

Article

Adaptively Lightweight Spatiotemporal Information-Extraction-Operator-Based DL Method for Aero-Engine RUL Prediction

Junren Shi ¹, Jun Gao ² and Sheng Xiang ^{1,*}

¹ School of Automation, Chongqing University of Posts and Telecommunications, Chongqing 400044, China; shijr@cqupt.edu.cn

² School of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400044, China; gaojun@cqupt.edu.cn

* Correspondence: xiangsheng@cqupt.edu.cn; Tel.: +86-133-6401-9853

Abstract: Accurate prediction of machine RUL plays a crucial role in reducing human casualties and economic losses, which is of significance. The ability to handle spatiotemporal information contributes to improving the prediction performance of machine RUL. However, most existing models for spatiotemporal information processing are not only complex in structure but also lack adaptive feature extraction capabilities. Therefore, a lightweight operator with adaptive spatiotemporal information extraction ability named Involution GRU (Inv-GRU) is proposed for aero-engine RUL prediction. Involution, the adaptive feature extraction operator, is replaced by the information connection in the gated recurrent unit to achieve adaptively spatiotemporal information extraction and reduce the parameters. Thus, Inv-GRU can well extract the degradation information of the aero-engine. Then, for the RUL prediction task, the Inv-GRU-based deep learning (DL) framework is firstly constructed, where features extracted by Inv-GRU and several human-made features are separately processed to generate health indicators (HIs) from multi-row data of aero-engines. Finally, fully connected layers are adopted to reduce the dimension and regress RUL based on the generated HIs. By applying the Inv-GRU-based DL framework to the Commercial Modular Aero Propulsion System Simulation (C-MAPSS) datasets, successful predictions of aero-engines RUL have been achieved. Quantitative comparative experiments have demonstrated the advantage of the proposed method over other approaches in terms of both RUL prediction accuracy and computational burden.

Keywords: RUL prediction; spatiotemporal information; aero-engine; deep learning



Citation: Shi, J.; Gao, J.; Xiang, S. Adaptively Lightweight Spatiotemporal Information-Extraction-Operator-Based DL Method for Aero-Engine RUL Prediction. *Sensors* **2023**, *23*, 6163. <https://doi.org/10.3390/s23136163>

Academic Editor: Mohammad Noori

Received: 29 May 2023

Revised: 26 June 2023

Accepted: 3 July 2023

Published: 5 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Remaining useful life (RUL) prediction, as a significant research domain in prognostics and health management (PHM) [1], offers the potential to forecast the future degradation trajectory of equipment based on its current condition. Transforming scheduled maintenance into proactive operations substantially mitigates the risks of personnel casualties and economic losses resulting from mechanical failures.

With the increasing complexity and sophistication of equipment, conventional PHM methods by dynamic models, expert knowledge, and manual feature extraction have become increasingly limited. Nowadays, fueled by rapid advancements in technologies such as sensors, the Internet of Things, and artificial intelligence, attention has been drawn to DL-based techniques with remarkable performance for RUL prediction [2–4]. Therefore, with industrial data accumulation, conducting DL-based RUL prediction research for equipment, which possess powerful feature extraction capabilities, has not only emerged as a hot research topic in academia but also holds significant practical implications for the industry.

DL-based methods enable the construction of deep neural network architectures, endowing them with more powerful feature extraction capabilities compared with shallow machine learning algorithms. Consequently, these methods can directly learn and optimize features from raw data obtained from complex equipment, as well as infer the RUL, thereby enhancing the accuracy and robustness of RUL estimation. Among various DL techniques, neural networks (NNs) have emerged as state-of-the-art models for addressing RUL prediction problems, attracting significant attention from researchers [5–8].

Recurrent neural networks (RNNs) use the input data and historical data as the final input matrix, which is different from other NNs. Based on this unique design, RNN is well-suited for processing sequential data and has been successfully applied in RUL prediction [9,10]. However, RNN also has its limitations, such as a long recursion time, which indirectly increases the depth and training time of the NN, as well as the frequently occurring issue of vanishing gradients [11]. Long short-term memory (LSTM) was deduced by Hochreiter and Schmidhuber to address the above issues in 1997 [12], and can mitigate the problem of long-term dependencies in RNN and has gained widespread application [13,14].

By comparing the aircraft engines' prediction performance of the vanilla RNN, LSTM, and gated recurrent unit (GRU), Yuan et al. [15] concluded that LSTM and GRU outperformed traditional RNNs. To solve the degradation problem in deep LSTM models, a residual structure is adopted [16]. Zhao et al. [17] conducted an empirical evaluation of an LSTM-based machine tool wear detection system. They applied the LSTM model to encode raw measurement data into vectors for corresponding tool wear prediction. Wu et al. [18] found that the fusion of multi-sensor inputs can enhance the long-term prediction capability of DLSTM. Guo et al. [19] proposed a novel artificial feature constructed from temporal and frequency domain features to boost the prediction accuracy of LSTM. As a commonly used variant of LSTM, GRU has attracted significant attention thanks to its simplified gating mechanism, which reduces the training burden without compromising the regression capability. Zhao et al. [20] presented a GRU model with local features for machine health monitoring. Zhou et al. [21] introduced an enhanced memory GRU network that utilizes previous state data for predicting bearings' RUL. He et al. [22] employed a fault-mode-assisted GRU method for RUL prediction to guide the initiation of the predictive maintenance time of machines. Que et al. [23] developed a combined method by stacked GRU, attention mechanism, and Bayesian methods to predict the RUL of bearings. A deep multi-scale feature fusion network based on multi-sensor data to predict the RUL of aircraft engines was proposed by Li et al. [24], with GRU replacing the commonly used fully connected layers for regression prediction. Ni et al. [25] used GRU to predict the RUL of bearing systems and adaptively adjusted the optimal hyper-parameters using a Bayesian optimization algorithm. Zhang et al. [26] proposed a dual-task network structure based on bidirectional GRU and multi-gate expert fusion units, which can simultaneously assess the health condition of aircraft engines and predict their RUL. Ma et al. [27] introduced a novel deep wavelet sequence GRU prediction model to predict the RUL of rotating machinery, where the proposed wavelet sequence GRU generates wavelet sequences at different scales through wavelet layers.

CNN exhibits powerful spatial feature extraction capabilities and is suitable for classification tasks such as fault diagnosis [28]. However, it is rarely adopted alone for RUL prediction. To enhance the model's ability to extract temporal and spatial information in the RUL prediction task, combining the CNN with RNN or adopting the convolution operators to replace the operations in RNN is the common approach [29,30]. Some researchers combine these two classical models serially and in parallel to construct novel models. Wang et al. [31] replaced the conventional full connections of forward and recurrent processes of GRU with convolutional operators. Similarly, Ma et al. [32] further replaced the full connection on the state-to-state transitions of LSTM as a convolution connection to boost the feature extraction ability. To improve the RUL prediction accuracy, Li et al. [33] presented a combination method by the ConvLSTM and self-attention mechanism. Cheng

et al. [34] introduced a new LSTM variant to predict the RUL of aircraft engines by combining autoencoders and RNNs. The proposed method conducted the pooling operation with LSTM's gating mechanism while retaining the convolutional operations, allowing for parallel processing. Dulaimi et al. [35] proposed a parallel DL framework based on CNN and LSTM to extract the temporal and spatial features from raw measurements. To solve the inconsistent problem of inputs, Xia et al. [36] proposed a CNN-BLSTM method, which has a different time scale processing ability. Xue et al. [37] introduced a data-driven approach for predicting the RUL, which incorporates two parallel pathways: one pathway combines multi-scale CNN and BLSTM, while the other pathway only utilizes BLSTM.

