Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8814658, 12 pages
https://doi.org/10.1155/2020/8814658

*Research Article*

# Developing Deep Survival Model for Remaining Useful Life Estimation Based on Convolutional and Long Short-Term Memory Neural Networks

**Chia-Hua Chu, Chia-Jung Lee, and Hsiang-Yuan Yeh**

*School of Big Data Management, Soochow University, Taipei City 111, Taiwan*

Correspondence should be addressed to Hsiang-Yuan Yeh; hyyeh.richard@gmail.com

Received 5 August 2020; Revised 16 October 2020; Accepted 20 November 2020; Published 7 December 2020

Academic Editor: Chao-Yang Lee

The application of mechanical equipment in manufacturing is becoming more and more complicated with technology development and adoption. In order to keep the high reliability and stability of the production line, reducing the downtime to repair and the frequency of routine maintenance is necessary. Since machine and components' degradations are inevitable, accurately estimating the remaining useful life of them is crucial. We propose an integrated deep learning approach with convolutional neural networks and long short-term memory networks to learn the latent features and estimate remaining useful life value with deep survival model based on the discrete Weibull distribution. We conduct the turbofan engine degradation simulation dataset from Commercial Modular Aero-Propulsion System Simulation dataset provided by NASA to validate our approach. The improved results have proven that our proposed model can capture the degradation trend of a fault and has superior performance under complex conditions compared with existing state-of-the-art methods. Our study provides an efficient feature extraction scheme and offers a promising prediction approach to make better maintenance strategies.

## 1. Introduction

With the advance of Internet of Things (IoT) technology and its applications to the industrial environments, data analytics methods can be applied to the mechanical equipment health and performance. In fact, any machine breakdown may lead to a huge loss on production yield. However, sometimes even a professional and experienced engineer cannot find where the fault is and also cannot figure out the main cause of the malfunction. In this case, the company has no choice but to suspend the production line for the thorough examination, which is certainly one of the disastrous situations for the company's business. In order to keep the high reliability and stability of the production line, reducing the downtime for fixing malfunction and the frequency of routine maintenance is necessary. The earliest maintenance technique takes place only when breakdown happens, which is called breakdown maintenance or run-to-failure maintenance. Later, companies tend to have time-based preventive maintenance. It means that engineers perform a preventive maintenance periodically, in spite of the status of the machine even though it is in healthy status. Preventive maintenance will cause a lot of cost and become a major expense of many companies. In order to save the cost, another maintenance strategy such as condition-based maintenance (CBM) is figured out to solve the situation. CBM proposes to reduce the number of unnecessary regular preventive maintenance and improves the reliability of machine by implementing maintenance only when there is an evidence that an exception occurs [1, 2]. Because it is effective in saving the cost to the companies, CBM has been more and more popular. The prognostics and health management (PHM) is one of the major tasks in CBM. The core of PHM is the estimation of the remaining useful life (RUL) of machines based on the collected information of the historical and ongoing degradation trends [2–4]. The flowchart of PHM contains five major processes as shown in Figure 1, including data acquisition, signal processing, diagnostic, prognostic, and maintenance decision. Data acquisition is the first process of PHM which is composed of sensors, data transmission, and data storage devices.

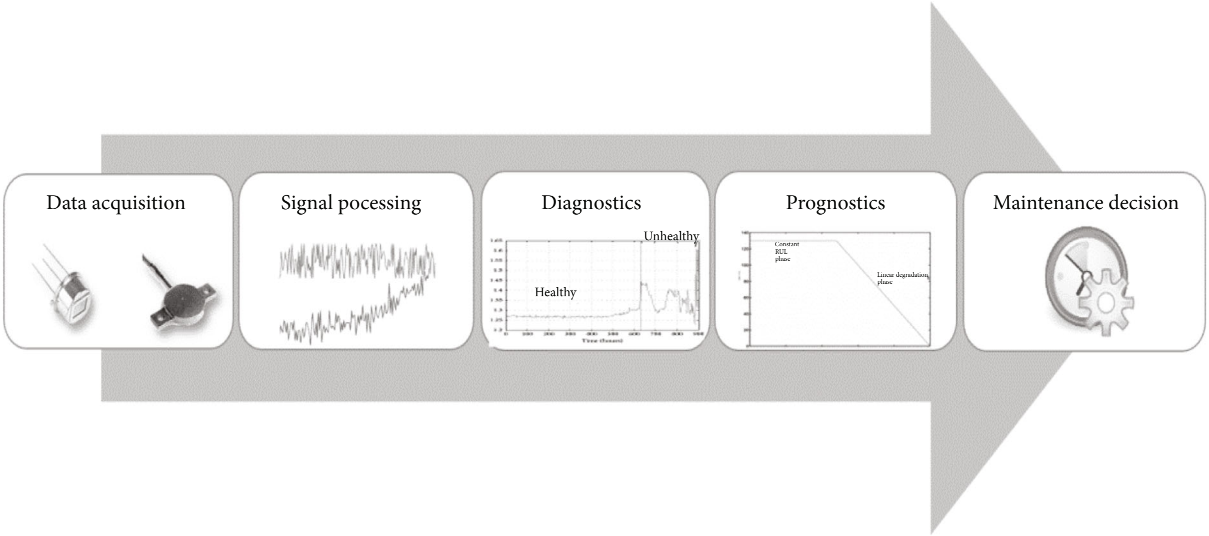


Figure 1: Flowchart of PHM processes.

Different kinds of sensors are used to collect different types of data, which are related to the health condition and able to reflect the degradation process of the monitored machine. Signal processing's task is to extract useful information from the data acquired from the previous step. Diagnostic is a process to divide the machine's whole lifetime into different health status. Prognostic is aimed at estimating the time length from current time to when it requires repair or replacement, which is also the definition of RUL. Maintenance decision is the final process in PHM and is used for analyzing the outputs from diagnostics and prognostics. If we can predict RUL, we are able to propose a strategy about scheduling maintenance, avoiding unplanned downtime, and optimizing operating efficiency and frequency to save the most cost. However, knowing that the machine and components' degradation are inevitable, the challenge of proper scheduling grows with the complexity of machines. One of the key problems in predictive maintenance is the prediction of the equipment failures should be early enough so that the proper maintenance could be scheduled before it happens. Therefore, predictive maintenance is based on the continual monitoring of the equipment in order to determine the right maintenance actions at the right times. The organization of this paper is as follows. Related works on RUL prediction is introduced in Section 2. Our proposed deep learning approach with a survival model is described in Section 3. In Section 4, the experimental results and evaluations are compared with existing state-of-the-art methods and show the effectiveness of the proposed approach. We conclude the contributions and limitations of our approach in Section 5.

