Uncertainty Prediction of Remaining Useful Life Using Long Short-Term Memory Network Based on Bootstrap Method

Conference Paper · June 2018

DOI: 10.1109/ICPHM.2018.8448804

CITATIONS
99

READS
2,280

3 authors, including:

Linxun Zhang
Tsinghua University
53 PUBLICATIONS **767** CITATIONS

SEE PROFILE

Chongdang Liu
Tsinghua University
19 PUBLICATIONS **405** CITATIONS

SEE PROFILE

Uncertainty Prediction of Remaining Useful Life Using Long Short-Term Memory Network Based on Bootstrap Method

Yuan Liao, Linxuan Zhang, Chongdang Liu
Department of Automation
Tsinghua University
Beijing, China
thu_ly@126.com

***Abstract*—The accuracy and reliability of remaining useful life (RUL) prediction have a significant influence on Condition Based Maintenance (CBM). Deep learning methods, especially Long Short-Term Memory networks (LSTM), have been proven to be effective in automatically mining the hidden features of sensors data, and then obtain a better point estimation for RUL forecast. However, it is pointless when the sensor data is full of noise. Estimated RUL often varies widely mainly due to the model parameters and the noise. In this case, it is inadequate to calculate the point estimation of RUL merely. For uncertainty prediction of RUL, this requires calculating not only the deterministic prediction value but also the estimation of the prediction interval. The reliability of deterministic RUL prediction, incorporated into the uncertainty prediction, is enhanced. In this paper, an LSTM-FNN (Feedforward Neural Network) based on bootstrap method (LSTMBS) for uncertainty prediction of RUL estimation is presented. The proposed method is tested on the benchmarking aircraft turbofan engines datasets (C-MAPSS). Different feature augmentation methods are applied to different sub-datasets of C-MAPSS due to the different operating conditions and fault modes. Results show that the proposed approach not only performs a competitive point estimation of RUL compared with other approaches but also obtains additional information on prediction interval of RUL to determine the appropriate maintenance time. In order to address this problem more universally, the unified LSTMBS model is constructed considering not discriminating the categories of new test trajectories in the practical system.**

Keywords-uncertainty prediction; RUL; LSTM; bootstrap; prediction interval

I. INTRODUCTION

Traditional maintenance based on regular strategy was time-consuming, costly and inefficient [1]. In order to overcome the shortcomings, Condition Based Maintenance (CBM) is gradually developed, which makes a timely and appropriate maintenance strategy through monitoring the current health condition of the system, thus reducing the overall downtime and cutting the costs [2]. RUL prediction is the key enablers of CBM, the research target of which is to evaluate the remaining useful life of system before system failure.

In recent years, many scholars have studied RUL estimation methods, mainly including model-based, data-driven and a hybrid of these two methods. Most model-based methods are dependent on the physical mechanism of equipment degradation which can be described by specific mathematical expressions. The prediction is relatively precise but this method is only used when the data is small and the physical model is already known [3]. However, it is very difficult to find a specific physical model or mechanism to describe the complicated system, which leads to the failure of model-based approaches. Under these circumstances, the data-driven approaches come into being for the RUL forecast without any knowledge of the physical model and failure mechanism [4], e.g. Support Vector Regression [5], particle filtering [6], Extreme Learning Machines [7] and so on. Hybrid prognostic methods combine the strengths of model-based and data-driven methods, which are also gaining in popularity.

Compared with traditional data-driven algorithms, deep learning technologies have obvious advantages faced with the huge sensor data [8,9]. There are some approaches including Deep Belief Networks (DBN) [10,11], Convolutional Neural Network (CNN) [8], Long Short-Term Memory networks (LSTM) [9,12] for RUL forecast, among which LSTM is good at dealing with time series problems of sensor data. Therefore, LSTM is used in this paper. In general, deep learning methods have the powerful ability to automatically extract latent fault features and degradation information when the prior expert knowledge is limited, which makes it a great way to handle the huge industrial sensor data with characteristics of low-value density, i.e. large amount of data with limited information included. Moreover, deep learning methods are no longer just for special failure problems, which can perform RUL prediction as relatively universal approaches.

However, these deep learning methods just achieve point estimation of RUL forecast, which is pointless with the sensor data full of noise. When the RUL prediction of engines on the test set is evaluated again under the same situations, it is seen that the estimated point values for RUL forecast are different each time and sometimes vary widely, which is mainly due to the uncertainty of the RUL prediction brought by the model parameters and noise. So only calculating the point estimation of RUL is not enough under multiple fault modes, multiple operation conditions and noise circumstances. For uncertainty

978-1-5090-0382-2/16/$31.00 ©2018 IEEE

978-1-5386-1164-7/18/$31.00 ©2018 IEEE

prediction of RUL, not only the deterministic predictive value but also its reliability, i.e. the estimation of prediction interval, needs to be calculated.

The purpose of constructing the prediction interval is to quantify the uncertainty of the point estimation [13]. The methods of constructing prediction interval of neural network include the delta [14], mean-variance estimation (MVE) [15], Bayesian [16], bootstrap [17,18,19], local uncertainty estimation model (LUEM) [13] and lower upper bound estimation (LUBE) [20,21]. The bootstrap method is simple and it doesn't depend on the distribution of data, which provides reliable uncertainty estimation considering the different initialization of network parameters [17]. LUBE method is said to generate higher quality prediction intervals than delta, Bayesian and bootstrap methods [20]. However, the cost function of LUBE method is discontinuous and highly nonlinear, resulting in the need for simulated annealing algorithm to train the network. Here it is not suitable for LSTM network training because it has ten thousands of parameters when there are multiple LSTM memory blocks, which make it difficult to converge for a short time.

LSTM-FNN (Feedforward Neural Network) has strong ability to perform the point evaluation of RUL, but it lacks the capability of uncertainty quantification and representation. However, the bootstrap method has the capability to perform the uncertainty prediction and representation of RUL. By combining their strengths, an LSTM-FNN based on bootstrap method (LSTMBS) for uncertainty prediction of RUL estimation is proposed in this paper. This approach has two advantages: 1) The reliability of deterministic RUL prediction is enhanced due to incorporating the uncertainty prediction. 2) For choosing an appropriate maintenance time, it will obtain more effective strategies based on the prediction interval of RUL than that based on the point evaluation of RUL.

