Fusion-Based Dual-Task Architecture for Predicting the Remaining Useful Life of an Aeroengine

Xiao Du[1]; Jiajie Chen, Ph.D.[2]; Jiqiang Wang, Ph.D.[3]; Haibo Zhang, Ph.D.[4]; and Junhao Wen, Ph.D.[5]

Abstract: The prediction of the remaining useful life (RUL) of aeroengines is crucial for ensuring their safe operation and reducing maintenance costs. However, aeroengines are complex nonlinear systems that exhibit multiple degradation modes, making it challenging to extract predictive information from diverse feature fields. To address this issue, we propose a data-driven fusion-based dual-task architecture that explicitly models the degradation modes of aeroengines by leveraging degradation information to enhance RUL prediction. In our dual-task model, a residual network extracts feature information from observable values within a single flight, whereas a long short-term memory network captures feature information across multiple flights. These two models are fused into a unified RUL prediction model, which is subsequently fine-tuned using RUL labels reconstructed from degradation data. Evaluation of a data set containing comprehensive flight information demonstrates that the proposed method improves prediction performance by 13% compared to RUL prediction models that do not incorporate degradation information. Furthermore, comparisons with commonly used state-of-the-art methods confirm the superior performance and robustness of the proposed method. **DOI: 10.1061/JAEEEZ.ASENG-5887.**

Introduction

Background and Motivation

Aeroengines are intricate thermomechanical systems that operate under high-temperature and high-pressure conditions. These harsh environments contribute to the degradation of various components (Chen et al. 2021c, a, 2020). Degradation that exceeds critical thresholds can lead to failures. Given their central role in aircraft operations, unexpected aeroengine failures can have grave consequences. Thus, precise prediction of failure timings, or the remaining useful life (RUL), of aeroengines is crucial for formulating effective preventive maintenance strategies, ensuring flight safety, and reducing operational and maintenance costs.

Aeroengines consist of numerous components, each of which may experience varying degrees of degradation, leading to different degradation modes (DMs), such as high-pressure compressor flow reduction, low-pressure turbine efficiency loss, fan flow reduction, and their combinations. Each DM represents a distinct operating mode of the aeroengine, leading to variations in sensor data distributions. For instance, even if two aeroengines possess identical RUL and undertake the same flight missions, differences in their DMs result in divergent sensor data distributions, including variations in mean, variance, and other high-dimensional features. When using data-driven prediction methods to model these differently distributed data, the performance of the prediction models is inevitably impacted (Wang 2013; Son 2011). Furthermore, as a crucial tool in data-driven methods, artificial neural networks (ANNs) typically assume that the training, validation, and test data sets share the same data distribution. This assumption is compromised when different DMs are present. Although data-driven methods are widely used for RUL prediction, the impact of varying DMs on model performance has not been thoroughly explored. In practice, approaches such as transfer learning (Tan et al. 2018) and fusion (Chao et al. 2022) models aim to improve RUL prediction by identifying the current DM. For instance, in fusion models, the calibrated health index output by the model (Chao et al. 2022) reflects the degradation level of each component and serves as input for the prediction network. However, these methods do not explicitly model DMs. Identifying DMs and incorporating degradation information into the prediction framework remains a key issue for accurate RUL prediction.

Literature Review

In addition to empirical models based on prior knowledge, RUL prediction primarily involves model-based approaches, data-driven approaches, and their fusion. Model-based methods build nonlinear physical models using physical laws and calibrate them with actual data to estimate unobservable variables, such as RUL (Du et al. 2022). Techniques include physical degradation models (Qin et al. 2011; Qian et al. 2017) and Kalman and particle filters (Lei et al. 2016; Wei et al. 2013; Zio and Peloni 2011). However, due to the complexity of aeroengines, constructing accurate and widely applicable physical models is challenging.

The advancement of sensor technology has led to an abundance of condition monitoring data (Chen et al. 2021b), fueling the development of data-driven methods for engine health management

[1]Ph.D. Student, School of Big Data and Software Engineering, Chongqing Univ., Chongqing 401331, China. ORCID: https://orcid.org/0000-0003-0175-7630. Email: duxiao@cqu.edu.cn

[2]Assistant Professor, Research Center for Special Aircraft Systems Engineering Technology, Ningbo Institute of Materials Technology and Engineering, Chinese Academy of Sciences, Ningbo 315201, China (corresponding author). ORCID: https://orcid.org/0000-0001-8293-8526. Email: chenjiajie@nimte.ac.cn

[3]Professor, Research Center for Special Aircraft Systems Engineering Technology, Ningbo Institute of Materials Technology and Engineering, Chinese Academy of Sciences, Ningbo 315201, China. ORCID: https://orcid.org/0000-0002-1317-0880. Email: wangjiqiang@nimte.ac.cn

[4]Professor, College of Energy and Power Engineering, Nanjing Univ. of Aeronautics and Astronautics, Nanjing 210016, China. Email: zhhbason@nuaa.edu.cn

[5]Professor, School of Big Data and Software Engineering, Chongqing Univ., Chongqing 401331, China. Email: jhwen@cqu.edu.cn

Note. This manuscript was submitted on May 15, 2024; approved on October 1, 2024; published online on January 20, 2025. Discussion period open until June 20, 2025; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Aerospace Engineering***, © ASCE, ISSN 0893-1321.

(Haidong et al. 2020a). Data-driven methods rely on observable data to build nonlinear models (Schwabacher and Goebel 2007). Deep learning techniques, which can derive information directly from raw data, are prominent in this field (Khan and Yairi 2018; Zhao et al. 2019). Notable models include ANNs (Chen et al. 2021e, f, d), long short-term memory (LSTM) networks (Xiang et al. 2021), gated recurrent units (GRUs) (Chen et al. 2019), convolutional neural networks (CNNs) (Li et al. 2018b, 2022a), graph neural networks (Ma et al. 2021), deep autoencoders (Haidong et al. 2020b; Li et al. 2021), and deep belief networks (Pan et al. 2020).

For instance, Huang et al. (2019) used Bidirectional LSTM networks to integrate sensor signals and environmental information. Zhang et al. (2022) developed a novel loss function to simultaneously evaluate health state and RUL prediction in a CNN-GRU series. Cheng et al. (2021) applied multiple-kernel maximum mean discrepancies and transfer learning to adapt their fusion prediction model. Zhu et al. (2020) utilized a hidden Markov model and multilayer perceptrons (MLP) with transfer learning to address differences in feature space distribution.

Fusion methods integrate model outputs as inputs to data-driven algorithms, expanding the feature space (Zhang et al. 2017; Dourado and Viana 2019). For aeroengines, a fusion method first calibrates a degradation model with observable data through reinforcement learning, then uses outputs such as virtual sensor values and health indices in a one-dimensional CNN network for RUL prediction (Chao et al. 2022). Fusion methods generally outperform purely data-driven methods due to the expanded feature space but inherit the limitations of model-based approaches. If the model deviates significantly from actual conditions, performance may be inferior to purely data-driven methods.

The method proposed in this article is data-driven but utilizes fusion concepts to enhance the observation space, integrating two distinct task models to improve RUL prediction accuracy and benefiting real-world industrial applications.

Scope and Contribution

This paper proposes a fusion-based dual-task to identify DMs and mitigate the adverse effects of different data distributions, thereby improving the accuracy of RUL predictions and ensuring applicability in real-world scenarios. In this architecture, aeroengine RUL prediction is divided into two primary tasks: DM identification and preliminary RUL prediction. The neural network combines a residual network (ResNet) and LSTM network in series for both tasks. ResNet extracts features from single-flight observable data, whereas LSTM captures high-dimensional temporal features from these ResNet-extracted features. These enriched features are then fed into a MLP, which maps observable data to RUL and DM in both tasks. Subsequently, by removing the original output layer of the task models and adding a new output layer, the fused model is obtained. The Dual-task ResNet-LSTM (DT-ResLSTM) model, which accounts for the effects of different DMs, is then developed by retraining the new output layer of the fused model while keeping the original parameters fixed. Additionally, RUL label reconstruction using the DM identification model, which is also part of the fusion process, further enhances the accuracy of RUL predictions.

Most RUL prediction studies for aeroengines utilize the publicly available data set generated by the National Aeronautics and Space Administration (NASA) in 2008 (Saxena et al. 2008) based on the Commercial Modular Aeropropulsion System Simulation (C-MAPSS) model (Frederick et al. 2007), known as the CMAPSS data set. However, this data set provides only four snapshots per flight, which limits the applicability of more advanced and comprehensive neural network models. In contrast, the data set used in this paper, also generated using the C-MAPSS model, incorporates real flight conditions (Matthews 2012), resulting in complete flight data (Chao et al. 2021).

The main contributions of this paper are summarized as follows:

1. This paper proposes a fusion-based dual-task architecture that integrates aeroengine DM information into the RUL prediction model through a combination of network architecture fusion and RUL label reconstruction. This approach enables the model to learn the differences among various DMs, thereby further enhancing the generalization and accuracy of the RUL prediction model.
2. The paper uses a ResNet-LSTM architecture to build the model network, where ResNet extracts features within the flight, and LSTM captures interflight features. This approach fully leverages the characteristics of both ResNet and LSTM to better extract useful information from original historical operation data. Because of the network architecture design, the model has a deeper structure that is better suited to fit larger data sets and is more applicable to real-world scenarios.
3. Experiments are performed using more realistic aeroengine simulation data under actual operating conditions to validate the effectiveness and robustness of the proposed method. Additionally, the effects of two critical parameters, namely, the single-flight data sampling size and the time step within the flight dimension, on the model's predictive performance are also analyzed.
4. The proposed method directly predicts RUL from the original historical data rather than constructing physical models or using health parameters, making it more suitable for real operational environments.

Methodology

This section first introduces an overview of the fusion-based dual-task architecture, followed by a detailed description of the network structure and the core network of the task. Finally, the steps for reconstructing the RUL are provided.

