each sub-dataset are concatenated along the temporal axis, resulting in a final compiled matrix of dimension $h \times 26$. The matrix is organized such that the first column represents the unit or engine number, followed by the operational cycles in the second. The third, fourth, and fifth columns chronicle the varying operational conditions. Different sensor readings are present in the subsequent 21 columns. The actual RUL values are given independently. The training set includes the complete

Fig. 4 The overall distribution of maximum time cycles for all the engines in FD001 dataset zero. Thus, this paper uses the piece-wise linear degradation function [39] as shown in Fig. 2b to set the RUL labels for the training data



lifecycle of the engines until their ultimate failure, whereas the test set engines cease operation at an arbitrary time before the point of failure. In the context of this experiment, our performance evaluation is solely based on the training set. A random selection process is implemented, reserving 90 engines for training and preserving the remaining 10 for testing purposes. Also, according to [11] the degradation occurs during the latter stages of the 120th to 130th cycles. Therefore, we set the maximum value of RUL to 125.

5.2 Sensor Selection

There are a total of 21 sensors that monitor engine degradation. However, the distribution of health cycles of all the engines is not the same. In the given dataset we have 81%

of engines that are healthy meaning that they operate at their full capacity, 14.2% unhealthy engines that operate close or little over 50% of their full capacity, and finally, we have 0.4% of engines whose health conditions are not determined.

The overall data distribution is shown in Figs. 5, 6 and 7. Some sensors do not show any change in their behaviour over time, thus providing no useful information like setting 1, Sensor 1, Sensor 5, Sensor 10, Sensor 16, Sensor 18, and Sensor 19, as shown in Fig. 7. Since these sensors do not provide any information regarding the engine states, they can cause issues during modelling. Therefore, we drop these sensors and consider the remaining 14 sensors, as described in [51]. Three operational settings, s^1 , s^2 , and s^3 impact the performance of each engine. There are 21 sensors: m_1, m_2, \dots, m_{21} collect different information related to the engines during run-time.

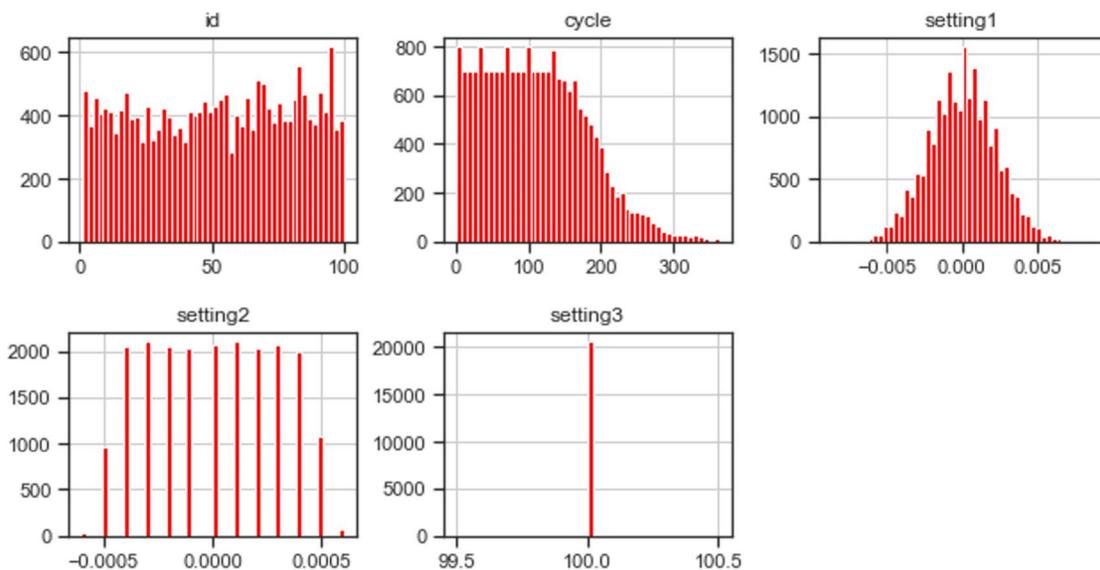


Fig. 5 The figure illustrates the distribution of individual signals like run-to-failure cycles, and three operational settings in the FD001 data subsets, respectively. The id shows the maximum lifetime of the 100

different units within FD001. It is seen in the figure that setting 3 displays no information

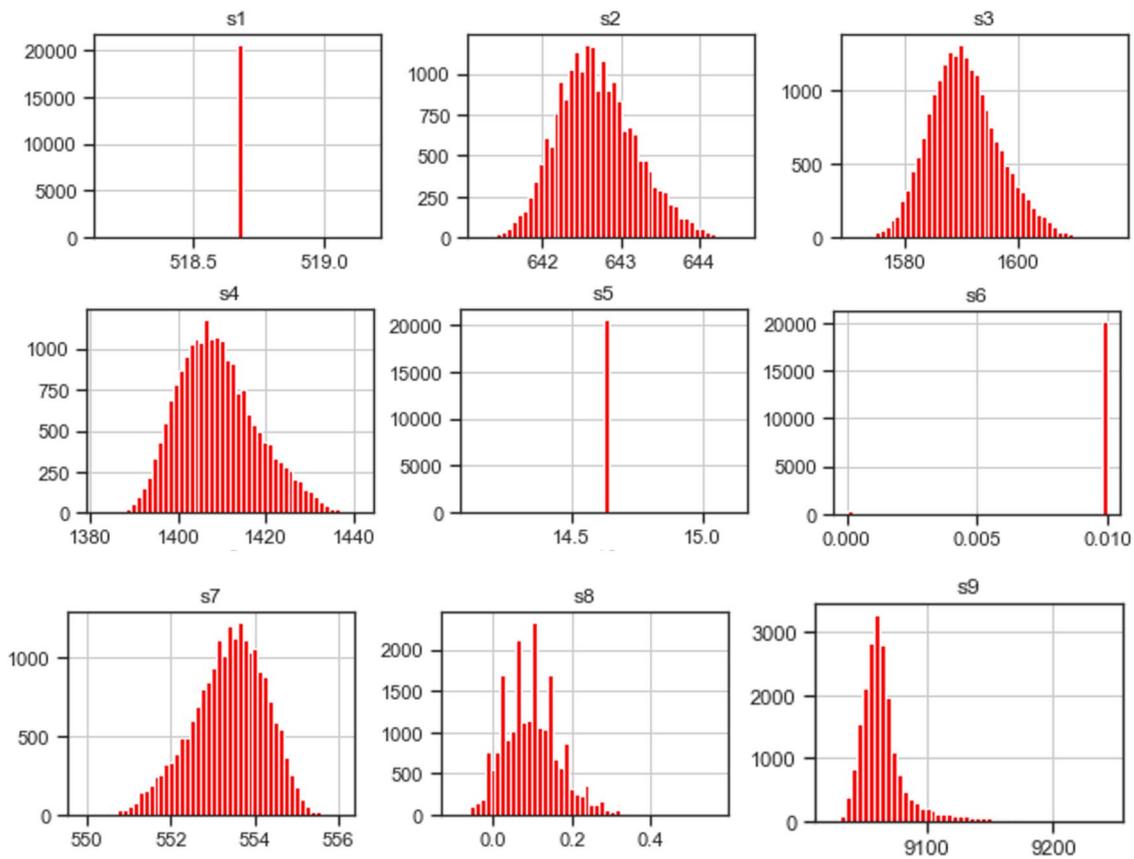


Fig. 6 The figure illustrates the distribution of individual signals in the FD001 dataset. Sensors [1, 2, 3, 4, 5] record the temperature measurements of different types of fan inlets and outlets. Sensor [6,

7, 8, and 9] measure the pressure, speed of physical fan, and physical fan speed. It is seen in the figure that sensors [1, 3, 5, and 6] display no information

However, the collected data is not free from noise. Each engine has its own unique lifespan and failure trend, indicating that its momentum does not synchronize over time.