The rest of the paper is organized as follows: Section II describes the approach of LSTM-FNN based on bootstrap method (LSTMBS) for uncertainty prediction of RUL. In section III, the LSTMBS models are applied on the C-MAPSS dataset to demonstrate the effectiveness of the approach. Section IV summarizes the paper with some conclusions.

II. METHODOLOGY

An LSTM-FNN based on bootstrap method is constructed to predict uncertainty of RUL at the current step before system failure. And then, LSTM, LSTM-FNN, and LSTM-FNN based on bootstrap method are explained.

A. Long Short-Term Memory networks（LSTM）

LSTM model is composed of repeating LSTM memory blocks, each of which contains three gates (input gate, forget gate and output gate) to decide what information was renewed, discarded and outputted from the memory cell. Gates mechanism ensures that memory block state won't be directly renewed like Recurrent Neural Networks (RNN) so that LSTM model is capable to learn long-time relevant connections. Given an observed sequence of sensor data, each of which is an m-dimensional vector $x_t \in \mathbb{R}^{m\times 1}$ at time t. Let denote k as

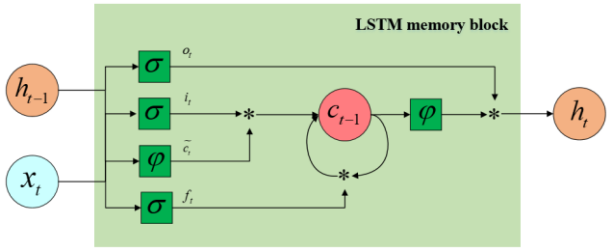

Figure 1. LSTM memory block

size of hidden output $h_{t-1} \in \mathbb{R}^{k\times 1}$ and state $c_{t-1} \in \mathbb{R}^{k\times 1}$ of previous memory block at time $t-1$. For every LSTM memory block in Figure 1, its update formula is as follows [9,12,22]:

$$i_t = \sigma(W_i x_t + R_i h_{t-1} + b_i), \quad (1)$$

$$f_t = \sigma\left(W_f x_t + R_f h_{t-1} + b_f\right), \quad (2)$$

$$o_t = \sigma(W_o x_t + R_o h_{t-1} + b_o), \quad (3)$$

$$\tilde{c_t} = \varphi(W_c x_t + R_c h_{t-1} + b_c), \quad (4)$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c_t}, \quad (5)$$

$$h_t = o_t * \varphi(c_t), \quad (6)$$

where $W_j \in \mathbb{R}^{k\times m}$, $R_j \in \mathbb{R}^{k\times k}$, $b_j \in \mathbb{R}^{k\times 1}$ are input weights of gates, recurrent weights and bias (j represents i, f, o, c). i_t, f_t, o_t, $\tilde{c_t}$ are input gate, forget gate, output gate and candidate values of memory block state, respectively. σ and φ are nonlinear activation functions. The new state c_t of the current memory block is decided by two parts: one is forgetting something from previous old state c_{t-1} through forget gate, the other is adding new information from the scaled candidate values $\tilde{c_t}$. Then, the output gate has an effect on final hidden output h_t.

B. LSTM-FNN

With the continuous use of the system, its performance will gradually degenerate until the threshold of failure. The degradation evolution of the system can be indirectly reflected from the collected sensor data. An LSTM-FNN model is constructed to handle time series of sensor data since LSTM is capable to learn latent fault features from long-time context, while FNN is good at mapping latent features from LSTM into regression outcomes, i.e. the observed target RUL of the current time before system failure.

C. LSTM-FNN based on bootstrap method for RUL uncertainty estimation

According to historical run-to-failure sensor data, the degradation evolution of the system is modeled to evaluate the RUL of the current time. Because the degradation state of the machine is intrinsically random and the sensor data is full of noise, it is difficult to reflect the degradation state of the machine using a time step of the sensor data. In order to contain more degradation information, the sliding window technique [2,9] is used. Assuming that there are M machines, each contains N_i run-to-failure cycles collected by multiple sensors, i.e. $\left(z_1^i, z_2^i, \cdots, z_{N_i}^i\right)$, $i = 1,2,\cdots, M$, where $z_t^i = \{x_{t1}^i, x_{t2}^i, \cdots, x_{tm}^i\}$, $t = 1,2,\cdots, N_i$, is an m-dimensional vector of

sensor data at time t. Let denote L as the length of the sliding window. For every machine i, $(z_1^i, z_2^i, \cdots, z_L^i)$ belongs to the first sliding window, i.e. the first training sample X_1 and $(z_2^i, z_3^i, \cdots, z_{L+1}^i)$ belongs to the second sliding window, i.e. the second training sample X_2 and so on. According to this rule, N_i continuous sensor data can generate $N_i - L + 1$ training samples for every machine i, i.e. X_1, X_2,⋯, X_{N_i-L+1}, $i = 1,2,\cdots, M$. An LSTM-FNN model $\psi(\cdot)$ was constructed during samples training phase, and (z_{t-L+1}^i, $z_{t-L+2}^i, \cdots, z_{t-1}^i, z_t^i$) is as its input and the observed target RUL_t is as its output, i.e.

$$\widehat{rul}_t = \psi(z_{t-L+1}^i, z_{t-L+2}^i, \cdots, z_{t-1}^i, z_t^i), \tag{7}$$

where $\widehat{rul}_t$ denotes the RUL prediction value at time t.

The bootstrap method provides a good solution to the uncertainty prediction, as it can theoretically reduce bias between the prediction value and regression mean [19]. As is shown in Figure 2, the basic idea of bootstrap is to resample B times with replacement from source data (source data means training data). Then the model $\psi_b(\ b = 1,2,\cdots,B)$ is obtained by using the resampling data $S_b(\ b = 1,2,\cdots,B)$ each time. Eventually, the ensemble of multiple models from different parameter spaces will generate the mean and variance of the prediction value. The formulas are as follows [17, 18,19]:

$$\bar{\psi} = \frac{1}{B}\sum_{b=1}^{B}\psi_b(S_b), \tag{8}$$

$$\hat{\sigma}^2 = \frac{1}{B-1}\sum_{b=1}^{B}(\psi_b(S_b) - \bar{\psi})^2, \tag{9}$$

where $\bar{\psi}$ and $\hat{\sigma}^2$ are the prediction mean and variance by using the bootstrap technique.

