RMixMLP:An all-MLP Block for machine remaining useful life prediction

Jinning Qin[a], Yanyan Wang[a,b,*]

[a]School of Control Science and Engineering, Shandong University, Jinan 250100, China
[b]Shenzhen Research Institute of Shandong University, Shandong University, Shenzhen 518000, China

Abstract

Convolutional neural networks (CNNs) and long short-term memory (LSTM) are the go-to models for machine remaining useful life prediction. Recently, attention-based networks, especially transformer-based ones, seem more sought after. Herein we found that while all the abovementioned methods are sufficient to achieve good performance, neither is necessary. In this paper, an all multi-layer perceptrons (MLPs) block called RMixMLP is proposed, which consists of three parts: one with MLPs applied to extract time series information, one with MLPs applied across channels, and the last is a self-recurrent framework that shares parameters. Several RMixMLP blocks (or only one RMixMLP block) are stacked without any CNN/LSTM module and connect the output to a fully connected layer for Remaining useful life (RUL) prediction. We trained on the aircraft turbofan engine dataset and achieved surprising performance, more than 13% better than state-of-the-art approaches, demonstrating that the RMixMLP is better than all existing approaches. Code is available at:`https://github.com/qinjinning/Obsidian_jinn.git`.

Keywords:
MLPs, Remaining useful life (RUL) prediction, Deep learning, C-MAPSS

1. Introduction

Remaining useful life (RUL) prediction is crucial for prognostics and health management (PHM) of mechanical systems. PHM focuses on improving system safety that utilizes condition-based predictions of failure made using observed data such as temperature, noise, and vibration; and provides a mechanism to plan and schedule maintenance actions to maximize its RUL[1, 2]. Thanks to the development of computer and sensor technology, a large amount of data collection has also been realized in the industry. The increase in the availability of industrial data has paved the way for the research and application of data-driven RUL prediction.

The data-driven RUL prediction method aims to establish the mapping relationship between RUL value and features of the target machine[3], which does not need to establish a mathematical model, and it can successfully construct an end-to-end predictive model through data analysis and feature extraction methods with the help of massive historical data. As one of the advanced machine-learning technologies, Deep learning (DL) has been widely used in RUL prediction research. CNN is an expert in feature extraction from multi-dimension[4]. Therefore, CNN has been applied in RUL prediction. Sateesh et al.[5] attempt to adopt CNN to estimate the RUL of turbofan engines and demonstrate CNN is not only more efficient but also more accurate. Subsequently, CNN shined in RUL research[6]. LSTM is highly suitable for processing sequence information and is good at learning long-term information or capturing sequential information[7]. Like CNN, LSTM and some of its variants have become popular research methods for RUL prediction[8, 9, 10, 11, 12, 13]. For instance, To extract spatial and temporal features for efficient prediction, Kong et al. proposes an RUL prediction method combined with CNN and LSTM neural networks[14]. Nowadays, attention-based models are the most popular method because of their ability to process information globally and allow the model to focus more on the important parts of the input data[15]. A large number of works in RUL based on attention have emerged. Zhang et al. propose a novel deep RUL prediction method, an encoder–decoder structure purely based on self-attention[16]. Combining the convolution-based branch with an attention-based branch, Pan proposed a multi-head attention network for accurate RUL prediction[17].

Many deep learning models, such as CNNs, LSTM and Transformer, have shown good performance in RUL prediction, but the application of the MLP model is still in the blank stage. The RUL prediction of mechanical equipment is a long sequence time-series forecasting (LSTF) task. Not surprisingly, the transformer model has always been the most popular method for LSTF[18, 19, 20], but Zeng et al. proposes DLinear, a simple linear model that surprisingly outperforms many of the Transformer based models, which challenge the effectiveness of the booming Transformers for the LSTF task[21], Google has also followed up on this research and proposed the TiDE model[22]. Again, MLP has been proven comparable to Transformer models on large data sets in vision[23, 24]. Still, there is very little discussion of the application of MLP on RUL prediction. Firstly, compared with pure LSTF, although RUL also inputs a series of time data, the output is not a predicted fu-

*Corresponding author
Email addresses: `qinjinning@mail.sdu.edu.cn` (Jinning Qin), `yanyan.wang@sdu.edu.cn` (Yanyan Wang)