5.3 Normalization

The raw sensor data is complex, with fluctuating magnitudes and multidimensional values, making it difficult to train a neural network. Therefore, we normalize every feature value within the range of [0, 1] by implementing min–max normalization. The new feature distribution is shown in Fig. 8. We use the sliding window approach [51] for the input time series to collect as much information as possible. We formulate the input multi-variate time series according to [65]. Let $X_i = X_i^T$ be the input time series of the i th engine. The sliding window generates an input data stream, which is expressed as:

$$T(X_i) = \left\{ (x_i^{t-T_{tw}+1}, x_i^{t-T_{tw}}, \dots, x_i^t) \right\}_{i=T_{tw}}^{T_i} \tag{10}$$

where T denotes the segmentation function and T_{tw} denotes the size of the time window. The normalization is performed as follows.

$$y_j^i = \frac{x_j^i - x_{min}^i}{x_{max}^i - x_{min}^i} \tag{11}$$

where x_j^i denotes the j th output of the i th sensor, x_{min}^i denotes the minimum values of all the outputs at i th sensor, and x_{max}^i denotes the maximum values of all outputs at i th sensor. The FD001 and FD003 datasets have a process of filtering out non-essential sensor information to increase computational speed and shorten training time. The datasets include 21 sensors, identified by numbers 1–21, and have three modes of operation, designated as $s^1, s^2,$ and s^3 .

5.4 Correlations with Engine RUL

Feature selection is very important during the pre-learning stage, as it removes unnecessary computation by selecting the optimal subset and therefore improves the model’s accuracy. Pearson’s correlation coefficient analysis [66] is a filter

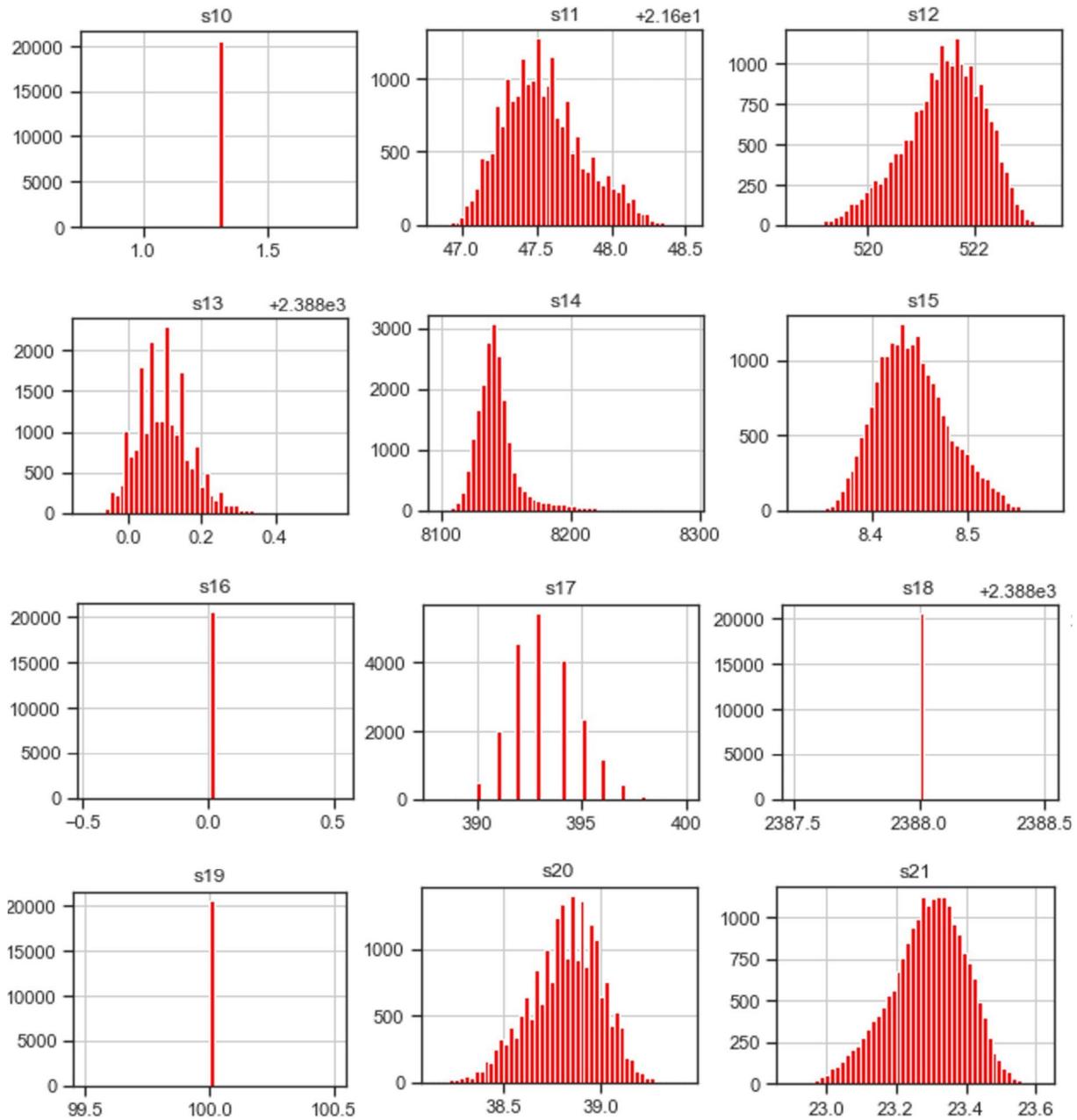


Fig. 7 The figure illustrates the distribution of individual signals in the FD001 dataset. Sensors [10, 11, 12, 13, 14] record the engine pressure, static pressure, the ratio of fuel flow, correct fan speed, and correct core speed. Sensors [15, 16, 17, 18, 19, 20, and 21] meas-

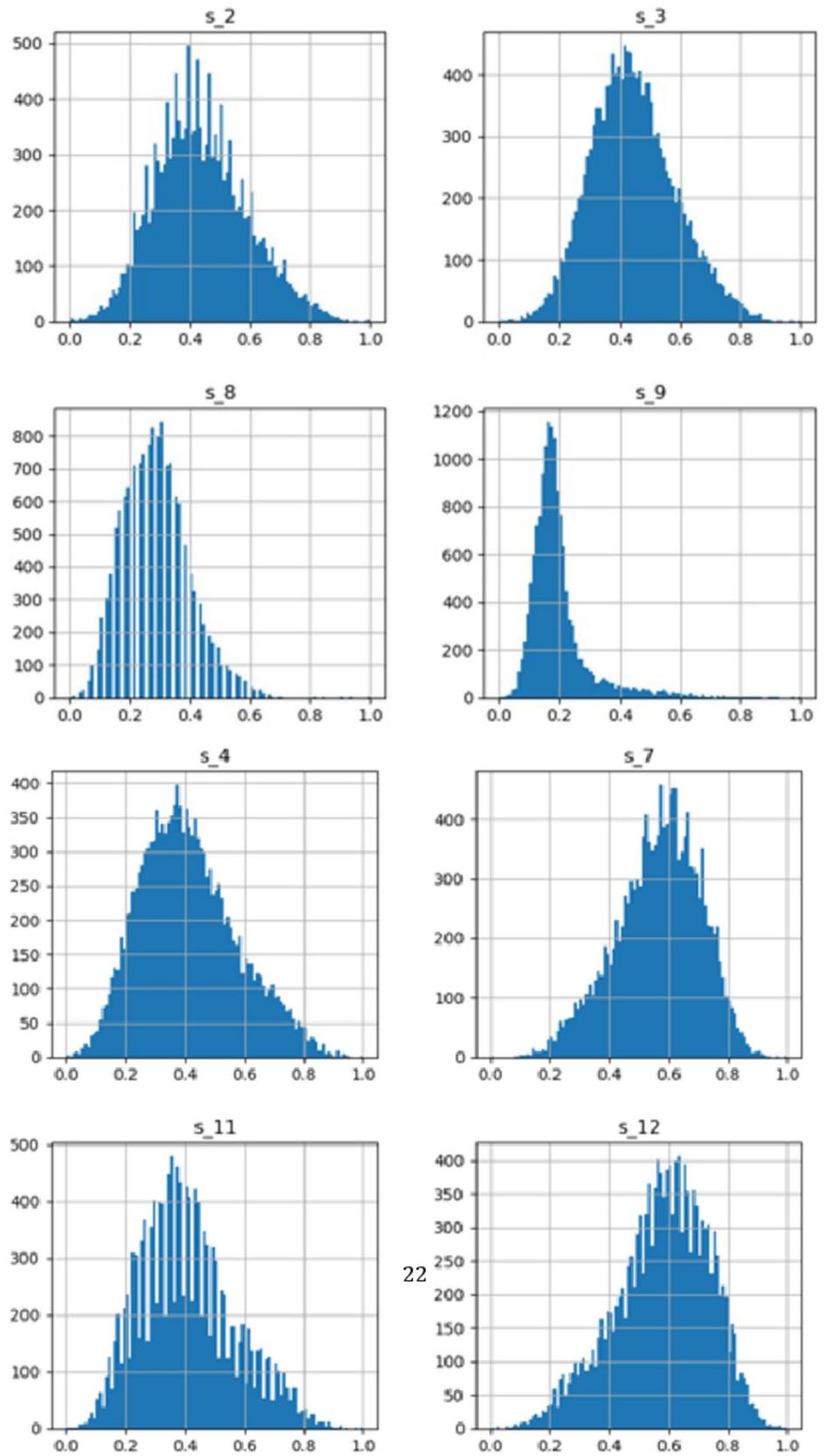
ure the bypass ratio, burner fuel–air ratio, bleed enthalpy, demanded fan speed, demanded speed of the physical fan, physical fan speed, demanded corrected fan speed, and coolant blends. It is seen in the figure that sensors [10, 16, 18, and 19] display no information