Therefore, the prediction interval for RUL estimation can be obtained by [18,19]:

$$\bar{\psi}_{\widehat{rul}_t} + t_{df}^{1-\frac{\alpha}{2}}\sqrt{\sigma_{\widehat{rul}_t}^2 + \sigma_{\varepsilon_t}^2}, \tag{10}$$

where $(1-\alpha)100\%$ is the prescribed confidence level. $t_{df}^{1-\frac{\alpha}{2}}$ is the $1-\frac{\alpha}{2}$ quantile of t-test function and its degree of free df is set to B.

As we can see in (10), the key to constructing prediction intervals is to find the prediction mean $\bar{\psi}_{\widehat{rul}_t}$, the variance $\sigma_{\widehat{rul}_t}^2$ and the noise $\sigma_{\varepsilon_t}^2$ on the target data (target data means test data). Figure 3 illustrates the procedure of LSTM-FNN based on bootstrap method (LSTMBS) for uncertainty prediction of RUL, which is performed as follows [19]:

Step 1. The continuous run cycles of machines containing fault features are mapped into the matrix with m features. Then the sensor data is processed by the sliding window technique, and the processed samples are obtained as source data T.

Step 2. The regression mean $\bar{\psi}_{T_1}$ and variance $\sigma_{T_1}^2$ are evaluated by using (8) and (9), where B regression LSTM-FNN models are constructed on the source data T_1 from 70% of source data T at random. In this part, the training samples X_{T_1} are as the input of LSTM-FNN and the observed target RUL_{T_1} will be mapped as output.

$$\bar{\psi} = \frac{1}{B}\sum_{b=1}^{B}\psi_b(S_b)$$

$$\hat{\sigma}^2 = \frac{1}{B-1}\sum_{b=1}^{B}(\psi_b(S_b) - \bar{\psi})^2$$

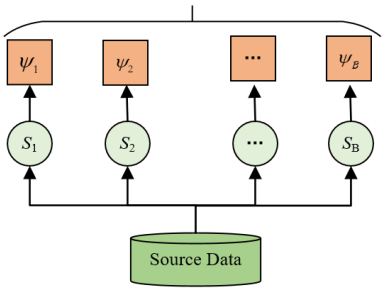

Figure 2. Bootstrap method

Step 3. The noise $\sigma_{\varepsilon_t}^2$ is obtained on the remaining source data T_2. Firstly, regression mean $\bar{\psi}_{T_2}$ and variance $\hat{\sigma}_{T_2}^2$ of source data T_2 are calculated using trained B regression LSTM-RNN models in step 2. Then, there will be [19]

$$R_{T_2}^2 = max\left(\left(RUL_{T_2} - \bar{\psi}_{T_2}\right)^2 - \hat{\sigma}_{T_2}^2, 0\right), \tag{11}$$

where RUL_{T_2} denotes observed target RUL on the source data T_2. Finally, a regression task called noise-forecast model [19] using LSTM-FNN is modeled between the training samples X_{T_2} and noise term $R_{T_2}^2$ on the source data T_2.

Step 4: The prediction intervals will be calculated from above $B+1$ models. The regression mean $\bar{\psi}_{\widehat{rul}_t}$ and the variance $\sigma_{\widehat{rul}_t}^2$ on the target data can be obtained by an ensemble of B LSTM-FNN models, while the noise item $\sigma_{\varepsilon_t}^2$ is calculated using another noise-forecast model. Eventually, prediction intervals on the target data can be obtained using (10).

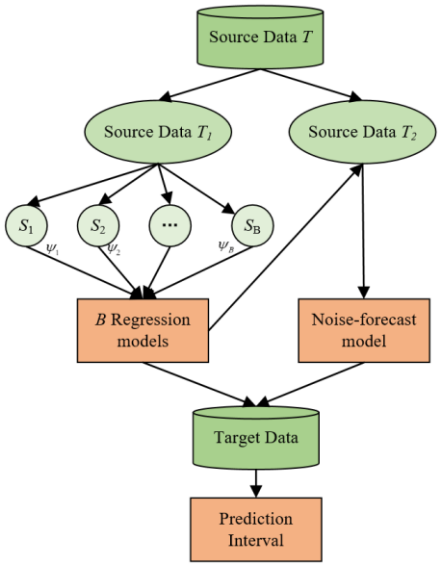

Figure 3. The procedure of LSTM-FNN based on bootstrap method (LSTMBS) for uncertainty prediction of RUL

III. EXPERIMENTS AND DISCUSSIONS

In the experiments, the constructed LSTM-FNN models based on bootstrap method are applied on the C-MAPSS sub-datasets to implement the performance evaluation of point forecast and prediction intervals of RUL.

A. Data description

Aircraft turbofan engines datasets [23] from NASA generated by Commercial Modular Aero-Propulsion System Simulation(C-MAPSS) are described in Table I. It contains 4 sub-datasets: single operation condition and single fault mode issue, multiple operation conditions and single fault mode issue, single operation condition and hybrid fault modes issue, multiple operation conditions and hybrid fault modes issue [12]. Each dataset contains a training set and a test set, where each row is a snapshot of data from the machine running cycle. There are 26 columns: the 1st and 2nd columns represent engine ID and running cycle, the next 3 columns are operating conditions setting indicating that the machine is running in the current operation condition, the last 21 columns are sensor readings corresponding to degeneration information of machines.

The trajectories of the training sets simulate the entire life cycles of engines. For each trajectory, the initial states are different, but all are considered as healthy. And the last cycle of the trajectory is declared as the system fault. The machine running terminates for some time prior to the system fault on the test sets, in which the target is to predict the remaining useful life of each engine. In order to validate the forecast results, the true RUL values are provided on the test set [23].