Research works based on LSTM variants and convolution operators have achieved significant success in RUL prediction, but they still have some gaps. The convolutional kernel exhibits redundancy in the channel dimension, and the extraction features lack the ability to adapt flexibly based on the input itself [38]. Moreover, the ability to capture flexible spatiotemporal features not only saves computational resources, but also enables the extraction of rich features, thereby improving the accuracy of mechanical RUL prediction. Additionally, the computation burden is also an important requirement for mechanical RUL prediction. Therefore, it is worth investigating how to enhance the spatiotemporal capturing capability of prediction models while minimizing model parameters to improve prediction speed.

Consequently, considering the aforementioned limitations, a lightweight operator with adaptive feature capturing capabilities named involution GRU (InvGRU) is proposed, and a deep learning framework is constructed based on this operator to predict the RUL of aircraft engines. The RUL prediction results of the C-MAPSS dataset [24] demonstrate that the proposed method outperforms other publicly available methods in terms of prediction accuracy and computational burden.

Below are the contributions of the article:

- **Introducing InvGRU:** We propose a novel operator called InvGRU, which replaces the connection operator in GRU and allows for adaptive capture of spatiotemporal information based on the input itself. InvGRU demonstrates the ability to extract spatiotemporal information with fewer parameters compared with other models.
- **Constructing a deep learning framework:** Building upon InvGRU, we construct a deep learning framework that achieves higher prediction accuracy.
- **Experimental validation:** The experimental results on aircraft engine RUL prediction validate the effectiveness and superiority of the proposed InvGRU-based deep learning framework. It outperforms other models in terms of prediction accuracy and showcases the potential for improved RUL estimation in practical applications.

The outline of the article is as follows. Section 1 introduces the research topic. Section 2 presents a concise explanation of the fundamental principles of GRU and involution. In Section 3, the novel operator InvGRU, which has the adaptively spatiotemporal information extraction ability, is introduced. Then, the proposed methods are thoroughly validated and compared through experiments on the C-MAPSS dataset in Section 4. Finally, Section 5 presents the conclusion.

2. Theoretical Basis

2.1. Inverse Convolution

Thanks to its spatial invariance and channel specificity, CNN has been widely employed for feature extraction. The formula for CNN is as follows:

$$\mathbf{Y}_{i,j,k} = \sum_{c=1}^{C_i} \sum_{(u,v) \in \Delta K} \mathbf{F}_{k,c,u+[K/2],v+[K/2]} \mathbf{X}_{i+u,j+v,c} \quad (1)$$

$$\Delta k = [-[K/2], \dots, [K/2]] \times [-[K/2], \dots, [K/2]] \quad (2)$$

where $\mathbf{X} \in \mathbf{R}^{H \times W \times C_i}$ and $\mathbf{Y} \in \mathbf{R}^{H \times W \times C_o}$ are the input tensor and the output tensor, respectively; $\mathbf{F} \in \mathbf{R}^{C_o \times C_i \times K \times K}$ denotes the kernel of convolution; c_o , c_i , and K denote the output channel number, input channel number, and kernel size, respectively; while H and W represent the spatial dimensions of the output and input channels. Although sharing spatial parameters alleviates some computational burden, it also introduces certain drawbacks. For instance, the extracted features tend to be relatively simplistic, and the convolution kernel lacks flexibility in adapting to input data [38]. Furthermore, the convolutional kernel exhibits redundancy in the channel dimension [38]. The recently proposed inverse convolutional neural network (INN) [38] addresses the aforementioned limitations in a manner that preserves channel invariance and spatial specificity. For the channel dimension, INN allows for sharing of involution kernels, causing INN to provide more flexible modeling of the involution kernels in the spatial dimension, thereby exhibiting characteristics opposite to those of convolutional neural networks. The mathematical expression of INN is as follows:

$$\mathbf{Y}_{i,j,k} = \sum_{(u,v) \in \Delta K} \mathbf{H}_{i,j,u+[K/2],v+[K/2],[kG/C]} \mathbf{X}_{i+u,j+v,k} \quad (3)$$

where $\mathbf{H} \in \mathbf{R}^{H \times W \times K \times K \times G}$ represents the kernel of involution, G represents that all of the channels share G involution kernels, and it is noted that $G \ll C$. Compared with CNN, INN cannot utilize fixed weight matrices as learnable parameters. Instead, it generates corresponding involution kernels by the input features.

$$\mathbf{H}_{i,j} = \Phi(\mathbf{X}_{\Psi_{i,j}}) = \mathbf{W}_1 \text{Relu}(\text{BN}(\mathbf{W}_0 \mathbf{X}_{i,j})) \quad (4)$$

where $\mathbf{W}_0 \in \mathbf{R}^{r \times c}$ and $\mathbf{W}_1 \in \mathbf{R}^{(K \times K \times G) \times r}$ denote the linear transformation matrix, r is the channel reduction rate, BN is the batch normalization, Relu is the Relu activation function, and $\mathbf{X}_{\Psi_{i,j}}$ denotes the index set of coordinate (i, j) . The principle of INN is shown in Figure 1, which is demonstrated as the example when $G = 1$.

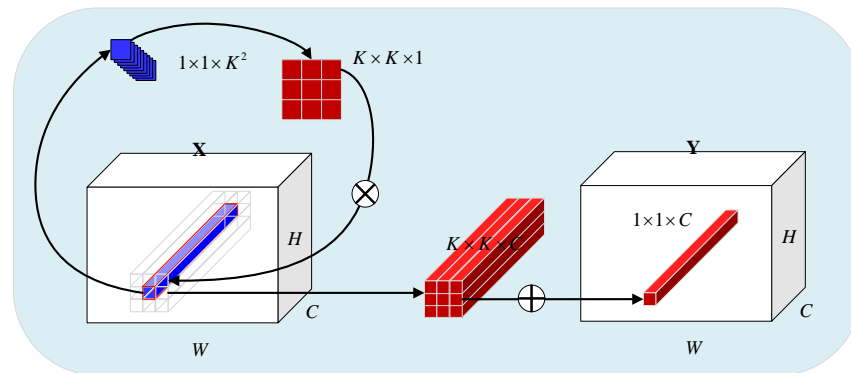


Figure 1. Principle of involution ($G = 1$).

2.2. GRU

GRU, which had fewer parameters compared with LSTM, only has the reset gate \mathbf{r}_t and an update gate \mathbf{z}_t . The structure of GRU is demonstrated in Figure 2. The output \mathbf{h}_t of GRU at the current time step t can be represented by the following equation:

$$\begin{aligned} \mathbf{z}_t &= \sigma(\mathbf{w}_{zx}\mathbf{x}_t + \mathbf{w}_{zh}\mathbf{h}_{t-1} + \mathbf{b}_z) \\ \mathbf{r}_t &= \sigma(\mathbf{w}_{rx}\mathbf{x}_t + \mathbf{w}_{rh}\mathbf{h}_{t-1} + \mathbf{b}_r) \\ \bar{\mathbf{h}}_t &= \tanh(\mathbf{u}_h\mathbf{x}_t + \mathbf{w}_h(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h) \\ \mathbf{h}_t &= (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \bar{\mathbf{h}}_t \end{aligned} \quad (5)$$

where \mathbf{w} denotes the weight matrix of the input data \mathbf{x}_t and recurrent data \mathbf{h}_{t-1} ; \mathbf{b} is the bias; $\bar{\mathbf{h}}_t$ represents the hidden state; \odot is the dot product operator; \tanh and σ are the activation functions; and \mathbf{h}_t denotes the output data.

