RMixMLP: An all-MLP block for remaining useful life prediction

***Abstract*—Remaining Useful Life (RUL) estimation has attracted significant attention, due to its importance for designing and managing complex systems. Although models based on Convolutional Neural Networks (CNNs) or Transformers have achieved good performance, their complexity may be unnecessary. In this study, a novel block structure referred to as a Recurrent Mixing Multilayer Perceptron (RMixMLP) is proposed to collect both global and time information, which consists of two distinct components. The first (MixMLP) is an improved block element that incorporates two types of MLPs: 1) a series-mixing MLP used to extract time series information and 2) a channel-mixing MLP employed to integrate channel information. The second, a self-recurrent framework, was included to facilitate parameter sharing within the MixMLP block, thus enabling the implementation of selectable cycles. These machine data, after being processed by several stacked RMixMLP blocks (or a single RMixMLP block), were input to a fully connected layer used to predict the RUL without employing other Deep Learning (DL) methods. The effectiveness of the proposed technique was validated through a series of tests conducted with an aircraft turbofan engine dataset. Experimental results demonstrated that RMixMLP not only achieved better performance compared with existing state-of-the-art methods but also reduced the complexity of the model, thereby highlighting the viability of the proposed MLP structure for RUL prediction.**

***Index Terms*—Remaining Useful Life (RUL) prediction, Deep Learning, MLPs, Turbofan engines.**

I. Introduction

THE estimation of Remaining Useful Life (RUL) plays a critical role in the prognostics and health management (PHM) of complex systems. PHM aims to improve safety and longevity by generating condition-based predictions of failure, using measured data such as temperature, noise, and vibrations. It also provides a mechanism for planning and maintenance scheduling actions used to maximize system RUL [1], [2]. Advances in computer and sensor technology have also facilitated the collection of large data quantities, which has enabled the development and application of data-driven RUL prediction models. Specifically, data-driven RUL prediction methods aim to establish a mapping relationship between RUL values and original features from the target machine [3]. This approach does not require the establishment of a mathematical model, but instead constructs an end-to-end predictive algorithm using data analysis and feature extraction methods, by employing massive historical data. As an advanced machine-learning technology, Deep Learning (DL) has been widely utilized in RUL prediction research. Convolutional Neural Networks (CNNs), which are particularly adept at extracting features from multi-dimensional data [4], have often been utilized in this process. For example, Sateesh et al. [5] used a CNN to estimate RUL values for turbofan engines, thereby demonstrating the efficiency and accuracy of this approach. Other promising RUL results have since been reported [6], [7], in which the size of a convolutional kernel often limits the capacity for extracting long-term dependent information from time series data. In contrast, Long Short-Term Memory (LSTM) is well suited for processing sequence information and excels in learning long-term and sequential data [8]. As with CNNs,LSTM networks and their variants have gained popularity as research tools for RUL predictions [9]–[12]. In their study, Kong et al. [13] proposed a method for RUL prediction that combined CNN and LSTM neural networks to extract spatial and temporal features for more accurate predictions. However, this hybrid CNN-LSTM approach faced significant challenges in terms of computational complexity and the required processing of long sequence data, thereby lacking generalizability.

Attention-based models, especially transformer-based methods, have gained significant popularity due to their ability to process global information while focusing on the critical components of input data [14]. Multiple studies on RUL prediction have recently utilized attention-based techniques [15]–[18]. For instance, Zhang et al. [19] introduced a new deep RUL prediction method, employing an encoder-decoder structure, based solely on self-attention. In addition, Pan [20] proposed a multi-head attention network that combined a convolution-based branch with an attention-based branch to achieve accurate RUL predictions for rocket engines, reducing the RMSE by 50% compared to a CNN. While positional encoding and token embedding sub-sequences used in transformers help to preserve organizational information, it is important to consider that self-attention mechanisms inevitably lead to a loss of temporal information. As such, researchers have turned to Graph Neural Networks (GNNs) to resolve this issue. GNN-based approaches have produced promising results and satisfactory performance, especially when compared to methods that do not utilize spatial relationships [21]–[23]. In their work, Li et al. [24] proposed a comprehensive framework for RUL prediction using a GNN, introducing three graph construction methods, seven graph convolution networks, and four graph pooling techniques. Similarly, Wang et al. [25] proposed a Comprehensive Dynamic Structure GNN (CDSG), employing a designed Dynamic Graph Learning (DGL) module to capture potential dynamic relationships among time series data. In addition, this model incorporated global temporal features obtained through a Visual Graph (VG) algorithm and a Graph Convolutional Network (GCN). However, it is important to note that GNNs are generally shallow networks with limited depth compared to most neural networks.

Several deep learning models (e.g., CNNs, LSTM, GNNs, and attention-based methods) have demonstrated strong performance in RUL prediction. Transformer models have been