## 2. Related Works

Generally, the methods of the RUL estimation problems can be categorized into model-based, data-driven, and hybrid approaches. Model-based prediction applies a physical model of the system for degradation [5]. This approach can be further divided into microlevel models [6] and macrolevel models [7]. Microlevel models need to consider the assumptions and simplifications in uncertain environments. A macrolevel model is constructed under different operational conditions of the physical system which includes the relationships among input variables, state variables, and system outputs. However, model-based methods require a large amount of prior knowledge. The physical models are difficult to build under many components which limit the effectiveness of the methods. The data-driven approaches detect the state of the system via large number of sensor monitoring, which are more suitable for the complex system and do not require a comprehensive understanding of the physical understanding [8]. Currently, high-dimensional data collected in real-life PHM applications makes it difficult to directly discover the trends for the prognostic algorithm. There are various operational conditions and health states in the same type of the system which may cause different degradation processes as well as unit-to-unit variability (UtUV) [9]. This situation brings difficulty to RUL estimation. Javed et al. contributed a data-driven prognostics approach based on extreme learning machine (ELM), which is able to model degrading states without assuming a homogeneous pattern [10]. Liu and Chen combined indirect health indicator (HI) and the smooth monotonic signals from sensory data and multiple Gaussian process regression (GPR) models to achieve the RUL prediction [11]. Previous works constructed a model based on Box–Cox transformation (BCT) and Monte Carlo (MC) simulation to predict the battery RUL [12]. Khelif et al. developed a procedure to estimate the RUL directly from sensor values using the support vector regression method which models the direct relationship among sensor values or health indicators [13]. However, the dramatic changes and variations of the indicators make a well-trained prediction model that may not be suitable for the practical applications. The traditional feature extraction method is hard to get high-level representations from measurements, and the poor prognostic performance may be achieved. Therefore, capturing hidden patterns from high-dimensional data efficiently is a necessary procedure in the feature extraction procedure [14]. Hybrid approaches are to complement the superiority of model-based and data-driven methods [15]. However, it still remains a challenging work to utilize the advantages and to avoid the disadvantages of both approaches.

Recently, data-driven prediction methods focused on the flexible deep learning models to capture useful information from high-dimensional data efficiently. Zhang et al. employed a multiobjective evolutionary algorithm with traditional deep belief networks (DBN) for RUL estimation in prognostics [16]. Sequence learning methods such as the hidden Markov model (HMM) were applied to capture time series information from the sensory data [17–19]. In the model, each state can only depend on the immediately previous state and the hidden states must be drawn from the discrete space. However, modeling long time dependencies may lead to high computational complexity while the set of the hidden stages grows larger. Recurrent neural network (RNN) can model time sequence data as well, and some work applied it to estimate RUL [20]. But RNN has its limitation to capture the long-term time dependencies, the gradients propagated over many hidden layers tend to either vanish or explode [21]. Long short-term memory (LSTM) network is a significant branch of RNN, can learn long-term dependencies, and avoid gradient disappearance and explosion in long sequence training. Previous studies have shown that LSTM networks can expose hidden patterns from the sequential sensor data with multiple operating conditions, fault, and degradation models [22, 23]. Some new approaches based on LSTM, such as bidirectional long short-term memory (BLSTM) network [24] and vanilla LSTM [25], were proposed. Wang et al. proposed a transfer learning algorithm based on BLSTM networks, which can be first trained on the datasets and fine-tuned the model with a different but related dataset [26]. Recent works also enhanced LSTM networks with attention mechanism and generative adversarial network (GAN) to improve the interpretability and accuracy of the deep networks [27–29]. The convolutional neural network (CNN) architecture has been proven to be effective for extracting abstract information on multichannel sequential sensor data [30–32]. Although LSTM networks enable us to build and capture long-term time dependencies, its feature extraction capabilities are marginally lower than CNN [33]. CNN can extract the spatial feature while LSTM can learn temporal features. Therefore, it is better to learn temporal features from the slow inherently long-term degradation process by combing those two structures. Recent paper proposed a deep neural network structure using both LSTM and CNN which can be combined in a serial or parallel manner to improve the accuracy of the RUL prediction of the equipment [34, 35].

Most of the prior works have focused on the RUL prediction problem which present one numeric RUL value only. However, it is nearly impossible to find an approach that can predict RUL exactly the same as the real one. If the variance is large, it is hard to have confidence on the predicted result. The RUL prediction problem is also similar to the survival analysis which is commonly used to model time-to-death events in the healthcare domain [34]. For example, the model predicts the failure will happen in 8 days with 80% probability is much better than predict 10 days until the failure. Martinsson proposed the Weibull time-to-event recurrent neural network, which is a simple framework for time series prediction of the time to the next event applicable [36]. Aggarwal et al. used the Weibull distribution assumption on the time-to-failure event with a linear hazard rate corresponding to the linear degradation model that most of the literature makes [34].

Due to the complicated environments in real-life PHM applications, monitoring sequential sensor data is subjected to the operating conditions and fault modes for the prognostic algorithm. The existing data-driven methods often rely on the sensor measurements as a whole data for training that may cause less effect and bias. To cope with this issue, there is a great potential to improve the RUL estimation by extracting latent patterns from partial information that is a necessary procedure to capture useful information from high-dimensional sequential sensor data. It is also valuable to estimate how much time is left of the equipment and the probability of a failure together. Therefore, we integrate CNN and LSTM with a deep survival model to enhance the ability of feature extraction and capture the degradation trend of a fault with a reasonable prediction horizon.

## 3. Materials and Methods

The overall workflow of our approach is shown in Figure 2. Different sensors may have different physical meanings and numerical ranges. In order to eliminate the influence of ranges of value, we first apply a min-max normalization method as feature scaling to adjust the range of sensor values between 0 and 1. Second, the training data and test data are prepared with a sliding time window (TW) to generate the sequential samples. Third, we use 1D temporal convolutions to learn hidden patterns in those sequential samples without any interference from the other sensor values. Forth, the extracted temporal patterns from the 1D convolution would be fed into LSTM networks to learn the long short-term time dependencies. Fifth, we use both regression and survival analyses with the discrete Weibull distribution to estimate the RUL and failure probability in the training phase. Finally, we can predict the RUL and the probability of a failure with test data in the trained model in the testing phase.

*3.1. Data Preparation with Time Window.* The input sequential sensor data from an engine are assumed to be a matrix $X(n) = [x^1, x^2, \cdots, x^{L_s}]$ with $k$ sensors (measurements), where $n$ denotes engine ID and $L_s$ denotes the last observed cycle or the cycle that fault occurs in Figure 3. A sliding time window strategy is adopted to generate the temporal sequence data except sampled at a single time step which may conduct better feature extraction efficiently. Taking $X(n)$ as an input and extracting sequential $X_i(n)$ as Equation (1) by sliding the fixed time window (TW) with length $m$, this can be presented in Figure 3:

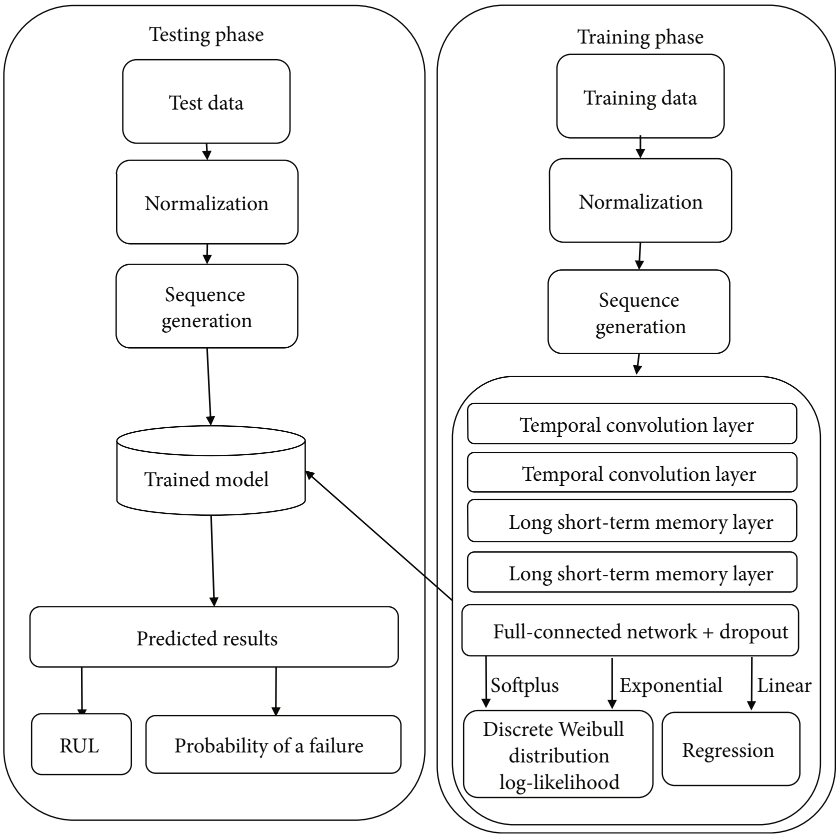


Figure 2: The workflow of our approach.

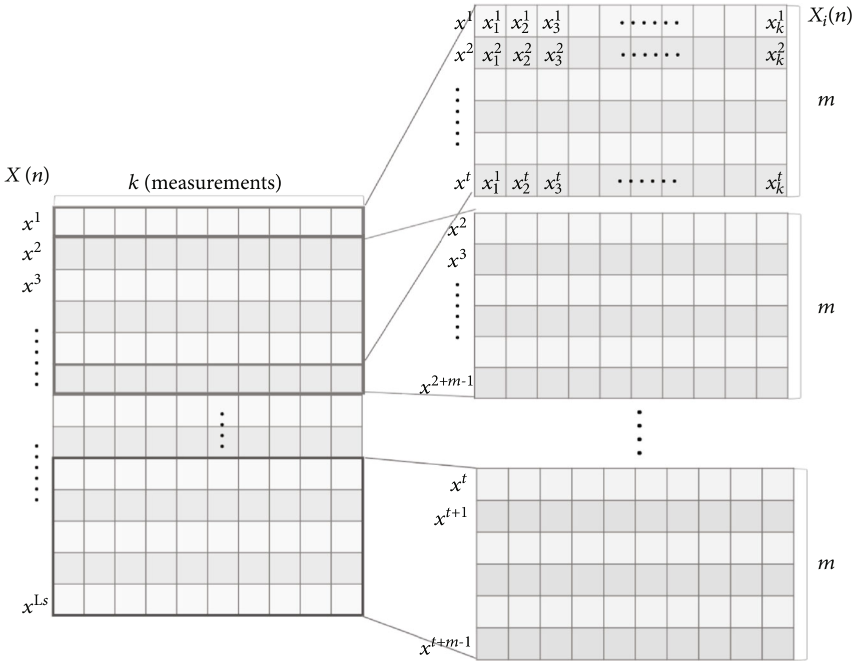

$X(n)$
$k$ (measurements)
$x^1$
$x^2$
$x^3$
$x^{Ls}$
$X_i(n)$
$x^1$ $x_1^1$ $x_2^1$ $x_3^1$ ······ $x_k^1$
$x^2$ $x_1^2$ $x_2^2$ $x_3^2$ ······ $x_k^2$
$x^t$ $x_1^t$ $x_2^t$ $x_3^t$ ······ $x_k^t$
$m$
$x^2$
$x^3$
$x^{2+m-1}$
$m$
$x^t$
$x^{t+1}$
$x^{t+m-1}$
$m$

Figure 3: Sliding time windows.

$$X_i(n) = x^{t:t+m-1} = \left[x^t, x^{t+1}, x^{t+2}, \cdots, x^{t+m-1}\right], \qquad (1)$$

where $x^t = [x_1^t, x_2^t, \cdots, x_k^t]$ represents a $k$-dimensional array of measurements at time (cycle) $t$ through a TW size of $m$. Therefore, the size of each array extracted each time by TW is $m \times k$ (TW $\times$ numbers of sensors), and the number of arrays is $L_s$-$m$ (lifetime cycles-TW).

*3.2. Temporal Convolutional Layer.* To deal with sequential information more effectively, a CNN layer can be used to extract abstract and high-level features before LSTM layers. The temporal convolutional layer consists of three layers, starting with 1D-convolution, 1D-max-pooling, and followed by activation function. 1D convolution represents a filtering window length and moves towards the depth across the data. We consider that the input data matrix size is $m \times k$ array in $X_i(n)$, and there are $d$ kinds of the feature detectors with $w$ kernel size. So, each feature detector has to move $(m - w) + 1$ times and generates the set of feature detector regions $X_{dw}$. Then, we add 1D-convolution weight kernel $\vec{w}$ and bias $b$ so the convolution operation is shown as

$$C_i = f\left(X_{dw} \bigotimes \vec{\mathrm{w}} + b\right). \qquad (2)$$

Where $\bigotimes$ denotes the Hadamard product (element-wise product) and $f$ represents the nonlinear activation function, *ReLu*. Accordingly, the output feature maps of a 1D-convolution layer will be the size of $((m - w) + 1) * d$. Since we have multiple temporal convolution layers, we let $C^{(l-1)}$ and $C^{(l)}$ be the input and output of the $l$th layer, respectively. We denote the $j$th feature map of layer $l$ as $C_j^{(l)}$ which can be computed by

$$C_j^{(l)} = f\left(\sum_i C_i^{(l-1)} \bigotimes \vec{w}_{i,j}^{(l)} + b_j^{(l)}\right). \qquad (3)$$