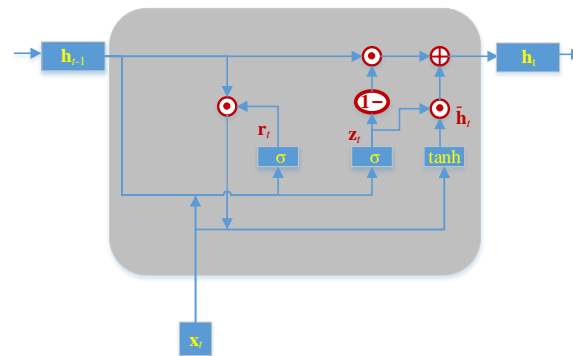


Figure 2. Schematic diagram of GRU, in which \oplus is the addition operator.

3. Proposed Methodology

3.1. Proposed InvGRU

Using convolutional operations to learn representations from multi-source raw data has been shown to outperform hand-crafted features in machine diagnosis and prognosis [28–30]. Recent studies have proposed combining RNN models with CNN representations to capture spatio-temporal information [35–37]. This approach improves the model's ability to understand patterns and relationships over space and time, leading to better analysis and prediction in various domains. A novel operator called Involution GRU (InvGRU) is proposed to address the limitations of the convolution operator. InvGRU introduces involution operations in both the input-to-state and state-to-state transitions, enabling adaptive feature extraction from multi-source raw data while reducing model parameters. This approach enhances the model's ability to capture spatio-temporal information effectively. The diagram of InvGRU is shown in Figure 3.

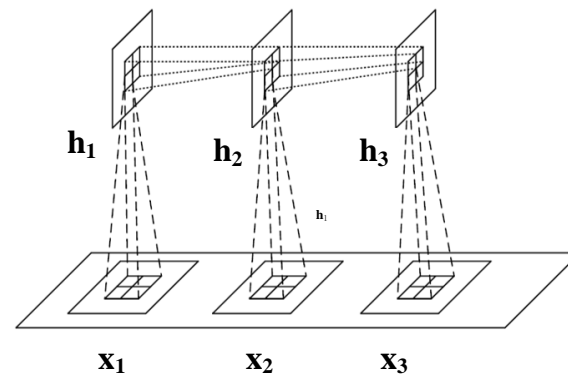


Figure 3. Schematic diagram of InvGRU, where X_1, X_2, X_3 are the time series input matrix and h_1, h_2, h_3 are the hidden features extracted by the involution operator.

To enhance the feature processing method for one-dimensional time series data, a one-dimensional involution algorithm based on one-dimensional vectors as inputs, namely 1D-INN, is adopted. The mathematical expression of 1D-INN is presented below:

$$Y_{i,k} = \sum_{u \in \Delta K} H_{i,u+[K/2],[KG/C]} X_{i+u,k} \quad (6)$$

$$H_i = \Phi(X_{\Psi_i}) = W_1 \text{Mish}(\text{BN}(W_0 X_i)) \quad (7)$$

where $H \in \mathbf{R}^{H \times K \times G}$ is the kernel of 1D-INN, X_{Ψ_i} is the index set of coordinate $(i, 1)$, $W_0 \in \mathbf{R}^{\xi \times c}$ and $W_1 \in \mathbf{R}^{(K \times G) \times \xi}$ are the weight connection matrixes to make a linear transformation, and Mish is the Mish activation function. The other parameters are the same as raw INN. We enhance the feature representation using INN to incorporate longer temporal convolutions, allowing for the prediction of RUL at a larger temporal scale. In the

article, the INN kernel is set to 5 and the size of max-pooling is set to 2, while r is set to 2. InvGRU, similar to the conventional GRU, comprises update gates, reset gates, and cells. The forward process of InvGRU, responsible for computing the output, is defined by the following equations:

Update gates:

$$\mathbf{A}_z^t = \mathbf{w}_{zx} * \mathbf{x}_t + \mathbf{w}_{zh} * \mathbf{h}_{t-1} + \mathbf{b}_z \quad (8)$$

$$\mathbf{z}_t = \sigma(\mathbf{A}_z^t) \quad (9)$$

Reset gates:

$$\mathbf{A}_r^t = \mathbf{w}_{rx} * \mathbf{x}_t + \mathbf{w}_{rh} * \mathbf{h}_{t-1} + \mathbf{b}_r \quad (10)$$

$$\mathbf{r}_t = \sigma(\mathbf{A}_r^t) \quad (11)$$

Cells:

$$\mathbf{A}_h^t = \mathbf{u}_h * \mathbf{x}_t + \mathbf{w}_h * (\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h \quad (12)$$

$$\bar{\mathbf{h}}_t = \tanh(\mathbf{A}_h^t) \quad (13)$$

Cell outputs:

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \bar{\mathbf{h}}_t \quad (14)$$

where $*$ is the operator of the ID-INN; \mathbf{w} and \mathbf{b} terms are the learnable weights and biases, respectively; and the other parameters are the same as GRU.

3.2. The Adopted DL Framework

Based on the proposed InvGRU, a DL framework is adopted to estimate the aero-engines' RUL. The framework diagram in Figure 4 integrates HIs from both neural networks (NNs) and human-made features, enabling a comprehensive approach to RUL prediction. First, InvGRU is employed to extract features based on multi-row measurements, including multiple sensors' data and engine operational condition (OC) information. The attention weights [39] are calculated by the obtained hidden features and combined with the hidden features, and the merged features are input into the following FC layers to generate the HIs from the NN. In the next step, commonly used handcrafted features such as the mean and trend coefficient are calculated from the raw data. The mean represents the average value of a window, while the trend coefficient corresponds to the slope coefficient derived from linear regression on the windowed time series. To obtain the HIs of human-made features, these handcrafted features are then fed into a new fully connected FC layer. Finally, HIs obtained from the neural network and human-made features are concatenated to form the HI set. This concatenated set is inputted into the regression layer, which predicts the RUL.

During the training, supposed that N represents the sample number and the loss function is the mean square error (MSE), which is adopted to evaluate the similarity between the predicted RUL \overline{Rul}_i and the true RUL Rul_i of each sample i . The MSE is calculated using Equation (15) as follows:

$$L(MSE, \theta) = \frac{1}{2} \sum_{i=1}^N (\overline{Rul}_i - Rul_i)^2 \quad (15)$$

Adam is used as the optimization method to tune the parameters θ of the proposed method based on the error gradients during the back-propagation processing. Dropout, a technique for preventing overfitting, is implemented in the model during training. Table 1 shows the hyper-parameters of the proposed DL framework based on InvGRU.