method that is used to determine the degree of correlation between two variables in a dataset. This method helps us to identify variables that have a strong linear relationship with each other, which is very helpful in selecting features for machine learning models or in identifying patterns in data. According to [67], Pearson’s correlation coefficient is a statistical measure that quantifies the level of linear correlation between two variables.

$$\hat{a} = \frac{\sum_{l=1}^n (Q_l - \bar{Q})(P_l - \bar{P})}{\sqrt{\sum_{l=1}^n (Q_l - \bar{Q})^2} \sqrt{\sum_{l=1}^n (P_l - \bar{P})^2}} \tag{12}$$

Here, \hat{a} symbolizes a correlation coefficient, which illustrates the relationship between two variables, Q and P. This coefficient can vary between -1 and 1 . A value of -1

Fig. 8 The normalized data distribution of FD001 dataset



22

signifies a perfect negative relationship, meaning if one variable increases, the other decreases. A value of 0 indicates no correlation or relationship at all. A value of 1 demonstrates a perfect positive correlation, implying that an increase in one variable will increase the other variable as well. In this equation, Q_i and P_i denote two sample points, while n stands for the total number of samples. The symbols \bar{Q} and \bar{P} are used for the average values of Q and P , respectively. When the correlation between Q and P gets close to 1, it suggests that the two variables tend to increase together. Figure 9 shows the heat map for the C-MAPSS dataset. The correlation between sensor data from s_1 to s_{21} is the most important input variable to consider. In Fig. 9, the correlation coefficient is highlighted in dark green whenever the number of rows and columns is equal, as it represents the correlation coefficient for itself. Green denotes positive correlations, pink denotes negative correlations, and white denotes no correlation. Due to their propensity to increase

or decrease together, Q and P have a high correlation in the correlation analysis. But P doesn't always tend to increase in relation to Q .

5.5 RUL Labelling

The process of correctly adjusting or "rectifying" the RUL values, also known as training labels, from the training data-set is crucial. A method called the piece-wise linear degradation model is typically used to figure out these RUL labels. This approach is effective when applied to the C-MAPSS dataset [68]. The task of correctly adjusting the values of RUL from the training data set is extremely important. We often use a method known as the piece-wise linear degradation model to do this. This method has been tested and shown to work well with the C-MAPSS dataset. A fixed threshold of 125 as suggested by the literature [68] is assigned to RUL for several cycles at the beginning known

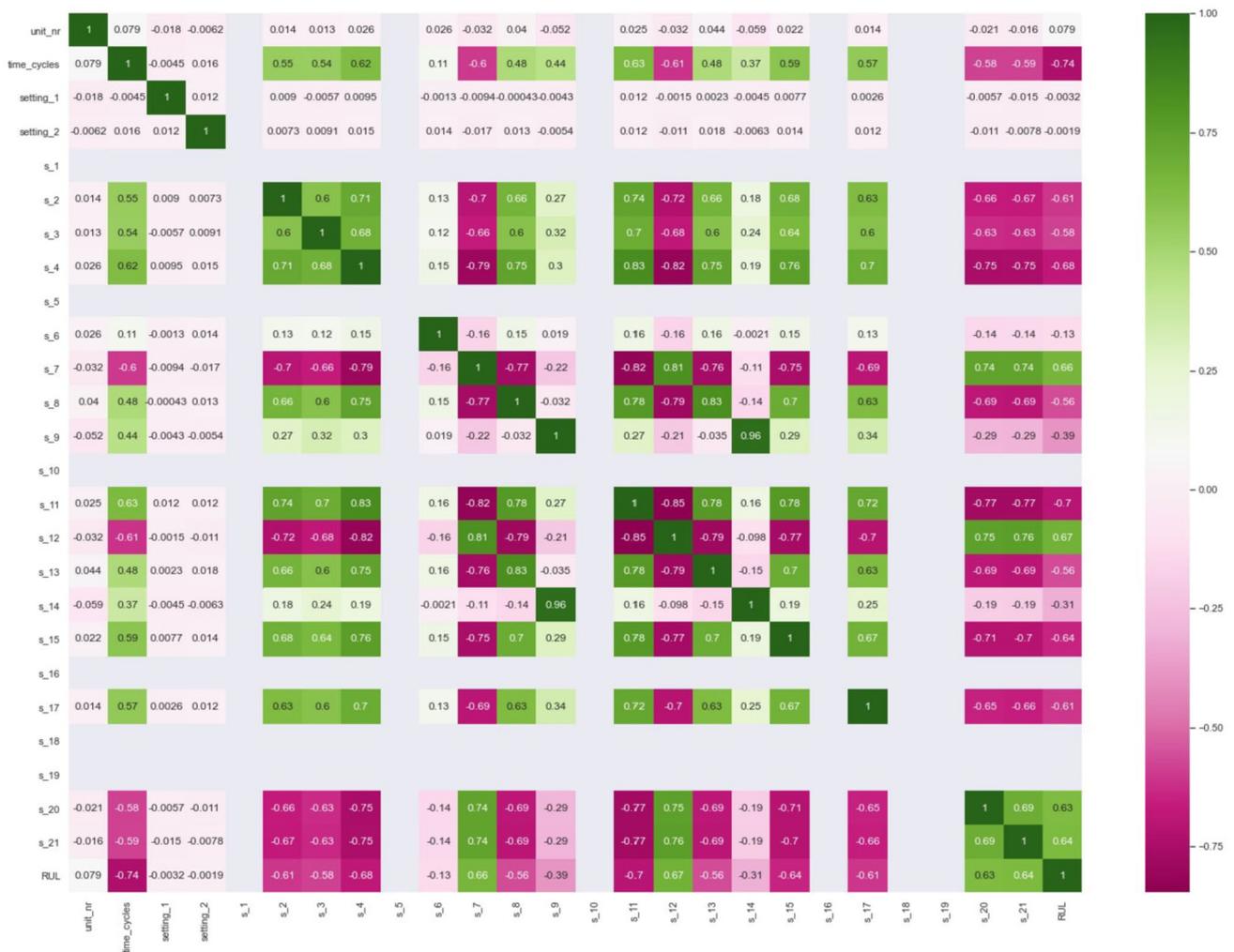


Fig. 9 Heat maps diagram of the correlation matrix for the C-MAPSS dataset

as early RULs. This rectification is necessary since the RUL should not decrease at the start of the degradation because the system is always deemed to be in good working order. Each engine follows a linear degradation as it reaches the early RUL, as shown in Fig. 2b. The rectified RUL labels are computed as follows:

$$l_r = \begin{cases} l_i, & l_r > l_i \\ l_i, & L_r \leq l_i \end{cases} \quad (13)$$

where l_r denotes the rectified RUL labels, l_i denotes the original RUL labels of the training dataset, and l_i denotes the initial RULs.