*3.3. Long Short-Term Memory.* LSTM cell state relies on three control gates: input gate, forget gate, and output gate. Input gate controls the extent to which incoming data flows into the cell. Forget gate judges which data from the foregoing cell state to be taken in consideration or be ignored. Output gate decides whether the value in the cell is used to compute the output. In the LTSM layer, it performs multiple internal equations as described below as

$$f_l = \sigma\left(W_f \bullet \left[h_{l-1}, C_j^{(l)}\right] + b_f\right), \qquad (4)$$

$$i_l = \sigma\left(W_i \bullet \left[h_{l-1}, C_j^{(l)}\right] + b_i\right), \qquad (5)$$

$$\widetilde{Z_l} = \tanh\left(W_c \bullet \left[h_{l-1}, C_j^{(l)}\right] + b_c\right), \qquad (6)$$

$$Z_l = f_l \otimes Z_{l-1} + i_l \otimes \widetilde{Z_l}, \qquad (7)$$

$$o_l = \sigma\left(W_o \bullet \left[h_{l-1}, C_j^{(l)}\right] + b_o\right), \qquad (8)$$

$$h_l = o_l \otimes \tanh\,(Z_l), \qquad (9)$$

where $f_l$ denotes the forget gate, and its main function is to neglect the data from the previous LSTM cell state. $\sigma(\,)$ is an activation function *sigmoid*. $W_f$ is the weight matrix of the forgot gate, $h_{l-1}$ denotes the short-term state of previous layer in the LSTM cell, the feature map of layer $C_j^{(l)}$ is the input of LSTM cell, and $b_f$ is the bias vector of forget gate. The input gate is composed of two parts, $i_l$ is a vector that determines which data in the short-term state $h_{l-1}$ is used to update the new cell state. After being selected by $i_l$, $\widetilde{Z_l}$ will be added to the long-term cell state and tan $h$ is an activation function. $W_i$ and $W_c$ denote the weight matrixes and both $b_i$ and $b_c$ denote the bias vectors of the input gate. Then, the forget gate and the input gate will be used to update the long-term state of the previous LSTM cell. The output gate $o_l$ is also composed of variables where $W_o$ denotes the weight matrixes and $b_o$ denote the bias vectors.

Finally, the LSTM layer connects to the fully connected layer for estimating the output target RUL value. Dropout technique is a regularization technique which randomly drops the hidden nodes with a given probability during training. It forms neural networks with different architectures in parallel and then takes an ensemble of them to prevent coadaptation. In order to alleviate the overfitting problems, the dropout is used between the final LSTM layer and the first fully connected layer [37].

*3.4. Loss Function.* Survival analysis is also called time-to-event analysis that is a subfield of statistics for analyzing the expected time duration until one or more events happen [38]. This approach calculates the probability of the subject to 'survive' the number of days or cycles [39]. One of the most commonly used distributions in the survival analysis is the discrete Weibull distribution which can be presented as Equation (10). The time-to-failure is modeled with a random variable $T$ giving the probability of failure time between $t$ and $t + 1$. The probability mass function (PMF) of a discrete random variable is characterized by two parameters: alpha ($\alpha$) is a scale parameter that denotes that the expected value and mode of the distribution are positioned in time, while the parameter beta ($\beta$) is an indicator of the shape as well as the variance of our prediction.

$$\Pr\,(t \le T \le t + 1) = e^{-(t/\alpha)^{\beta}} - e^{-((t+1)/\alpha)^{\beta}}. \qquad (10)$$

We have to utilize a special log-likelihood as loss-function, called the discrete Weibull distribution log-likelihood [34]. The discrete Weibull distribution log-likelihood punishes the model for predicting high probabilities of failures occurring during the lifetime without failures for all samples. In addition, the discrete Weibull distribution log-likelihood will reward distributions that give high probabilities of the event happening at that point in time for samples where the failure time is known. The discrete Weibull distribution log-likelihood can be defined as follows:

$$d(y) = \Lambda(y+1) - \Lambda(y) = \left(\frac{y+1}{\alpha}\right)^{\beta} - \left(\frac{y}{\alpha}\right)^{\beta}, \tag{11}$$

$$\begin{aligned} \log\left(\mathfrak{L}_d\right) &= \bar{u} \bullet \log\left(e^{d(y)} - 1\right) - \Lambda(y+1) = \left(\frac{y+1}{\alpha}\right)^{\beta} - \left(\frac{y}{\alpha}\right)^{\beta} \\ &= \alpha^{-\beta}\left((y+1)^{\beta} - y^{\beta}\right), \end{aligned} \tag{12}$$

where $y$ denotes the time-to-event value (cycle) and $u$ denotes either a 0 or 1 machinery health event indicator. Since it is an average value, we express it by $\bar{u}$. In each training step, we apply two types of loss functions. We apply linear activation function for the output value in regression analysis approach and use mean squared errors (MSE), which measures the average squared difference between the estimated RUL values and the true RUL value as loss function. On the other hand, the activation layer of the discrete Weibull distribution is a costumed function that is set to use an exponential function for alpha and *softplus* function for beta [34]. We use the discrete Weibull distribution log-likelihood as loss function in failure probability in the survival analysis approach and estimate Weibull parameters be the outputs of the layer giving us a distribution of the training data.

*3.5. Performance Evaluation.* For the sake of comparability with other existing state-of-the-art methods, the same metrics are used to evaluate the performance. While using the model to predict the RUL with regression and survival analysis approaches, there is an error between the predicted RUL and the actual RUL called root mean square error (RMSE) as Equation (13). The late prediction might delay the schedule of the proper maintenance operations, and too early prediction might not be harmful but still wastes more maintenance resources. Since the key aspect is to avoid the failure, early prediction is generally more desirable than late prediction. The scoring function as Equation (14) penalizes late predictions more than early predictions to evaluate the model. In addition, we also calculate MAE as Equation (15) and $R^2$ ($R$ squared) coefficient of determination which is a statistical measure of how well the predictions approximate the real data points for more comparison between the two analysis approaches. The higher $R^2$ value means more information about the fit of a model can be explained.

$$\text{RMSE} = \sqrt{\frac{1}{N} \bullet \sum_{i=1}^{N} \left(\text{RUL}_{\text{True}} - \text{RUL}_{\text{Predicted}}\right)^2}, \tag{13}$$

$$\text{score} = \sum_{j=1}^{N} \begin{cases} e^{-(\text{RUL}_{\text{True}} - \text{RUL}_{\text{Predicted}})/13} - 1, \text{if } \left(\text{RUL}_{\text{True}} - \text{RUL}_{\text{Predicted}}\right) < 0, \\ e^{-(\text{RUL}_{\text{True}} - \text{RUL}_{\text{Predicted}})/10} - 1, \text{if } \left(\text{RUL}_{\text{True}} - \text{RUL}_{\text{Predicted}}\right) \geq 0, \end{cases} \tag{14}$$

$$\text{MAE} = \frac{1}{N} \bullet \sum_{i=1}^{N} \left|\text{RUL}_{\text{True}} - \text{RUL}_{\text{Predicted}}\right|. \tag{15}$$

## 4. Results and Discussion

*4.1. Datasets.* We adopt the dataset from the Commercial Modular Aero-Propulsion System Simulation (CMAPSS) dataset provided by NASA [40]. The CMAPSS dataset is well known in the PHM community and has been widely used for evaluating predictive performance. The dataset consists of simulated degradation data of turbofan engines generated with the thermodynamical simulation model. This dataset includes four subdatasets with different operating conditions and fault modes. A description of four subdatasets is shown in Table 1. A fleet of engines in the FD001 dataset suffered the high-pressure compressor failure under a single operating condition. For FD002, the sample suffered the high-pressure compressor failure under six operating conditions. In the FD003 situation, the sample suffered high-pressure compressor and fan degradations under a single operating condition, while in FD004, the sample suffered under six operating conditions. There are 26 columns in each subdataset, including engine unit number, number of cycle, three operational settings, and 21 sensor measurements. Each subdataset is divided into a training set, a testing set, and actual RUL corresponding to the test data. The different subsets have different numbers of engines with various operational cycles. As the operating time increases, the engines start to degrade until a failure occurs in the training datasets, while the degradation in the test datasets ends prior to the occurrence of a failure. The purpose of the proposed algorithm is to predict the RULs of the test datasets, and the true RUL targets of the test datasets are also provided to calculate the prognostic performance.