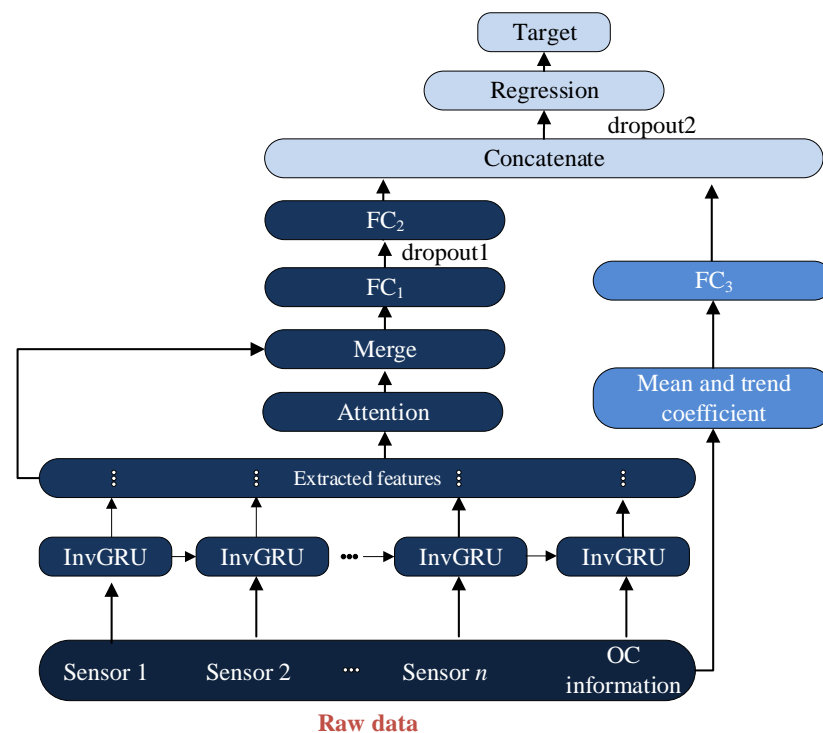


Figure 4. InvGRU-based DL framework.

Table 1. The hyper-parameters of the proposed DL framework based on InvGRU.

Sub Layer	Hyperparameter Value	Sub Layer	Hyperparameter Value
InvGRU	70	Regression (Linear)	1
FC1 (Relu)	30	Learning rate	0.005
FC2 (Relu)	30	Dropout1	0.5
FC3 (Relu)	10	Dropout2	0.3

4. Experimental Analysis

4.1. Evaluation Indexes

The RUL prediction performance of the method is quantitatively characterized using score and root mean square error (RMSE), which are defined by the following formulas:

$$A_i = \begin{cases} \exp(-((\overline{Rul}_i - Rul_i)/13)) - 1, & \overline{Rul}_i < Rul_i \\ \exp((\overline{Rul}_i - Rul_i)/10) - 1, & \overline{Rul}_i \geq Rul_i \end{cases} \quad (16)$$

$$Score = \sum_{i=1}^N A_i \quad (17)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (Rul_i - \overline{Rul}_i)^2} \quad (18)$$

These metric values are inversely proportional to the RUL prediction performance. In other words, a lower value indicates better model performance. Score penalizes delayed predictions more heavily than RMSE, as shown in Figure 5, making it more aligned with engineering practices. Therefore, the score is more reasonable, especially when the RMSE values are close. In the figure, the vertical axis represents the value of RMSE and score, while the horizontal axis represents the errors between the predicted RUL and actual RUL.

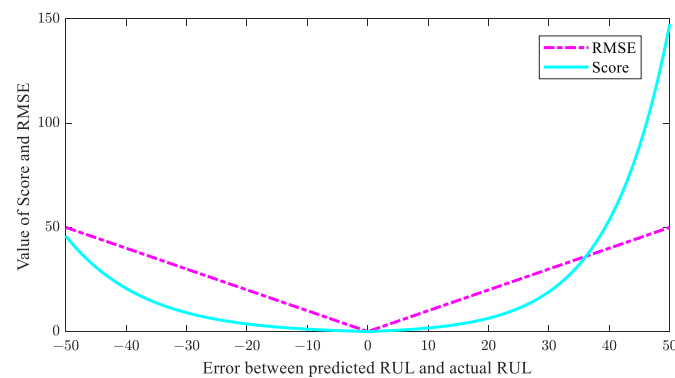


Figure 5. The curves of the two evaluation indexes.

4.2. The Details of the C-MAPSS Dataset

The C-MAPSS dataset, developed by NASA, simulates degradation data for turbofan engines, whose structure is shown in Figure 6. The C-MAPSS dataset can be divided into four subsets based on different operating conditions and fault modes, as described in Table 2. The 21 simulation outputs of C-MAPSS are listed in Table 3, in which \sim , \uparrow and \downarrow represent the stable, upward and descend trends of the sensor measurements.

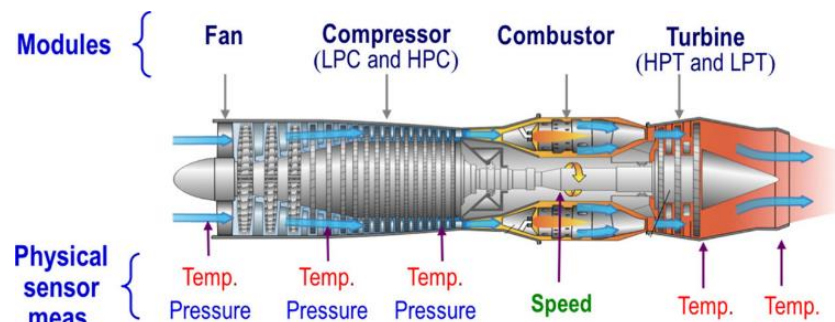


Figure 6. Diagram of the aircraft engine.

Table 2. The details of dataset C-MAPSS.

Subset	FD001	FD002	FD003	FD004
Total number of engines	100	260	100	249
Operating condition	1	6	1	6
Type of fault	1	1	2	2
Maximum cycles	362	378	525	543
Minimum cycles	128	128	145	128

Each subset of the dataset consists of training data, testing data, and corresponding actual RUL values. The training data comprise all engine data from a healthy state to failure, while the testing data include data from engines that were operated before failure. In both the training and testing datasets, a diverse set of engines with varying initial health states is included. This results in variations in the operating cycles of different engines within the same dataset, reflecting the heterogeneous nature of the engine population. To demonstrate the effectiveness of the proposed method, experiments are conducted on all subsets of the dataset.

Table 3. Sensors of C-MAPSS.

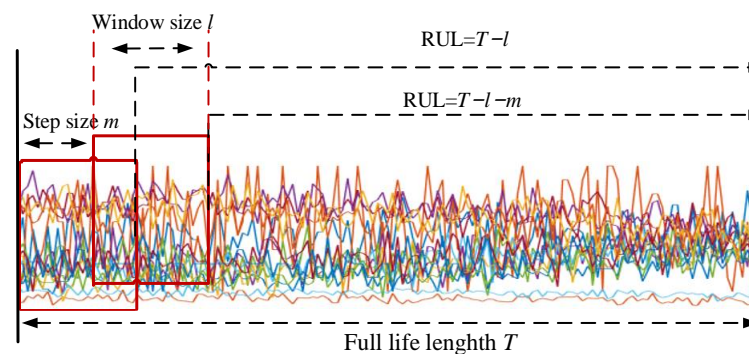
Number	Symbol	Description	Unit	Trend	Number	Symbol	Description	Unit	Trend
1	T2	Total fan inlet temperature	°R	~	12	Phi	Fuel flow ratio to Ps30	pps/psi	↓
2	T24	Total exit temperature of LPC	°R	↑	13	NRf	Corrected fan speed	rpm	↑
3	T30	HPC total outlet temperature	°R	↑	14	NRc	Modified core velocity	rpm	↓
4	T50	Total LPT outlet temperature	°R	↑	15	BPR	Bypass ratio	--	↑
5	P2	Fan inlet pressure	psia	~	16	farB	Burner gas ratio	--	~
6	P15	Total pressure of culvert pipe	psia	~	17	htBleed	Exhaust enthalpy	--	↑
7	P30	Total outlet pressure of HPC	psia	↓	18	NF_dmd	Required fan speed	rpm	~
8	Nf	Physical fan speed	rpm	↑	19	PCNR_dmd	Modify required fan speed	rpm	~
9	Nc	Physical core velocity	rpm	↑	20	W31	HPT coolant flow rate	lbm/s	↓
10	Epr	Engine pressure ratio	--	~	21	W32	LPT coolant flow rate	lbm/s	↓
11	Ps30	HPC outlet static pressure	psia	↑					