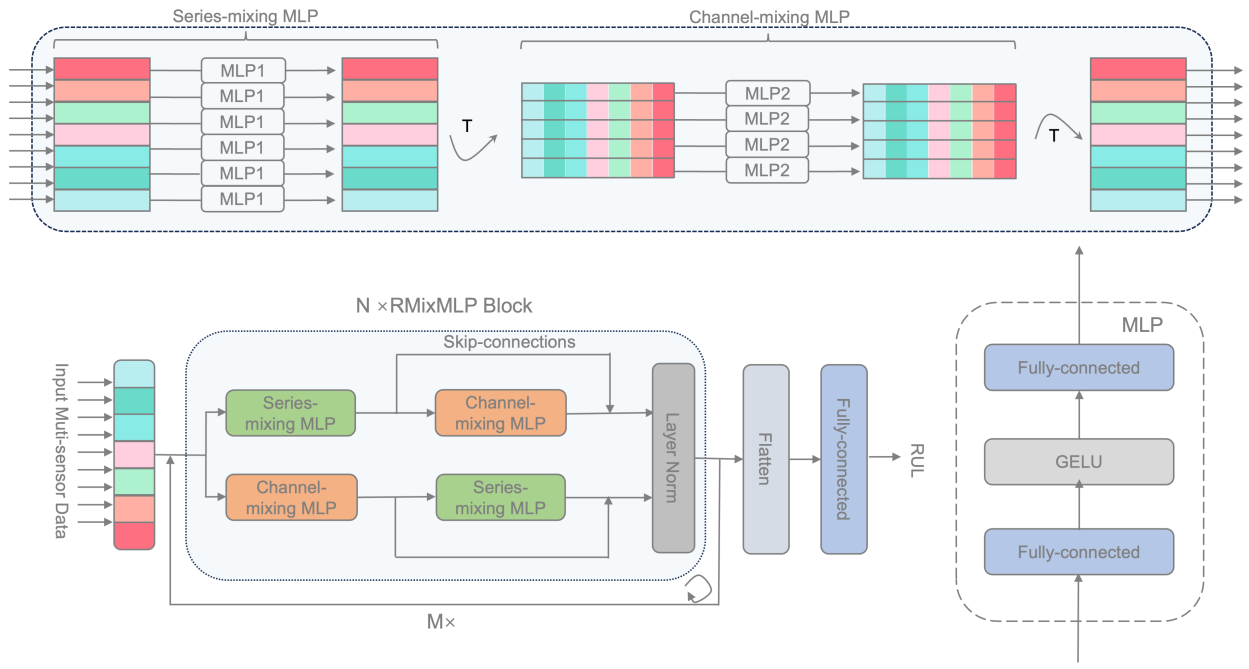

Fig. 1. The proposed model, which consists primarily of several RMixMLP blocks. Each block comprises one series-mixing MLPs and one channel-mixing MLPs, which are composed of two fully-connected layers and a GELU non-linearity. The input data to the RMixMLP undergo M cycles (or no cycles if direct output is required, indicated by $M = 0$). RUL values for a machine are predicted using a fully connected layer and linear fitting in a regression layer.

especially effective and are currently among the most popular methods used for long sequence time forecasting (LSTF) [26]–[28]. However, these models are also complex, and the required computational loads may not be needed if models based on Multilayer Perceptron (MLP) can achieve similar RUL prediction results. For example, a simple linear model proposed by Zeng et al. (DLinear) has surprisingly outperformed many transformer-based models, challenging the efficiency of these algorithms for LSTF tasks [29]. However, this algorithm was unable to address the nonlinear relationship among predictions, while MLP has achieved excellent results for both LSTF tasks [30] and nonlinear problems. MLP techniques have been demonstrated to be comparable to transformers for large vision datasets [31], [32]. MLPs have also been shown to exhibit superior global learning capabilities [33], compared to CNNs or LSTM. Unlike transformers, which are a type of global operator, MLPs do not require position encoding and naturally involve spatial properties. Another significant advantage of MLPs is their simplicity, which reduces the complexity of algorithm implementation. Despite this, there has been limited research on the application of MLP models for RUL prediction, primarily for three reasons. First, unlike LSTF, RUL does not generate a predicted future time series as output. Second, RUL prediction requires extracting critical information from each channel, while LSTF often assumes that channel independence will result in better performance. Third, acquiring mechanical data in engineering applications is more challenging than in computer vision, making it more difficult to train with large quantities of data. MLPs are also prone to overfitting in scenarios involving limited training samples, high model complexity, and noise in the training data. As a result, conventional MLPs often encounter the following challenges:

1) MLPs face several limitations when extracting information between channels, as spatial information is lost when 2D data are flattened from a matrix to a vector. In addition, directly unfolding data from multiple channels can produce highly complex spaces.
2) While MLPs can achieve impressive performance with small models, they can also cause significant overfitting as the model size increases [34]. Therefore, determining an appropriate model size requires careful consideration.

To address these issues, this paper introduces a novel MLP block termed Recurrent Mixing Multilayer Perceptron (RMixMLP), which primarily employs the components of MLPMixer [31] and consists of two components. First, an enhanced MLP block called Mixing Multilayer Perceptron (MixMLP) is introduced, which incorporates two types of MLPs: one for extracting time series information (a series-mixing MLP) and another for processing information across channels (a channel-mixing MLP). The structure of these two components is similar to that of token mixing and channel mixing MLPs in MLPmixer, established by removing skip connections and a layer norm from the original basis. Second, a self-recurrent framework is devised, wherein MixMLP block parameters are shared to improve MLP model performance without altering the network size. This sharing step is a critical component of the self-recurrent framework [35], [36]. Experimental results from small datasets in the RUL domain (e.g., the C-MAPSS data) confirmed that this MLP architecture outperformed CNN, LSTM, GNN, and transformer models, reducing the root mean square error (RMSE) by 9.7%. The primary contributions of this study can be summarized as follows:

1) An RMixMLP block is introduced and the applicability of the MLP architecture for RUL prediction tasks is investigated. Both global and time information can be collected from machine data by utilizing a straightforward MLP network, leading to satisfactory RUL prediction performance.
2) Compared to similar MLPmixer architectures, the RMixMLP model achieved a 20% performance improvement using either the same or fewer parameters.
3) The effectiveness of the proposed RMixMLP was verified experimentally using the C-MAPSS benchmark, producing state-of-the-art performance.

The remainder of this article is organized as follows. In Section II, we first present our proposed RMixMLP block and then propose our RMixMLP-based network. Experimental settings, network hyperparameters, evaluation methods, and our experimental results are discussed in Section III. Conclusions are then provided in Section IV.

II. Proposed Method

This section first introduces the typical algorithm called MLPMixer that uses full-MLP in the field of image processing, and then formalizes the RUL prediction problem by studying a new full-MLP module named RMixMLP Block with the same parameters. The proposed RMixMLP module includes two primary components: a MixMLP block and a self-recurrent framework. These elements are designed to reduce model parameters and improve model performance. In the proposed RUL prediction model, several RMixMLP blocks were connected to a fully connected layer and then to an output layer, as illustrated in Figure 1.