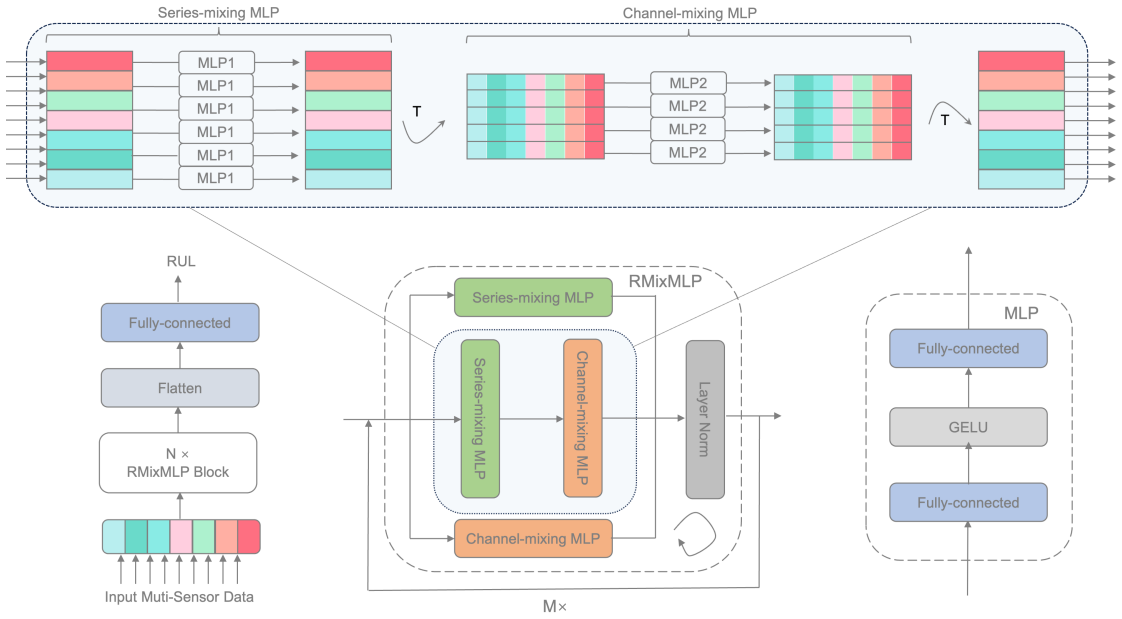

Figure 1: The model is stacked with RMixMLP block. RMixMLP contains two series-mixing MLP and two channel-mixing MLP, each consisting of two fully-connected layers and a GELU nonlinearity, Data input RMixMLP will go through N cycles (or no cycle, direct output, i.e., the number of cycles is 0).

ture time series; secondly,RUL needs to extract the connection between each data, and LSTF currently believes that channel independence will achieve better performance; thirdly, compared to the field of vision, mechanical data in the engineering field is more difficult to obtain, it's difficult to train with large amounts of data, MLP is easy overfitting. Inspired by the above motivations, the traditional MLP has the following problems:

1. MLP does not extract information between channels well, and directly unfolding data from multiple channels creates an unacceptably complex space.
2. MLP can achieve strong performance at the size of small models, but it can also cause serious overfitting as the model size increases[25], and it is not easy to find a suitable size for a data set.

To solve the above problems, we present RMixMLP, an all multi-layer perceptrons (MLPs) Block, which consists of three parts: one with MLPs applied to extract time series information called it Series-Mixing MLP, one with MLPs applied across channels called channel-mixing MLP, and the last is a self-recurrent framework that can improve the performance of the MLP model to a certain extent without changing the model size. Weight sharing is the core of the self-recurrent framework[26, 27]. This paper confirmed that even on small data sets in the RUL field, experimental results represent that the MLP architectures can also achieve performance comparable to CNNs and transformers and, in some cases, even better performance. The significant contributions of this paper are summarized as follows.

1. RMixMLP, a new RUL prediction that consists entirely of MLPs is proposed.The two MixMLPs can extract the connections between the channels well and can reducing the complexity exponentially. Self-recurrent framework experiments have shown that they have a certain role in solving overfitting.
2. On popular RUL prediction benchmarks, the RMixMLP model achieves better performance compared to prior neural network-based baselines.

2. RMixMLP Block

In this section, we describe our proposed RMixMLP block for RUL prediction. It consists of two key components, MixMLP and self-recurrent framework, to improve training stability and model performance.

2.1. MixMLP

Usually, traditional MLP models always expand a set of two-dimensional input data directly first and then input it into the fully connected layer. The MixMLP proposed in this paper is to fully connect the two dimensions of the input separately, which can better mix channel information and greatly reduce complexity. MixMLP consists of the most basic MLP block connections. Figure 1 summarizes the architecture.

MixMLP takes as input a s a two-dimensional (2-D) matrix $X \in R^{S \times C}$, where S is the length of the signal series, and C represents the number of channels. A MixMLP block consists of 4 MLP blocks and a normalization layer. An MLP block can be written as follows:

$$\boldsymbol{H} = \phi\left(\boldsymbol{X}\boldsymbol{W}_{\mathrm{h}} + \boldsymbol{b}_{\mathrm{h}}\right) \tag{1}$$

$$\boldsymbol{O} = \boldsymbol{H}\boldsymbol{W}_{\mathrm{o}} + \boldsymbol{b}_{\mathrm{o}} \tag{2}$$