5.6 Piece-Wise Linear Degradation Model

The piecewise linear degradation model, as introduced in [68], enables us to estimate the RUL of a system. This model assumes that the deterioration rate remains constant over time. Predicting the exact RUL of a well-functioning engine can be challenging, particularly when the engine fails suddenly without any prior signs of issues. However, the linear degradation model provides a reasonable estimate of the RUL based on the current condition of the engine and the rate at which it is deteriorating. In this study, the RUL of an engine was analysed. As depicted in Fig. 2b, the RUL initially remains constant and then gradually decreases. To help the network converge, a successful approach is to use a piece-wise RUL model, which has been proven to work well with the C-MAPSS dataset [60]. For this experiment, we have set the maximum RUL to 125, following the examples of other research studies.

5.7 Evaluation Metrics

The two commonly used evaluation metrics used in the literature are the root mean square error (RMSE) and the scoring function. We adopted the same approach for evaluating the proposed method. The error of the i^{th} component of an input time-series is estimated as follows:

$$E_i = \text{Estimated}_{RUL} - \text{Actual}_{RUL} \quad (14)$$

The RMSE is most commonly used in regression and is calculated as follows.

$$RMSE = \sqrt{\frac{i}{n} \sum_{i=1}^N E_i^2} \quad (15)$$

where N denotes the total number of samples and E_i denotes the error between the true RUL labels and the estimated ones. The smaller value of RMSE indicates higher estimation accuracy and lower estimation error.

Another metric adopted is the scoring function, which is defined as follows:

$$\text{Score} = \begin{cases} \sum_{i=1}^N (e - \frac{E_i}{13}), & E_i < 0 \\ \sum_{i=1}^N (e - \frac{E_i}{10}), & E_i \geq 0 \end{cases} \quad (16)$$

The lower score value represents higher prediction accuracy and lower estimation error, the same as RMSE. However, unlike the RMSE scoring function, which involves penalizing early and late predictions, because late predictions are more likely to result in serious disasters and significant economic losses, they must be penalised more severely than early predictions. As a result, the scoring system will severely penalise large errors. It's worth noting that the scoring function is a cumulative function with a value proportional to the number of samples. Moreover, one of the major drawbacks of the scoring function is its sensitivity to outliers, as a single outlier point can be responsible for a drastic change in the scoring function. Thus, making RMSE the most favourable metric for evaluation. Figure 10 shows the comparison between RMSE and the scoring function. Since the scoring function is exponential and lacks error normalisation, a single outlier has a substantial impact on its value. As a result, it is noted that a singular scoring function cannot accurately assess an algorithm's performance. Therefore, both the performance metrics (RMSE) and scoring functions are aggregated to ensure that the algorithm predicts accurately.

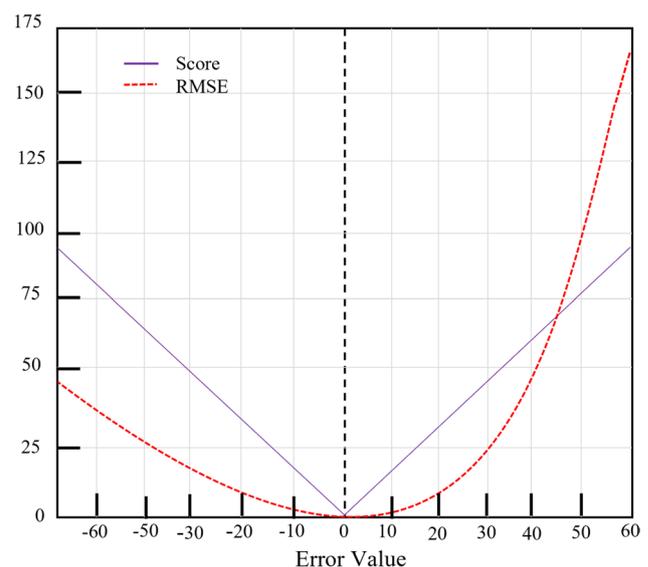


Fig. 10 Comparison between RMSE and scoring function

6 Network Architecture

Our proposed network structure is illustrated in Fig. 11. It has three main parts: 1D convolutional layers, an RSC, and an attention mechanism. First, the 1D CNN is used to recognise and extract temporal features, or time-dependent patterns, from the input time series data. These patterns help in understanding the health status of the machine, detected through sensor readings. The outputs from this stage are what we refer to as abstract features. Next, the RSC, which uses an LSTM, learns the time related dynamics of the machine's health status. The RSC output and the abstract features from the 1D CNN are combined through a fully connected layer, resulting in a set of enhanced feature maps. Lastly, we have the attention mechanism, which plays a vital role in identifying the most relevant features to estimate the RUL of an engine. The RSC output is fed into the attention mechanism, which generates a set of weighted feature maps. Each weight in these maps reflects the importance of a specific feature. The final step is a fully connected layer, which uses these weighted features to predict the RUL. The dataset we use includes three operational conditions and 21 sensor readings, as discussed in Sect. 5. However, based on the evaluation in Sect. 5.2, we only selected 14 of those sensor readings. Hence, our input comprises 17 channels in total, with a sequence length of 90. The subsequent sections offer

more detailed explanations of each part of our proposed architecture.

1D convolutional layers: TCRSCANet employs three 1D CNN layers, followed by batch normalisation and the ReLU activation function. These layers can find spatial correlations on the C-MAPSS dataset and extract useful information from the input sequence by using sensor readings from different engine parts. Each convolutional layer performs a set of convolutional operations on the input data, followed by a pooling operation to down-sample the feature maps. The 1D CNN generates high-level feature maps that capture important temporal patterns in the input sequence.

Recurrent Block: The recurrent block models temporal dependencies in input sequences using multiple LSTM cells, which are a type of RNN. The LSTM cells have an internal state that can store information about past inputs, which allows the network to maintain a memory of the temporal patterns in the sequence. The output of the recurrent block is a sequence of hidden states, which contain information about the temporal dynamics of the input sequence.