A. Problem setting

The goal of aircraft engine RUL prediction is to forecast the remaining service life of a machine at any given time, by considering various sensor data collected previously. These input data can be represented as $X = (x_1, x_2, ..., x_T)^{\mathrm{T}}$, where T represents the time window duration, $x_i = (x^1, x^2, ..., x^m)$, and m denotes the number of sensor data. The output is typically a number Y, which represents the RUL. This predicted value can be expressed as a mapping function, given by:

$$\boldsymbol{Y} = f(\boldsymbol{X}). \tag{1}$$

B. MLPMixer

MLPMixer [31] is an architecture that relies exclusively on MLPs and is comprised of two types of layers. The first applies MLPs independently to image patches, thereby "mixing" location-specific features. The other applies MLPs across patches, thus mixing spatial information. In essence, MLPMixer separates the two tasks typically performed by CNNs and employs two MLP networks to process the results. This is achieved using 1) token mixing, which combines different positions, and 2) channel mixing, which combines different channels at the same position. MLPMixer achieved competitive scores on image classification benchmarks when trained on large datasets or with modern regularization schemes. However, research on pure MLP architecture models applied to small datasets is currently lacking, particularly in the context of industrial machine operation data. As such, this study aims to address this gap by applying an MLP architecture model to the field of RUL prediction.

C. MixMLP block

The conventional MLP architecture typically adopts a strategy of direct flattening for two-dimensional data. This paper proposes the MixMLP block, which (as in MLPMixer models) trains with input 2D data by dividing it into two independent dimensions. This approach effectively mixes channel information and significantly reduces complexity. This MixMLP block is composed of basic MLP blocks, as illustrated in Figure 1. MixMLP takes a 2D matrix $X \in R^{S\times C}$ as input, where S represents the length of the signal series, and C denotes the number of channels. A MixMLP block then consists of four MLP blocks and a normalization layer. A channel-mixing MLP block can be defined as follows:

$$\boldsymbol{H} = \phi\left(\boldsymbol{X}\boldsymbol{W}_{\mathrm{h}} + \boldsymbol{b}_{\mathrm{h}}\right). \tag{2}$$

$$\boldsymbol{O} = \boldsymbol{H}\boldsymbol{W}_{\mathrm{o}} + \boldsymbol{b}_{\mathrm{o}}. \tag{3}$$

where ϕ is an element-wise non-linearity (i.e., a Gaussian Error Linear Unit (GELU) [37]),$\boldsymbol{W}_{\mathrm{h}} \in R^{C\times H}$,$\boldsymbol{W}_{\mathrm{o}} \in R^{H\times O}$. Since MLP blocks perform feature extraction in different dimensions, the channel-mixing MLP (MLP_c) is then responsible for extracting channel features, while the series-mixing MLP (MLP_s) is used to extract sequence features. The structure of these two MLP blocks is similar, with the only difference being the dimensions of the fully connected layer and the size of the hidden layer. As such, the MixMLP block can be represented as:

$$\begin{aligned}\boldsymbol{Y} = \mathrm{LayerNorm}\left(MLP_{\mathrm{s}}(\boldsymbol{X}) + MLP_{\mathrm{c}}(\boldsymbol{X})\right.\\ \left.+MLP_{\mathrm{s}}(MLP_{\mathrm{c}}(\boldsymbol{X})) + MLP_{\mathrm{c}}(MLP_{\mathrm{s}}(\boldsymbol{X}))\right).\end{aligned} \tag{4}$$

In this process, the inputs and outputs of each MLP block remain unchanged, allowing for easy addition. Once the data have traversed two MLP branches, they are output after undergoing layer normalization [38] in the MixMLP block.

D. Self-recurrent framework

The combination of a MixMLP block and a self-recurrent framework is referred to as an RMixMLP block. The self-recurrent framework involves passing the output data from one layer to the input for use in retraining, as illustrated in Figure 1. Parameter sharing also plays a vital role in this self-recurrent framework, as it provides a straightforward yet effective approach to enhancing parameter efficiency. Parameter sharing can mathematically be expressed as a recursive update of a single RMixMLP block (i.e., one shared block) as follows:

$$\mathbf{Z}_{i,j} = f\left(\mathbf{Z}_{i,j-1}; \boldsymbol{\theta}_i\right), \quad i = 1, \ldots, N; j = 0, \ldots, M. \tag{5}$$

The output of feature embedding with sequences in block i, and cyclic training conducted j times, is denoted by $\mathbf{Z}_{i,j+1}$, where L represents the total number of blocks and $\boldsymbol{\theta}$ denotes shared parameters in the RMixMLP block. The effectiveness of parameter sharing has been investigated in various complex transformer models. Previous research has also demonstrated that this process can help reduce the number of parameters in a neural network without significantly impacting performance. However, most of this research has focused on training large models [39]–[41]. In this study, we explore the impact of parameter sharing on small models, demonstrating that it can actually improve model training and performance to some extent. We also propose a self-recurrent framework in the RMixMLP block, in which the concept of parameter sharing is first applied. Validation experiments with four datasets demonstrated that adding a self-recurrent framework produced two of the best outcomes among comparative algorithms. In the following sections, we define a Cycle as the implementation of one self-recurrent process.

III. Experimental Validation and Analysis

A. Dataset description

The company-modular aero-propulsion system simulation (C-MAPSS) dataset from NASA is a widely used benchmark for RUL prediction [42]. The C-MAPSS data consist of four subsets comprising degradation measurements made under different operating conditions. Multiple fault models are also represented during normal operation, as shown in Table I. Among these, the operational conditions and failure modes for FD001 are the simplest, while those of FD004 are the most complex and challenging. Each C-MAPSS subset includes a training set, a test set, and actual RUL samples, all collected from the same type of aero-engine. Both the training and test sets contain sensor monitoring data for several engine operation modes. Each engine exhibits a series of time cycles, each of which is represented by 26 columns of information. Column 1 provides the engine serial number, column 2 includes the number of operating cycles, columns 3-5 denote three operating settings (i.e., flight altitude, Mach number, and throttle stick angle), and columns 6-26 represent 21 sets of sensor monitoring data. In this study, the proposed RMixMLP-based model utilizes all operational settings and sensor data as input, rather than selected sensor data.