*4.2. Performance of Regression and Survival Analyses.* We randomly select 80 percent of the samples from the training set to train the models and the remaining 20 percent of the instances is used as the validation set to select the parameters in the training phase. We predict the RUL from the test data using a trained model and denote the performance comparing with state-of-the-art methods in Table 2. In Table 2, the italicized numbers denote the top 3 ranked results among those methods. The first three are regression-related algorithms including the multilayer perception (MLP) [41], support vector regression (SVR) [42], and relevance vector regression (RVR) [43], and the others are deep neural network related. The deep learning methods show better performance than the traditional machine learning methods. Our proposed approach achieves the lowest RMSE values and scoring function based either on regression analysis or on the discrete Weibull distribution for the FD002 and has superior performance for scoring function in FD004. We perform significantly reduce 5.19 and 0.15 in terms of RMSE and show improvement $1.75^*10^3$ and $1.11^*10^3$ in terms of scoring function, for FD002 and FD004, respectively. The samples working in multimodal switching in FD002 and FD004 datasets are more challenging for obtaining accurate prediction results. Our proposed network structure is able to find hidden patterns, and the prediction capability of the proposed method is better than the existing RUL prediction methods under the complex conditions. According to the

Table 1: The description of the CMAPSS dataset.

| Subdataset | FD001 (training/test) | FD002 (training/test) | FD003 (training/test) | FD004 (training/test) |
|---|---|---|---|---|
| Engines | 100/100 | 260/259 | 100/100 | 249/248 |
| Operating condition | 1 | 6 | 1 | 6 |
| Fault mode | 1 | 1 | 2 | 2 |

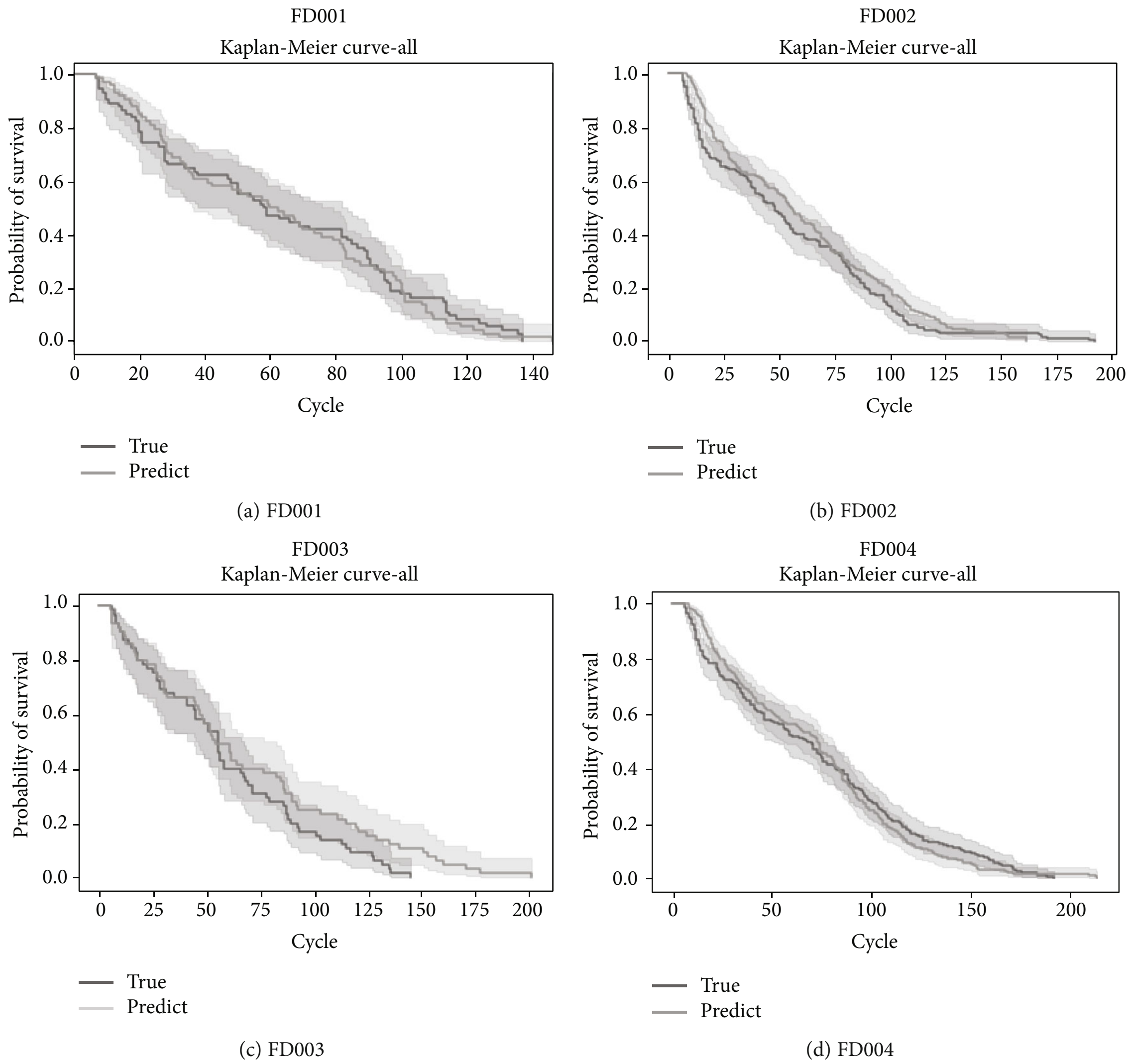
Table 2: The performance comparing our approach and state-of-the-art methods.