4.3. Data Preprocessing

Firstly, not all sensor measurements are included as inputs in the RUL prediction model. Some stable measurements (sensors 1, 5, 6, 10, 16, 18, and 19) are excluded in advance. These sensor measurements contain limited degradation information of the engine and are not suitable for predicting the RUL. Additionally, operating condition information affects the predictive capability of the model. Therefore, the 14 selected sensor measurements and operating condition information serve as the final input for the model. Secondly, we segment the data using the technique demonstrated in Figure 7. T , l , and m represent the total lifecycle, the window size, and the sliding step, respectively. The size of the i -th input is $l \times n$, where n represents the dimension number of the final input of the proposed model. The RUL at this point is $Ts - l - (i - 1) \times m$. Based on the results of the experiments, the sliding window size l is set to 30 and the sliding step m is set to 1. Finally, the linear piecewise RUL technique is used to construct the RUL labels as follows:

$$Rul = \begin{cases} Rul, & \text{if } Rul \leq Rul_{\max} \\ Rul_{\max}, & \text{if } Rul > Rul_{\max} \end{cases} \quad (19)$$

where the preset Rul_{\max} is 125.

**Figure 7.** Processing of data segmentation.

4.4. The Analysis and Comparison of RUL Prediction Results

First, the proposed InvGRU-based DL framework is trained using the training sets from all of the subsets. Then, the test set of the subsets is adopted to test the predictive performance of InvGRU-based DL framework. The prediction results are shown in Figures 8–12. In the figures, the x-axis is the tested aircraft engine unit number and the y-axis denotes the RUL cycles. The predicted RUL and the actual RUL are represented by the solid blue line and dashed green line, respectively.

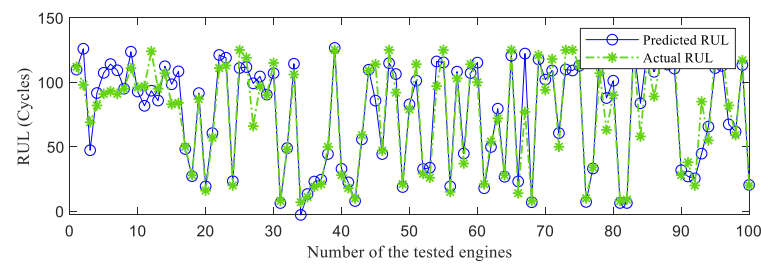


Figure 8. RUL prediction performance on FD001.

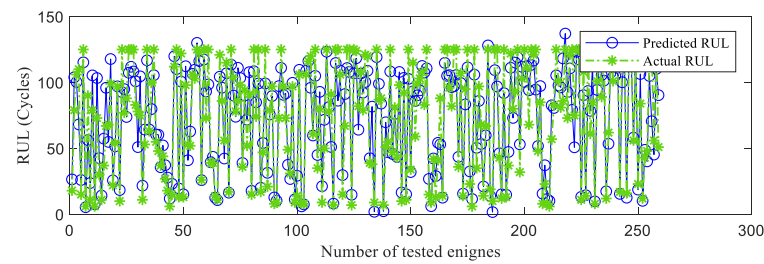


Figure 9. RUL prediction performance on FD002.

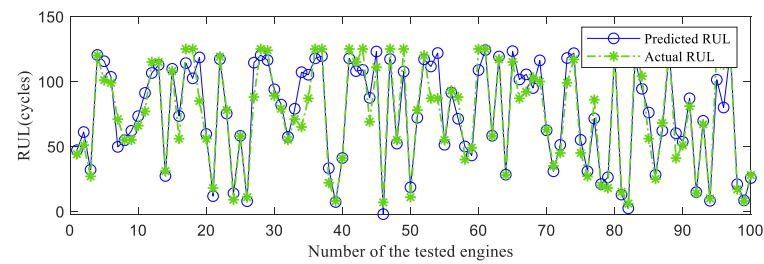


Figure 10. RUL prediction performance on FD003.

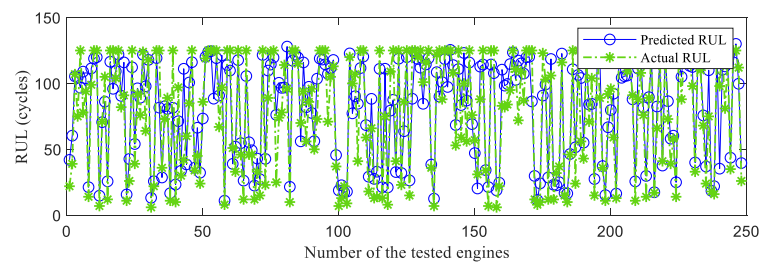


Figure 11. RUL prediction performance on FD004.

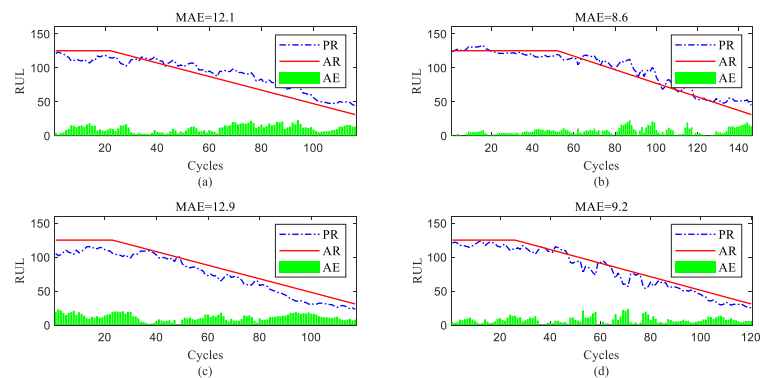


Figure 12. RUL prediction performance of engines of FD001 ((a) engine #46, (b) engine #58, (c) engine #66, and (d) engine #92).

From Figures 8–12, it can be observed that, across all subsets (FD001, FD002, FD003, and FD004), the proposed model demonstrates a consistent prediction of the RUL that aligns closely with the actual RUL for the majority of the tested aircraft engine units. This is evident from the substantial overlap between the blue and green data points, indicating the high accuracy of the proposed model in predicting RUL. Upon closer examination, Figure 8 shows a closer proximity between the RUL and the actual RUL compared with Figures 9–11. This indicates that the proposed model achieves its best performance on the FD001 dataset. Additionally, the RUL prediction performance of the proposed method is superior on the FD003 dataset compared with the FD002 dataset, while it performs worst on the FD004 dataset. Moreover, the RUL prediction effectiveness of the proposed model is higher on the FD001 and FD003 datasets compared with the FD002 and FD004 datasets, highlighting its superior performance under consistent failure modes (FD001 and FD003) compared with multiple operating conditions (FD002 and FD004). This is attributed to the relatively simpler degradation trend of engines under a single operating condition, coupled with significant overlap between the training and testing sets. Furthermore, the accuracy of RUL prediction results is higher for the FD001 dataset than for the FD003 dataset, and higher for the FD002 dataset than for the FD004 dataset. This suggests that, under consistent operating conditions, the proposed model exhibits better RUL prediction performance for single failure modes (FD001 and FD002) compared with composite failure modes (FD003 and FD004). Hence, the proposed model demonstrates higher RUL prediction accuracy for single failure modes compared with multiple failure modes. Additionally, the RUL prediction results on the FD003 dataset surpass those on the FD002 dataset, indicating that the complex failure mode in the C-MAPSS dataset has less influence on the RUL prediction of the proposed model compared with the operating conditions of the aircraft engine units.