Overview of Fusion-Based Dual-Task Architecture

According to the section "Introduction," different DMs result in varying distributions of aeroengine operational history data. The core idea of the proposed fusion-based dual-task architecture is that, during RUL prediction, the model can identify and correct the effects of different data distributions caused by different DMs, thereby improving prediction performance. The implementation approach is as follows. First, the DM identification model detects the varying data distributions introduced by different DMs in the input data and extracts them into high-dimensional features. These features are then fused into the RUL prediction model to enhance its ability to adjust for different input data distributions. The fusion occurs in two main aspects: (1) integrating the network architectures of the DM identification and RUL prediction models, and (2) using the DM identification model to reconstruct the RUL labels. This will be elaborated in this section.

Fig. 1 presents a detailed flowchart of the fusion-based dual-task architecture, which is divided into two stages: offline training and online prediction. In the offline training stage, the model updates its parameters using known data, including historical operating data, scrap time, and component degradation patterns of aircraft engines, which are readily obtainable through inspections and routine maintenance. In the online prediction stage, observable variables such as sensor measurements and flight altitude are utilized as

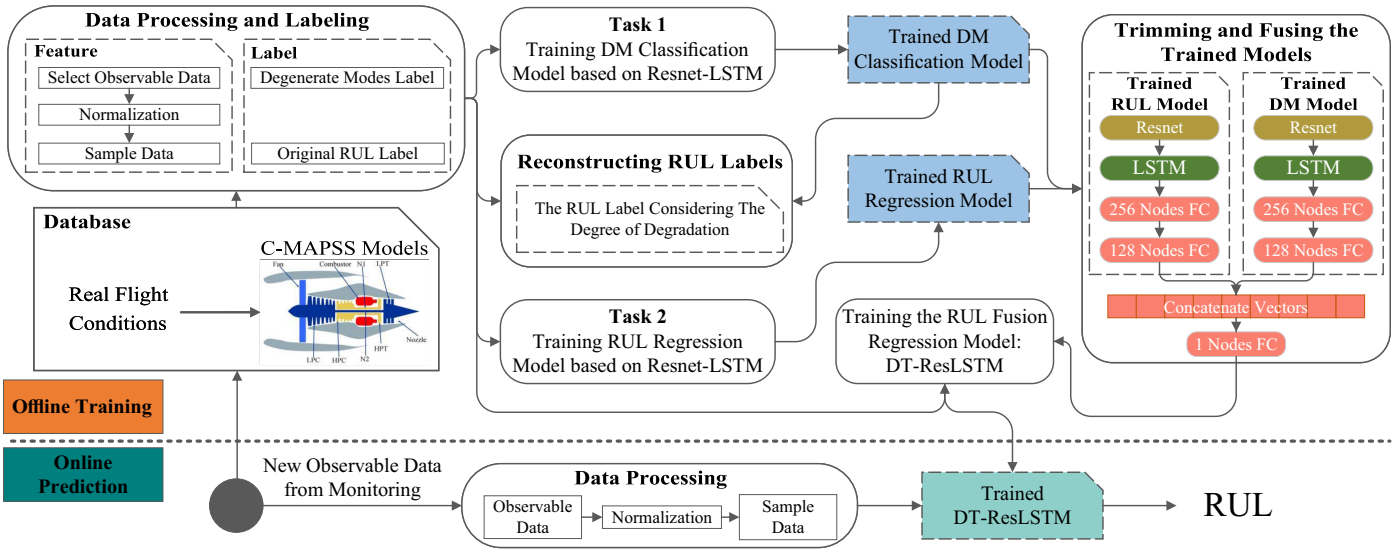

Fig. 1. Schematic diagram of fusion-based dual-task architecture.

inputs to generate predicted RUL values. This study employs the N-CMAPSS data set (Chao et al. 2021), generated by a simulated commercial aircraft model based on a real flight trajectory, to simulate the entire experimental process.

The offline training phase encompasses five key steps:

1. Data Processing and Labeling: Observable data from the simulated data set is normalized, sampled, and transformed into model input features. Additionally, original RUL prediction and DM identification labels are constructed.
2. Training DM Classification Model: The DM labels and feature information are used to train a classification model (Task 1) for DM classification.
3. Reconstructing RUL labels: The trained DM classifier and original RUL labels are employed to reconstruct RUL labels that account for the degree of degradation.
4. Training RUL Regression Model: The reconstructed RUL labels and feature information are then used to train the RUL regression model.
5. Model Trimming and Fusion: The trained models from both tasks undergo pruning to remove redundant parameters and are subsequently fused into the final RUL prediction model, DT-ResLSTM. This fusion involves retraining DT-ResLSTM with the reconstructed RUL labels and feature information.

The online prediction stage involves two main steps.

1. Data Preprocessing: New observations are normalized using the parameters obtained from offline data normalization and are sampled to match the length of the training data.
2. RUL Prediction: The processed feature information is fed into the trained DT-ResLSTM model to generate the predicted RUL value.

The DT-ResLSTM is obtained through the following.

1. Model Pruning: For the models trained in both tasks, the network layers leading to the output layer are pruned, while the overall architecture is retained and the weights in the remaining layers are fixed.
2. Feature Concatenation: The output vectors from the final layer of the pruned network architectures are then concatenated.
3. Final Layer Addition: A fully connected layer with a single output node is added at the network's end.
4. Model Retraining: Finally, the combined DT-ResLSTM model is retrained for optimal performance.

The mathematical expression of this process is shown in Eq. (1)

$$\hat{r} = FC_1(V) \qquad V = concate(V_r, V_d)$$
$$V_r = F^r(X) = O^r\left(L^r\left(\overset{T}{\underset{t=1}{concate}}(R_t^r(x_t))\right)\right)$$
$$V_d = F^d(X) = O^d\left(L^d\left(\overset{T}{\underset{t=1}{concate}}(R_t^d(x_t))\right)\right)$$
$$X = \{x_1, x_2, \ldots, x_T\} \tag{1}$$

where $\hat{r}$ = RUL prediction value output by the model, which is a scalar; V_r = output vector of the pruned network model in task 2; V_d is the same; and V = new vector after concatenating V_r, V_d. The function symbols with superscripts represent the functions corresponding to each trained network architecture with fixed weights. For example, F^d = function corresponding to the trained and cropped DM classification model in Task 1; and L^r = function corresponding to the trained LSTM network in Task 2. For an explanation of other variables, please refer to the next subsection.

Network Structure of Two Tasks

Our proposed fusion-based dual-task architecture incorporates two preliminary tasks: DM identification and preliminary RUL prediction. The detailed network structures for these tasks are presented in Fig. 2.

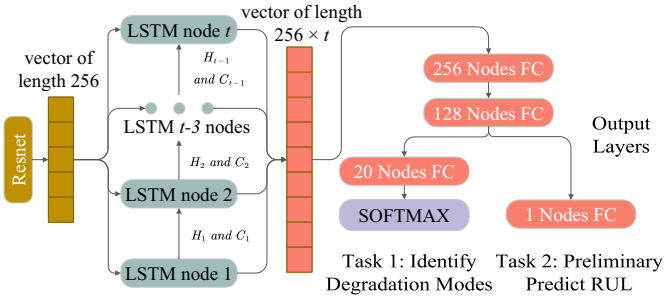

Fig. 2. Schematic diagram of network architecture in task.

J. Aerosp. Eng., 2025, 38(3): 04025004

The network structure consists of three components: ResNet, LSTM, and output layers. ResNet initially extracts features from the observable data of a single flight. LSTM then models these features while preserving the flight sequence. Finally, various output layers are connected to complete the network construction.

Identifying DMs is a classification task that uses a fully connected layer with 20 nodes and a softmax layer as the final two layers. Each pair of output nodes from the fully connected layer corresponds to a component's DM, such as the efficiency degradation of the high-pressure turbine. The softmax layer normalizes the output to the (0, 1) interval, producing a failure probability for each component's DM. Predicting the RUL is a regression task, with the final layer being a fully connected layer with a single output node.

Assume a multidimensional feature matrix $X \in \mathbb{R}^{T\times N_s\times N_f}$ as the input to the network, where T = time dimension (measured in flight cycles); N_s = data length dimension; and N_f = feature dimension. The mathematical description of the network output is given by Eq. (2)

$$\begin{aligned} \hat{r} = FC_1(F(X)),\ \ \hat{D} = soft\max(FC_{20}(F(X)) \\ F(X) = O(L(V_R)) \\ V_R = \overset{T}{\underset{t=1}{concate}}(R_t(x_t)) \\ X = \{x_1, x_2, \ldots, x_T\} \end{aligned} \tag{2}$$

where $\hat{D}$ = aeroengine DM output by the entire network, which is a vector containing the degradation probability of each component; FC_1 and FC_{20} = functions of the fully connected layers with 1 and 20 output nodes, respectively; F = function of the remaining network components excluding the task-specific output layers, which include ResNet, LSTM network, and remaining output layer; O = function of the shared network layers for both tasks in the output layer comprising two fully connected layers with 256 and 128 output nodes, as shown in Fig. 2; L = LSTM network function; and R_t = ResNet function that processes the input feature matrix x_t. The *concate* operation represents vector concatenation.

Specifically, as illustrated in Fig. 2, the final layer of ResNet is a fully connected layer with 256 output nodes producing an output vector $v_t \in \mathbb{R}^{256}$. This vector is concatenated across the other dimension, resulting in the output matrix $V_R \in \mathbb{R}^{256\times T}$. It is important to note that all the aforementioned function symbols correspond to the untrained network architecture.

Core Network in Task Model

ResNet

The core concept of ResNet, a deep CNN introduced by He et al. (2016), is to ensure that the function space of the network with additional layers encompasses the function space of the original network. A schematic diagram of the core idea is shown in Fig. 3.

Let F denote the function space that a neural network can fit, and f^* be the objective function to be approximated. For a given data set, there exists a function f_F^* in F that is closest to f^* and can be approximated by the network. As illustrated in Fig. 3, increasing network depth expands the function space F. However, deeper ordinary neural networks may lead to a greater distance from f^*. In contrast, ResNet avoids this issue because each additional layer retains the original function space F. Consequently, although deeper networks can suffer from overfitting due to excessive complexity, ResNet is less prone to overfitting despite increased depth.