TABLE I
Details of the C-MAPSS Datasets

Data		Number of Units	Operation Conditions	Failure Models	Minimum Cycle	Maximum Cycle
FD001	Train	100	1	1	128	362
	Test	100	1	1	31	303
FD002	Train	260	6	1	128	378
	Test	259	6	1	21	267
FD003	Train	100	1	2	145	525
	Test	100	1	2	38	475
FD004	Train	249	6	2	128	543
	Test	248	6	2	19	486

B. Evaluation metrics

Two objective performance metrics [6], [43] were selected to evaluate RUL prediction performance: Root Mean Square Error (RMSE) and a scoring function (Score). These terms can be defined as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\overline{RUL_i} - RUL_i\right)^2}. \tag{6}$$

$$Sorce = \begin{cases} \sum_{i=1}^{N} e^{-\left(\frac{\overline{RUL_i}-RUL_i}{13}\right)} - 1, & \left(\overline{RUL_i} - RUL_i \le 0\right) \\ \sum_{i=1}^{N} e^{\left(\frac{\overline{RUL_i}-RUL_i}{10}\right)} - 1, & \left(\overline{RUL_i} - RUL_i > 0\right). \end{cases} \tag{7}$$

where i is the serial number, N is the number of test samples, and $\overline{RUL_i}$ and RUL_i are the predicted and true RUL values for the ith sample, respectively. Smaller RMSE and score values indicate better performance by the predictive model. RMSE gives equal weight to the bias in lag forecasting, while score penalizes lagging prediction bias more severely. Although leading forecasting may result in increased maintenance costs, it is considered preferable to the potential risks associated with lag forecasting.

C. Data normalization

As stated in Section III-A, the proposed method did not involve the direct filtering of raw input data. However, it was still necessary to normalize the input data. After comparing Z-score normalization methods, we determined that the maximum-minimum technique was the most suitable for our purposes. This term can be expressed as:

$$x_{\text{norm}}^{i,j} = \frac{x^{i,j} - x_{\text{min}}^{i}}{x_{\text{max}}^{i} - x_{\text{min}}^{i}}. \tag{8}$$

where $x^{i,j}$ represents the jth data for the ith sensor, $x_{\text{norm}}^{i,j}$ denotes the normalized $x^{i,j}$ result, and x_{max}^{i} and x_{min}^{i} represent the maximum and minimum value of the ith sensor observations, respectively.

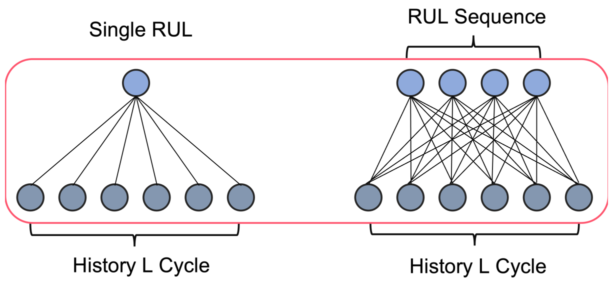

Fig. 2. A sequence of output RUL values.

D. Time window for RUL prediction

Unlike the sampling of data at single time steps, sliding time windows allow for the acquisition of additional time series information. For a multi-sensor time series of length L, sliding the input data one step at a time (using a window of length T) results in $L - T + 1$ time windows. In this paper, a novel prediction technique is introduced [44]. Rather than predicting single RUL values, the aim is to predict a continuous sequence of RUL results, with the last number in the sequence representing the actual RUL. Figure 2 illustrates the last two layers of this proposed RUL prediction model. The C-MAPSS turbofan engine dataset contains only training and test samples. Due to the absence of validation data, 5% of the training data were allocated as a validation set to assess performance. The remaining 95% of the training set was then used to train the network, while the test set remained unchanged. This division of training, test, and validation sets is illustrated in Figure 3.

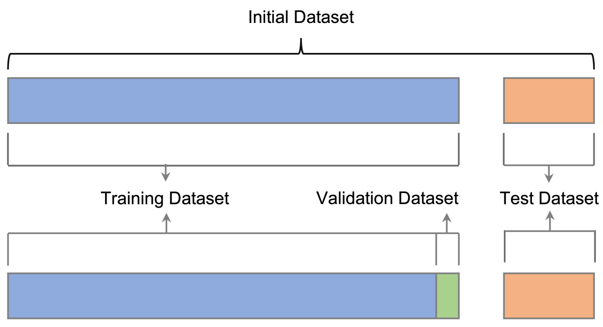

Fig. 3. The division of training, testing, and validation sets.

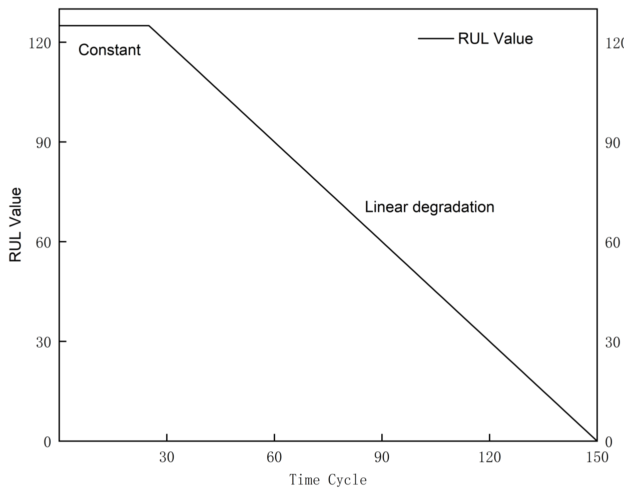

Fig. 4. The piecewise linear degradation process.

E. Piecewise linear RUL

Li et al. employed a piecewise linear degradation model to establish an RUL target, demonstrating the effectiveness of piecewise linear labeling for analyzing C-MAPSS engine degradation data [6], [45]. This procedure can be accurately described using a piecewise linear model, in which the RUL target function can be described as:

$$RUL = \begin{cases} RUL, \text{ if } RUL \leq RUL_{\max} \\ RUL_{\max}, \text{ if } RUL > RUL_{\max}. \end{cases} \tag{9}$$

RUL decline is a linear process in which the life of an engine decreases by one cycle for each cycle of engine operation. However, during the early stages of use, or for short time periods, the engine remains in a healthy state and generally does not experience performance issues. In this stage, the RUL was assumed to be a constant value of 125, set by analyzing the total number of cycles in the engine training set. The details of piece-wise linear degradation are illustrated in Figure 4.