To further show the InvGRU-based DL framework performance in predicting the RUL of individual engine units during the overall degradation process, four test engine units randomly selected from all subsets were used to showcase the full-life estimation process shown in Figures 12–15. The blue line in the figures represents the predicted RUL (PR) of the engine unit, while the red line represents the actual RUL (AR). The green bars represent the absolute error (AE) between PR and AR for each cycle. Additionally, the mean of the absolute errors (MAE) between PR and AR across all cycles of the engine unit was computed to evaluate the average prediction error.

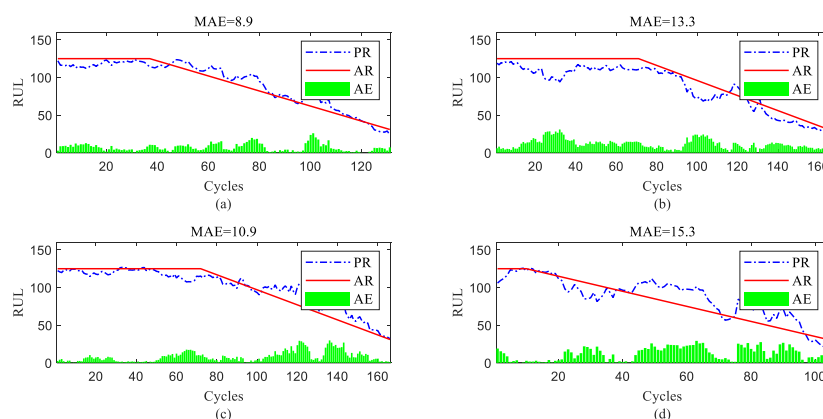


Figure 13. RUL prediction performance of engines of FD002 ((a) engine #9, (b) engine #45, (c) engine #150, and (d) engine #182).

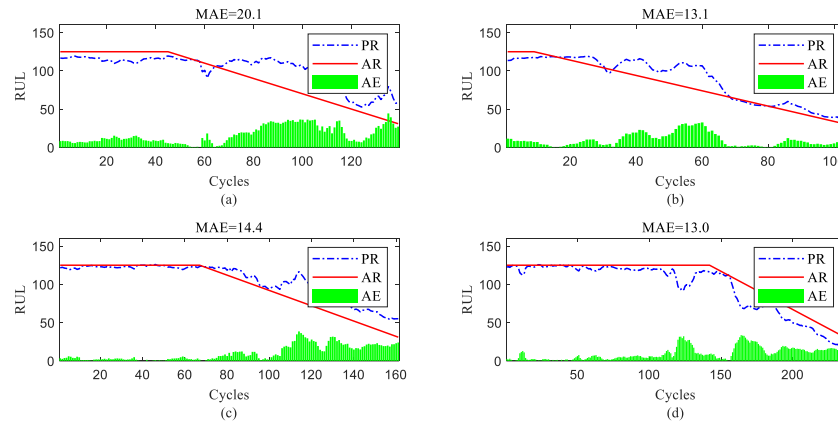


Figure 14. RUL prediction performance of engines of FD003 ((a) engine #25, (b) engine #38, (c) engine #75, and (d) engine #92).

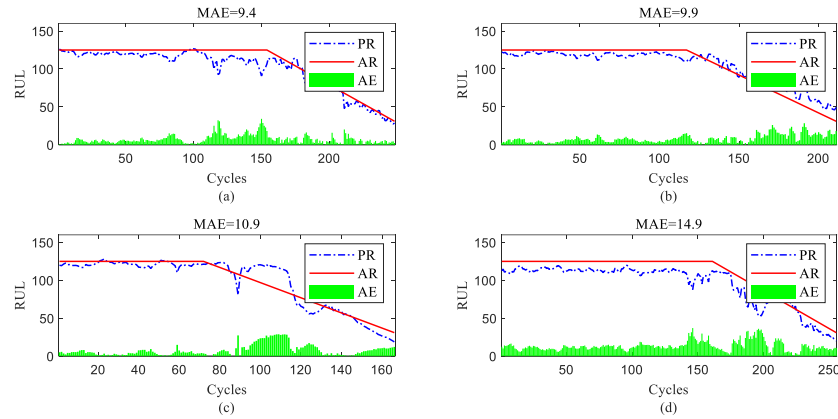


Figure 15. RUL prediction performance of engines of FD004 ((a) engine #35, (b) engine #68, (c) engine #100, and (d) engine #151).

It can be observed from Figures 12–15 that the predicted RUL of the selected test engine units closely aligns with the actual RUL, effectively revealing their degradation trends. Moreover, considering the average values of the MAE in Figures 12–15, the average MAE on the FD001 dataset is 10.7, while the average MAE on the FD002, FD003, and FD004 datasets are 12.1, 15.2, and 11.3, respectively. This indicates that the proposed model exhibits significantly better RUL prediction performance on the FD001 dataset compared with the FD002, FD003, and FD004 datasets. As the number of engine cycles gradually increases, the degradation process begins to manifest and worsen. For most engines, the accuracy of predicting the RUL in the later stages of the degradation process tends to be higher than in the earlier stages. This is evident in Figure 12c, Figure 13a–c, Figure 14b,d and Figure 15a,c,d.

To demonstrate the lightweight nature of the proposed methods and illustrate the lower computational resource consumption, we compare the parameter count and computational cost of the models. For general validation purposes, INN and CNN are employed in a two-dimensional configuration. The parameter count of INN is $\frac{K^2GC+C^2}{r}$, while the computational burden of INN can be divided into two parts: the involution kernel generation component, which is $HW \times \frac{K^2GC+C^2}{r}$, and the multiplication-addition component, which is $HW \times K^2C$. On the other hand, CNN has a parameter count and computational burden of K^2C^2 and $HW \times K^2C^2$, respectively, which is higher than that of INN. This indicates that, under the same hyper-parameters, INN has a smaller computational load compared with CNN. Simultaneously, GRU has a parameter count of $9 \times c_n$ and a computational burden of $8 \times (c_n \times i_l) + 3 \times (c_n \times i_l)^2 + 3 \times (c_n^3 \times i_l^2) + 9 \times (c_n^3 \times i_l^4)$, while LSTM has a

parameter count of $12 \times c_n$ and a computational burden of $12 \times (c_n \times i_l) + 18 \times (c_n \times i_l)^2 + 54 \times (c_n \times i_l)^3$, where c_n represents the number of hidden neurons and i_l represents the input length. GRU exhibits lower computational costs compared with LSTM. From this observation, it is evident that the computational complexity of InvGRU is lower than that of ConvLSTM.

To evaluate the computational efficiency, we selected the challenging FD004 dataset for performance testing. Specifically, we compared the runtime of InvGRU with ConvLSTM on the FD004 dataset. Using the same computing device consisting of Nvidia GeForce RTX2060, Intel(R) Core(TM) i7-10875H, and 16 GB RAM, InvGRU achieved a remarkable 16% reduction in time per epoch, taking only 4 s. In the training stage, each epoch required 8 s and a total of 32 epochs were executed, resulting in a cumulative training time of 256 s. In the testing stage, the inference time was exceptionally fast, with a calculation time of just 0.07 s per sample. Therefore, the proposed method is more concise.