Here ϕ is an element-wise nonlinearity (GELU[28]). Because MLP blocks perform feature extraction on different dimensions,

the MLP_c is the MLP block for channel feature extraction and MLP_s is the MLP block for sequence feature extraction.The structure of these two MLP blocks is exactly the same, except that the dimensions of the fully connected layer and the size of the hidden layer are different. Thus a MixMLP block can be represented by:

$$\boldsymbol{Y} = LayerNorm\left(MLP_{\mathrm{s}}\left(\boldsymbol{X}\right) + MLP_{\mathrm{s}}\left(MLP_{\mathrm{c}}\left(\boldsymbol{X}\right)\right) + MLP_{\mathrm{c}}\left(\boldsymbol{X}\right)\right) \quad (3)$$

In this paper, Instead of dimensionally transforming the input, we used a little trick, which can be referred to our code. The inputs and outputs of each MLP block are exactly the same, so they can be easily added together. After the data is branched through three MLP blocks, it is output after a layer normalization[29], which is MixMLP.

2.2. Self-recurrent framework

MixMLP plus self-recurrent framework is RMixMLP mentioned above. The self-recurrent framework is a method that passes the output data of one layer to the input for retraining, as shown in Figure 1; parameter sharing is the core of the self-recurrent framework, which is a simple but effective way to improve parameter efficiency. Mathematically, Parameter sharing can be formulated as a recursive update of one RMixMLP block (i.e., one shared block):

$$\mathbf{Z}_{i,j} = f\left(\mathbf{Z}_{i,j-1}; \boldsymbol{\theta}_i\right), \quad i = 0, \ldots, L; j = 1, \ldots, N \quad (4)$$

where $\mathbf{Z}_{i,j+1}$ denotes the feature embedding of the sequence in layer i and trains in this block cyclically j times, L is the total number of layers, and $\boldsymbol{\theta}$represents the shared weights of the RMixMLP block across all layers. The efficacy of weight sharing has been explored and proved in natural language transformer models[30, 31, 32, 33]. It can prevent the number of parameters from growing with the depth of the network without seriously hurting the performance, thus improving parameter efficiency. Still, this prior work has focused on training on the large model rather than a small model. This paper finds that the use of parameter sharing on small models may positively affect the model's training and improve the model's performance to a certain extent, so the idea of parameter sharing can be applied to the model as a hyperparameter. Two of the best performances in the four datasets in this paper are the addition of parameter sharing. In the following, one *cycle* is defined that implementing one self-recurrent.

3. Experimental validation and analysis

3.1. Dataset description

The Company-Modular Aero-Propulsion System Simulation (CMAPSS) dataset from NASA is a widely used benchmark data in RUL prediction[34]. The CMAPSS dataset consists of four sub-datasets that record degradation data under different operating conditions and fault models during engine operation, as shown in Table 1. Among these four conditions, the operational conditions and failure modes of FD001 are the simplest, while those of FD004 are the most complicated and challenging. Each subdataset in the CMAPSS dataset contains three parts of the RUL data for the training set, the test set, and the true test RUL, all derived from the same type of aero-engine. Both the training and test sets of the dataset contain sensor monitoring data for multiple engine operations, each engine consists of a series of time cycle cycles, and each cycle data contains 26 columns of data; column 1 is the engine serial number, column 2 is the number of operating cycles, columns 3-5 are three operating settings (flight altitude, Mach number, and throttle stick angle), and columns 6-26 are 21 monitoring sensors data. It is worth mentioning that in this paper, the input of RMixMLP is all operating settings and sensor data, Instead of selected sensor data.

Table 1: Details of the C-MAPSS datasets

Data		Number of unit	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

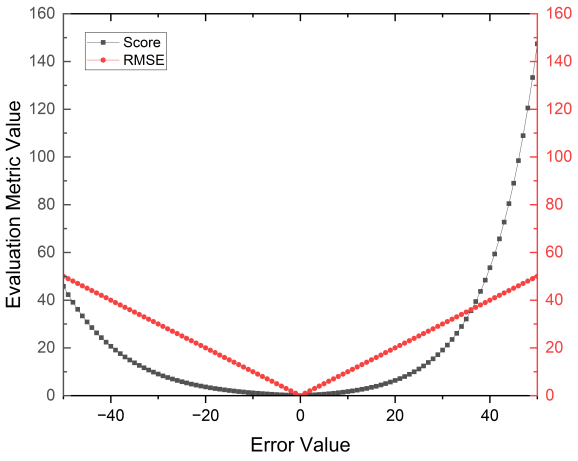

Figure 2: The comparison of scoring function and RMSE function.