Skip component Block: The skip component is a key part of our network, which helps to improve how information moves through the system. It does this by creating a direct connection, or "shortcut," between the input and output of the 1D CNN layers and the output from the recurrent component. This shortcut allows the network to learn from both immediate and distant relationships within the input data, which can boost the accuracy of the network's predictions. This means our network is capable of understanding

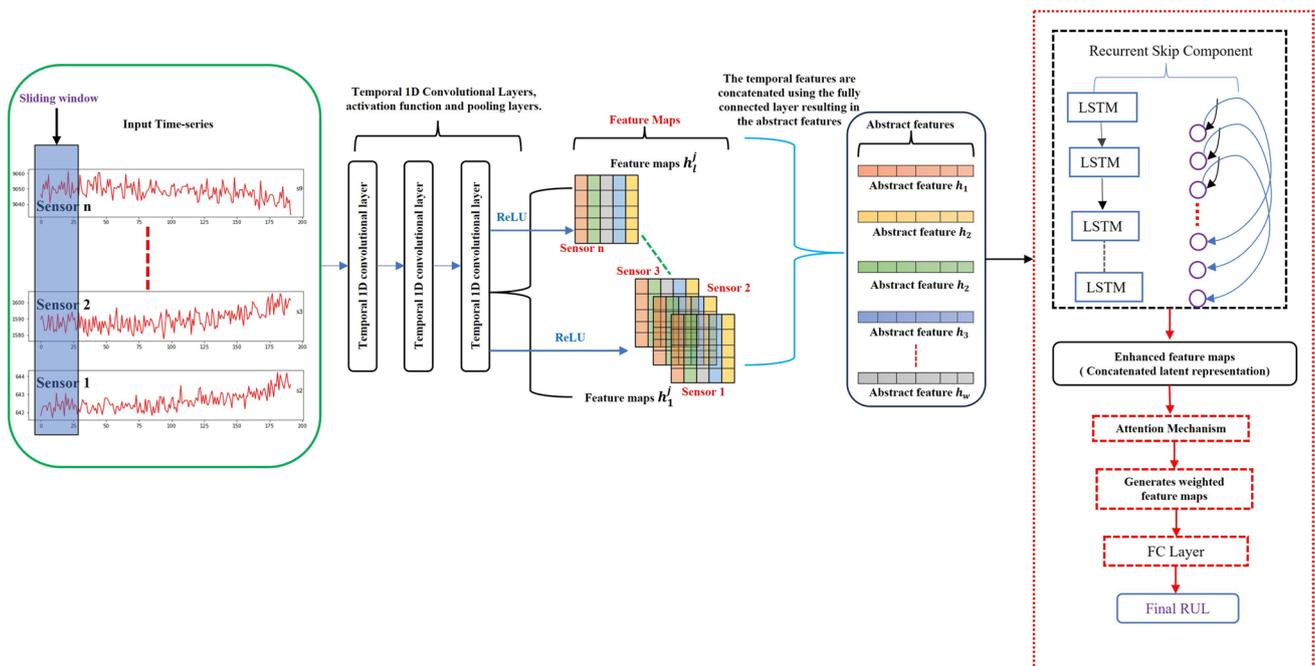


Fig. 11 The overall architecture of the proposed TCRSCANet

both immediate patterns and changes over time, making its predictions more reliable.

Attention Mechanism: An attention mechanism is a tool that helps the network concentrate on the most important parts of the data when trying to predict the RUL. It's a bit like highlighting the most important points in a textbook. The mechanism uses something called attention weights, which are like markers showing how relevant each piece of data is for making the prediction. The network learns these weights as it's trained and bases them on how similar the current data is to past data. The final output from the attention mechanism is a weighted sum of the hidden states; in simple terms, it takes all the data, weighs it based on importance, and sums it up. This output represents the most relevant information for predicting the RUL.

6.1 Effect of Time Window Size

The sliding time window method is an important coefficient in RUL prediction. It captures the crucial degradation

details over time. The size of the time window is measured in cycles, and it greatly impacts the prediction accuracy. In our case time window creates meaningful features within a window that serve as input. It also analyses the relationship between different data points and their relative position in time. We tested various window sizes: 10, 20, 30, 40, 50, and 60 cycles. However, some engines lacked sufficient data for the larger window sizes, so we excluded them for a more accurate analysis. In previous research, it's been found that a window size of 30 does a better job than a window size of 15 when trying to predict the RUL of a machine [27]. Also, because the test data sequences have a minimum cycle length of 31 cycles, it's crucial to pick a window size that's less than 31 cycles. In our study, we tried different window sizes ranging from 20, 30, 40, 50, to 60 on all four sub-datasets. The RMSE values with different window sizes on C-MAPSS datasets are shown in Fig. 12. We found that at first, a larger window size helps to make predictions more accurate (the RMSE, a measure of accuracy, decreases), but after a point, accuracy starts to drop off. Why is that? Well,

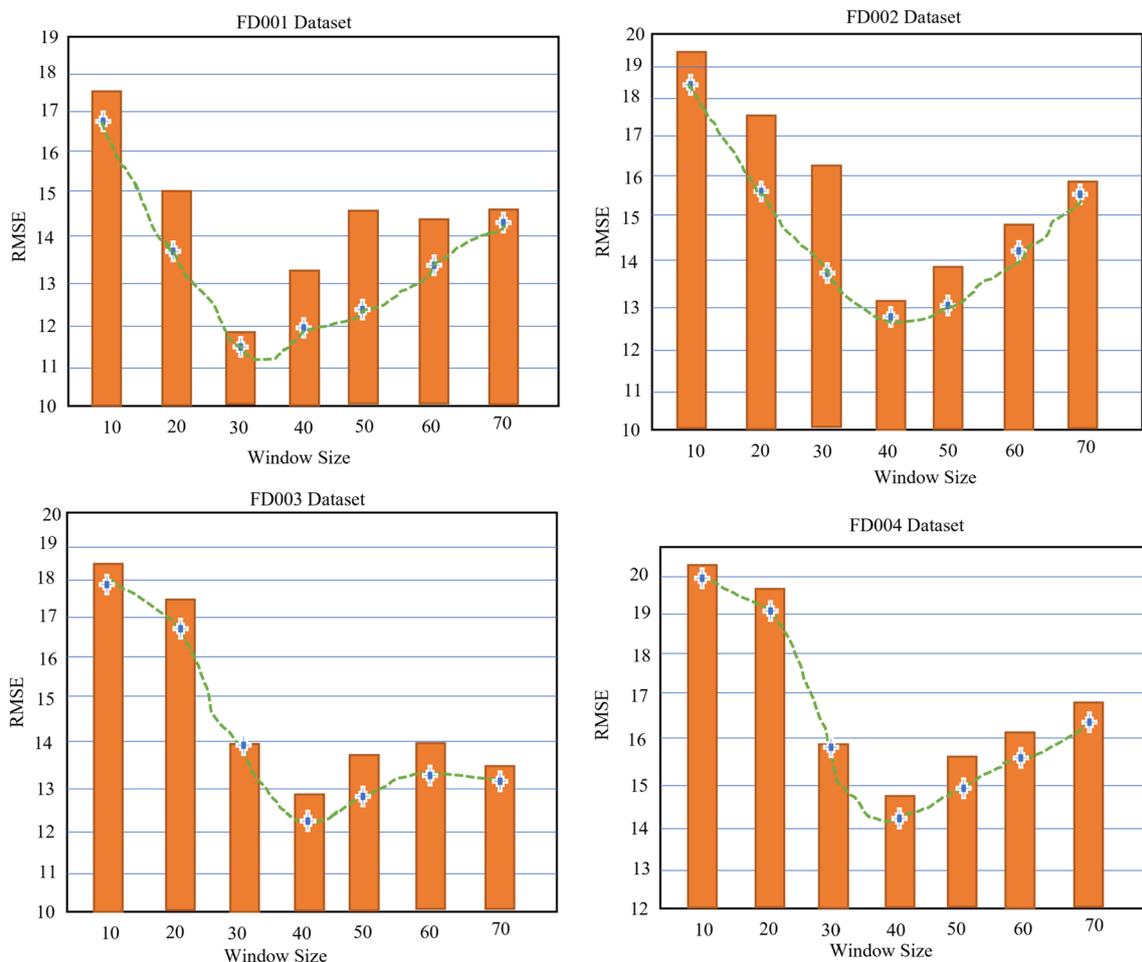


Fig. 12 Impact of time-window on all four sub-datasets of C-MAPSS dataset

it seems that the current state of a machine is mostly linked to the most recent data, and this link gets weaker the farther back you go. So, when the window size is too big, it's like trying to understand the current page of a book by reading too many pages that came before it—the extra information just becomes noise and makes things harder. Our experiments show that as the window size increases, the results vary more, indicating that the extra noise is indeed making it harder to train the model. For the first dataset (FD001), a window size of 30 gave the best results, while a window size of 40 worked best for the other three datasets.