Residual block is the core module of ResNet. Compared with the ordinary block, the residual block adds a shortcut to input X such that the function to be fitted by the network layer is changed from $f(X)$ to $f(X) - X$. There are two advantages to this change. (1) Fitting $f(X) - X$ is more sensitive to data changes than directly fitting $f(X)$ and can capture subtle fluctuations in mapping. (2) The input X can bypass the network layer, directly influencing the subsequent layer and mitigating the risk of model overfitting.

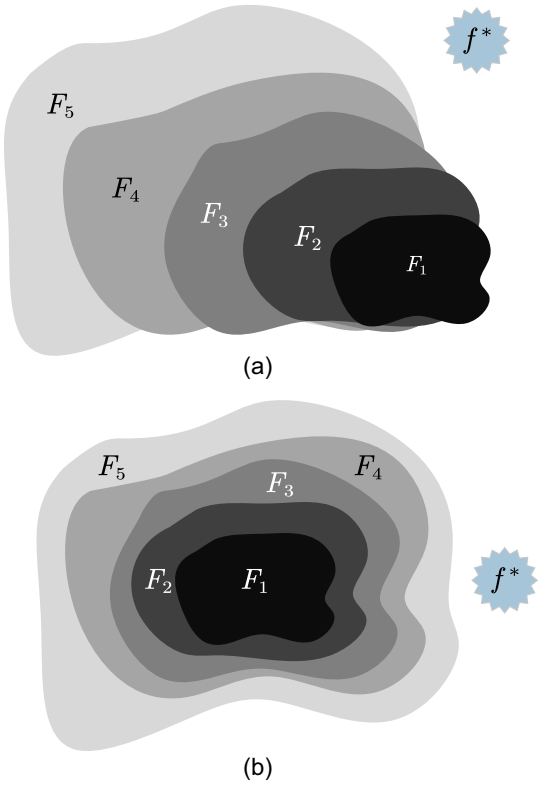

Fig. 3. Schematic diagram of core design idea of ResNet: (a) nonnested function classes (ordinary neural network); and (b) nested function classes (ResNet).

Given the extensive feature space of the input data in this study, a sufficiently deep network is necessary to capture the data's feature information. However, it is also crucial to prevent severe overfitting. Thus, ResNet is an appropriate choice for constructing the network model.

This study employs two types of residual blocks in ResNet, as illustrated in Fig. 4. The primary distinction between two blocks is the use of a 1×1 convolutional layer to modify the number of channels. In both block types, the 3×3 convolutional layer has a stride of 2 and padding of 1, whereas the 1×1 convolutional layer has a stride of 1 and no padding. Additionally, neither block uses bias parameters.

The network architecture of ResNet used in this study is depicted in Fig. 5. It begins with a 7×7 convolutional layer with 64 output channels and a stride of 2, followed by a 3×3 max pooling layer with a stride of 2. This is succeeded by eight residual blocks. The network concludes with a 1×1 global average pooling layer and a fully connected layer with 256 output nodes. The input and output channel parameters of each residual block and whether the shortcut has a 1×1 convolutional layer are shown in Table 1.

LSTM

Long short-term memory (LSTM) is a type of latent variable model well-suited for time series data (Hochreiter and Schmidhuber 1997). Unlike traditional hidden variable models, LSTMs incorporate a memory cell C_t that governs the update of the hidden state H_t

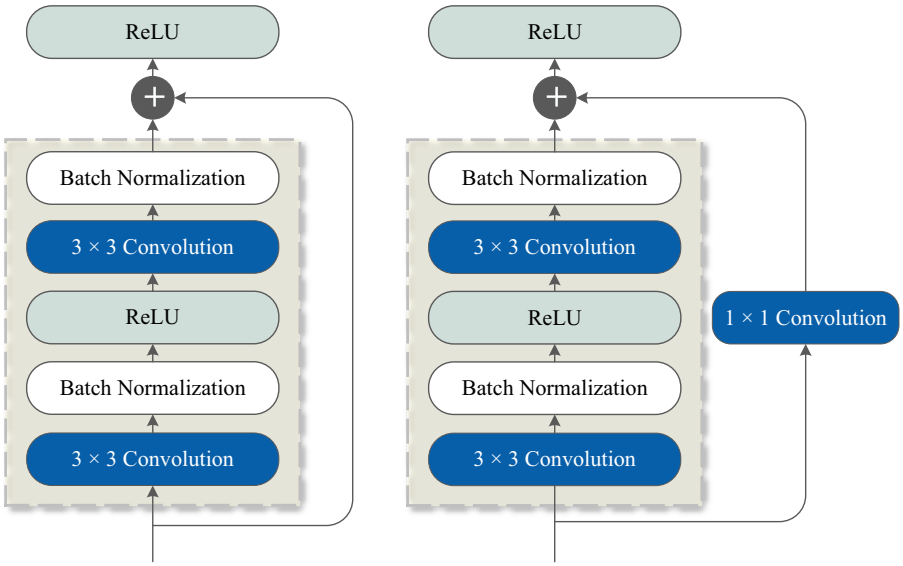

Fig. 4. Schematic diagram of specific structure of two residual blocks in network architecture.

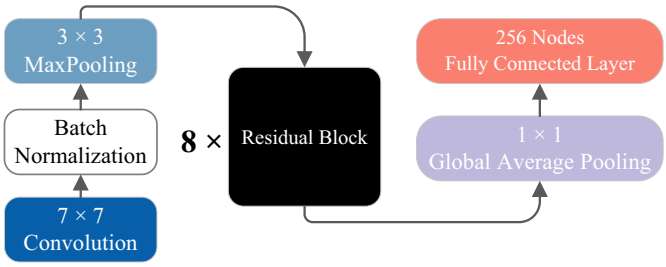

Fig. 5. Schematic diagram of ResNet used.

Table 1. ResNet's network parameters

Block	Input channels	Output channels	1 × 1 convolutional layer
1	64	64	N
2	64	64	N
3	64	64	E
4	128	128	N
5	128	128	E
6	256	256	N
7	256	256	E
8	512	512	N

Note: E = exist; and N = none.

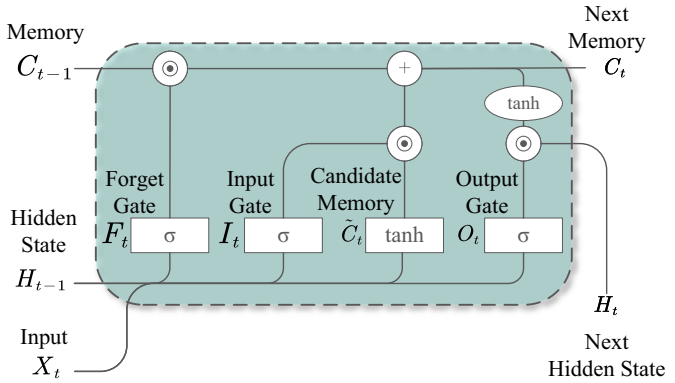

Fig. 6. Schematic diagram of LSTM calculation process.

while enabling the inclusion of more features. The detailed architecture of an LSTM node is illustrated in Fig. 6.

The forget gate, input gate, and output gate are three fully connected layers with *sigmoid* activation functions. The candidate memory cell is a fully connected layer with *tanh* as the activation function.

Reconstructing RUL Labels

The definition of RUL adopted in this paper is typically the number of flight cycles until failure. In aeroengine RUL prediction, labels are often set as either a linear or piecewise linear function. The piecewise linear function divides the operation process into two phases: healthy and fast degradation. In the healthy phase, RUL is fixed at a constant maximum value r_{health}, whereas in the fast degradation phase, it decreases linearly with the number of flight cycles. Due to the low degradation rate in the early operation stages, using a piecewise function is common. For instance, in the CMAPSS data set, r_{health} is typically set to 125 cycles (Li et al. 2018a, b). However, this uniform setting may obscure valuable degradation information due to differences among individual aeroengines and their unknown initial health states. To better utilize degradation information and enhance model performance, we use a DM identification model to reconstruct RUL for each aeroengine in the data set.

The RUL reconstruction process utilizes the trained DM identification model in three steps.

1. Component Degradation Probability Estimation: We apply the model to each engine in the data set and obtain the degradation probability for each component across all flight cycles (denoted as $M_{\hat{D}} = \{\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{N_d}\}$, $\hat{D} \in \mathbb{R}^{n \times N_d}$). Here, n = engine's total flight cycles; and N_d = number of components.
2. Dominant DM Identification: In this paper, we assume that the higher the degradation probability output by the model is,

Table 2. Overview of N-CMAPSS data set used

Group	#Unit	Flight classes	FanE	FanF	LPCE	LPCF	HPCE	HPCF	HPTE	HPTF	LPTE	LPTF
DS01	10	1,2,3	—	—	—	—	—	—	1	—	—	—
DS02	9	1,2,3	—	—	—	—	—	—	1	—	1	1
DS03	15	1,2,3	—	—	—	—	—	—	1	—	1	1
DS04	10	2,3	1	1	—	—	—	—	—	—	—	—
DS05	10	1,2,3	—	—	—	—	1	1	—	—	—	—
DS06	10	1,2,3	—	—	1	1	1	1	—	—	—	—
DS07	10	1,2,3	—	—	—	—	—	—	—	—	1	1
DS08a	9	1,2,3	1	1	1	1	1	1	1	1	1	1
DS08c	6	1,2,3	1	1	1	1	1	1	1	1	1	1

Note: "1" indicates that the DM exists in the data set. LPC = low-pressure compressor; HPC = high-pressure compressor; HPT = high-pressure turbine; and LPT = low-pressure turbine.

the more severely the corresponding component has degraded. Since engine life is determined by the first failing component, we identify the component with the highest average degradation probability ($\hat{D}_i = \{d_1, d_2, \ldots, d_n\}_i$, where $i \in \{1, 2, \ldots, N_d\}$) over its entire lifecycle. This component's degradation probability is then used as an indicator for RUL reconstruction.

3. RUL Label Assignment: We define a threshold $d_{\hat{D}_i}$ based on the accuracy of the DM identification model. For the current and subsequent $m-1$ flights (where $m = 10$ in this study), we calculate the average degradation probability $\bar{d}_j = 1/m \sum_{a=j}^{j+m} d_a$. If $\bar{d}_j < d_{\hat{D}_i}$, the engine is considered healthy, and its RUL remains unchanged. Conversely, the engine is assumed to enter a rapid degradation phase, with a linearly decreasing RUL until the end of its life.