TABLE II
DETAILS OF A PARAMETER TUNING EXPERIMENT DURING A GRID SEARCH

Parameter	Range	Configuration
Time window length	[10,20,**30**,40,50,60,70,80]	60
RUL predicted length	[**1**,10,20,30,40,50,60]	40
Batch size	[8,16,**32**,64,128,256]	16
Learning rate	[0.0005,**0.001**,0.005,0.01,0.05]	0.001
Number of blocks	[**1**,2,3,4,5]	1/1/2/1
Number of $cycles$	[**0**,1,2,3,4]	0/0/1/2

TABLE III
RMIXMLP PARAMETER CONFIGURATIONS

Parameter	Parameter Configuration
Number of input variables	24
Hidden dim of MLP_c	128
Hidden dim of MLP_s	256
Epoch	100
Optimizer	Adam
Activation function	GELU
Loss function	RMSE

F. Tuning parameter experiment

Reproducibility of the experiments was ensured by setting a random seed to guarantee consistency in the results of each trial [29]. Additionally, all parameters for these four working cases were adjusted to the same values to ensure consistency in the model, excluding the number of cycles and the number of RMixMLP blocks. The C-MAPSS dataset was used as an example to validate the effectiveness of the proposed RMixMLP approach and investigate the impact of various parameters, such as time window length, RUL prediction length, batch size, learning rate, number of RMixMLP blocks, number of cycles, choice of activation function, and selected loss function. In Table II, initial parameters are highlighted in bold, and an optimal configuration was selected from the range. Figure 5 compares corresponding RUL prediction results for different parameter combinations. It is noteworthy that RMixMLP performed well during feature extraction in long sequences. Also, a positive correlation was observed between the time window length and model performance for window lengths of less than 60. However, performance was negatively affected by window lengths exceeding 60. This effect was analyzed further, primarily for two reasons. First, a longer time window will result in fewer training samples. Second, as shown in Table I, if the minimum cycle in a test sample set is too small, a long time window will prevent the extraction of additional test samples. Further analysis of block and cycle quantities will be presented in Section III-G. Other important parameters are provided in Table III.

G. Experimental results

The number of blocks and cycles, the two most important parameters in the RMixMLP-based RUL prediction model,

TABLE IV
RUL PREDICTION RESULTS FOR DIFFERENT NUMBERS OF BLOCKS AND CYCLES

		FD001		FD002		FD003		FD004	
Blocks	Cycles	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
1	0	**12.08**	261.82	**11.72**	**627.26**	11.40	189.01	13.70	1006.44
	1	12.21	**258.67**	12.71	678.20	11.12	257.82	14.01	1076.60
	2	12.64	302.95	13.23	854.78	11.50	190.12	**13.63**	**968.81**
	3	12.39	299.26	12.63	750.19	14.16	281.91	14.16	1044.02
2	0	12.76	321.58	12.00	685.43	10.83	209.77	13.86	1020.09
	1	13.29	322.33	12.64	803.00	**10.40**	**184.37**	13.82	1009.76
	2	12.72	315.31	12.91	799.88	12.21	220.48	14.66	1129.44
	3	12.86	327.25	42.97	39128.77	11.29	217.49	15.28	1337.57
Chosen		**12.08**	**261.82**	**11.72**	**627.26**	**10.40**	**184.37**	**13.63**	**968.81**

TABLE V
A COMPARISON OF RUL PREDICTION PERFORMANCE

Category	Method	FD001		FD002		FD003		FD004		Average	
		RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
MLP	MLP (this paper)	12.76	256.28	22.49	3442.36	13.70	358.51	27.34	7982.37	19.07	3009.88
	MLPMixer (this paper)	13.03	282.59	12.99	777.78	10.83	204.56	23.33	10636.16	15.05	2975.27
CNN	DCNN [46]	12.61	273.70	22.36	10412.00	12.64	284.41	23.31	12466.00	17.73	5859.03
LSTM	Bi-LSTM [12]	11.96	206.33	14.48	726.82	13.41	223.36	15.11	892.39	13.74	512.23
Attention-based Model	IMDSSN [44]	12.14	206.11	17.40	1775.15	12.35	229.54	19.78	2852.81	15.42	1265.90
GNN	AdaNet [22]	13.12	248.45	15.20	890.71	12.41	231.06	15.02	**883.21**	13.93	563.35
	CDSG [25]	**11.26**	188.00	18.13	1740.00	12.03	218.00	19.73	2332.00	15.28	1119.50
	DVGTformer [23]	11.33	**179.75**	14.28	797.26	11.89	254.55	15.50	1107.50	13.25	584.76
Hybrid CNN-LSTM	BLCNN [47]	13.18	302.28	19.09	1557.55	13.75	381.37	20.97	3858.78	16.75	1524.99
	HDNN [43]	13.02	245.00	15.24	1282.42	12.22	287.72	18.16	1547.42	14.66	840.64
Hybrid LSTM-Attention	LSTM-MLSA [48]	11.57	252.86	14.02	899.18	12.13	370.39	17.21	1558.48	13.73	770.23
Hybrid CNN-Attention	AGCNN [15]	12.42	225.51	19.43	1492.00	13.39	227.09	21.50	3392.00	16.69	1334.15
Hybrid CNN-LSTM-Attention	MSIDSN [49]	11.74	205.55	18.26	2046.65	12.04	196.42	22.48	2910.73	16.13	1339.84
	SAM-CNN-LSTM [18]	12.60	261.00	15.30	1156.00	13.80	253.00	18.60	2425.00	15.08	1023.75
	ADLDNN [50]	13.05	238.00	17.33	1149.00	12.59	281.00	16.95	1371.00	14.98	759.75
	ABiLSTM-CNN [16]	12.13	233.12	16.01	1230.00	11.96	242.00	18.10	1513.00	14.55	804.53
Transformer	MHT [51]	11.92	215.20	13.70	746.70	10.63	**150.50**	17.73	1572.00	13.50	671.10
Proposed	**RMixMLP**	12.08	261.82	**11.72**	**627.26**	**10.40**	184.37	**13.63**	968.81	**11.96**	**510.57**