| Methods | FD001 | | FD002 | | FD003 | | FD004 | |
|---|---|---|---|---|---|---|---|---|
| | RMSE | Score | RMSE | Score | RMSE | Score | RMSE | Score |
| MLP [41] | 37.56 | $1.80 \times 10^4$ | 80.03 | $7.80 \times 10^6$ | 37.39 | $1.74 \times 10^4$ | 77.37 | $5.62 \times 10^6$ |
| SVR [42] | 20.96 | $1.38 \times 10^3$ | 42.00 | $5.90 \times 10^5$ | 21.05 | $1.60 \times 10^3$ | 45.35 | $3.71 \times 10^5$ |
| RVR [43] | 23.80 | $1.50 \times 10^3$ | 31.30 | $1.74 \times 10^4$ | 22.37 | $1.43 \times 10^3$ | 34.34 | $2.65 \times 10^4$ |
| CNN [31] | 18.45 | $1.29 \times 10^3$ | 30.29 | $1.36 \times 10^4$ | 19.82 | $1.60 \times 10^3$ | 29.16 | $7.89 \times 10^3$ |
| LSTM [22] | 16.14 | $3.38 \times 10^2$ | 24.49 | $4.45 \times 10^3$ | 16.18 | $8.52 \times 10^2$ | 28.17 | $5.55 \times 10^3$ |
| ELM [10] | 17.27 | $5.23 \times 10^2$ | 37.28 | $4.98 \times 10^5$ | 18.47 | $5.74 \times 10^2$ | 30.96 | $1.21 \times 10^5$ |
| DBN [15] | 15.21 | $4.18 \times 10^2$ | 27.12 | $9.03 \times 10^3$ | 14.71 | $4.42 \times 10^2$ | 29.88 | $7.95 \times 10^3$ |
| MODBNE [16] | 15.04 | $3.34 \times 10^2$ | 25.05 | $5.59 \times 10^3$ | *12.51* | $4.22 \times 10^2$ | 28.66 | $6.56 \times 10^3$ |
| RNN [19] | *13.44* | $3.39 \times 10^2$ | 24.03 | $1.43 \times 10^4$ | 13.36 | $3.47 \times 10^2$ | 24.02 | $1.43 \times 10^4$ |
| DCNN [32] | *12.61* | $2.74 \times 10^2$ | 22.36 | $1.04 \times 10^4$ | *12.64* | $2.84 \times 10^2$ | 23.31 | $1.25 \times 10^4$ |
| BiLSTM [26] | 13.65 | $2.95 \times 10^2$ | 23.18 | $4.13 \times 10^3$ | 13.74 | $3.17 \times 10^2$ | 24.86 | $5.43 \times 10^3$ |
| Aug+CNN+LSTM [30] | 23.57 | $1.22 \times 10^3$ | 20.45 | $3.10 \times 10^3$ | 21.17 | $1.30 \times 10^3$ | *21.03* | $4.00 \times 10^3$ |
| DAG [35] | *11.96* | $2.29 \times 10^2$ | *20.34* | $2.73 \times 10^3$ | 12.46 | $5.35 \times 10^2$ | *22.43* | $3.37 \times 10^3$ |
| CNN+LSTM w/regression | 14.04 | $3.10 \times 10^2$ | *15.15* | $1.08 \times 10^3$ | 14.62 | $3.25 \times 10^2$ | *21.92* | $2.26 \times 10^3$ |
| CNN+LSTM w/Weibull | 13.98 | $2.31 \times 10^2$ | *15.77* | $9.79 \times 10^2$ | 15.55 | $6.72 \times 10^2$ | 23.05 | $3.67 \times 10^4$ |

defining of RMSE in Equation (13), the extrapolated RUL value as shown in FD001 and FD003 leads to little larger RMSE in the results. Although those subsets did not reach the best performance in RMSE, the score values are close to the best one. The experimental results also show that the integrated deep learning-related methods such as DAG and our method get better performance than previous single CNN or LSTM methods. Based on the comparison of the scoring function evaluation criteria, it can be seen that our proposed method appears in the top 3 ranked results among all four benchmark datasets, but the DAG method get three out of four in the experimental results. It shows that our proposed method can avoid failure with the early prediction.

One engine unit may have its own historical sensor data, and we apply the fixed sliding time window to scan the historical data to generate the several sequential sensor data. We estimate the RUL and its probability of failure corresponding to each sequential data and then construct the Kaplan-Meier survival curve. Figure 4 shows the predicted and the actual degradation process of the four randomly selected engines in each subdataset from the test data with Kaplan-Meier curve. We can find that the engines have 20% probability to keep running after 100 cycles, namely, the engines have 80% probability that it will have failure after operating 100 cycles in Figure 4(a). The survival probability of these engines decreases by 10% ~20% after every 20 cycles. From FD002 in Figure 4(b) and FD004 in Figure 4(d), the degradation trends have a fast decrease at the beginning and become steady when left 10% probability of availability, which is very close to failure. As for FD003 in Figure 4(c), there is a plateau in the right extrapolated tail of the Kaplan-Meier curve and it may cause the error between the predicted and true values to become bigger with cycles. This is the reason that RMSE of FD003 cannot surpass other approaches.

We describe the difference of our7 approach based on the discrete Weibull distribution and regression analysis with the linear model to predict the failure cycle. In Figures 5–8, we randomly selected four engines in each subdataset as examples and showed Kaplan-Meier curve with confidence intervals and probabilities based on the discrete Weibull distribution and RUL value based on linear regression analysis under time cycles. The results in Figures 5–8(b) have more variance and more error at the beginning of the estimation in the regression analysis approach, especially when the conditions become more complicated. It is easy to find that the prediction errors are greater in the early stage of degradation than in the late stage of degradation in the regression analysis approach. Due to the late stage with more sequential information comparing to the early stage, the predicted results can get better performance. The traditional RUL methods based on the regression analysis might lead to inconsistent