To further highlight the advantages of the InvGRU-based DL framework in predicting RUL, this study conducted comparative experiments on RUL prediction capabilities between the proposed model and several other models, including statistical-based models [34], shallow machine learning models [39], classical deep models [40–42], and recently published deep learning models [4,14,34,43]. To obtain comprehensive performance results, these models were subjected to 10 parallel experiments for RUL prediction on each subset. Subsequently, performance evaluation metrics, namely score and RMSE values, were computed based on the prediction results and are presented in Tables 4–6. Table 4 displays the evaluation metric values for the compared methods on the FD001 and FD002 datasets, Table 5 presents the evaluation metric values for the compared methods on the FD003 and FD004 datasets, and Table 6 represents the mean evaluation metric values for the compared methods across all subsets, providing an average performance assessment of the predictive capabilities of the compared methods on the C-MAPSS dataset.

Table 4. The RUL prediction comparisons of different methods on subset FD001 and FD002.

Model	FD001		FD002	
	Score	RMSE	Score	RMSE
Cox's regression [34]	28,616	45.10	N/A	N/A
SVR [39]	1382	20.96	58,990	41.99
RVR [39]	1503	23.86	17,423	31.29
RF [39]	480	17.91	70,456	29.59
CNN [40]	1287	18.45	17,423	30.29
LSTM [42]	338	16.14	4450	24.49
DBN [41]	418	15.21	9032	27.12
MONBNE [41]	334	15.04	5590	25.05
LSTM+attention+ handcraft feature [20]	322	14.53	N/A	N/A
Acyclic Graph Network [43]	229	11.96	2730	20.34
AEQRNN [34]	N/A	N/A	3220	19.10
MCLSTM-based [4]	260	13.21	1354	19.82
SMDN [14]	240	13.72	1464	16.77
Proposed	238	12.34	1205	15.59

Table 5. The RUL prediction comparisons of different methods on subset FD003 and FD004.

Model	FD003		FD004	
	Score	RMSE	Score	RMSE
Cox's regression [34]	N/A	N/A	1,164,590	54.29
SVR [39]	1598	21.04	371,140	45.35
RVR [39]	17,423	22.36	26,509	34.34
RF [39]	711	20.27	46,568	31.12
CNN [40]	1431	19.81	7886	29.16
LSTM [42]	852	16.18	5550	28.17
DBN [41]	442	14.71	7955	29.88
MONBNE [41]	422	12.51	6558	28.66
LSTM+attention+ handcraft feature [20]	N/A	N/A	5649	27.08
Acyclic Graph Network [43]	535	12.46	3370	22.43
AEQRNN [34]	N/A	N/A	4597	20.60
MCLSTM-based [4]	327	13.45	2926	22.10
SMDN [14]	305	12.70	1591	18.24
Proposed	292	13.12	1020	13.25

Table 6. The comparisons of different methods for RUL prediction based on the C-MAPSS dataset.

Model	Mean Performance	
	RMSE	Score
Cox's regression [34]	49.70	596,603
SVR [39]	32.335	108,277
RVR [39]	27.96	11,716
RF [39]	24.72	29,553
CNN [40]	24.42	7006
LSTM [42]	21.25	2797
DBN [41]	21.73	4461
MONBNE [41]	20.32	3225
LSTM+attention+ handcraft feature [20]	20.80	2985
Acyclic Graph Network [43]	16.80	1716
AEQRNN [34]	19.85	3908
MCLSTM-based [4]	17.40	1216
SMDN [14]	15.36	900
Proposed	13.58	689

Moreover, from Tables 4–6, it can be observed that the proposed model exhibits favorable predictive performance and a significant improvement compared with other deep learning models. This demonstrates that the utilization of spatiotemporal information of the input leads to feature diversification and enhances the model's RUL predictive capability. The proposed Inv-GRU adopted an involution operator to replace the information connection in the gated recurrent unit, enabling the adaptively spatiotemporal information extraction ability and reducing the parameters, and further enhancing the prediction performance of aircraft engine RUL. Based on the aforementioned analysis, it can be concluded that the proposed model exhibits satisfactory universality and accuracy in predicting RUL on the C-MAPSS dataset. Thus, the proposed method can be successfully applied in the aero-engine RUL prediction tasks.

5. Conclusions

To overcome the complexity and limited feature extraction capability of conventional models used for processing spatiotemporal information, a lightweight operator called InvGRU is introduced to enhance the prediction of RUL for aero-engines. InvGRU replaces the information connection in the gated recurrent unit with an adaptive feature extraction

operator known as Involution. By doing so, InvGRU can dynamically extract spatiotemporal information while reducing the number of parameters involved. The output of InvGRU is then passed through a neural network (NN) to transform it into aero-engine health features. These health features, along with manually crafted features, are concatenated and fed into FC layers for dimension reduction and subsequent RUL estimation. The proposed model is trained using existing data and, once trained, it can be utilized to estimate the RUL of aero-engines using new measurements. The proposed method exhibits a 23.44% improvement in the score metric and an 11.58% improvement in the RMSE metric compared with other methods, highlighting its superiority. These results demonstrate the advantages of the proposed approach in accurately predicting the RUL of aero-engines.

Author Contributions: S.X. conceived and designed the experiments; J.S. conducted the programming; J.G. performed the experiments. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by [The National Key Research and Development Program of China] grant number [No. 2022YFE0101000] and partially supported by [school-level research projects] grant number [22XJZXZD05].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: <https://data.nasa.gov/dataset/C-MAPSS-Aircraft-Engine-Simulator-Data/xaut-bemq>.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, R.; Yan, R.; Chen, Z.; Mao, K.; Wang, P.; Gao, R.X. Deep learning and its applications to machine health monitoring. *Mech. Syst. Signal Process.* **2019**, *115*, 213–237. [CrossRef]
2. Khan, S.; Yairi, T. A review on the application of deep learning in system health management. *Mech. Syst. Signal Process.* **2018**, *107*, 241–265. [CrossRef]
3. 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. [CrossRef]
4. Xiang, S.; Qin, Y.; Luo, J.; Pu, H.; Tang, B. Multicellular LSTM-based deep learning model for aero-engine remaining useful life prediction. *Reliab. Eng. Syst. Saf.* **2021**, *216*, 107927. [CrossRef]
5. 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*, 694–700. [CrossRef]
6. Zhou, Z.; Li, T.; Zhao, Z.; Sun, C.; Chen, X.; Yan, R.; Jia, J. Time-varying trajectory modeling via dynamic governing network for remaining useful life prediction. *Mech. Syst. Signal Process.* **2023**, *182*, 109610. [CrossRef]
7. Herzog, M.A.; Marwala, T.; Heyns, P.S. Machine and component residual life estimation through the application of neural networks. *Reliab. Eng. Syst. Saf.* **2009**, *94*, 479–489. [CrossRef]
8. Ren, L.; Cui, J.; Sun, Y.; Cheng, X. Multi-bearing remaining useful life collaborative prediction: A deep learning approach. *J. Manuf. Syst.* **2017**, *43*, 248–256. [CrossRef]
9. Xu, D.; Xiao, X.; Liu, J.; Sui, S. Spatio-temporal degradation modeling and remaining useful life prediction under multiple operating conditions based on attention mechanism and deep learning. *Reliab. Eng. Syst. Saf.* **2023**, *229*, 108886. [CrossRef]
10. Yu, W.; Kim, I.Y.; Mechefske, C. An improved similarity-based prognostic algorithm for RUL estimation using an RNN autoencoder scheme. *Reliab. Eng. Syst. Saf.* **2020**, *199*, 106926. [CrossRef]
11. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [CrossRef]
12. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
13. Xiang, S.; Qin, Y.; Liu, F.; Gryllias, K. Automatic multi-differential deep learning and its application to machine remaining useful life prediction. *Reliab. Eng. Syst. Saf.* **2022**, *223*, 108531. [CrossRef]
14. Xiang, S.; Qin, Y.; Luo, J.; Pu, H. Spatiotemporally multidifferential processing deep neural network and its application to equipment remaining useful life prediction. *IEEE Trans. Ind. Inf. Inform.* **2021**, *18*, 7230–7239. [CrossRef]
15. Yuan, M.; Wu, Y.; Lin, L. Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network. In Proceedings of the 2016 IEEE International Conference on Aircraft Utility Systems (AUS), Beijing, China, 10–12 October 2016; pp. 135–140.