6.2 Results and Comparison

We compare how well the TCRSCANet works with other methods on all parts of the C-MAPSS dataset using the performance metrics of RMSE and Score shown in Table 3. Our findings show that the suggested strategy improves the RUL prediction significantly and performs better in various situations. In particular, the suggested technique obtains smaller RUL standard deviations during the degradation process, even if its score is comparable to the top score in prior research on the FD001 and FD003 datasets. With the lowest prediction errors, the suggested technique performs noticeably better than previous methods, according to the results of the FD002 and FD004 datasets. It decreased noticeably when compared to the best earlier outcomes. More than 23% for RMSE and 49% for the score on the FD002 dataset, and 31% for RMSE and 34% for the score on the FD004 dataset, respectively. This successfully validates the accuracy and dependability of the proposed TCRSCANet in practical PHM applications. Also, FD002 and FD004 have bigger prediction errors for all known methods compared to the two other datasets. This is because the operating conditions are more complicated. Because it extracts the characteristics related to degradation, the TCRSCANet has the smallest

difference in prediction error of the four subsets. Furthermore, the TCRSCANet can match the prediction accuracy of other approaches under normal and complex circumstances. The notable performance gains highlight the TCRSCANet's superiority in challenging situations. Additionally, it should be noted that most of the industrial machinery operates under a range of challenging circumstances in the actual world, making the suggested approach well-suited for these applications.

One sample engine from each of the four datasets—FD001, FD002, FD003, and FD004—is chosen to represent the prediction results. The evolution of the anticipated RUL around the actual RUL is depicted in Fig. 13. TCRSCANet is capable of accurately capturing the degradation development trend. At the final failure stage of the degradation progression, the fault characteristics amass. When an engine is getting close to the final failure stage, the prediction error gradually drops. In Fig. 13 it is also seen that the anticipated RUL is typically lower than the actual RUL, reducing the possibility that overestimating the RUL may result in needless maintenance expenses. Remarkable prediction performance is achieved by the proposed TCRSCANet in the datasets FD001, FD002, FD003, and FD004 under diverse operating conditions.

7 Ablation Study

We proposed a new architecture called TCRSCANet, which is designed to predict the RUL of turbo-jet engines. The architecture is composed of three key components: attention mechanisms, 1D CNNs, and the RSC. To better understand the contributions and significance of these modules in the model's predictive capacity, we conducted a systematic ablation study.

Table 3 RUL prediction results in comparison with different methods for the C-MAPSS dataset

Methods	RMSE				Score			
	FD001	FD002	FD003	FD004	FD001	FD002	FD003	FD004
LSTM [46]	16.14	24.49	16.18	28.17	338	4450	852	5550
DCNN [51]	12.61	22.36	12.64	23.31	273.7	10,412	284.1	12,466
DAG [69]	11.96	20.34	12.46	18.67	229	2730	284	3370
HDNN [27]	13.02	15,024	12.22	18.16	245	1282.42	287.72	1527.42
DATCN [70]	11.78	16.95	11.56	18.23	229.48	1842.38	257.11	2317.32
RBPF [71]	15.94	17.15	16.17	20.72	383.39	1226.97	375.29	2071.51
Attent-Based LSTM [60]	14.53	–	–	27.08	322	–	–	5649
MDSA [72]	11.71	13.32	11.47	14.38	209	1058	1618	
BIGRU-TSAM [38]	12.56	18.94	12.45	20.47	213.35	2264.13	232.86	3610.34
KGHM [73]	13.18	13.25	13.54	19.86	250.99	1131.03	333.44	3356.10
DTBA [74]	12.25	17.08	13.39	19.86	198	1575	290	1741
Proposed TCRSCANet	10.43	11.02	10.03	16.23	217.02	789.32	216.79	1107.96

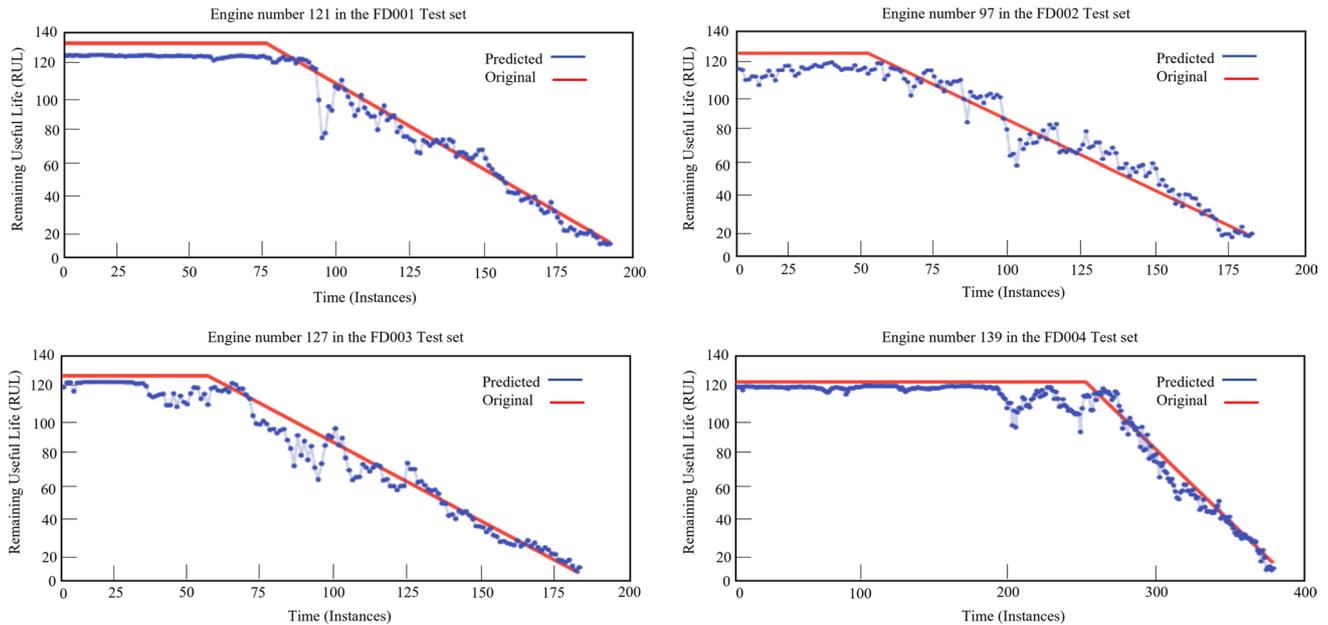


Fig. 13 Remaining useful life (RUL) prediction results on four sub-datasets

RSC: The RSC is an integration of LSTM that aims to capture the time-related complexities of machine health dynamics. One important feature of the RSC is its ability to create a shortcut connection between the 1D CNN layers and the recurrent output, which helps the model understand both immediate and temporally distant relationships in the input data. It was found that the prediction accuracy decreased when the RSC was not included, and the model was trained without it. Comparing the RSC-integrated model with its modified version brought attention to the critical role of the component in improving the accuracy of the model.

Attention Mechanism: Removing the attention mechanism and training the model resulted in reduced prediction accuracy because it was unable to identify essential data points. This comparative analysis reinforced the component's vital role in the architecture. Another aspect of our research explored the influence of window sizes on prediction accuracy. Through systematic testing, we identified the best window sizes for different datasets, emphasising the significance of this parameter in timeseries data tasks. This module is responsible for determining the relative significance of features and directing the model's focus towards salient aspects of data relevant to RUL prediction. To create weighted feature maps, the mechanism emphasises features vital for prediction.

In the TCRSCANet design, the ablation analysis clearly shows that the RSC and attention mechanism play the most important roles. Combining these two components is not only a good idea, but it is crucial for getting the best possible

predictive performance when it comes to estimating the RUL.