3.2. Evaluation metrics

Referring to some previous work, two objective performance measures is secected, ie., Root Mean Square(RMSE), and scoring function(Score), to evaluate the RUL prediction performance. These indexes are defined by

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\overline{RUL_i} - RUL_i\right)^2} \quad (5)$$

$$Sorce = \begin{cases} \sum_{i=1}^{N} e^{-\left(\frac{\overline{RUL_i} - RUL_i}{13}\right)} - 1 & \left(\overline{RUL_i} - RUL_i \leq 0\right) \\ \sum_{i=1}^{N} e^{\left(\frac{\overline{RUL_i} - RUL_{ti}}{10}\right)} - 1 & \left(\overline{RUL_i} - RUL_i > 0\right) \end{cases} \quad (6)$$

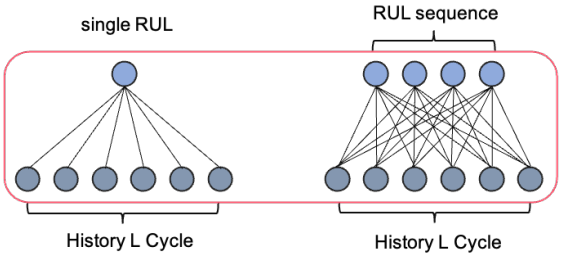

Figure 3: The output RUL value is not a single one, but a sequence

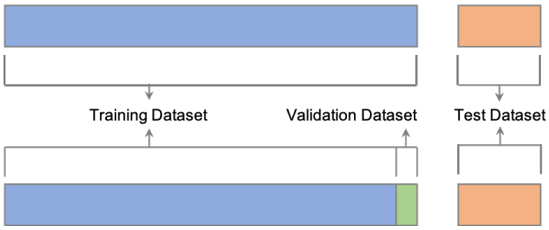

Figure 4: The division of the training, test, and validation datasets

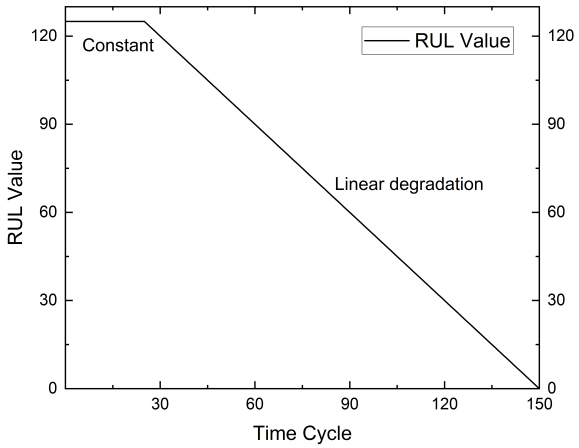

Figure 5: Details of piece-wise linear degradation.

where i is the serial number, N is the number of test samples,$\overline{RUL_i}$ and RUL_i are the RUL predicted value and the truth value of the jth sample. Smaller values for RMSE and Score indicate better performance of the predictive model. The main difference between the two evaluation metrics is shown in Figure 2; RMSE gives equal weight to the bias of lagging forecasting, while Score penalizes lagging prediction bias more severely, although leading may increase maintenance costs, but far better than the risk posed by lagging forecasting.

3.3. Data normalization

As mentioned in section 3.1, this paper will not filter the raw data and input directly, but the data must be normalized, which is still necessary. After a comparison of some normalization methods, the maximum–minimum normalization is adopted to achieve data normalization, which can be shown in Eq. (5):

$$x_{\text{norm}}^{i,j} = \frac{x^{i,j} - x_{\text{min}}^{i}}{x_{\text{max}}^{i} - x_{\text{min}}^{i}} \tag{7}$$

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

3.4. Time window and out of RUL prediction

Compared with the data sampled by a single time step, more time series information can be obtained by sliding the time window. For a multi-sensor time series of length L, $L - T + 1$ time windows can be obtained by using a time window of length T to slide the input data one-time step at a time along the time axis. in this paper, for the prediction of RUL, compared to all previous methods, a different approach is tried, not just to predict a single value but to a continuous sequence of RUL predictions, the last value of the sequence is the real RUL. Figure 3 shows the model's last two layers. Since the CMAPSS turbofan engine dataset contains only the training set and the test machine, the validation set is missing. Therefore, when building the dataset for each operation condition, after the original data is generated through the time window, 5% of the training set is used as the validation dataset to verify the performance, and only the remaining 95% is used as training samples to train the network model, the testing set remains unchanged. The division of the training, test, and validation datasets is shown in Figure 4.