B. Performance evaluation

This paper is intended to evaluate the performance of uncertainty prediction of RUL, which contain two parts: one is for point evaluation, i.e. scoring function and Root Mean Square Error (RMSE), and the other is for prediction interval, i.e. coverage probability of prediction intervals (PICP) and normalized mean prediction interval width (NMPIW).

(1) Scoring function [8,9]:

$$S = \begin{cases} \sum_{i=1}^{n}(e^{-d_i/13} - 1), & d_i < 0 \\ \sum_{i=1}^{n}(e^{d_i/10} - 1), & d_i \geq 0. \end{cases} \tag{12}$$

n denotes the number of test samples, d_i is equal to the estimated RUL minus true RUL for every test sample i. As is seen in (12), scoring function let late prediction bear greater penalty than early prediction even if the error d_i is same, as late prediction may lead to more serious consequences.

(2) RMSE [8,9]:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} {d_i}^2}. \tag{13}$$

RMSE gives the equal penalty for early prediction and late prediction if the error d_i is same.

(3) PICP [20,21]:

$$PICP = \frac{1}{n}\sum_{i=1}^{n} rul_i, \tag{14}$$

$$rul_i = \begin{cases} 1, L_i \leq RUL_i \leq H_i \\ 0, \quad otherwise. \end{cases} \tag{15}$$

TABLE I. DETAILS DESCRIPTION OF C-MAPSS DATASET

Data Set	FD001	FD002	FD003	FD004
Train trajectories	100	260	100	249
Train Maximum lifespan (Cycles)	362	378	525	543
Train Minimum lifespan (Cycles)	128	128	145	128
Test trajectories	100	259	100	248
Maximum length of test trajectories(Cycles)	303	367	475	486
Minimum length of test trajectories(Cycles)	31	21	38	19
Operating conditions	1	6	1	6
Fault modes	1	1	2	2

L_i and H_i are the lower and upper bounds of prediction interval for every test sample i. PICP measures whether the observed target RUL lies within the prediction interval with a probability (1-α).

(4) NMPIW [20,21]:

$$MPIW = \frac{1}{n}\sum_{i=1}^{n}(H_i - L_i), \tag{16}$$

$$NMPIW = MPIW/(max(RUL_i) - min(RUL_i)). \tag{17}$$

The mean prediction interval width (MPIW) measures mean width of prediction intervals and NMPIW is the normalization of MPIW, indicating MPIW is as a percentage of a range of expected RUL. Generally speaking, PICP and NMPIW are conflicting, and a higher PICP will result in a wider NMPIW. In fact, we should first ensure that the PICP exceeds desired coverage probability. Then on this basis, the narrower NMPIW is expected.

C. Data preprocessing

(1) Data Normalization: The different ranges of sensor readings have an effect on the accuracy and convergence speed of the model, so the normalization of data is needed prior to training and testing, which ensures that all features under different operation and fault modes have an equal contribution. The normalization method in this paper is used by zero-mean normalization:

$$X^* = \frac{X - \mu}{\sigma}, \tag{18}$$

where X^* is the normalization of raw features X. μ and σ are the mean and variance of X.

(2) Operating Conditions: Operating setting values on FD002 and FD004 have been clustered into 6 clusters in [8,24], which can be replaced by six different discrete values. Then we apply one-hot method to encode these six operating conditions.

(3) Length of the sliding window: It has proved that the longer the sliding window become, the more information it contains, which leads to lower scores and RMSE [9]. However, as is seen in table I, the minimum length of test trajectories limits the length of the sliding window, i.e. 31,21,38,19, respectively, as every engine on the test sets should be predicted.

(4) The number of test samples: The estimated RUL value of each engine on the test sets is dependent on the last running cycles as these contains deterioration information. Specifically, the number of the last running cycles is equal to the length of

the sliding window. So the number of test samples is equal to the number of test trajectories (Table I), i.e. 100, 259, 100, 248 respectively.

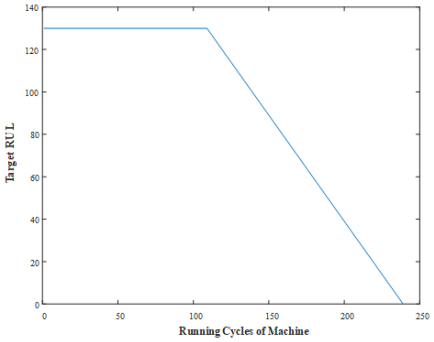

Figure 4. RUL Target function on the C-MAPSS dataset (The initial RUL is set as 130)

(5) RUL target function: Determining the expected output value for each input sample is a challenging job. However, a sensible solution is to simply assign the actual time before system failure as the desired output. It has been pointed out that piece-wise linear function as RUL target value will obtain better performance in [24]. As is shown in Figure 4, the first half is restricted to a constant, indicating that the initial degradation is ignored, and then it begins to degenerate linearly at a certain point. Here, we set the constant in the first half as 130 [24].

D. Feature augmentation and Network structure

In [12], forward difference method between current sensor readings and previous sensor readings based on the same operating conditions was proposed to generate new features, which has been proved to be more effective for RUL prediction [12]. However, for the single operating condition, this method doesn't work as there are not multiple operating conditions. This paper proposes utilizing forward difference method between current sensor readings and previous sensor readings based on the running cycles applied to the single operating condition and based on the hybrid of running cycles and the same operating conditions applied to the multiple operating conditions. And then the effect of different feature augmentation methods under different network structures on performance evaluation of RUL will be discussed.

In order to find the best network structure under different training features and make the research results more convincing, 10-fold cross-validation was performed randomly on 70% of training samples of FD001~FD004 to evaluate the performance of models based on different network structures and different feature augmentation methods (the reason for the 70% is that 70% of source data is used to train *B* bootstrap models in Figure 3). As is shown in Table III-VI, the mean scores and RMSE on the validation set are displayed under different training features and different network structures. For the sake of layout, the abbreviation symbols are used in Table II.