Data Set and Data Processing

This section presents a detailed description of the simulated data sets and the data processing procedures.

Data Set

This study utilizes the N-CMAPSS data set created by Chao et al. (2021) to assess the effectiveness of the proposed method. Similar to the CMAPSS data set (Saxena et al. 2008), the N-CMAPSS data set is generated using NASA's C-MAPSS model (Frederick et al. 2007). The N-CMAPSS data set includes degradation trajectories for a fleet of turbofan engines with unknown initial health states, extending to failure. The data set comprises nine groups and 89 engine units, with each group having different aeroengine DMs.

The N-CMAPSS data set offers the following several advantages over the CMAPSS data set. (1) It provides the complete flight envelope for each flight, whereas the CMAPSS data set offers only four snapshots per flight. (2) It includes a higher number of DMs. (3) It utilizes real flight trajectories from DASHlink (Matthews 2012) to generate data, encompassing three different flight classes.

Table 2 provides details of the DMs and flight classes for each data group in the N-CMAPSS data set. The terms FanE, FanF, LPCF, and others represent the types of degradation in various aeroengine components. For example, FanE refers to fan efficiency degradation, whereas LPCF refers to low pressure compressor flow degradation. The term "#Unit" indicates the number of simulated aeroengines within each data set group. Simulated flights are categorized into three classes based on their duration: Flight Class 1 for 1–3 h, Flight Class 2 for 3–5 h, and Flight Class 3 for more than 5 h.

Table 3. Specific information of operating condition and sensor measurement signal

#	Type	Symbol	Description	Units
1	w	alt	Altitude	ft
2		Mach	Flight Mach number	—
3		TRA	Throttle–resolver angle	%
4	x_s	T2	Total temperature at fan inlet	°R
5		Wf	Fuel flow	Pps
6		N1	Physical fan speed	Rpm
7		N2	Physical core speed	Rpm
8		T24	Total temperature at LPC outlet	°R
9		T30	Total temperature at HPC outlet	°R
10		T48	Total temperature at HPT outlet	°R
11		T50	Total temperature at LPT outlet	°R
12		P15	Total pressure in bypass-duct	psia
13		P2	Total pressure at fan inlet	psia
14		P21	Total pressure at fan outlet	psia
15		P24	Total pressure at LPC outlet	psia
16		Ps30	Static pressure at HPC outlet	psia
17		P40	Total pressure at burner outlet	psia
18		P50	Total pressure at LPT outlet	psia

Data Selection

The N-CMAPSS data set comprises six distinct data types: operating condition w, sensor measurement signal x_s, virtual sensor signal x_v, engine health parameter θ, RUL label r, and auxiliary data such as the number of flight cycles c.

The task of this paper is to establish a mapping relationship between observable data and RUL. Therefore, to match the real situation, the operating condition w and the sensor measurement signal x_s are selected as the feature data of the neural network model. The specific information of the operating condition w and sensor measurement signal x_s is shown in Table 3. Fig. 7 (Saxena et al. 2008) shows the simplified diagram of engine simulated in the C-MAPSS model.

Data Normalization

Due to significant discrepancies in variable scales, their influence on network weight updates during training can vary considerably. To address this, we normalize each variable using the Z-Score normalization method. The mathematical expression is shown in Eq. (3)

$$X_{norm}^{train} = \frac{X^{train} - \mu^{train}}{\sigma^{train}} \tag{3}$$

where μ^{train} and σ^{train} = mean and variance, respectively, of the observable variables X^{train} within the training set. During predictions,

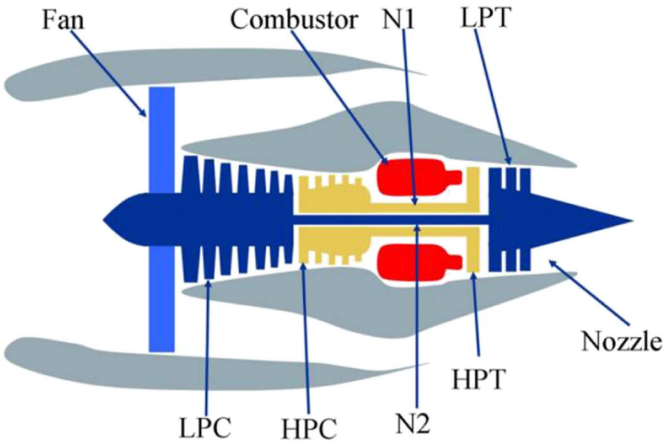

Fig. 7. Simplified diagram of engine simulated in C-MAPSS model.

we employ the same μ^{train} and σ^{train} that are used in Eq. (3) to normalize the test data X^{test}, preserving the data distribution. This normalization process is mathematically expressed in Eq. (4)

$$X_{norm}^{test} = \frac{X^{test} - \mu^{train}}{\sigma^{train}} \tag{4}$$

Data Sampling

The N-CMAPSS data set adopts the real flight trajectory—the flight time of each flight is different, making the simulated data length of each flight different. However, variable length time series as samples are unacceptable for neural networks. In addition, unifying the samples to the same length N_s is beneficial to balance model performance and computer computing performance. To retain more original information about the flight, this paper uses nearest-neighbor interpolation to downsample the time series length. The mathematical expression is shown in Eq. (5)

$$I_s = \left[I_o \times \frac{N_s}{N_o} \right] \tag{5}$$

where I_o and I_s = original and sampled data point indexes, respectively; and N_o and N_s = data length of the original and target samples, respectively.

Fig. 8 illustrates the sampling process, during which raw data sequences of variable length N_s are downsampled to a fixed length N_0 using nearest-neighbor interpolation. This approach effectively preserves feature information compared to direct data cropping.

The network receives input data denoted as $X \in \mathbb{R}^{T \times N_f \times N_s}$. Here, T represents the number of flights included in the input features, referred to as the time steps, which correspond to the number of LSTM nodes. For example, $T = 4$ indicates that the data from the current and previous three flights are used as the model's input. N_f is the number of features, including the operating condition w and measurement sensors w_x; therefore, $N_f = 4 + 14$. N_s is the fixed sequence length of the flight data after sampling. The impact of T and N_s on prediction results is further explored in Section 4.

Degradation Modes Labels

The DM label $D \in \mathbb{R}^{N_d}$ is a binary vector, where N_d = number of degradation types. In this paper, there are five degradable components, each with two degradation types: efficiency and flow rate. Therefore, $N_d = 10$. The DM label reflects the status of the aircraft engine, where 1 indicates degradation and 0 indicates health. For example, the label $D = [0, 0, 0, 0, 0, 0, 1, 0, 1, 1]$ indicates high-pressure turbine (HPT) efficiency degradation, low-pressure turbine (LPT) efficiency degradation, and LPT flow rate degradation, whereas the other components indicate health.

The reason we set the DM label D this way rather than directly using the health parameter θ is to better adapt to real-world scenarios. In this paper, we use θ provided by the data set to set D, whereas in real-world situations, D can be constructed during the maintenance process. This flexibility allows the proposed network architecture to be trained and used for prediction on data sets built from actual flight and maintenance data.

Experiments

This section details the evaluation metrics, experimental settings, and key findings. We begin by presenting the evaluation metrics and experimental setup. Following this, we analyze the impact of crucial hyperparameters during the training process. Next, we assess the effectiveness of the DM identification model (Task 1) and the quality of the reconstructed RUL labels that leverage degradation information. Finally, we demonstrate the effectiveness and robustness of the proposed DT-ResLSTM RUL prediction model and analyze the obtained prediction results.

Evaluation Metrics

During the training phase of deep learning, loss functions guide parameter updates through backpropagation. Typically, the root mean square error (RMSE) loss function is used for regression tasks, whereas the cross-entropy loss function is used for classification tasks. Given that the labels for the DM identification task are vector-based, this paper modifies the cross-entropy loss function to accommodate vector evaluation. The mathematical definitions of the RMSE loss function and the modified cross-entropy loss function used in this work are provided in Eqs. (6) and (7), respectively

$$rmse(V_r, V_{\hat{r}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r - \hat{r})^2} \tag{6}$$

$$CE(M_D, M_{\hat{D}}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{N_d}(-(d_j \log \hat{d}_j + (1 - d_j)\log(1 - \hat{d}_j)) \tag{7}$$

where V_r and $V_{\hat{r}}$ = vectors of true r and predicted RUL $\hat{r}$, respectively, V_r, $V_{\hat{r}} \in \mathbb{R}^n$; M_D and $M_{\hat{D}}$ = matrices composed of the real DM vector D and the identified DM vector $\hat{D}$, respectively, M_D, $M_{\hat{D}} \in \mathbb{R}^{n \times N_d}$; N_d = number of component degradation types, which is 10 in this case; and d and $\hat{d}$ = elements in vectors D and $\hat{D}$ corresponding to the probability of the degradation of each component. When training, n = batch size $n = N_b$. When estimating model performance, n = data set size.