3.5. Piece-wise linear RUL

Wu et al. used a piece-wise linear degradation model to determine the RUL target, demonstrating that piece-wise linear labelling of CMASS engine degradation data is effective[35, 36]. The degradation process of the engine RUL can be described as a piece-wise linear mode. The formulation of the RUL target function is given by

$$RUL = \begin{cases} RUL, \text{ if } RUL \leq RUL_{\text{max}} \\ RUL_{\text{max}}, \text{ if } Rul > RUL_{\text{max}} \end{cases}, \tag{8}$$

In reality, the decline of RUL is a linear process, and for each cycle of engine operation, its life decreases by one cycle. Still, the engine is in the early stages of operation or only for a short period of time, during which the engine is in a healthy state and generally does not have performance degradation problems. At this stage, RUL is assumed to be constant. By analyzing the overall number of cycles of the engine training set, a constant value of 125 can be set. Details of piece-wise linear degradat is shown in Figure 5

3.6. Tuning parameter experiment

Since in this paper, several new concepts is introduced, there are many parameters. To maintain the consistency of the model, all parameters are adjusted to the same for these four working cases, except for the number of *Cycle* and the number of the RMixMLP block. To verify the validity of the proposed

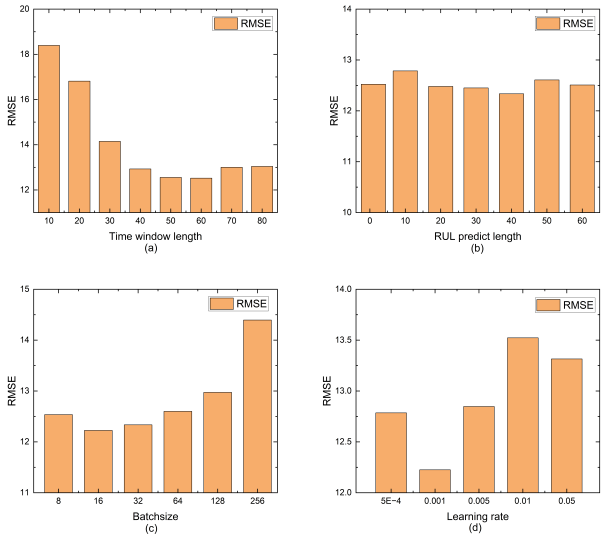

Figure 6: Influence of model parameters. (a) Results of different time window length.(b) Results of different RUL predict length. (c) Results of different batch sizes. (d) Results of different learning rate.

RMixMLP approach, this section takes the C-MPASS dataset as an instance to discuss the influence of parameters, namely, time window length,RUL predict length, batch size, learning rate, number of RMixMLP block, number of *cycle*, activation function and loss function; details are shown in Table 2.

Table 2 in bold is the initial parameter, which is finally configured as the optimal parameter selected from the range. RUL prediction results of different parameters are compared in Figure 6; it is worth noting that the RMixMLP has a relatively good feature extraction for long sequences; when the length of the time window is lower than 60, the time window length and the performance of the model are positively correlated, and the performance decreases slightly after more than 60, we analyze mainly for two reasons: firstly, because the longer time window will lead to fewer training samples, and secondly, because when the model performance is verified with the test set, more samples will not be able to extract the long time window directly. This paper verifies the test set sample with a time window less than the given value after making up the average. The number of blocks and *cycle*s will be analysed in Section 3.7. Additional important parameters are given in Table 3.

3.7. Experimental results

For our proposed RMixMLP, the number of blocks and cycles is the most important, and these two parameters need to be combined to continue the analysis of the alternative parameters in Table 2, and the experimental results are shown in Table 4. At the same time, Figure 7 represents the RUL prediction results of the proposed RMixMLP under four operational conditions FD001–FD004.

As mentioned in the introduction, deep learning has a lot of research on RUL prediction, and in this paper, some classic or relatively new work is selected to compare, and the selected

Table 2: The details of parameter tuning experiment of the grid search

Parameter	Range	Configuration
Time window length	[10,20,**30**,40,50,60,70,80]	60
RUL predict 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 block	[**1**,2,3,4,5]	[1,2]
Number of cycle	[**0**,1,2,3,4]	[0,1,2,3]

Table 3: Parameters configuration of the RMixMLP

Parameter	Parameter configuration
Number of input variables	24
hidden dim of channel	128
hidden dim of series	256
Epoch	100
Optimizer	Adam
Activation function	GELU
Loss function	RMSE

method is applied to all four subdatasets of C-MPASS and the maximum RUL label is set to 125. It is eas to find that these works are basically based on CNN, LSTM or attention; most of them are a combination of these methods, such as song proposed attention-based bidirectional LSTM-CNN model[37]is a combination of these three, the model that can jump out of these methods is very rare, the RMixMLP is completely based on MLP that brings new inspiration to RUL prediction. A comparison of RMixMLP and some advanced RUL prediction methods is shown in Table 5, it can be seen that RMixMLP performance is surprisingly, which is a 13% improvement in RMSE compared to the previous sota method, Score performed slightly poorly, but it also achieved the best results, only on the FD001 did not achieve sota results, and the overall performance was better than all current methods.

3.8. Discussion

In the RUL prediction experiment of aircraft engines, we found that none of the previous work had entered the number of cycles of the engine as a static covariate into the model for training. In fact, this situation can be considered. In industry, it is not very difficult for us to get the number of cycles of the engine. Table 6 shows the results of training the number of engine cycles entered into the original model as a static covariate, about parameter configuration, batchsize is 32, RUL prediction length is 1, the number of blocks and *cycle*s are shown in Table 7, and the rest are the same as section 3.6.