In Table III and Table IV, the newly added features of forward difference between current sensor readings and previous sensor readings based on the running cycles used on

TABLE II. THE MEANING OF ABBREVIATION SYMBOLS

abbreviation symbols	meaning
L(x)	x nodes in the first LSTM layer
F(x)	x nodes in the second FNN layer
S	21 sensor readings
O	3 operating settings
H	One-hot encoding for 6 discrete operating conditions
T	Forward difference method between current sensor readings and previous sensor readings based on the running cycles
C	Forward difference method between current sensor readings and previous sensor readings based on the same operating conditions
S(x)	Mean score of 10-fold cross-validation(Unit: $\times 10^x$)
R(x)	RMSE of 10-fold cross-validation(Unit: $\times 10^x$)

single operating condition datasets (FD001 and FD003) make it have slower scores and RMSE of 10-fold cross-validation under all network structures. In table V and table VI, for multiple operating conditions datasets, the FD002 performs slightly better on "S+H+C" features than that on "S+H+T+C" features, while the FD004 performs best on "S+H+T+C" features under most of the network structures. Here machine running cycles and operating settings are regarded as background information. When there is a single operating condition, the sensor readings may mainly depend on machine running cycles, so the FD001 and FD003 perform better on "S+O+T" features. When there are multiple operating conditions, the sensor readings may mainly depend on different operating conditions [12]. Moreover, dynamic difference method based on the same operating conditions makes the degradation information apparent [12]. So the newly added "C" features can bring lower mean scores and RMSE than the newly added "T" features. As for the "S+H+C" and "S+H+T+C" features where the model performs best, this may be related to the fault modes.

TABLE III. MEAN SCORES AND RMSE OF 10-FOLD CROSS-VALIDATION UNDER DIFFERENT TRAINING FEATURES AND DIFFERENT NETWORK STRUCTURES ON FD001 TRAINING SET (THE OPTIMAL VALUE IS IN BOLD)

FD001	S+O		S+O+T	
	S(3)	R(1)	S(3)	R(1)
L(32)F(4)	5.683	1.423	**3.603**	**1.228**
L(32)F(8)	4.630	1.415	**3.483**	**1.202**
L(64)F(4)	6.242	1.452	**2.581**	**1.052**
L(64)F(8)	4.868	1.386	**3.034**	**1.183**
L(128)F(8)	5.631	1.433	**2.955**	**1.144**
L(128)F(16)	4.899	1.409	**2.706**	**1.103**
L(256)F(8)	4.407	1.406	**2.539**	**1.047**
L(256)F(16)	4.630	1.487	**2.775**	**1.095**

TABLE IV. MEAN SCORES AND RMSE OF 10-FOLD CROSS-VALIDATION UNDER DIFFERENT TRAINING FEATURES AND DIFFERENT NETWORK STRUCTURES ON FD003 TRAINING SET (THE OPTIMAL VALUE IS IN BOLD)

FD003	S+O		S+O+T	
	S(3)	R(1)	S(3)	R(1)
L(32)F(4)	8.125	1.377	**3.522**	**1.080**
L(32)F(8)	6.160	1.325	**3.111**	**1.064**
L(64)F(4)	6.987	1.408	**4.881**	**1.158**
L(64)F(8)	4.947	1.230	**2.080**	**0.881**
L(128)F(8)	4.530	1.231	**2.032**	**0.844**
L(128)F(16)	4.802	1.237	**3.079**	**0.981**
L(256)F(8)	5.288	1.309	**2.918**	**0.957**
L(256)F(16)	5.677	1.321	**2.384**	**0.889**

TABLE V. MEAN SCORES AND RMSE OF 10-FOLD CROSS-VALIDATION UNDER DIFFERENT TRAINING FEATURES AND DIFFERENT NETWORK STRUCTURES ON FD002 TRAINING SET (THE OPTIMAL VALUE IS IN BOLD)

FD002	S+H		S+H+T		S+H+C		S+H+T+C	
	S(4)	R(1)	S(4)	R(1)	S(4)	R(1)	S(4)	R(1)
L(32)F(4)	6.837	2.135	5.283	2.149	1.408	1.336	**1.101**	**1.282**
L(32)F(8)	7.297	2.156	7.126	2.126	**1.370**	1.358	1.425	**1.353**
L(64)F(4)	9.233	2.240	6.410	2.091	**0.671**	**1.034**	1.026	1.216
L(64)F(8)	7.141	2.136	5.961	2.058	**0.605**	**1.021**	0.816	1.081
L(128)F(8)	7.600	2.139	7.168	2.106	0.475	0.916	**0.381**	**0.816**
L(128)F(16)	5.550	2.066	7.468	2.098	**0.537**	0.894	0.557	**0.851**
L(256)F(8)	7.534	2.132	5.961	2.118	**0.264**	**0.653**	0.278	0.674
L(256)F(16)	6.954	2.108	6.372	2.088	**0.207**	**0.601**	0.370	0.682

TABLE VI. MEAN SCORES AND RMSE OF 10-FOLD CROSS-VALIDATION UNDER DIFFERENT TRAINING FEATURES AND DIFFERENT NETWORK STRUCTURES ON FD004 TRAINING SET (THE OPTIMAL VALUE IS IN BOLD)

FD004	S+H		S+H+T		S+H+C		S+H+T+C	
	S(5)	R(1)	S(5)	R(1)	S(4)	R(1)	S(4)	R(1)
L(32)F(4)	1.316	2.249	1.351	2.178	**3.034**	**1.458**	3.232	1.502
L(32)F(8)	1.249	2.116	1.376	2.140	3.654	1.376	**3.305**	**1.371**
L(64)F(4)	1.214	2.086	1.319	2.103	2.339	1.235	**1.800**	**1.199**
L(64)F(8)	1.037	2.132	1.272	2.087	2.528	1.255	**1.167**	**1.034**
L(128)F(8)	1.323	2.119	1.177	2.128	**0.926**	**0.905**	1.469	0.975
L(128)F(16)	1.000	2.066	1.381	2.102	1.743	1.056	**1.034**	**0.917**
L(256)F(8)	1.012	2.157	1.019	2.082	1.843	0.928	**1.150**	**0.767**
L(256)F(16)	1.147	2.104	1.164	2.093	1.121	0.887	**0.814**	**0.852**

After selecting features on FD001~FD003, and then the best network structure is decided on each dataset. Comparison of mean scores and comparison of RMSE under different network structures on each dataset are presented in Figure 5 and Figure 6. The boxplot can reflect the minimum, median, maximum and two quartiles without any assumption of data distribution, through which the best network structure can be decided, mainly based on the smallest median and the smallest range between maximum and minimum value. The best network structures related to feature augmentation on FD001~FD004 are shown in Table VII.