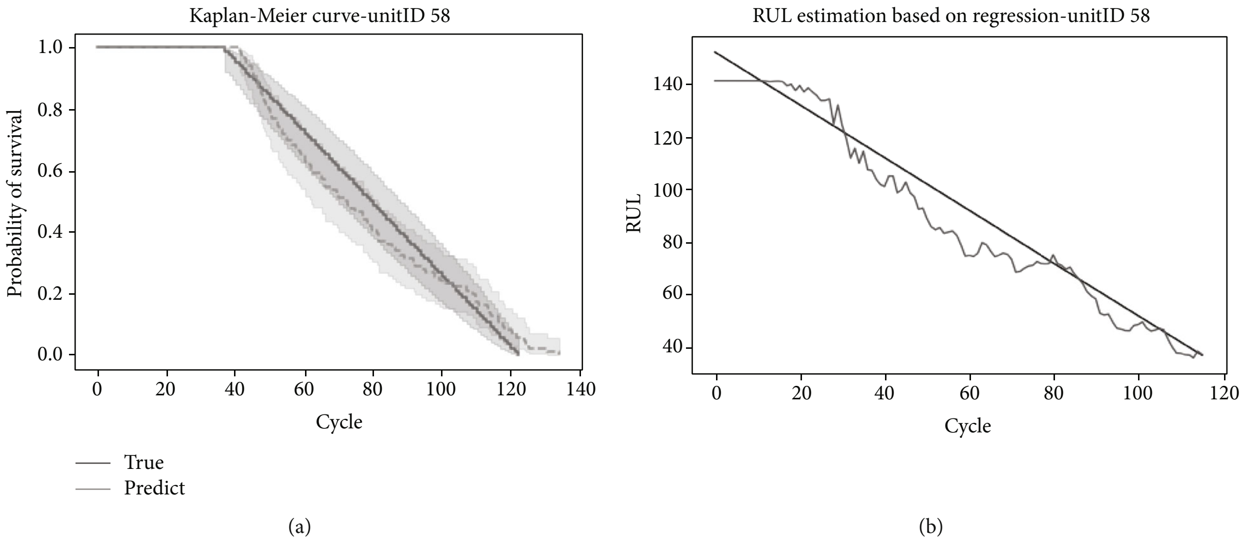
(a) FD001

(b) FD002

(c) FD003

(d) FD004

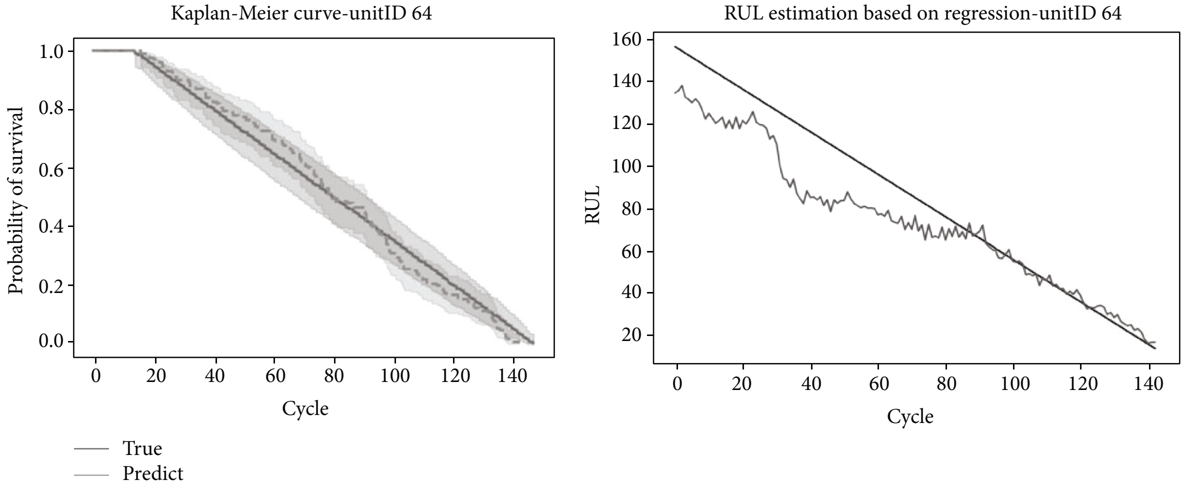
Figure 4: Results of the discrete Weibull distribution with Kaplan-Meier curve for all engines in each subdataset.



(a)

(b)

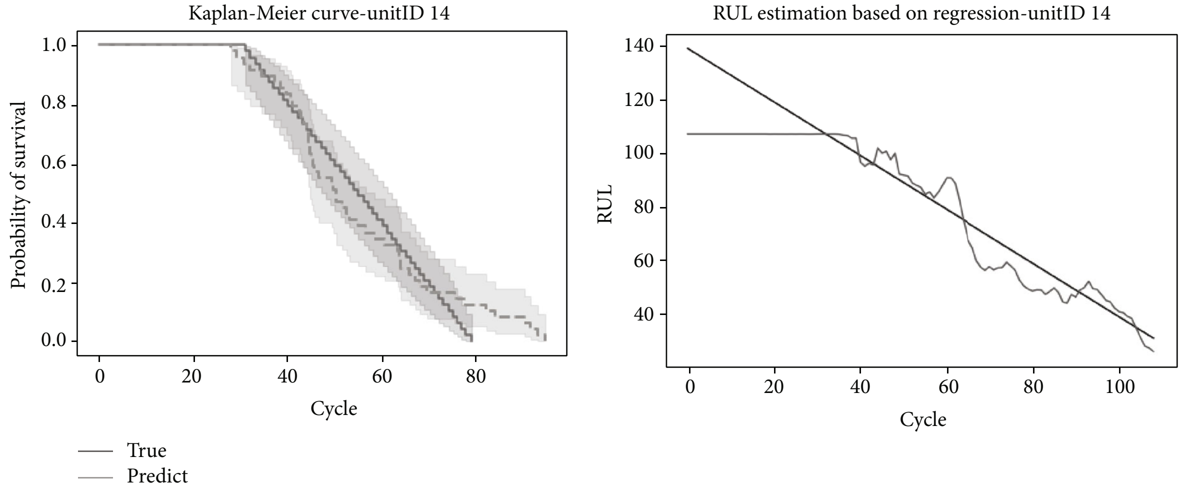
Figure 5: Result of engine ID 58 in FD001 (a) based on the Weibull distribution and (b) based on regression.



(a)

(b)

Figure 6: Result of engine ID 64 in FD002 (a) based on the Weibull distribution and (b) based on regression.



(a)

(b)

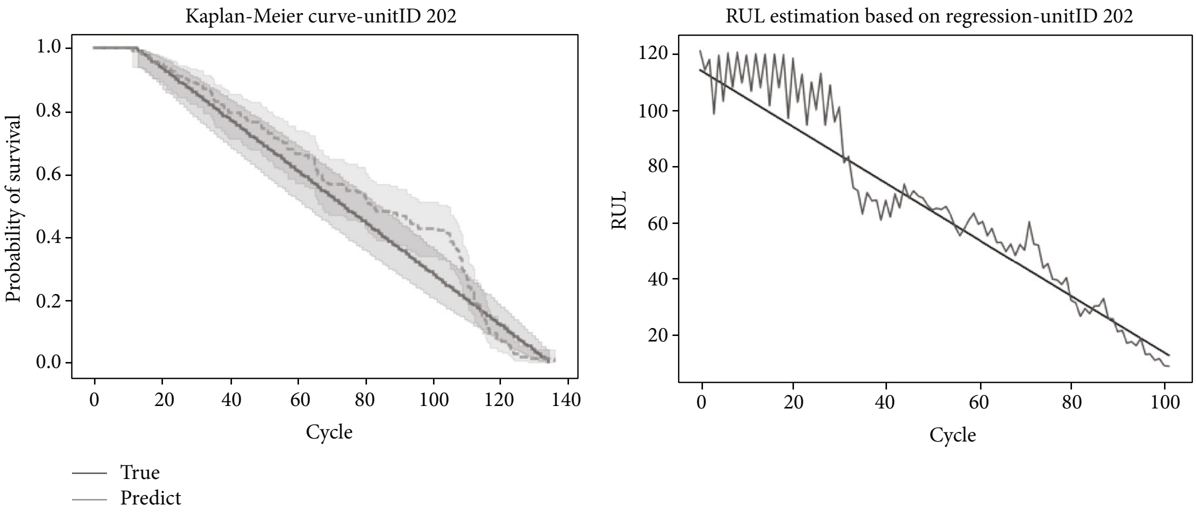
Figure 7: Result of engine ID 14 in FD003 (a) based on the Weibull distribution and (b) based on regression.

predictions that are posing a dilemma for the maintenance decision. Instead, the survival analysis based on the discrete Weibull distribution has a smoother trend with Kaplan-Meier curve in Figures 5–8(a). On the other hand, previous works based on the regression analysis only evaluate the performance at the last time step of each engine for RUL estimation and do not achieve confidence intervals for the corresponding RUL prediction. Our model based on the discrete Weibull distribution represents the probabilities of the failure and the trustworthy confidence intervals for the inherent uncertainties with the degradation process.

*4.3. Performance with Different Time Windows.* We apply more reference to compare the performance with different TW lengths based on two analysis approaches for each subdataset in Tables 3–6. The italicized numbers denote the best results among different TW under two analyses. As for MAE and RMSE, regression analysis approach usually has better performance than the Weibull distribution. As for $R^2$, it shows that the discrete Weibull distribution is much more explainable and also denotes that the degradation trends of the turbofan engines are more similar to the Weibull distribution.