In addition to the loss functions, we employ Score [Saxena et al. (2008) and Eq. (8)] as an asymmetric evaluation metric that penalizes overestimating RUL, aligning better with real-world scenarios. The parameter definitions in Eq. (8) are consistent with those previously described

$$score(V_r, V_{\hat{r}}) = \begin{cases} \sum_{i=1}^{n} e^{-\frac{r-\hat{r}}{10}} - 1 for\,(r - \hat{r}) < 0 \\ \sum_{i=1}^{n} e^{\frac{r-\hat{r}}{13}} - 1\, for (r - \hat{r}) < 0 \end{cases} \tag{8}$$

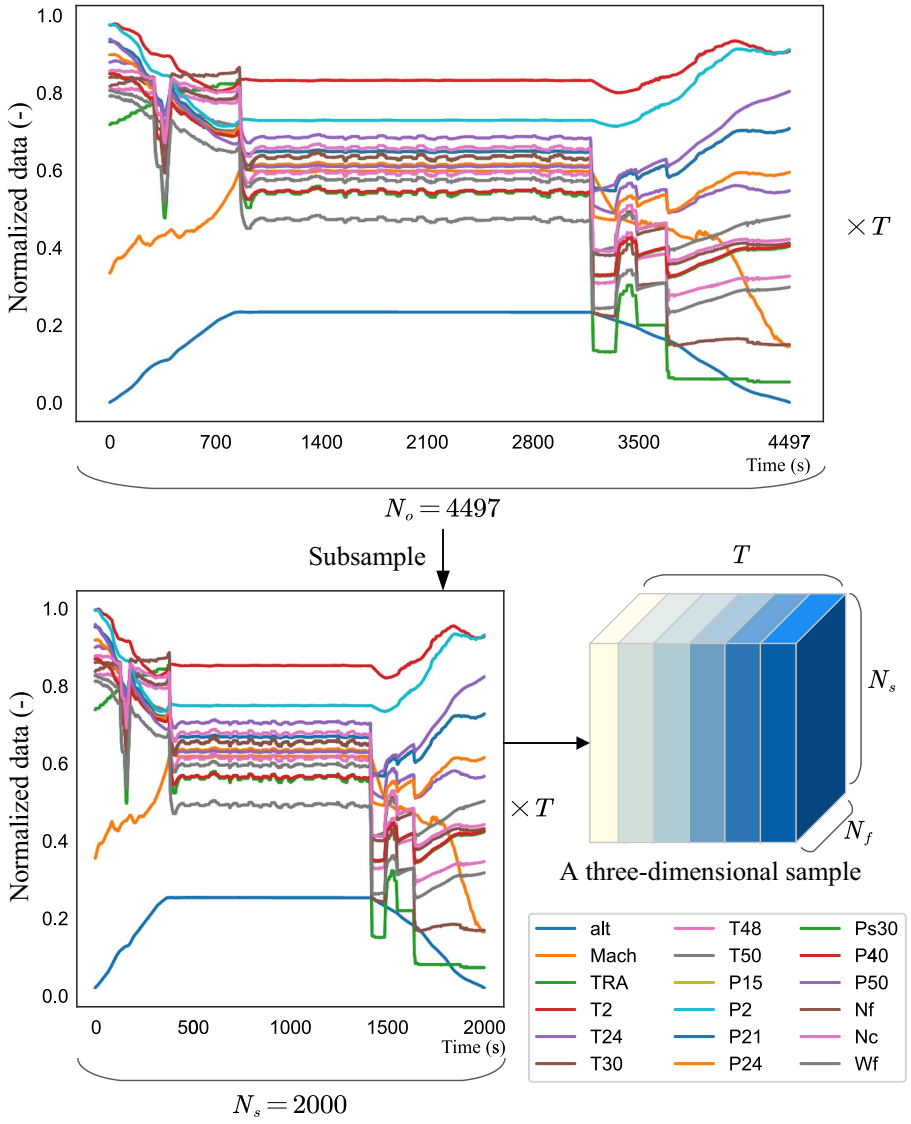

Fig. 8. Schematic diagram of process of sampling data and forming sample.

Experimental Setting

Experiments were conducted in a Linux 5.4 environment using Python 3.7, PyTorch 1.12, and CUDA 11.2, with a fixed random seed of 42 for reproducibility. The N-CMAPSS data set is predivided into "development" and "test" subsets. We further split the "development data set" into 80% for training and 20% for validation, with the "test data set" reserved for final evaluation.

For network training, we used a batch size of 32, a total of 300 epochs, and the adaptive differential mutation algorithm (ADAM) optimizer for parameter updates. The initial learning rate was set to 0.0001 with a cosine decay function. All activation functions within the network, except where explicitly mentioned, were rectified linear unit (ReLU).

Influence of Key Parameters

The fixed sampling length of the flight data, N_s, and the time step (the number of flights), T, are key hyperparameters that affect model performance. Longer N_s leads to less information compression but increases storage and computational requirements. Similarly, a larger T captures more temporal features between flights but at the cost of higher complexity.

To identify an optimal parameter combination, we conduct a grid search over $T \in \{2, 4, 6, 8, 10\}$ and $N_s \in \{128, 256, 384, 512, 640\}$. To expedite the grid search process, we train a simpler RUL regression model (instead of the full DT-ResLSTM) on the original RUL labels for 50 epochs. Each parameter combination result is obtained by averaging the predictions from 10 independent training runs.

The grid search results (Fig. 9) reveal that increasing the time step T has a more significant initial performance enhancement compared to increasing the sample size N_s. However, these gains diminish with larger T and N_s values, following the Markov assumption. Balancing computational cost and performance, we select $T = 8$ and $N_s = 512$ for training the models in both tasks, including DT-ResLSTM.

Task1: Identifying Degradation Modes

The trained DM identification model achieves a final accuracy of **93.18%** on the entire test set. This accuracy remains consistent

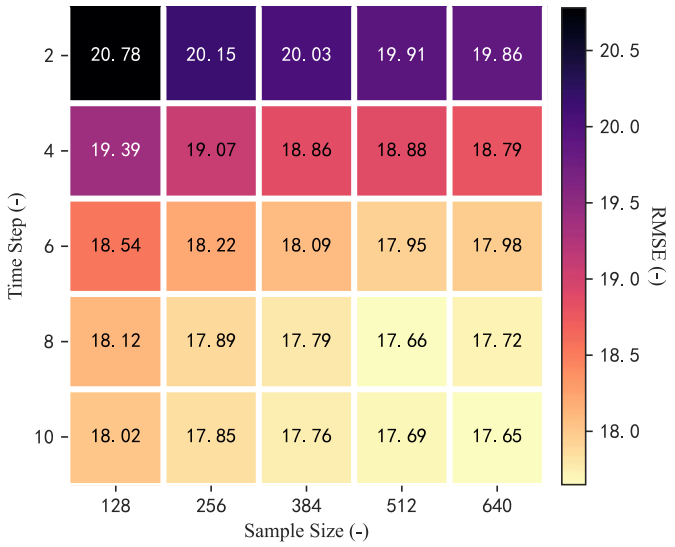

Fig. 9. Grid search results.

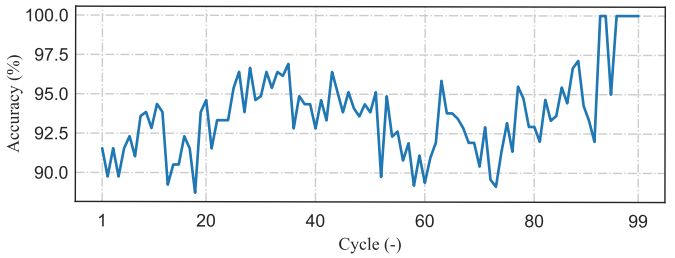

Fig. 10. Changes in accuracy of DM identification model on test data set as flight increases.

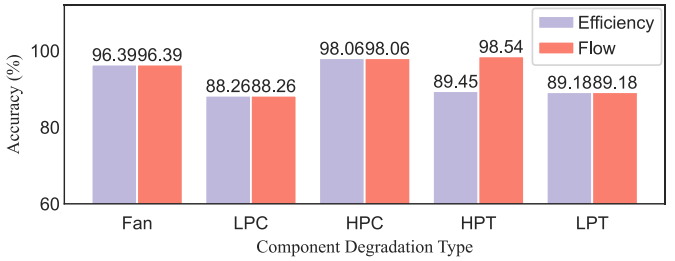

Fig. 11. Accuracy of DM identification model for degradation of various components.

across increasing flight cycles, demonstrating its effectiveness over time (Fig. 10). The model also exhibits high accuracy for various degradation types, with nearly identical performance in identifying flow and efficiency degradation within the same component, except for the HPT, as shown in Fig. 11. This indicates similar degradation patterns across these categories.

RUL Labels

We follow the method described in Section 2 and use the degradation information provided by the DM identification model in Task 1 to reconstruct the remaining useful life (RUL) label. In this paper

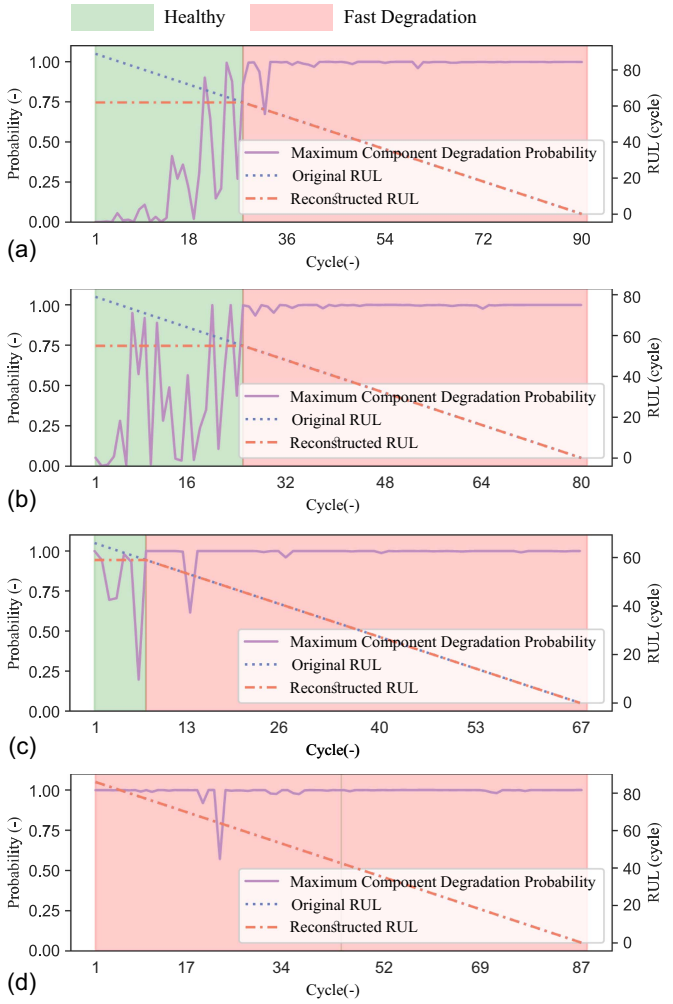

Fig. 12. Schematic diagram of RUL reconstruction in (a) Unit 7 DS01; (b) Unit 9 DS01; (c) Unit 15 DS03; and (d) Unit 10 DS07.