The LSTMBS models on FD001~FD004 are constructed using information in Table VII, during which the bias of LSTM's forget gate is set as 1 [25]. Eventually, the optimizer, objective function, and epoch are set as Adam, mean-square error (MSE) and 10, respectively. At the same time, dropout and early stopping are used to prevent overfitting. The constructed

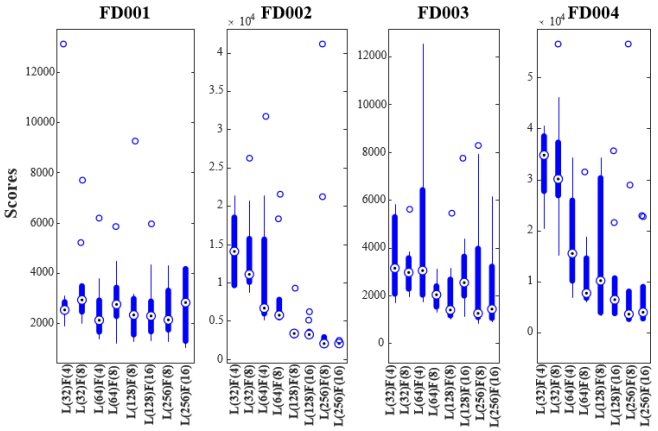

Figure 5. Comparison of mean scores under different network structures on each dataset

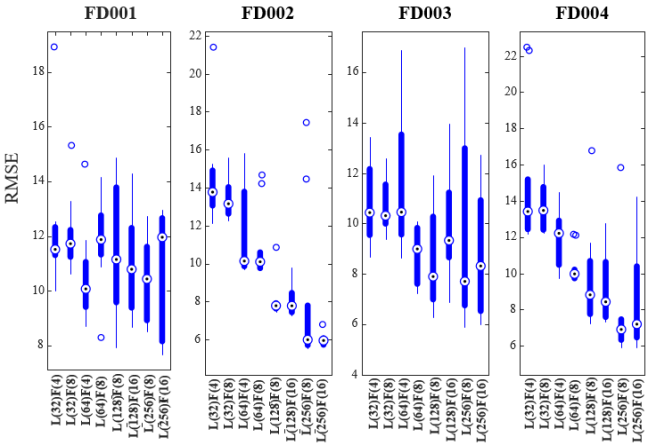

Figure 6. Comparison of RMSE under different network structures on each dataset

TABLE VII. THE BEST NETWORK STRUCTURES RELATED TO FEATURE AUGMENTATION ON FD001~FD004

Data Set	FD001	FD002	FD003	FD004
feature augmentation	S+O+T	S+H+C	S+O+T	S+H+T+C
Network structure	L(64)F(4)	L(256)F(16)	L(128)F(8)	L(256)F(8)

models on FD001~FD004 are applied to their test sets, where the estimated RUL and their prediction intervals will be obtained. All experiments are repeated three times and the best results are recorded. Scores and RMSE of point estimation of RUL on the C-MAPSS dataset are shown in table VIII and table IX, which is compared with other methods [8,9]. Reference [9] has demonstrated on the CMAPSS dataset the scores and RMSE of point estimation of deep LSTM method outperform that of RUL forecast methods like MLP, SVR, RVR, CNN. The proposed LSTMBS method in this paper is slightly better than deep LSTM method in terms of RMSE and scores of RUL estimation except for score and RMSE indexes on FD002 and score index on FD001. The RMSE of LSTMBS on FD001 is lower than that of deep LSTM, but its score is relatively higher, which indicates that the predicted RUL using LSTMBS method is slightly greater than the true RUL, thus coming with a greater penalty. The reason for the poor performance on FD002 may be related to the selection of the length of the sliding window. If the length of the sliding window is greater than the minimum length of test sets, that leads to the elimination of engines on the test sets. The shortest trajectory on the FD002 test set is 21, so the sliding window is set as 21 on FD002.

In addition, PICP and NMPIW of prediction interval of RUL are shown in Table X. The prescribed confidence level is 95%, i.e. $\alpha = 0.05$. Due to large number of test trajectories on the FD001~FD004 test sets, the prediction intervals of the first 50 engines on their behalf on the C-MAPSS sub-datasets are shown in Figure 7, which is seen that when desired PICP reaches 95% the more complex degradation information the sub-datasets contains, i.e. the more the operating conditions and fault modes increase, the relatively wider the prediction intervals become.

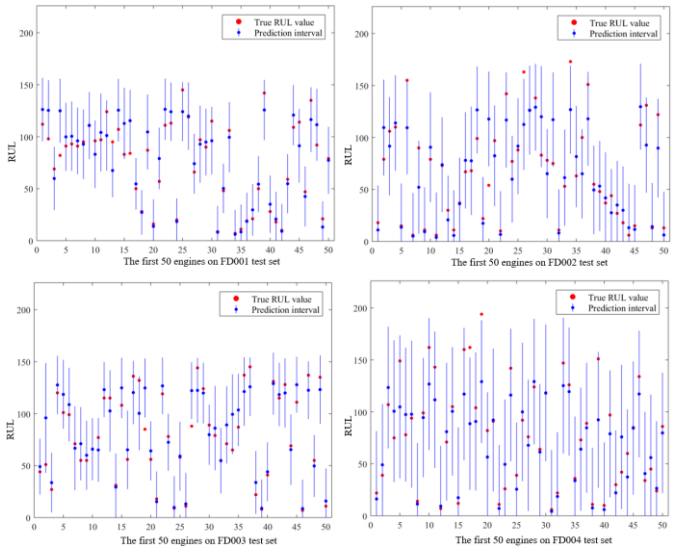

Figure 7. The prediction intervals of the first 50 engines on the C-MAPSS sub-datasets