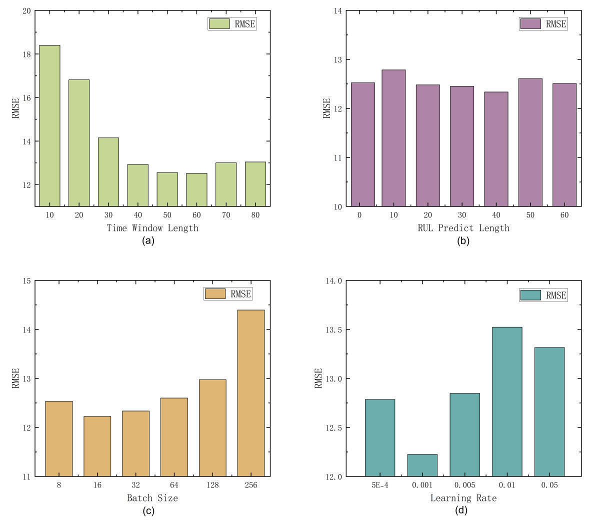

Fig. 5. An illustration of model parameter influence on prediction results, involving various (a) time window lengths, (b) RUL prediction lengths, (c) batch sizes, and (d) learning rates.

were combined for further analysis of the parameters shown in Table II. Corresponding experimental results are presented in Table IV, while Figure 6 provides RUL prediction results for the proposed RMixMLP-based model in four operational conditions (FD001-FD004). Table V compares the RMixMLP-based model with other advanced RUL prediction methods. The experiments reported in this paper demonstrate the superiority of the RMixMLP architecture over the original MLP and MLPMixer models. Notably, RMixMLP outperformed MLPMixer by more than 20%. In addition, we compared several state-of-the-art deep learning models used for RUL predictions. These techniques were applied to all four sub-datasets from C-MAPSS, with a maximum RUL label set to 125. It was evident that these methods relied primarily on CNN, LSTM, GNN, or attention mechanism models, with most of them being a combination of these algorithms. For instance, Song et al. proposed an attention-based bidirectional LSTM-CNN framework [16], which incorporated all three techniques. However, models that deviate from these common approaches are rare. Notably, RMixMLP, which is based solely on MLP, provided new insights into RUL prediction. The results indicate that RMixMLP performed exceptionally well, achieving a 9.7% reduction in RMSE compared to a previous state-of-the-art technique. Although the score model performed slightly worse, it still achieved the best overall results. Only in the FD001 sub-dataset did RMixMLP not outperform other algorithms. Overall, RMixMLP exhibited superior performance compared with existing methods.

We also conducted a detailed parameter comparison analysis with MLP, MLPMixer, and RMixMLP. As seen in Table VI, the parameter counts for RMixMLP and MLPmixer were reduced significantly compared to MLP. This indicates the proposed mixing MLP model not only effectively improved prediction performance, but also successfully reduced the

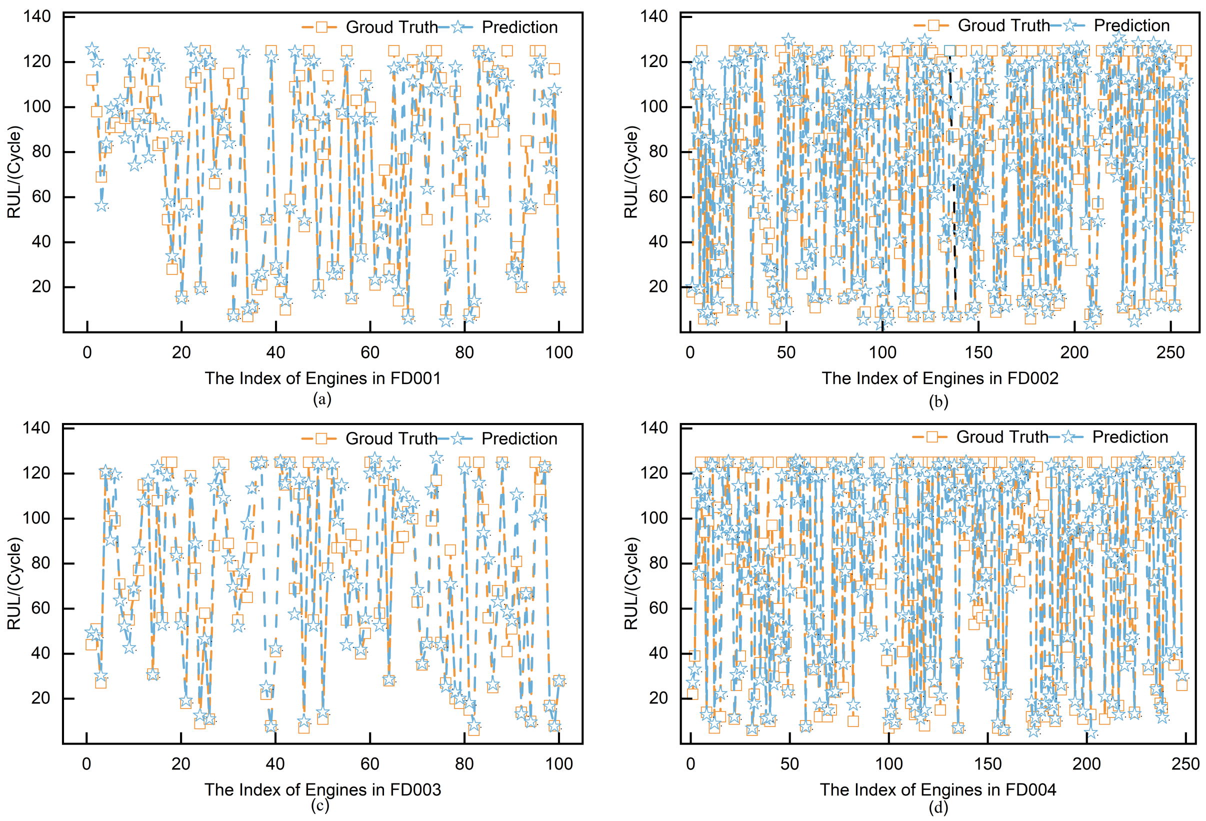

Fig. 6. Predictions for the last recorded data point from different test engine units in FD001-FD004, including (a) 100 test engine units in FD001, (b) 256 units in FD002, (c) 100 units in FD003, and (d) 248 units in FD004.

complexity of the model. In addition, the RMixMLP model achieved a 20% performance improvement using the same or fewer parameters.