It's not hard to find that even if the model has not been carefully tuned, and the accuracy has far exceeded that of the original performance in section 3.7. In fact, in the RUL prediction, all the data obtained is a series of time data. Regarding the dataset used in this article, the number of cycles of the engine is the only time data and should be considered. Still, in order to be fair, the experiment in this paper continues the previous work and does not enter this data.

Table 4: The RUL prediction results of the different number of blocks and *cycle*s.

		FD001		FD002		FD003		FD004	
number of block	number of cycle	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	12.24	266.62	11.79	669.23	11.90	187.68	14.02	1020.71
	2	**12.07**	267.40	12.92	837.07	13.51	258.10	13.84	1017.77
	3	12.71	320.37	12.43	801.64	11.19	258.90	14.17	1054.91
2	0	12.39	289.88	11.91	689.19	11.12	254.89	**13.61**	**971.44**
	1	12.77	298.33	12.96	802.79	**10.29**	181.23	14.01	1032.84
	2	12.38	268.17	13.09	819.10	10.63	**178.75**	13.92	1054.56
	3	13.08	288.36	12.24	749.95	11.16	234.52	14.98	1128.76
chosen		**12.07**	**267.40**	**11.72**	**627.26**	**10.29**	**181.23**	**13.61**	**971.44**

Table 5: The comparison of the RUL prediction performance.

Methods	FD001		FD002		FD003		FD004		Average	
	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
DCNN[36]	12.61	273.70	22.36	10412.00	12.64	284.41	23.31	12466.00	17.73	5859.03
BLCNN[38]	13.18	302.28	19.09	1557.55	13.75	381.37	20.97	3858.78	16.75	1524.99
AGCNN[39]	12.42	225.51	19.43	1492.00	13.39	227.09	21.50	3392.00	16.69	1334.15
MSIDSN[40]	11.74	205.55	18.26	2046.65	12.04	196.42	22.48	2910.73	16.13	1339.84
IMDSSN[41]	12.14	206.11	17.40	1775.15	12.35	229.54	19.78	2852.81	15.42	1265.90
Double Attention[42]	12.25	**198.00**	17.08	1575.00	13.39	290.00	18.86	1741.00	15.40	951.00
SAM-CNN-LSTM[43]	12.60	261.00	15.30	1156.00	13.80	253.00	18.60	2425.00	15.08	1023.75
ADLDNN[44]	13.05	238.00	17.33	1149.00	12.59	281.00	16.95	1371.00	14.98	759.75
HDNN[45]	13.02	245.00	15.24	1282.42	12.22	287.72	18.16	1547.42	14.66	840.64
ABiLSTM-CNN[37]	12.13	233.12	16.01	1230.00	11.96	242.00	18.10	1513.00	14.55	804.53
Bi-LSTM[13]	11.96	206.33	14.48	726.82	13.41	223.36	15.11	892.39	13.74	512.23
LSTM-MLSA[46]	**11.57**	252.86	14.02	899.18	12.13	370.39	17.21	1558.48	13.73	770.23
RMixMLP	12.07	267.40	**11.72**	**627.26**	**10.29**	**181.23**	**13.61**	**971.44**	**11.92**	**511.83**

Table 6: The RUL prediction results of the model that had entered the number of cycles of the engine.

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	10.8225	437.525

Table 7: Experimental parameters of the four working conditions

FD001		FD002		FD003		FD004	
blocks	*cycles*	blocks	*cycles*	blocks	*cycles*	blocks	*cycles*
1	0	1	1	1	0	2	0

4. Conclusion

The most important contribution of this paper is to propose an MLP-based RUL prediction method, which provides a new idea for RUL prediction. The RMixMLP performance we present is excellent, surpassing state-of-the-art CNNs, LSTMs, attention-based models and their variants. In terms of details, there are two different points for the previous work: 1. We did not screen the data of FD001 and FD003, and FD002 and FD004 are the same as the strategy; all of them are input orginal data directly; 2. We also provide a new idea for RUL prediction, using sequence prediction instead of single numerical prediction. Finally, the importance of time labels for model training is discussed, which is the number of cycles of the engine for the C-MPASS dataset, and a simple comparison proves that it does have a very important impact on RUL prediction. Regarding future work, for the experiment in this paper, although the performance of Score has reached the optimal, the effect is obviously not as good as RMSE, and the loss function can be designed to make it pay more attention to reducing the error of later prediction. In view of our proposed research on RMixMLP in the field of RUL prediction, we hope to further study the work of MLP in RUL prediction, including the study of self mechanism and the traditional methods commonly used for RUL prediction.