8 Conclusion

This paper introduces a data-driven deep learning framework, TCRSCANet, designed to enhance the accuracy of RUL prediction. This new method uses the best parts of 1D CNNs along with recurrent skip parts and an advanced attention mechanism to efficiently process and analyse data. Our testing of the model on the standard C-MAPSS dataset shows that it can identify and prioritise key moments in time that have a big effect on RUL outcomes. The recurrent skip part of the TCRSCANet finds the temporal dependencies in the data, and the attention mechanism learns the links between different features. It then uses attention to sequence to learn how the weights affect things at different times. This enables the model to focus on the sequences that significantly affect the weights at different time instances, pinpointing the key degradation trends that are vital for precise RUL estimation. We also investigated the piece-wise degradation method and different window sizes that can impact the RUL prediction accuracy of the proposed model. Our results show that TCRSCANet predicts the RUL more effectively when compared to other state-of-the-art methods. Deep learning methods assume that online data must have a distribution that is comparable to the distribution of historical data. However, in real industrial processes, it is challenging to accomplish. In our future work, we would like to investigate

the imbalanced training datasets when it comes to RUL prediction using generative adversarial networks (GANs). Also, we would like to explore GNN-based architectures and try to adapt our models to more industrial-focused scenarios.

Acknowledgements This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Numbers SFI/16/RC/3918 (Confirm) and SFI/12/RC/2289 P2 (Insight). For the purpose of Open Access, the author has applied a CC BY public copyright license to any Author Accepted Manuscript version arising from this submission.

Author Contributions AW: Conceptualization, methodology, modeling, data curation, writing an original draft, visualization, investigation, validation, and comparative analysis. JGB: Supervision, reviewing and editing. MAI: Supervision, reviewing and editing.

Funding Science Foundation Ireland (SFI) under Grant Numbers SFI/16/RC/3918 (Confirm) and SFI/12/RC/2289 P2 (Insight).

Availability of data and materials The authors declare that the data (C-MAPSS dataset) supporting the findings of this study are available. <https://github.com/AbNeil/RUL-Estimation>.

Declarations

Conflict of interest The authors report no conflicts of interest, either financial or otherwise. Regarding the subject matter of this work, the authors have no competing interests to disclose. The authors declare that they have no relationships with any organizations that would benefit monetarily or otherwise from the publication of this manuscript's contents. The authors of this manuscript have no financial or proprietary stake in any of the topics covered.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Solis-Martin D, Galán-Páez J, Borrego-Díaz J. A stacked deep convolutional neural network to predict the remaining useful life of a turbofan engine. 2021. arXiv preprint [arXiv:2111.12689](https://arxiv.org/abs/2111.12689).
- Ran Y, Zhou X, Lin P, Wen Y, Deng R. A survey of predictive maintenance: systems, purposes and approaches. 2019. arXiv preprint [arXiv:1912.07383](https://arxiv.org/abs/1912.07383).
- Da Xu L, He W, Li S. Internet of things in industries: a survey. *IEEE Trans Ind Inf*. 2014;10(4):2233–43.
- Susto GA, Schirru A, Pampuri S, Pagano D, McLoone S, Beghi A. A predictive maintenance system for integral type faults based on support vector machines: an application to ion implantation. In: 2013 IEEE international conference on automation science and engineering (CASE). IEEE; 2013. pp. 195–200.
- Köksal G, Batmaz I, Testik MC. A review of data mining applications for quality improvement in manufacturing industry. *Expert Syst Appl*. 2011;38(10):13448–67.
- Hanachi H, Liu J, Banerjee A, Chen Y, Koul A. A physics-based modeling approach for performance monitoring in gas turbine engines. *IEEE Trans Reliab*. 2014;64(1):197–205.
- Vogl GW, Weiss BA, Helu M. A review of diagnostic and prognostic capabilities and best practices for manufacturing. *J Intell Manuf*. 2019;30:79–95.
- Zhang C, Yao X, Zhang J, Jin H. Tool condition monitoring and remaining useful life prognostic based on a wireless sensor in dry milling operations. *Sensors*. 2016;16(6):795.
- Lasheras FS, Nieto PJG, Cos Juez FJ, Bayón RM, Suárez VMG. A hybrid PCA-CART-MARS-based prognostic approach of the remaining useful life for aircraft engines. *Sensors*. 2015;15(3):7062–83.
- Atamuradov V, Medjaher K, Dersin P, Lamoureux B, Zerhouni N. Prognostics and health management for maintenance practitioners—review, implementation and tools evaluation. *Int J Prognost Health Manag*. 2017;8(3):1–31.
- Heimes FO. Recurrent neural networks for remaining useful life estimation. In: 2008 International conference on prognostics and health management, IEEE; 2008. pp. 1–6.
- Liao L, Köttig F. Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction. *IEEE Trans Reliab*. 2014;63(1):191–207.
- Chao MA, Kulkarni C, Goebel K, Fink O. Fusing physics-based and deep learning models for prognostics. *Reliab Eng Syst Saf*. 2022;217: 107961.
- Gebraeel N, Lawley M, Liu R, Parmeshwaran V. Residual life predictions from vibration-based degradation signals: a neural network approach. *IEEE Trans Ind Electron*. 2004;51(3):694–700.
- Khelif R, Chebel-Morello B, Malinowski S, Laajili E, Fnaiech F, Zerhouni N. Direct remaining useful life estimation based on support vector regression. *IEEE Trans Ind Electron*. 2016;64(3):2276–85.
- Wu D, Jennings C, Terpenney J, Gao RX, Kumara S. A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. *J Manuf Sci Eng*. 2017. <https://doi.org/10.1115/1.4036350>.
- LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521(7553):436–44.
- Lei Y, Li N, Guo L, Li N, Yan T, Lin J. Machinery health prognostics: a systematic review from data acquisition to RUL prediction. *Mech Syst Signal Process*. 2018;104:799–834.
- Zang Y, Shangguan W, Cai B, Wang H, Pecht MG. Hybrid remaining useful life prediction method a case study on railway D-cables. *Reliab Eng Syst Saf*. 2021;213:107746.
- Yin W, Kann K, Yu M, Schütze H. Comparative study of CNN and RNN for natural language processing. 2017. arXiv preprint [arXiv:1702.01923](https://arxiv.org/abs/1702.01923).
- Khan S, Yairi T. A review on the application of deep learning in system health management. *Mech Syst Signal Process*. 2018;107:241–65.
- Remadna I, Terrissa SL, Zemouri R, Ayad S. An overview on the deep learning based prognostic. In: 2018 international conference on advanced systems and electric technologies (IC ASET). IEEE; 2018. pp. 196–200.
- Sun W, Shao S, Zhao R, Yan R, Zhang X, Chen X. A sparse auto-encoder-based deep neural network approach for induction motor faults classification. *Measurement*. 2016;89:171–8.
- Zhao R, Wang D, Yan R, Mao K, Shen F, Wang J. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Trans Ind Electron*. 2017;65(2):1539–48.