16. Wang, J.; Peng, B.; Zhang, X. Using a stacked residual LSTM model for sentiment intensity prediction. *Neurocomputing* **2018**, *322*, 93–101. [\[CrossRef\]](#)
17. Zhao, R.; Wang, J.; Yan, R.; Mao, K. Machine health monitoring with LSTM networks. In Proceedings of the 2016 10th International Conference on Sensing Technology (ICST), Nanjing, China, 11–13 November 2016; pp. 1–6.
18. Wu, J.; Hu, K.; Cheng, Y.; Zhu, H.; Shao, X.; Wang, Y. Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network. *ISA Trans.* **2020**, *97*, 241–250. [\[CrossRef\]](#)
19. Guo, L.; Li, N.; Jia, F.; Lei, Y.; Lin, J. A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing* **2017**, *240*, 98–109. [\[CrossRef\]](#)
20. 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*, 1539–1548. [\[CrossRef\]](#)
21. Zhou, J.; Qin, Y.; Chen, D.; Liu, F.; Qian, Q. Remaining useful life prediction of bearings by a new reinforced memory GRU network. *Adv. Eng. Inform.* **2022**, *53*, 101682. [\[CrossRef\]](#)
22. He, X.; Wang, Z.; Li, Y.; Khazhina, S.; Du, W.; Wang, J.; Wang, W. Joint decision-making of parallel machine scheduling restricted in job-machine release time and preventive maintenance with remaining useful life constraints. *Reliab. Eng. Syst. Saf.* **2022**, *222*, 108429. [\[CrossRef\]](#)
23. Que, Z.; Jin, X.; Xu, Z. Remaining useful life prediction for bearings based on a gated recurrent unit. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 3511411. [\[CrossRef\]](#)
24. Li, X.; Jiang, H.; Liu, Y.; Wang, T.; Li, Z. An integrated deep multiscale feature fusion network for aeroengine remaining useful life prediction with multisensor data. *Knowl. Based Syst.* **2022**, *235*, 107652. [\[CrossRef\]](#)
25. Ni, Q.; Ji, J.; Feng, K. Data-driven prognostic scheme for bearings based on a novel health indicator and gated recurrent unit network. *IEEE Trans. Ind. Inf. Inform.* **2022**, *19*, 1301–1311. [\[CrossRef\]](#)
26. Zhang, Y.; Xin, Y.; Liu, Z.-W.; Chi, M.; Ma, G. Health status assessment and remaining useful life prediction of aero-engine based on BiGRU and MMoE. *Reliab. Eng. Syst. Saf.* **2022**, *220*, 108263. [\[CrossRef\]](#)
27. Ma, M.; Mao, Z. Deep wavelet sequence-based gated recurrent units for the prognosis of rotating machinery. *Struct. Health Monit.* **2021**, *20*, 1794–1804. [\[CrossRef\]](#)
28. Zhao, M.; Zhong, S.; Fu, X.; Tang, B.; Dong, S.; Pecht, M. Deep residual networks with adaptively parametric rectifier linear units for fault diagnosis. *IEEE Trans. Ind. Electron.* **2020**, *68*, 2587–2597. [\[CrossRef\]](#)
29. Ren, L.; Dong, J.; Wang, X.; Meng, Z.; Zhao, L.; Deen, M.J. A data-driven auto-CNN-LSTM prediction model for lithium-ion battery remaining useful life. *IEEE Trans. Ind. Inf. Inform.* **2020**, *17*, 3478–3487. [\[CrossRef\]](#)
30. Zhao, C.; Huang, X.; Li, Y.; Iqbal, M.Y. A double-channel hybrid deep neural network based on CNN and BiLSTM for remaining useful life prediction. *Sensors* **2020**, *20*, 7109. [\[CrossRef\]](#)
31. Wang, B.; Lei, Y.; Yan, T.; Li, N.; Guo, L. Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery. *Neurocomputing* **2020**, *379*, 117–129. [\[CrossRef\]](#)
32. Ma, M.; Mao, Z. Deep-convolution-based LSTM network for remaining useful life prediction. *IEEE Trans. Ind. Inf. Inform.* **2020**, *17*, 1658–1667. [\[CrossRef\]](#)
33. Li, B.; Tang, B.; Deng, L.; Zhao, M. Self-attention ConvLSTM and its application in RUL prediction of rolling bearings. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 3518811. [\[CrossRef\]](#)
34. Cheng, Y.; Hu, K.; Wu, J.; Zhu, H.; Shao, X. Autoencoder quasi-recurrent neural networks for remaining useful life prediction of engineering systems. *IEEE ASME Trans. Mechatron.* **2021**, *27*, 1081–1092. [\[CrossRef\]](#)
35. 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*, 186–196. [\[CrossRef\]](#)
36. Xia, T.; Song, Y.; Zheng, Y.; Pan, E.; Xi, L. An ensemble framework based on convolutional bi-directional LSTM with multiple time windows for remaining useful life estimation. *Comput. Ind.* **2020**, *115*, 103182. [\[CrossRef\]](#)
37. Xue, B.; Xu, Z.-B.; Huang, X.; Nie, P.-C. Data-driven prognostics method for turbofan engine degradation using hybrid deep neural network. *J. Mech. Sci. Technol.* **2021**, *35*, 5371–5387. [\[CrossRef\]](#)
38. Li, D.; Hu, J.; Wang, C.; Li, X.; She, Q.; Zhu, L.; Zhang, T.; Chen, Q. Involution: Inverting the inference of convolution for visual recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12321–12330.
39. 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*, 2521–2531. [\[CrossRef\]](#)
40. Sateesh Babu, G.; Zhao, P.; Li, X.-L. Deep convolutional neural network based regression approach for estimation of remaining useful life. In Proceedings of the Database Systems for Advanced Applications: 21st International Conference, DASFAA 2016, Dallas, TX, USA, 16–19 April 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 214–228.
41. Zhang, C.; Lim, P.; Qin, A.K.; Tan, K.C. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 2306–2318. [\[CrossRef\]](#)

42. Zheng, S.; Ristovski, K.; Farahat, A.; Gupta, C. Long. short-term memory network for remaining useful life estimation. In Proceedings of the 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), Detroit, MI, USA, 19–21 June 2017; pp. 88–95.
43. 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*, 75464–75475. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.