E. Unified LSTMBS model on the C-MAPSS dataset

It may seem less than universal to construct different LSTM-FNN models for each sub-dataset. Moreover, it is difficult to discriminate the categories of new running trajectories of the machine in the practical system [12]. So it is necessary to build a model, called unified LSTMBS model, utilizing the whole C-MAPSS dataset to estimate RUL forecast and prediction intervals. Here the four data sets are stacked together as the whole training set. As is shown in Figure 8, the operating settings on the whole training set, similar to operating settings on FD002 and FD004, are still drawn into six clusters in a three-dimensional coordinate system. So "S+H+C" and "S+H+T+C" features are applied to unified LSTMBS model (they are marked as Unified_C and Unified_T+C, respectively), and network architectures are decided using 10-fold cross-validation randomly on 70% of the whole training set. Eventually, the relatively optimal network structure is 800 nodes in the first LSTM layer and 300 nodes in the second FNN layer. The length of the sliding window of unified LSTMBS models is 19 for the minimum length of test trajectories on the C-MAPSS dataset is 19, which ensures that RUL prediction on the whole test sets is performed. The performance evaluation of the unified LSTMBS models including scores, RMSE, PICP, NMPIW is displayed in Table VIII, Table IX, and Table X.

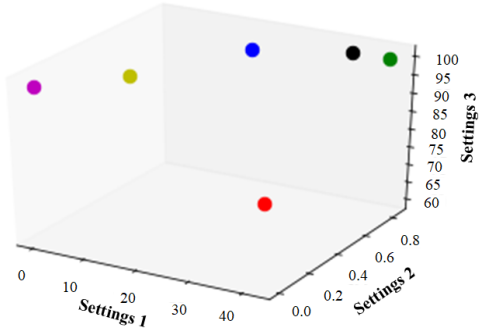

Figure 8. The operating settings on the whole training set are drawn into six clusters in a three-dimensional coordinate system.

For mean scores and RMSE of point estimation of RUL, compared with LSTMBS model utilizing the separate C-MAPSS sub-datasets, these two unified LSTMBS models have a poorer performance on the FD001 test set and a better performance on the FD004 test set, but both exist on the FD002 and FD003 test sets. The FD004 is most complicated because of involving multiple conditions and hybrid fault modes, including all kinds of situations on FD001~FD003. As is seen in Table VI, the performance using the "S+H+C" and "S+H+T+C" features is better than that using "S+H+T" and "S+H" features. Moreover, the number of the whole training samples increases due to the stacked sub-datasets. So these lead to a better performance of point evaluation on the FD004 test set using the unified LSTMBS models. Oppositely, the FD001 is simplest because of involving the single condition and the single fault mode, which indicates "S+O+T" features, not "S+H+C" or "S+H+T+C" features, are more suitable in Table III. Therefore, although the number of the whole training samples increases, the unsuitable features make the performance of point estimation on the FD001 test set slightly worse. As for the complexity, FD002 and FD003 are located between FD001 and FD004, the performance of point estimation is also a trade-off between two situations above. In general, the performance of RUL forecast of the Unified_T+C model is slightly better than that of the Unified_C model on the C-MAPSS dataset mainly due to huge scores difference on FD002 in Table VIII.

For PICP and NMPIW of the prediction interval of RUL, the coverage probabilities of the two unified LSTMBS models on FD002 and FD004 are lower than those on FD001 and FD003 under the same parameters, the reason for which is that "S+H+C" and "S+H+T+C" features of two unified LSTMBS models are more suitable for FD002 (in Table V) and FD004 (in Table VI) and not suitable for FD001 (in Table III) and FD003 (in Table IV). As is known in (11), this results in a larger $R_{S_2}^2$ and noise item $\sigma_{\varepsilon_t}^2$ obtained by the noise-forecast model on FD001 and FD003 test sets. According to (10), the wider prediction intervals of RUL forecast result in higher coverage probabilities on FD001 and FD003.

TABLE VIII. COMPARISON OF SCORES ON THE C-MAPSS DATASET

Data Set	FD001	FD002	FD003	FD004
CNN[8]	1.287×10^{3}	1.357×10^{4}	1.596×10^{3}	7.886×10^{3}
Deep LSTM[9]	3.380×10^{2}	4.450×10^{3}	8.520×10^{2}	5.550×10^{3}
LSTMBS	4.811×10^{2}	7.982×10^{3}	4.934×10^{2}	5.200×10^{3}
Unified_C	7.902×10^{2}	1.044×10^{4}	4.736×10^{2}	4.436×10^{3}
Unified_T+C	7.472×10^{2}	7.656×10^{3}	4.967×10^{2}	4.805×10^{3}

TABLE IX. COMPARISON OF RMSE ON THE C-MAPSS DATASET

Data Set	FD001	FD002	FD003	FD004
CNN[8]	1.845×10^{1}	3.029×10^{1}	1.982×10^{1}	2.916×10^{1}
Deep LSTM[9]	1.614×10^{1}	2.449×10^{1}	1.618×10^{1}	2.817×10^{1}
LSTMBS	1.489×10^{1}	2.686×10^{1}	1.511×10^{1}	2.711×10^{1}
Unified_C	1.814×10^{1}	2.801×10^{1}	1.605×10^{1}	2.661×10^{1}
Unified_T+C	1.789×10^{1}	2.758×10^{1}	1.622×10^{1}	2.673×10^{1}

TABLE X. COMPARISON OF PICP AND NMPIW ON THE C-MAPSS DATASET (THE PRESCRIBED CONFIDENCE LEVEL IS 95%)

FD004	FD001		FD002		FD003		FD004	
	PICP (%)	NMPIW(%)	PICP (%)	NMPIW(%)	PICP (%)	NMPIW(%)	PICP (%)	NMPIW(%)
LSTMBS	96.0	37.7	89.2	47.2	96.0	45.9	95.2	65.4
Unified_C	95.0	63.3	87.6	41.8	99.0	61.0	83.5	41.0
Unified_T+C	95.0	61.1	84.9	42.5	98.0	58.3	82.7	42.0

IV. CONCLUSIONS

LSTM, which is good at discovering hidden features of time series, has a powerful ability for RUL forecast due to its gates mechanism. The bootstrap method is very simple to obtain the prediction interval of RUL without any hypothesis of sensor data distribution. The paper proposes an LSTM-FNN based on bootstrap method (LSTMBS) for uncertainty prediction of RUL estimation by combining their advantages. Experiments on the benchmarking aircraft turbofan engines datasets (C-MAPSS) show that the LSTMBS models utilizing the separate C-MAPSS sub-datasets perform a very competitive point estimation of RUL forecast compared with other approaches, which owes to an appropriate feature augmentation method according to the characteristics of the four sub-datasets. And more importantly, the LSTMBS method also provides additional information on prediction interval of RUL, which is a benefit for the next maintenance strategy.