$m = 10$, $d_{\hat{D}_i}$ equals the accuracy of the DM identification model on the data set. Fig. 12 shows the process of reconstructing RUL for four typical aeroengines.

Predicting Remaining Useful Life

Effectiveness

Fig. 13 compares the mean square error (MSE) changes on the validation data set during training for DT-ResLSTM and baseline models, including an eight-layer 2D-CNN, eight-layer 3D-CNN, and ResNet-LSTM sequential model for RUL prediction in Task 2. Although the loss reduction rates are similar for the baseline models, the sequential model achieves a lower converged loss than does the CNN-based models, demonstrating its effectiveness. DT-ResLSTM surpasses the baselines in convergence speed and final loss due to retaining the trained parameters of the two task-specific models and only requiring updates to the output layer's parameters. The lower converged loss is attributed to the fusion of DM information, enhancing the model's performance.

Table 4 shows the effectiveness of the RUL prediction model in Task 2 and DT-ResLSTM on the test data set. DT-ResLSTM significantly improves RUL prediction accuracy by incorporating the DM identification model, with a 13% increase in RMSE and a 19% increase in Score. This demonstrates the value of DM information

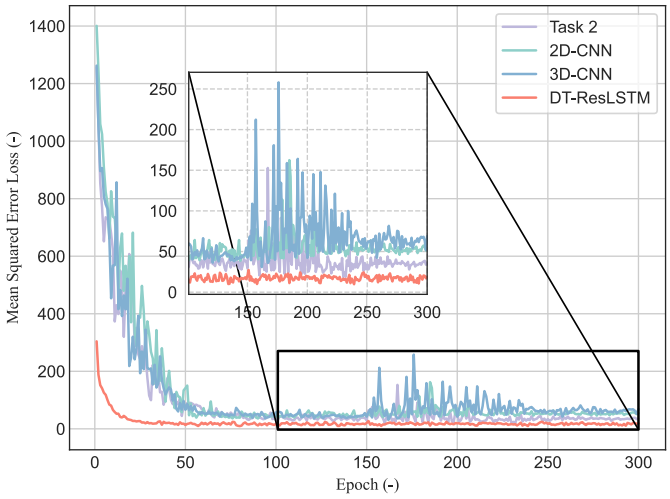

Fig. 13. Comparison of MSE evolution during training between DT-ResLSTM and baseline models.

in enhancing RUL prediction and mitigating the impact of the domain shift due to varying DMs.

Tables 5 and 6 highlight the clear performance advantage of DT-ResLSTM on the N-CMAPSS and CMAPSS data sets, achieving accuracy close to models that utilize both observable and unobservable inputs, despite relying solely on observable data.

In the CMAPSS data set, each flight provides only four data points. We first expand this data to 512 dimensions using the nearest neighbor algorithm and then fine-tune the trained DT-ResLSTM on CMAPSS. The DT-ResLSTM achieves performance close to the state-of-the-art, demonstrating its effectiveness, generalization, and robustness.

Robustness

As shown in Figs. 14 and 15, DT-ResLSTM consistently outperforms the RUL prediction model in Task 2 across all data groups and flight categories, demonstrating superior robustness. Notably, DT-ResLSTM excels in DS05 and DS06, likely due to an imbalance in the number of samples with similar degradation characteristics in the data set. Although the DM-recognizing model mitigates this imbalance, DT-ResLSTM still exhibits this performance preference.

Moreover, in real-world aeroengines, the number of sensors is often fewer than those in the N-CMAPSS data set shown in Table 3. To further test the robustness of the proposed method, we referred to the control system of the CFM56-3 aeroengine (Linke-Diesinger 2008) and selected {T2, T50, P2, Ps30, Wf, N1, N2} as typical sensors under realistic conditions. We used the method outlined in Section 2 to build an RUL prediction model under limited sensors, and its performance on the test set is compared in Table 4. As shown in Table 4, when the number of sensors is limited, both RMSE and Score increase significantly, indicating that the difficulty of RUL prediction increases notably with fewer sensors. However, given limited sensor conditions, the performance of DT-ResLSTM still significantly outperforms the baseline model, demonstrating the adaptability and stability of the proposed method for various sensor configurations.

Table 5. RMSE comparisons with common method and related works in DS02 of N-CMAPSS

Model	Sources	RMSE (cycle)
FNN	Chao et al. (2022)	7.89
LSTM	Chao et al. (2022)	5.02
Hybrid input CNN	Chao et al. (2022)	4.14
DGP	Zeng and Liang (2023)	6.17
BLSTM	Xiang et al. (2024)	6.73
BGT	Xiang et al. (2024)	6.35
DAT	Li et al. (2022b)	5.34
MR-LSTM	Xu et al. (2023)	4.52
Transformer	Xu et al. (2024)	5.34
MSBLS	Xu et al. (2024)	4.86
Proposed-GAN	Wang et al. (2024)	5.81
MHT	Guo et al. (2024)	5.15
DLformer	Ren et al. (2024b)	6.79
DL-LSTM	Ren et al. (2024b)	6.57
DL-DNN	Ren et al. (2024b)	7.27
2D-CNN	In this work	5.45
3D-CNN	In this work	5.51
Task2 ResNet+LSTM	In this work	4.77
DT-ResLSTM	In this work	4.22

Table 6. RMSE comparisons with related works in FD001 of CMAPSS

Model	Source	RMSE (cycle)
MMoE-BiGRU	Zhang et al. (2022)	13.22
MT-CNN	Kim and Sohn (2021)	12.48
MCLSTM	Xiang et al. (2021)	13.71
IDMFFN	Li et al. (2022a)	12.18
MODBNE	Zhang et al. (2016)	15.04
MS-DCNN	Li et al. (2020)	11.44
MHT	Guo et al. (2024)	11.92
CapNet	Ruiz-Tagle Palazuelos and López Droguett (2020)	12.58
ATCN	Zhang et al. (2024)	11.48
BGT	Xiang et al. (2024)	12.09
IMDSSN	Zhao et al. (2023)	12.13
PSA-GHLSTM	Chen (2024	13.14
BTCAN	Ren et al. (2024a)	14.46
DT-ResLSTM	In this work	12.23

Analysis

Fig. 16 represents the prediction of DT-ResLSTM on the test data set. The upper and lower side graphs of the x-axis represent the

Table 4. Overview of performance of RUL prediction model in Task 2 and DT-ResLSTM on test set

Model	RMSE (cycle)	Score $\times 10^2$	rel. Delta in RMSE (%)	rel. Delta in score (%)
Task 2	6.51 ± 0.21	19.63 ± 1.22	0	0
DT-ResLSTM	5.67 ± 0.18	15.95 ± 0.93	−13	−19
Task 2 (limited sensors)	8.76 ± 0.52	26.72 ± 2.68	35	36
DT-ResLSTM (limited sensors)	7.43 ± 0.33	21.49 ± 1.42	14	9

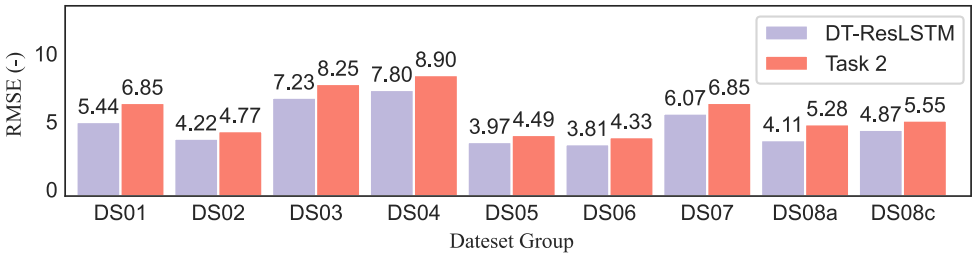

Fig. 14. Model predictions on different data set groups.

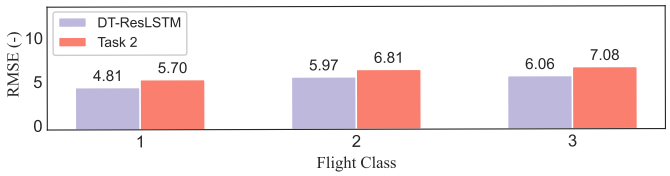

Fig. 15. Model predictions on different flight classes.

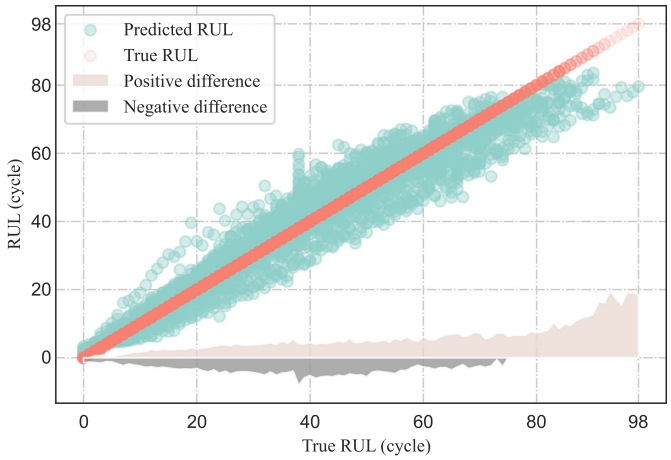

Fig. 16. As cycle increases, prediction of DT-ResLSTM on test data set.

positive and negative difference between the true value and the predicted RUL.

Larger prediction errors are observed for engines with a high RUL. This can be attributed to two factors: (1) Subtle Degradation: At the beginning of an engine's life, degradation is less pronounced, making it challenging for the model to distinguish between similar readings; and (2) Data Imbalance: The training data likely have fewer samples representing high-RUL scenarios compared to those with lower RUL. However, these errors are less critical in this context. Even if the predicted RUL is underestimated, exceeding it during early engine life might not necessitate additional maintenance or pose immediate safety risks.

Conversely, as the engine approaches its end-of-life, the abundance of training data and more pronounced degradation patterns lead to improved prediction accuracy. Moreover, there are far more cases for which the actual value exceeds the predicted value than vice versa. The bias toward underestimating RUL becomes favorable in practice because it prioritizes safety by prompting maintenance before a potential catastrophic event.