25. Wu Y, Yuan M, Dong S, Lin L, Liu Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*. 2018;275:167–79.
26. Ayodeji A, Wang W, Su J, Yuan J, Liu X. An empirical evaluation of attention-based multi-head models for improved turbofan engine remaining useful life prediction. 2021. arXiv preprint [arXiv:2109.01761](https://arxiv.org/abs/2109.01761).
27. Al-Dulaimi A, Zabihi S, Asif A, Mohammadi A. A multimodal and hybrid deep neural network model for remaining useful life estimation. *Comput Ind*. 2019;108:186–96.
28. Peng C, Chen Y, Chen Q, Tang Z, Li L, Gui W. A remaining useful life prognosis of turbofan engine using temporal and spatial feature fusion. *Sensors*. 2021;21(2):418.
29. Zhang S, Tong H, Xu J, Maciejewski R. Graph convolutional networks: a comprehensive review. *Comput Soc Netw*. 2019;6(1):1–23.
30. Yang X, Zheng Y, Zhang Y, Wong DS-H, Yang W. Bearing remaining useful life prediction based on regression shapale and graph neural network. *IEEE Trans Instrum Meas*. 2022;71:1–12.
31. Al-Dulaimi A, Zabihi S, Asif A, Mohammadi A. Hybrid deep neural network model for remaining useful life estimation. In: ICASSP 2019–2019 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2019. pp. 3872–3876.
32. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. 2014. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473).
33. Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. pp. 7132–7141.
34. Zhou G, Zhu X, Song C, Fan Y, Zhu H, Ma X, Yan Y, Jin J, Li H, Gai K. Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. 2018. pp. 1059–1068.
35. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I. Attention is all you need. In: Advances in neural information processing systems, vol. 30. 2017.
36. Chorowski JK, Bahdanau D, Serdyuk D, Cho K, Bengio Y. Attention-based models for speech recognition. In: Advances in neural information processing systems, vol. 28. 2015.
37. Huang L, Wang W, Chen J, Wei X-Y. Attention on attention for image captioning. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019. pp. 4634–4643.
38. Zhang J, Jiang Y, Wu S, Li X, Luo H, Yin S. Prediction of remaining useful life based on bidirectional gated recurrent unit with temporal self-attention mechanism. *Reliab Eng Syst Saf*. 2022;221: 108297.
39. Qin Y, Cai N, Gao C, Zhang Y, Cheng Y, Chen X. Remaining useful life prediction using temporal deep degradation network for complex machinery with attention-based feature extraction. 2022. arXiv preprint [arXiv:2202.10916](https://arxiv.org/abs/2202.10916).
40. Lai G, Chang W-C, Yang Y, Liu H. Modeling long-and short-term temporal patterns with deep neural networks. In: The 41st international ACM SIGIR conference on research & development in information retrieval. 2018. pp. 95–104.
41. Si X-S, Wang W, Hu C-H, Zhou D-H. Remaining useful life estimation—a review on the statistical data driven approaches. *Eur J Oper Res*. 2011;213(1):1–14.
42. Sateesh Babu G, Zhao P, Li X-L. Deep convolutional neural network based regression approach for estimation of remaining useful life. In: International conference on database systems for advanced applications. Springer; 2016. pp. 214–228.
43. Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans Neural Netw*. 1994;5(2):157–66.
44. Zhang Y, Xiong R, He H, Pecht MG. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Trans Veh Technol*. 2018;67(7):5695–705.
45. Wang F, Liu X, Deng G, Yu X, Li H, Han Q. Remaining life prediction method for rolling bearing based on the long short-term memory network. *Neural Process Lett*. 2019;50:2437–54.
46. Zheng S, Ristovski K, Farahat A, Gupta C. Long short-term memory network for remaining useful life estimation. In: 2017 IEEE international conference on prognostics and health management (ICPHM). IEEE; 2017. pp. 88–95.
47. Zhang A, Wang H, Li S, Cui Y, Liu Z, Yang G, Hu J. Transfer learning with deep recurrent neural networks for remaining useful life estimation. *Appl Sci*. 2018;8(12):2416.
48. Jayasinghe L, Wijerathne N, Yuen C. A deep learning approach for classification of cleanliness in restrooms. In: 2018 international conference on intelligent and advanced system (ICIAS). IEEE; 2018. pp. 1–6.
49. Malhotra KR, Davoudi A, Siegel S, Bihorac A, Rashidi P. Autonomous detection of disruptions in the intensive care unit using deep mask R-CNN. In: 2018 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW). IEEE; 2018. pp. 1944–19442.
50. Zhao R, Yan R, Chen Z, Mao K, Wang P, Gao RX. Deep learning and its applications to machine health monitoring. *Mech Syst Signal Process*. 2019;115:213–37.
51. Li X, Ding Q, Sun J-Q. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab Eng Syst Saf*. 2018;172:1–11.
52. Yao D, Li B, Liu H, Yang J, Jia L. Remaining useful life prediction of roller bearings based on improved 1d-cnn and simple recurrent unit. *Measurement*. 2021;175: 109166.
53. An Q, Tao Z, Xu X, El Mansori M, Chen M. A data-driven model for milling tool remaining useful life prediction with convolutional and stacked LSTM network. *Measurement*. 2020;154: 107461.
54. Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M, Inman DJ. 1d convolutional neural networks and applications: a survey. *Mech Syst Signal Process*. 2021;151: 107398.
55. Remadna I, Terrissa LS, Ayad S, Zerhouni N. RUL estimation enhancement using hybrid deep learning methods. *Int J Prognost Health Manag*. 2021. <https://doi.org/10.36001/ijphm.2021.v12i1.2378>.
56. Bai S, Kolter JZ, Koltun V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. 2018. arXiv preprint [arXiv:1803.01271](https://arxiv.org/abs/1803.01271).
57. Zhang S, Zhang S, Wang B, Habetler TG. Deep learning algorithms for bearing fault diagnostics—a comprehensive review. *IEEE Access*. 2020;8:29857–81.
58. Wang S, Zhang X, Gao D, Chen B, Cheng Y, Yang Y, Yu W, Huang Z, Peng J. A remaining useful life prediction model based on hybrid longshort sequences for engines. In: 2018 21st international conference on intelligent transportation systems (ITSC). IEEE; 2018. pp. 1757–1762.
59. Xia J, Feng Y, Lu C, Fei C, Xue X. LSTM-based multi-layer self-attention method for remaining useful life estimation of mechanical systems. *Eng Fail Anal*. 2021;125: 105385.
60. Chen Z, Wu M, Zhao R, Guretno F, Yan R, Li X. Machine remaining useful life prediction via an attention-based deep learning approach. *IEEE Trans Ind Electron*. 2020;68(3):2521–31.
61. Zhang Z, Song W, Li Q. Dual-aspect self-attention based on transformer for remaining useful life prediction. *IEEE Trans Instrum Meas*. 2022;71:1–11.
62. Hernández A, Amigó JM. Attention mechanisms and their applications to complex systems. *Entropy*. 2021;23(3):283.
63. Ermschaus A, Schäfer P, Leser, U. Window size selection in unsupervised time series analytics: a review and benchmark.

64. Saxena A, Goebel K, Simon D, Eklund N. Damage propagation modeling for aircraft engine run-to-failure simulation. In: 2008 International conference on prognostics and health management. IEEE; 2008. pp. 1–9.
65. Xu D, Qiu H, Gao L, Yang Z, Wang D. A novel dual-stream self-attention neural network for remaining useful life estimation of mechanical systems. *Reliab Eng Syst Saf.* 2022;222:108444.
66. Sedgwick P. Pearson's correlation coefficient. *BMJ.* 2012;345:e4483.
67. Friendly M. Corrgrams: Exploratory displays for correlation matrices. *Am Stat.* 2002;56(4):316–24.
68. Liu H, Liu Z, Jia W, Lin X. Remaining useful life prediction using a novel feature-attention-based end-to-end approach. *IEEE Trans Ind Inf.* 2020;17(2):1197–207.
69. Li J, Li X, He D. A directed acyclic graph network combined with CNN and LSTM for remaining useful life prediction. *IEEE Access.* 2019;7:75464–75.
70. Song Y, Gao S, Li Y, Jia L, Li Q, Pang F. Distributed attention based temporal convolutional network for remaining useful life prediction. *IEEE Internet Things J.* 2020;8(12):9594–602.
71. Cai H, Feng J, Li W, Hsu Y-M, Lee J. Similarity-based particle filter for remaining useful life prediction with enhanced performance. *Appl Soft Comput.* 2020;94: 106474.
72. Lai Z, Liu M, Pan Y, Chen D. Multi-dimensional self attention based approach for remaining useful life estimation. 2022. arXiv preprint [arXiv:2212.05772](https://arxiv.org/abs/2212.05772)
73. Li Y, Chen Y, Hu Z, Zhang H. Remaining useful life prediction of aero-engine enabled by fusing knowledge and deep learning models. *Reliab Eng Syst Saf.* 2023;229: 108869.
74. Liu L, Song X, Zhou Z. Aircraft engine remaining useful life estimation via a double attention-based data-driven architecture. *Reliab Eng Syst Saf.* 2022;221: 108330.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.