TABLE VI
A Comparison of Parameters for Three Models Applied to the FD001-FD004 Datasets

Dataset	FD001	FD002	FD003	FD004
MLP	15347944	11233132	11233132	11233132
MLPMixer	94704	168832	168832	131768
RMixMLP	**94704**	**94704**	**131768**	**94704**

H. Discussion

Previous studies involving RUL prediction for aircraft engines did not include the number of engine cycles as a dynamic covariate in the training model. However, this factor is critical, since it serves as the only time label in the dataset. In addition, identifying the number of engine cycles in practical applications is relatively simple. Table VII presents the results of training with the number of engine cycles as a static covariate in the original model. The corresponding parameter configuration included a batch size of 32, an RUL prediction length of 1, and the number of blocks and cycles shown in Table VIII. All remaining parameters were the same as in section III-F. It is evident that even without careful tuning, model accuracy surpassed that of the original framework presented in section III-G, as the RMSE was reduced by 9.5%. The collected RUL prediction samples consisted of a series of time data. In the training set used for this study, the number of engine cycles was the only time data that needed to be taken into consideration. As such, the experiment conducted in this paper built upon previous work by not incorporating these temporal data.

TABLE VII
RUL Prediction Results with Varying Engine Cycle Quantities

FD001		**FD002**		**FD003**		**FD004**		**Average**	
RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
10.03	174.74	11.82	610.54	8.94	142.38	12.5	822.44	10.82	437.52

TABLE VIII
Experimental Parameters for the Four Working Conditions

FD001		**FD002**		**FD003**		**FD004**	
Blocks	Cycles	Blocks	Cycles	Blocks	Cycles	Blocks	Cycles
1	0	1	1	1	0	2	0

IV. Conclusion

RUL predictions are essential for ensuring system reliability and safety, as they play a critical role in accident prevention. This paper introduced an improved MLP block called RMixMLP, which incorporates two mixing MLPs and a self-recurrent framework and represents a novel approach to RUL prediction research. The performance of this RMixMLP-based technique surpassed that of other state-of-the-art CNN, LSTM, GNN, and attention-based models and their variants. This study involved two distinct experimental aspects, compared to previous studies. First, the C-MAPSS dataset was utilized without any data screening, including all operational settings and sensor monitoring data, which were directly input to the model. Second, a new concept for RUL prediction was introduced, in which a sequence RUL target was used instead of a single RUL target. The significant impact of time labels on RUL predictions was then demonstrated, resulting in an RMSE reduction of 9.5% compared to results without time labels. In addition, despite the score model reaching its optimal performance levels, it was not as effective as RMSE in predicting RUL. To improve the effectiveness of the score metric, future work will involve designing a loss function that focuses on reducing errors in later predictions. Finally, parameter comparisons indicated the proposed RMixMLP model not only improved prediction performance, but also reduced model complexity.

References

[1] R. Siraskar, S. Kumar, S. Patil, A. Bongale, and K. Kotecha, "Reinforcement learning for predictive maintenance: a systematic technical review," *Artificial Intelligence Review*, pp. 1–63, 2023.

[2] D. Thomas and B. Weiss, "Economics of manufacturing machinery maintenance: A survey and analysis of us costs and benefits," 2020.

[3] Y. Qin, D. Chen, S. Xiang, and C. Zhu, "Gated dual attention unit neural networks for remaining useful life prediction of rolling bearings," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6438–6447, 2020.

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[5] G. Sateesh Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *Database Systems for Advanced Applications: 21st International Conference, DASFAA 2016, Dallas, TX, USA, April 16-19, 2016, Proceedings, Part I 21*. Springer, 2016, pp. 214–228.

[6] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.

[7] B. Wang, Y. Lei, N. Li, and T. Yan, "Deep separable convolutional network for remaining useful life prediction of machinery," *Mechanical systems and signal processing*, vol. 134, p. 106330, 2019.

[8] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[9] H. Miao, B. Li, C. Sun, and J. Liu, "Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5023–5032, 2019.

[10] C.-G. Huang, H.-Z. Huang, and Y.-F. Li, "A bidirectional lstm prognostics method under multiple operational conditions," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 11, pp. 8792–8802, 2019.

[11] Y. Cheng, J. Wu, H. Zhu, S. W. Or, and X. Shao, "Remaining useful life prognosis based on ensemble long short-term memory neural network," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–12, 2020.

[12] R. Jin, Z. Chen, K. Wu, M. Wu, X. Li, and R. Yan, "Bi-lstm-based two-stream network for machine remaining useful life prediction," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022.

[13] Z. Kong, Y. Cui, Z. Xia, and H. Lv, "Convolution and long short-term memory hybrid deep neural networks for remaining useful life prognostics," *Applied Sciences*, vol. 9, no. 19, p. 4156, 2019.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[15] H. Liu, Z. Liu, W. Jia, and X. Lin, "Remaining useful life prediction using a novel feature-attention-based end-to-end approach," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1197–1207, 2020.

[16] J. W. Song, Y. I. Park, J.-J. Hong, S.-G. Kim, and S.-J. Kang, "Attention-based bidirectional lstm-cnn model for remaining useful life estimation," in *2021 IEEE international symposium on circuits and systems (ISCAS)*. IEEE, 2021, pp. 1–5.

[17] W. Xu, Q. Jiang, Y. Shen, Q. Zhu, and F. Xu, "New rul prediction method for rotating machinery via data feature distribution and spatial attention residual network," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–9, 2023.