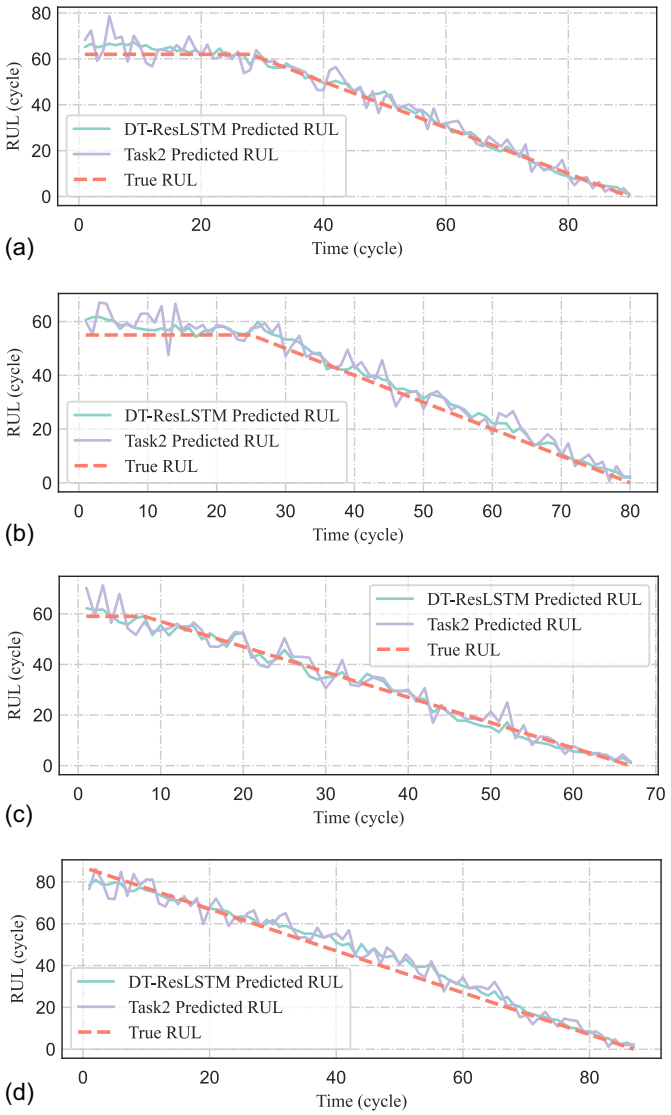

Fig. 17. Prediction results on (a) Unit 7 DS01; (b) Unit 9 DS01; (c) Unit 15 DS03; and (d) Unit 10 DS07.

Fig. 17 illustrates the RUL prediction results of DT-ResLSTM and the prediction model from Task 2 for four representative engine lifecycles. As shown in the figure, both DT-ResLSTM and the Task 2 prediction model effectively track the actual degradation trends, with accuracy improving as the engines approach the end of their service life, consistent with the results shown in Fig. 16. Compared to the model from Task 2, DT-ResLSTM's predictions are generally closer to the true values and exhibit less variability, which aligns

with the results in Table 4, confirming the superior performance of DT-ResLSTM.

Conclusion

This paper proposes a fusion-based dual-task architecture. Task 1 identifies the DM of the aeroengine, and Task 2 performs a preliminary prediction of the RUL. The RUL labels, reconstructed using the DM identification model, train a neural network architecture that integrates both task models, resulting in the final RUL prediction model, DT-ResLSTM. The main idea behind this architecture is to use the DM identification model from Task 1 to guide and correct prediction errors caused by data biases from different DMs.

The network architecture, combining ResNet and LSTM, offers sufficient parameter space to fit larger data sets, such as N-CMAPSS, which better reflect real-world conditions. A grid search evaluates how sample size and time step affect model performance. Compared to the Task 2 model without DM identification, DT-ResLSTM improves prediction performance by 13% in RMSE and 19% in Score.

The method's performance advantages are confirmed through comparisons with common and advanced methods from other studies. This data-driven approach achieves good results using only observable information from the data set, making it suitable for real-world applications.

Although this study shows the effectiveness of the proposed method with simulated data, further exploration is needed to assess its real-world applicability. Transfer learning or model calibration techniques could help bridge the gap between simulated and real-world engine data.

Data Availability Statement

Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments

The present work was supported by the Yongjiang Talent Project of Ningbo (No. 2022A-012-G).

References

Chao, M. A., C. Kulkarni, K. Goebel, and O. Fink. 2021. "Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics." *Data* 6 (1): 5. https://doi.org/10.3390/data6010005.

Chao, M. A., C. Kulkarni, K. Goebel, and O. Fink. 2022. "Fusing physics-based and deep learning models for prognostics." *Reliab. Eng. Syst. Saf.* 217 (Jan): 107961. https://doi.org/10.1016/j.ress.2021.107961.

Chen, C., N. Lu, B. Jiang, and C. Wang. 2021a. "A risk-averse remaining useful life estimation for predictive maintenance." *IEEE/CAA J. Autom. Sin.* 8 (2): 412–422. https://doi.org/10.1109/JAS.2021.1003835.

Chen, C., N. Lu, B. Jiang, and Y. Xing. 2020. "Condition-based maintenance optimization for continuously monitored degrading systems under imperfect maintenance actions." *J. Syst. Eng. Electron.* 31 (4): 841–851. https://doi.org/10.23919/JSEE.2020.000057.

Chen, C., N. Lu, B. Jiang, Y. Xing, and Z. H. Zhu. 2021b. "Prediction interval estimation of aeroengine remaining useful life based on bidirectional long short-term memory network." *IEEE Trans. Instrum. Meas.* 70 (Nov): 1–13. https://doi.org/10.1109/TIM.2021.3126006.

Chen, C., C. Wang, N. Lu, B. Jiang, and Y. Xing. 2021c. "A data-driven predictive maintenance strategy based on accurate failure prognostics." *Eksploatacja i Niezawodność* 23 (2): 387–394. https://doi.org/10.17531/ein.2021.2.19.

Chen, J., H. Jing, Y. Chang, and Q. Liu. 2019. "Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process." *Reliab. Eng. Syst. Saf.* 185 (May): 372–382. https://doi.org/10.1016/j.ress.2019.01.006.

Chen, X. 2024. "A novel transformer-based dl model enhanced by position-sensitive attention and gated hierarchical LSTM for aero-engine RUL prediction." *Sci. Rep.* 14 (1): 10061. https://doi.org/10.1038/s41598-024-59095-3.

Chen, Y., L. Liu, V. Phonevilay, K. Gu, R. Xia, J. Xie, Q. Zhang, and K. Yang. 2021d. "Image super-resolution reconstruction based on feature map attention mechanism." *Appl. Intell.* 51 (7): 4367–4380. https://doi.org/10.1007/s10489-020-02116-1.

Chen, Y., L. Liu, J. Tao, R. Xia, Q. Zhang, K. Yang, J. Xiong, and X. Chen. 2021e. "The improved image inpainting algorithm via encoder and similarity constraint." *Vis. Comput.* 37 (7): 1691–1705. https://doi.org/10.1007/s00371-020-01932-3.

Chen, Y., H. Zhang, L. Liu, J. Tao, Q. Zhang, K. Yang, R. Xia, and J. Xie. 2021f. "Research on image inpainting algorithm of improved total variation minimization method." *J. Ambient Intell. Hum. Comput.* 14 (5): 1–10. https://doi.org/10.1007/s12652-020-02778-2.

Cheng, H., X. Kong, G. Chen, Q. Wang, and R. Wang. 2021. "Transferable convolutional neural network based remaining useful life prediction of bearing under multiple failure behaviors." *Measurement* 168 (Jan): 108286. https://doi.org/10.1016/j.measurement.2020.108286.

Dourado, A., and F. A. Viana. 2019. "Physics-informed neural networks for corrosion-fatigue prognosis." In Vol. 11 of *Proc., Annual Conf. of the PHM Society.* Rochester, NY: PHM Society. https://doi.org/10.36001/phmconf.2019.v11i1.814.

Du, X., J. Chen, H. Zhang, and J. Wang. 2022. "Fault detection of aero-engine sensor based on inception-CNN." *Aerospace* 9 (5): 236. https://doi.org/10.3390/aerospace9050236.

Frederick, D. K., J. A. DeCastro, and J. S. Litt. 2007. *User's guide for the commercial modular aero-propulsion system simulation (C-MAPSS).* Houston: National Aeronautics and Space Administration.

Guo, J., S. Lei, and B. Du. 2024. "MHT: A multiscale hourglass-transformer for remaining useful life prediction of aircraft engine." *Eng. Appl. Artif. Intell.* 128 (Feb): 107519. https://doi.org/10.1016/j.engappai.2023.107519.

Haidong, S., C. Junsheng, J. Hongkai, Y. Yu, and W. Zhantao. 2020a. "Enhanced deep gated recurrent unit and complex wavelet packet energy moment entropy for early fault prognosis of bearing." *Knowl.-Based Syst.* 188 (Jan): 105022. https://doi.org/10.1016/j.knosys.2019.105022.

Haidong, S., D. Ziyang, C. Junsheng, and J. Hongkai. 2020b. "Intelligent fault diagnosis among different rotating machines using novel stacked transfer auto-encoder optimized by PSO." *ISA Trans.* 105 (Oct): 308–319. https://doi.org/10.1016/j.isatra.2020.05.041.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep residual learning for image recognition." In *Proc., IEEE Conf. on Computer Vision and Pattern Recognition*, 770–778. New York: IEEE. https://doi.org/10.1109/CVPR.2016.90.

Hochreiter, S., and J. Schmidhuber. 1997. "Long short-term memory." *Neural Comput.* 9 (8): 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Huang, C.-G., H.-Z. Huang, and Y.-F. Li. 2019. "A bidirectional LSTM prognostics method under multiple operational conditions." *IEEE Trans. Ind. Electron.* 66 (11): 8792–8802. https://doi.org/10.1109/TIE.2019.2891463.