In order to make the model more universal, the unified LSTMBS models are built based on "S+H+C" and "S+H+T+C" features. The performance evaluation of the unified LSTMBS model is related to the complexity involved on the C-MAPSS sub-datasets. In general, the unified LSTMBS model constructed by using "S+H+T+C" features is a relatively universal choice without discriminating the categories of new test trajectories in the practical system.

In the future, the performance evaluation of point estimation and prediction interval is seen as a whole optimization target, concerning the scores, RMSE, PICP and NMPIW, not just MSE, which will produce a better prediction model and maintenance strategy.

REFERENCES

[1] M. A. Zaidan, A. R. Mills, and R. F. Harrison, "Bayesian framework for aerospace gas turbine engine prognostics," in 2013 IEEE Aerospace Conference, 2013, pp. 1-8.

[2] P. Lim, C. K. Goh, and K. C. Tan, "A time window neural network based framework for Remaining Useful Life estimation," in 2016 International Joint Conference on Neural Networks (IJCNN), 2016, pp. 1746-1753.

[3] N. H. Kim, J. H. Choi, and D. An, Prognostics and Health Management of Engineering Systems. Springer International Publishing, 2017.

[4] L. Yongxiang, S. Jianming, W. Gong, and L. Xiaodong, "A data-driven prognostics approach for RUL based on principle component and instance learning," in 2016 IEEE International Conference on Prognostics and Health Management (ICPHM), 2016, pp. 1-7.

[5] R. Khelif, B. Chebel-Morello, S. Malinowski, E. Laajili, F. Fnaiech, and N. Zerhouni, "Direct Remaining Useful Life Estimation Based on Support Vector Regression," IEEE Transactions on Industrial Electronics, vol. 64, no. 3, pp. 2276-2285, 2017.

[6] A. Guha and A. Patra, "Particle filtering based estimation of remaining useful life of lithium-ion batteries employing power fading data," in 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), 2017, pp. 193-198.

[7] Z. X. Yang and P. B. Zhang, "ELM Meets RAE-ELM: A hybrid intelligent model for multiple fault diagnosis and remaining useful life prediction of rotating machinery," in 2016 International Joint Conference on Neural Networks (IJCNN), 2016, pp. 2321-2328.

[8] G. S. Babu, P. Zhao, and X. L. Li, "Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life," in International Conference on Database Systems for Advanced Applications, 2016, pp. 214-228.

[9] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long Short-Term Memory Network for Remaining Useful Life estimation," in 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), 2017, pp. 88-95.

[10] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics," IEEE Trans Neural Netw Learn Syst, vol. 28, no. 10, pp. 2306-2318, Oct 2017.

[11] G. Zhao, G. Zhang, Y. Liu, B. Zhang, and C. Hu, "Lithium-ion battery remaining useful life prediction with Deep Belief Network and Relevance Vector Machine," in 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), 2017, pp. 7-13.

[12] Y. Wu, M. Yuan, S. Dong, L. Lin, and Y. Liu, "Remaining Useful Life Estimation of Engineered Systems using vanilla LSTM Neural Networks," Neurocomputing, 2017.

[13] D. L. Shrestha and D. P. Solomatine, "Machine learning approaches for estimation of prediction interval for the model output," Neural Netw, vol. 19, no. 2, pp. 225-35, Mar 2006.

[14] J. T. Genehwang and A. Adamding, "Prediction Intervals for Artificial Neural Networks," Publications of the American Statistical Association, vol. 92, no. 438, pp. 748-757, 1997.

[15] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," in IEEE International Conference on Neural Networks, 1994. IEEE World Congress on Computational Intelligence, 1994, pp. 55-60 vol.1.

[16] D. J. C. Mackay, "The Evidence Framework applied to Classification Networks," Neural Computation, vol. 4, no. 5, pp. 720-736, 1992.

[17] E. Zio, "A study of the bootstrap method for estimating the accuracy of artificial neural networks in predicting nuclear transient processes," IEEE Transactions on Nuclear Science, vol. 53, no. 3, pp. 1460-1478, 2006.

[18] P. Baraldi, F. Mangili, E. Zio, and E. Zio, "Ensemble of bootstrapped models for the prediction of the remaining useful life of a creeping turbine blade," in 2012 IEEE Conference on Prognostics and Health Management, 2012, pp. 1-8.

[19] K. Li, R. Wang, H. Lei, T. Zhang, Y. Liu, and X. Zheng, "Interval prediction of solar power using an Improved Bootstrap method," Solar Energy, vol. 159, pp. 97-112, 2018.

[20] A. Khosravi, S. Nahavandi, D. Creighton, and A. F. Atiya, "Lower upper bound estimation method for construction of neural network-based prediction intervals," IEEE Trans Neural Netw, vol. 22, no. 3, pp. 337-46, Mar 2011.

[21] H. Zhang, J. Zhou, L. Ye, X. Zeng, and Y. Chen, "Lower Upper Bound Estimation Method Considering Symmetry for Construction of Prediction Intervals in Flood Forecasting," Water Resources Management, vol. 29, no. 15, pp. 5505-5519, 2015.

[22] M. Tan, C. D. Santos, B. Xiang, and B. Zhou, "LSTM-based Deep Learning Models for Non-factoid Answer Selection," Computer Science, 2016.

[23] E. Ramasso and A. Saxena, "Performance Benchmarking and Analysis of Prognostic Methods for CMAPSS Datasets," International Journal of Prognostics & Health Management, vol. 5, no. 2, pp. 1-15, 2016.

[24] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in International Conference on Prognostics and Health Management, 2008, pp. 1-6.

[25] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in International Conference on International Conference on Machine Learning, 2015, pp. 2342-2350.