CRediT authorship contribution statement

Jinning Qin:Data curation; Formal analysis; Investigation; Methodology; Project administration; Resources; Software; Validation; Visualization; Roles/Writing - original draft; and Writing - review & editing.Yanyan Wang:Conceptualization, Methodology, Review, Supervision,Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research is supported by the National Natural Science Foundation of China (No. 62273204, No. 52275104) and the Shenzhen Science and Technology Innovation Commission Foundation, China (No. JCYJ20220530141210023).

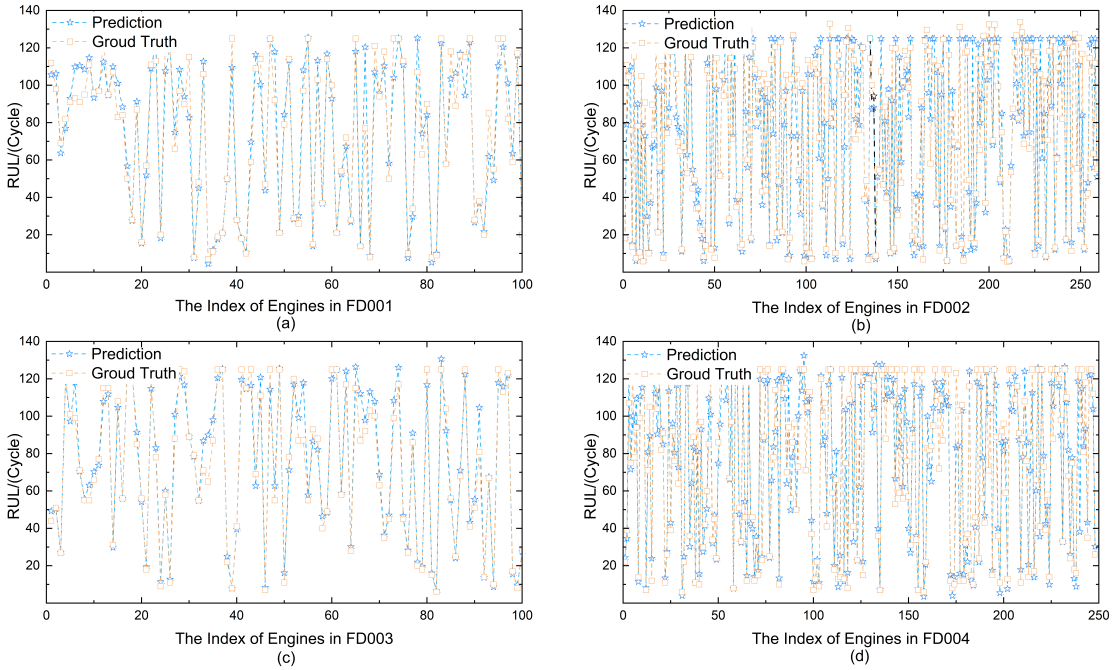

Figure 7: The prediction for the last recorded data point of different testing engine units in FD001-FD004. (a) Prediction for the 100 testing engine units in FD001. (b) Prediction for the 256 testing engine units in FD002. (c) Prediction for the 100 testing engine units in FD003. (d) Prediction for the 248 testing engine units in FD004.

Data Availability

Data openly available in a public repository. The dataset was provided by the Prognostics CoE at NASA Ames. We can visit `https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/` to get it, and the name is "Turbofan Engine Degradation Simulation Data Set." The dataset is in the text format and has been zipped including a readme file.

References

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

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

[3] Y. Qin, D. Chen, S. Xiang, C. Zhu, Gated dual attention unit neural networks for remaining useful life prediction of rolling bearings, IEEE Transactions on Industrial Informatics 17 (9) (2020) 6438–6447.

[4] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[5] G. Sateesh Babu, P. Zhao, 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] B. Wang, Y. Lei, N. Li, T. Yan, Deep separable convolutional network for remaining useful life prediction of machinery, Mechanical systems and signal processing 134 (2019) 106330.

[7] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE transactions on neural networks 5 (2) (1994) 157–166.

[8] S. Zheng, K. Ristovski, A. Farahat, C. Gupta, Long short-term memory network for remaining useful life estimation, in: 2017 IEEE international conference on prognostics and health management (ICPHM), IEEE, 2017, pp. 88–95.

[9] R. Zhao, R. Yan, J. Wang, K. Mao, Learning to monitor machine health with convolutional bi-directional lstm networks, Sensors 17 (2) (2017) 273.

[10] J. Zhang, P. Wang, R. Yan, R. X. Gao, Long short-term memory for machine remaining life prediction, Journal of manufacturing systems 48 (2018) 78–86.

[11] H. Miao, B. Li, C. Sun, J. Liu, Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks, IEEE Transactions on Industrial Informatics 15 (9) (2019) 5023–5032.

[12] C.-G. Huang, H.-Z. Huang, Y.-F. Li, A bidirectional lstm prognostics method under multiple operational conditions, IEEE Transactions on Industrial Electronics 66 (11) (2019) 8792–8802.