[18] J. Li, Y. Jia, M. Niu, W. Zhu, and F. Meng, "Remaining useful life prediction of turbofan engines using cnn-lstm-sam approach," *IEEE Sensors Journal*, 2023.

[19] Z. Zhang, W. Song, and Q. Li, "Dual-aspect self-attention based on transformer for remaining useful life prediction," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022.

[20] T. Pan, J. Chen, Z. Ye, and A. Li, "A multi-head attention network with adaptive meta-transfer learning for rul prediction of rocket engines," *Reliability Engineering & System Safety*, vol. 225, p. 108610, 2022.

[21] Z. Kong, X. Jin, Z. Xu, and B. Zhang, "Spatio-temporal fusion attention: a novel approach for remaining useful life prediction based on graph neural network," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[22] R. Jin, D. Zhou, M. Wu, X. Li, and Z. Chen, "An adaptive and dynamical neural network for machine remaining useful life prediction," *IEEE Transactions on Industrial Informatics*, 2023.

[23] L. Wang, H. Cao, Z. Ye, H. Xu, and J. Yan, "Dvgtformer: A dual-view graph transformer to fuse multi-sensor signals for remaining useful life prediction," *Mechanical Systems and Signal Processing*, vol. 207, p. 110935, 2024.

[24] T. Li, Z. Zhou, S. Li, C. Sun, R. Yan, and X. Chen, "The emerging graph neural networks for intelligent fault diagnostics and prognostics: A guideline and a benchmark study," *Mechanical Systems and Signal Processing*, vol. 168, p. 108653, 2022.

[25] H. Wang, Z. Zhang, X. Li, X. Deng, and W. Jiang, "Comprehensive dynamic structure graph neural network for aero-engine remaining useful life prediction," *IEEE Transactions on Instrumentation and Measurement*, 2023.

[26] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.

[27] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in Neural Information Processing Systems*, vol. 34, pp. 22 419–22 430, 2021.

[28] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," *arXiv preprint arXiv:2211.14730*, 2022.

[29] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 9, 2023, pp. 11 121–11 128.

[30] A. Das, W. Kong, A. Leach, R. Sen, and R. Yu, "Long-term forecasting with tide: Time-series dense encoder," *arXiv preprint arXiv:2304.08424*, 2023.

[31] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems*, vol. 34, pp. 24 261–24 272, 2021.

[32] H. Liu, Z. Dai, D. So, and Q. V. Le, "Pay attention to mlps," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9204–9215, 2021.

[33] X. Ding, C. Xia, X. Zhang, X. Chu, J. Han, and G. Ding, "Repmlp: Re-parameterizing convolutions into fully-connected layers for image recognition," *arXiv preprint arXiv:2105.01883*, 2021.

[34] Y. Zhao, G. Wang, C. Tang, C. Luo, W. Zeng, and Z.-J. Zha, "A battle of network structures: An empirical study of cnn, transformer, and mlp," *arXiv preprint arXiv:2108.13002*, 2021.

[35] R. Pfeifer, Z. Schreter, F. Fogelman-Soulié, and L. Steels, *Connectionism in perspective*. Elsevier, 1989.

[36] S. J. Nowlan and G. E. Hinton, "Simplifying neural networks by soft weight-sharing," *Neural computation*, vol. 4, no. 4, pp. 473–493, 1992.

[37] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[38] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[39] S. Bai, J. Z. Kolter, and V. Koltun, "Deep equilibrium models," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[40] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “Albert: A lite bert for self-supervised learning of language representations,” *arXiv preprint arXiv:1909.11942*, 2019.

[41] J. Zhang, H. Peng, K. Wu, M. Liu, B. Xiao, J. Fu, and L. Yuan, “Minivit: Compressing vision transformers with weight multiplexing,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 145–12 154.

[42] A. Saxena, K. Goebel, D. Simon, and N. Eklund, “Damage propagation modeling for aircraft engine run-to-failure simulation,” in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–9.

[43] A. Al-Dulaimi, S. Zabihi, A. Asif, and A. Mohammadi, “A multimodal and hybrid deep neural network model for remaining useful life estimation,” *Computers in industry*, vol. 108, pp. 186–196, 2019.

[44] J. Zhang, X. Li, J. Tian, H. Luo, and S. Yin, “An integrated multi-head dual sparse self-attention network for remaining useful life prediction,” *Reliability Engineering & System Safety*, vol. 233, p. 109096, 2023.

[45] Y. Wu, M. Yuan, S. Dong, L. Lin, and Y. Liu, “Remaining useful life estimation of engineered systems using vanilla lstm neural networks,” *Neurocomputing*, vol. 275, pp. 167–179, 2018.

[46] H. Li, W. Zhao, Y. Zhang, and E. Zio, “Remaining useful life prediction using multi-scale deep convolutional neural network,” *Applied Soft Computing*, vol. 89, p. 106113, 2020.

[47] H. Liu, Z. Liu, W. Jia, and X. Lin, “A novel deep learning-based encoder-decoder model for remaining useful life prediction,” in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[48] J. Xia, Y. Feng, C. Lu, C. Fei, and X. Xue, “Lstm-based multi-layer self-attention method for remaining useful life estimation of mechanical systems,” *Engineering Failure Analysis*, vol. 125, p. 105385, 2021.

[49] K. Zhao, Z. Jia, F. Jia, and H. Shao, “Multi-scale integrated deep self-attention network for predicting remaining useful life of aero-engine,” *Engineering Applications of Artificial Intelligence*, vol. 120, p. 105860, 2023.

[50] S. Xiang, Y. Qin, F. Liu, and K. Gryllias, “Automatic multi-differential deep learning and its application to machine remaining useful life prediction,” *Reliability Engineering & System Safety*, vol. 223, p. 108531, 2022.

[51] J. Guo, S. Lei, and B. Du, “Mht: A multiscale hourglass-transformer for remaining useful life prediction of aircraft engine,” *Engineering Applications of Artificial Intelligence*, vol. 128, p. 107519, 2024.