Khan, S., and T. Yairi. 2018. "A review on the application of deep learning in system health management." *Mech. Syst. Signal Process.* 107 (Jul): 241–265. https://doi.org/10.1016/j.ymssp.2017.11.024.

Kim, T. S., and S. Y. Sohn. 2021. "Multitask learning for health condition identification and remaining useful life prediction: Deep convolutional neural network approach." *J. Intell. Manuf.* 32 (8): 2169–2179. https://doi.org/10.1007/s10845-020-01630-w.

Lei, Y., N. Li, S. Gontarz, J. Lin, S. Radkowski, and J. Dybala. 2016. "A model-based method for remaining useful life prediction of machinery." *IEEE Trans. Reliab.* 65 (3): 1314–1326. https://doi.org/10.1109/TR.2016.2570568.

Li, F., L. Zhang, B. Chen, D. Gao, Y. Cheng, X. Zhang, Y. Yang, K. Gao, Z. Huang, and J. Peng. 2018a. "A light gradient boosting machine for remaining useful life estimation of aircraft engines." In *Proc., 2018 21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, 3562–3567. New York: IEEE.

Li, H., W. Zhao, Y. Zhang, and E. Zio. 2020. "Remaining useful life prediction using multi-scale deep convolutional neural network." *Appl. Soft Comput.* 89 (Apr): 106113. https://doi.org/10.1016/j.asoc.2020.106113.

Li, X., Q. Ding, and J.-Q. Sun. 2018b. "Remaining useful life estimation in prognostics using deep convolution neural networks." *Reliab. Eng. Syst. Saf.* 172 (Apr): 1–11. https://doi.org/10.1016/j.ress.2017.11.021.

Li, X., H. Jiang, S. Liu, J. Zhang, and J. Xu. 2021. "A unified framework incorporating predictive generative denoising autoencoder and deep coral network for rolling bearing fault diagnosis with unbalanced data." *Measurement* 178 (Jun): 109345. https://doi.org/10.1016/j.measurement.2021.109345.

Li, X., H. Jiang, Y. Liu, T. Wang, and Z. Li. 2022a. "An integrated deep multiscale feature fusion network for aeroengine remaining useful life prediction with multisensor data." *Knowl.-Based Syst.* 235 (Jan): 107652. https://doi.org/10.1016/j.knosys.2021.107652.

Li, X., J. Li, L. Zuo, L. Zhu, and H. T. Shen. 2022b. "Domain adaptive remaining useful life prediction with transformer." *IEEE Trans. Instrum. Meas.* 71 (Aug): 1–13. https://doi.org/10.1109/TIM.2022.3200667.

Linke-Diesinger, A. 2008. *Systems of commercial turbofan engines: An introduction to systems functions*. Berlin: Springer.

Ma, G., S. Xu, and C. Cheng. 2021. "Fault detection of lithium-ion battery packs with a graph-based method." *J. Storage Mater.* 43 (Nov): 103209. https://doi.org/10.1016/j.est.2021.103209.

Matthews, B. 2012. "Dashlink flight data for tail 687." Accessed September 15, 2023. https://c3.nasa.gov/dashlink/resources/664/.

Pan, Y., R. Hong, J. Chen, and W. Wu. 2020. "A hybrid DBN-SOM-PF-based prognostic approach of remaining useful life for wind turbine gearbox." *Renewable Energy* 152 (Jun): 138–154. https://doi.org/10.1016/j.renene.2020.01.042.

Qian, Y., R. Yan, and R. X. Gao. 2017. "A multi-time scale approach to remaining useful life prediction in rolling bearing." *Mech. Syst. Signal Process.* 83 (Jan): 549–567. https://doi.org/10.1016/j.ymssp.2016.06.031.

Qin, L., S. Wu, H. Zhao, and Y. Liu. 2011. "Determination of damage tolerance and remaining lifefor aeroengine crankcase." *J. Beijing Univ. Aeronaut. Astronaut.* 37 (7): 895–900. https://doi.org/10.13700/j.bh.1001-5965.2011.07.026.

Ren, L., S. Li, Y. Laili, and L. Zhang. 2024a. "BTCAN: A binary trend-aware network for industrial edge intelligence and application in aero-engine RUL prediction." *IEEE Trans. Instrum. Meas.* 73: 1–10. https://doi.org/10.1109/TIM.2024.3428643.

Ren, L., H. Wang, and G. Huang. 2024b. "Dlformer: A dynamic length transformer-based network for efficient feature representation in remaining useful life prediction." *IEEE Trans. Neural Networks Learn. Syst.* 35 (5): 5942–5952. https://doi.org/10.1109/TNNLS.2023.3257038.

Ruiz-Tagle Palazuelos, A., and E. López Droguett. 2020. "A novel deep capsule neural network for remaining useful life estimation." *Proc. Inst. Mech. Eng., Part O: J. Risk Reliab.* 234 (1): 151–167. https://doi.org/10.1177/1748006X19866546.

Saxena, A., K. Goebel, D. Simon, and N. Eklund. 2008. "Damage propagation modeling for aircraft engine run-to-failure simulation." In *Proc., Int. Conf. on Prognostics and Health Management*, 1–9. New York: IEEE.

Schwabacher, M., and K. Goebel. 2007. "A survey of artificial intelligence for prognostics." In *Proc., AAAI Fall Symp.: Artificial Intelligence for Prognostics*, 108–115. Washington, DC: Association for the Advancement of Artificial Intelligence.

Son, Y. K. 2011. "Reliability prediction of engineering systems with competing failure modes due to component degradation." *J. Mech. Sci. Technol.* 25 (7): 1717–1725. https://doi.org/10.1007/s12206-011-0415-y.

Tan, C., F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. 2018. "A survey on deep transfer learning." In *Proc., 27th Int. Conf. on Artificial Neural Networks: Artificial Neural Networks and Machine Learning–ICANN 2018: Proc., Part III 27*, 270–279. New York: Springer.

Wang, S. 2013. "Reliability model of mechanical components with dependent failure modes." *Math. Probl. Eng.* 2013 (1): 828407. https://doi.org/10.1155/2013/828407.

Wang, W., H. Song, S. Si, W. Lu, and Z. Cai. 2024. "Data augmentation based on diffusion probabilistic model for remaining useful life estimation of aero-engines." *Reliab. Eng. Syst. Saf.* 252: 110394. https://doi.org/10.1016/j.ress.2024.110394.

Wei, M., M. Chen, and D. Zhou. 2013. "Multi-sensor information based remaining useful life prediction with anticipated performance." *IEEE Trans. Reliab.* 62 (1): 183–198. https://doi.org/10.1109/TR.2013.2241232.

Xiang, F., Y. Zhang, S. Zhang, Z. Wang, L. Qiu, and J.-H. Choi. 2024. "Bayesian gated-transformer model for risk-aware prediction of aero-engine remaining useful life." *Expert Syst. Appl.* 238 (Mar): 121859. https://doi.org/10.1016/j.eswa.2023.121859.

Xiang, S., Y. Qin, J. Luo, H. Pu, and B. Tang. 2021. "Multicellular lstm-based deep learning model for aero-engine remaining useful life prediction." *Reliab. Eng. Syst. Saf.* 216 (Dec): 107927. https://doi.org/10.1016/j.ress.2021.107927.

Xu, T., G. Han, H. Zhu, C. Lin, and J. Peng. 2024. "Multiscale bls-based lightweight prediction model for remaining useful life of aero-engine." *IEEE Trans. Reliab* 73 (4): 1757–1767. https://doi.org/10.1109/TR.2023.3349201.

Xu, T., G. Han, H. Zhu, T. Taleb, and J. Peng. 2023. "Multi-resolution lstm-based prediction model for remaining useful life of aero-engine." *IEEE Trans. Veh. Technol* 73 (2): 1931–1941. https://doi.org/10.1109/TVT.2023.3319377.

Zeng, J., and Z. Liang. 2023. "A deep Gaussian process approach for predictive maintenance." *IEEE Trans. Reliab.* 72 (3): 916–933. https://doi.org/10.1109/TR.2022.3199924.

Zhang, C., P. Lim, A. K. Qin, and K. C. Tan. 2016. "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics." *IEEE Trans. Neural Networks Learn. Syst.* 28 (10): 2306–2318. https://doi.org/10.1109/TNNLS.2016.2582798.

Zhang, D., S. Dey, H. E. Perez, and S. J. Moura. 2017. "Remaining useful life estimation of lithium-ion batteries based on thermal dynamics." In *Proc., 2017 American Control Conf. (ACC)*, 4042–4047. New York: IEEE.

Zhang, Q., Q. Liu, and Q. Ye. 2024. "An attention-based temporal convolutional network method for predicting remaining useful life of aero-engine." *Eng. Appl. Artif. Intell.* 127 (Jan): 107241. https://doi.org/10.1016/j.engappai.2023.107241.

Zhang, Y., Y. Xin, Z.-W. Liu, M. Chi, and G. Ma. 2022. "Health status assessment and remaining useful life prediction of aero-engine based on BiGRU and MMoE." *Reliab. Eng. Syst. Saf.* 220 (Apr): 108263. https://doi.org/10.1016/j.ress.2021.108263.

Zhao, K., Z. Jia, F. Jia, and H. Shao. 2023. "Multi-scale integrated deep self-attention network for predicting remaining useful life of aero-engine." *Eng. Appl. Artif. Intell.* 120 (Apr): 105860. https://doi.org/10.1016/j.engappai.2023.105860.

Zhao, R., R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao. 2019. "Deep learning and its applications to machine health monitoring." *Mech. Syst. Signal Process.* 115 (Jan): 213–237. https://doi.org/10.1016/j.ymssp.2018.05.050.

Zhu, J., N. Chen, and C. Shen. 2020. "A new data-driven transferable remaining useful life prediction approach for bearing under different working conditions." *Mech. Syst. Signal Process.* 139 (May): 106602. https://doi.org/10.1016/j.ymssp.2019.106602.

Zio, E., and G. Peloni. 2011. "Particle filtering prognostic estimation of the remaining useful life of nonlinear components." *Reliab. Eng. Syst. Saf.* 96 (3): 403–409. https://doi.org/10.1016/j.ress.2010.08.009.