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

[14] Z. Kong, Y. Cui, Z. Xia, H. Lv, Convolution and long short-term memory hybrid deep neural networks for remaining useful life prognostics, Applied Sciences 9 (19) (2019) 4156.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

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

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

[18] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, W. Zhang, Informer: Beyond efficient transformer for long sequence time-series forecasting, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 35, 2021, pp. 11106–11115.

[19] H. Wu, J. Xu, J. Wang, M. Long, Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, Advances in

Neural Information Processing Systems 34 (2021) 22419–22430.
[20] Y. Nie, N. H. Nguyen, P. Sinthong, J. Kalagnanam, A time series is worth 64 words: Long-term forecasting with transformers, arXiv preprint arXiv:2211.14730 (2022).
[21] A. Zeng, M. Chen, L. Zhang, Q. Xu, Are transformers effective for time series forecasting?, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 37, 2023, pp. 11121–11128.
[22] A. Das, W. Kong, A. Leach, R. Sen, R. Yu, Long-term forecasting with tide: Time-series dense encoder, arXiv preprint arXiv:2304.08424 (2023).
[23] 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 34 (2021) 24261–24272.
[24] H. Liu, Z. Dai, D. So, Q. V. Le, Pay attention to mlps, Advances in Neural Information Processing Systems 34 (2021) 9204–9215.
[25] Y. Zhao, G. Wang, C. Tang, C. Luo, W. Zeng, Z.-J. Zha, A battle of network structures: An empirical study of cnn, transformer, and mlp, arXiv preprint arXiv:2108.13002 (2021).
[26] R. Pfeifer, Z. Schreter, F. Fogelman-Soulié, L. Steels, Connectionism in perspective, Elsevier, 1989.
[27] S. J. Nowlan, G. E. Hinton, Simplifying neural networks by soft weight-sharing, Neural computation 4 (4) (1992) 473–493.
[28] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), arXiv preprint arXiv:1606.08415 (2016).
[29] J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450 (2016).
[30] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, Ł. Kaiser, Universal transformers, arXiv preprint arXiv:1807.03819 (2018).
[31] S. Bai, J. Z. Kolter, V. Koltun, Deep equilibrium models, Advances in Neural Information Processing Systems 32 (2019).
[32] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, Albert: A lite bert for self-supervised learning of language representations, arXiv preprint arXiv:1909.11942 (2019).
[33] J. Zhang, H. Peng, K. Wu, M. Liu, B. Xiao, J. Fu, L. Yuan, Minivit: Compressing vision transformers with weight multiplexing, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 12145–12154.
[34] A. Saxena, K. Goebel, D. Simon, 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.
[35] Y. Wu, M. Yuan, S. Dong, L. Lin, Y. Liu, Remaining useful life estimation of engineered systems using vanilla lstm neural networks, Neurocomputing 275 (2018) 167–179.
[36] X. Li, Q. Ding, J.-Q. Sun, Remaining useful life estimation in prognostics using deep convolution neural networks, Reliability Engineering & System Safety 172 (2018) 1–11.
[37] J. W. Song, Y. I. Park, J.-J. Hong, S.-G. Kim, 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.
[38] H. Liu, Z. Liu, W. Jia, 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.
[39] H. Liu, Z. Liu, W. Jia, X. Lin, Remaining useful life prediction using a novel feature-attention-based end-to-end approach, IEEE Transactions on Industrial Informatics 17 (2) (2020) 1197–1207.
[40] K. Zhao, Z. Jia, F. Jia, H. Shao, Multi-scale integrated deep self-attention network for predicting remaining useful life of aero-engine, Engineering Applications of Artificial Intelligence 120 (2023) 105860.
[41] J. Zhang, X. Li, J. Tian, H. Luo, S. Yin, An integrated multi-head dual sparse self-attention network for remaining useful life prediction, Reliability Engineering & System Safety 233 (2023) 109096.
[42] L. Liu, X. Song, Z. Zhou, Aircraft engine remaining useful life estimation via a double attention-based data-driven architecture, Reliability Engineering & System Safety 221 (2022) 108330.
[43] J. Li, Y. Jia, M. Niu, W. Zhu, F. Meng, Remaining useful life prediction of turbofan engines using cnn-lstm-sam approach, IEEE Sensors Journal (2023).
[44] S. Xiang, Y. Qin, F. Liu, K. Gryllias, Automatic multi-differential deep learning and its application to machine remaining useful life prediction, Reliability Engineering & System Safety 223 (2022) 108531.
[45] A. Al-Dulaimi, S. Zabihi, A. Asif, A. Mohammadi, A multimodal and hybrid deep neural network model for remaining useful life estimation, Computers in industry 108 (2019) 186–196.
[46] J. Xia, Y. Feng, C. Lu, C. Fei, X. Xue, Lstm-based multi-layer self-attention method for remaining useful life estimation of mechanical systems, Engineering Failure Analysis 125 (2021) 105385.