## 5. Conclusions

With the growth of smart manufacturing in the industry, more and more data will be collected and deep learning models will be extremely applied to estimate the health statement of a machine for the maintenance strategy. The predictive maintenance can bring advantages for proposing strategies to optimize the maintenance schedule with a goal of reducing unplanned downtime, as well as needless preventive maintenance to save the most cost for the company. We propose an integrated deep learning approach with convolutional neural networks and long short-term memory networks to learn the latent features and estimate remaining useful life value with deep survival model based on the



(a)

(b)

Figure 8: Result of engine ID 202 in FD004 (a) based on the Weibull distribution and (b) based on regression.

Table 3: Results of FD001 with two analyses and different TW.

| | Loss function (regression) | | | | Loss function (Weibull) | | | |
|---|---|---|---|---|---|---|---|---|
| TW | MAE | RMSE | $R^2$ | Score | MAE | RMSE | $R^2$ | Score |
| 20 | 17.26 | 23.13 | 0.65 | 3,737 | 20.24 | 28.64 | 0.84 | 12,492 |
| 30 | 15.06 | 19.06 | 0.73 | 1,065 | 15.08 | 21.50 | 0.91 | 2,324 |
| 40 | 12.81 | 17.89 | 0.79 | 994 | 14.14 | 19.91 | 0.91 | 2,164 |
| 50 | 11.29 | 14.90 | 0.86 | 431 | 13.14 | 19.23 | 0.91 | 5,861 |
| *60* | *10.20* | *14.04* | *0.88* | *310* | 13.69 | 20.36 | 0.90 | 31,427 |
| 70 | 10.80 | 14.33 | 0.87 | 254 | 13.98 | 20.01 | 0.90 | 5,888 |
| 80 | 10.32 | 14.14 | 0.86 | 198 | 11.63 | 15.45 | 0.93 | 325 |
| *90* | 10.36 | 14.15 | 0.87 | 200 | *10.50* | *13.98* | *0.94* | *231* |
| 100 | 9.99 | 13.98 | 0.79 | 190 | 11.17 | 15.02 | 0.91 | 254 |
| 110 | 9.91 | 13.81 | 0.77 | 183 | 11.03 | 15.43 | 0.90 | 389 |
| 120 | 9.44 | 13.85 | 0.86 | 161 | 9.79 | 13.73 | 0.93 | 206 |

Table 4: Results of FD002 with two analyses and different TW.

| | Loss function (regression) | | | | Loss function (Weibull) | | | |
|---|---|---|---|---|---|---|---|---|
| TW | MAE | RMSE | $R^2$ | Score | MAE | RMSE | $R^2$ | Score |
| 20 | 23.85 | 32.58 | 0.50 | 44,982 | 22.88 | 33.89 | 0.81 | 46,493 |
| 30 | 24.09 | 33.36 | 0.59 | 41,297 | 21.94 | 30.90 | 0.84 | 29,696 |
| 40 | 22.22 | 31.75 | 0.63 | 36,752 | 20.86 | 28.92 | 0.86 | 21,125 |
| 50 | 18.94 | 27.62 | 0.70 | 12,390 | 18.24 | 25.60 | 0.88 | 7,232 |
| 60 | 16.04 | 22.42 | 0.81 | 4,696 | 17.31 | 24.60 | 0.89 | 5,494 |
| 70 | 13.98 | 20.34 | 0.84 | 3,232 | 16.11 | 21.87 | 0.90 | 3,497 |
| 80 | 13.43 | 20.03 | 0.80 | 5,478 | 15.19 | 20.12 | 0.90 | 2,293 |
| *90* | *10.71* | *15.15* | *0.87* | *1,085* | 13.57 | 17.58 | 0.91 | 1,349 |
| *100* | 10.37 | 15.32 | 0.84 | 1,476 | *11.70* | *15.77* | *0.91* | *979* |
| 110 | 9.96 | 15.13 | 0.84 | 1,414 | 13.53 | 17.97 | 0.88 | 1,701 |
| 120 | 9.64 | 14.27 | 0.84 | 790 | 12.69 | 16.39 | 0.88 | 1,810 |

Table 5: Results of FD003 with two analyses and different TW.

| | Loss function (regression) | | | | Loss function (Weibull) | | | |
|---|---|---|---|---|---|---|---|---|
| TW | MAE | RMSE | $R^2$ | Score | MAE | RMSE | $R^2$ | Score |
| 20 | 14.41 | 20.20 | 0.79 | 2,437 | 33.45 | 49.28 | 0.52 | 4,598,400 |
| 30 | 11.79 | 17.12 | 0.85 | 1,700 | 24.94 | 37.87 | 0.65 | 139,633 |
| 40 | 11.01 | 16.02 | 0.83 | 1091 | 15.86 | 24.33 | 0.88 | 9,330 |
| 50 | 11.24 | 15.41 | — | 679 | 14.45 | 21.82 | 0.90 | 8,597 |
| *60* | *10.80* | *14.62* | *0.86* | *325* | 14.69 | 23.57 | 0.87 | 32,361 |
| 70 | 12.08 | 15.94 | 0.80 | 401 | 14.11 | 22.73 | 0.88 | 22,581 |
| 80 | 11.87 | 16.54 | 0.82 | 379 | 13.69 | 23.01 | 0.87 | 30,087 |
| 90 | 12.25 | 17.25 | 0.79 | 353 | 14.00 | 23.42 | 0.86 | 39,985 |
| 100 | 11.88 | 17.53 | 0.78 | 366 | 14.60 | 25.52 | 0.83 | 30,360 |
| 110 | 12.58 | 19.41 | 0.76 | 463 | 12.21 | 20.98 | 0.89 | 11,536 |
| *120* | 13.45 | 21.20 | — | 625 | *9.81* | *15.55* | *0.93* | *672* |

Table 6: Results of FD004 with two analyses and different TW.

| | Loss function (regression) | | | | Loss function (Weibull) | | | |
|---|---|---|---|---|---|---|---|---|
| TW | MAE | RMSE | $R^2$ | Score | MAE | RMSE | $R^2$ | Score |
| 20 | 27.81 | 36.39 | 0.54 | 58,522 | 28.69 | 38.56 | 0.78 | 345,533 |
| 30 | 27.44 | 35.89 | 0.52 | 65,319 | 27.36 | 36.75 | 0.76 | 580,301 |
| 40 | 22.77 | 30.32 | 0.64 | 17,827 | 23.16 | 31.59 | 0.84 | 41,486 |
| 50 | 21.16 | 29.17 | 0.68 | 13,305 | 20.87 | 28.67 | 0.84 | 10,035 |
| 60 | 19.41 | 26.37 | 0.73 | 11,064 | 21.59 | 29.99 | 0.84 | 51,971 |
| 70 | 19.83 | 27.56 | 0.73 | 6,999 | 18.11 | 26.05 | 0.88 | 11,148 |
| 80 | 18.59 | 24.55 | 0.78 | 4,652 | 18.85 | 27.68 | 0.86 | 120,754 |
| 90 | 18.41 | 24.93 | 0.77 | 4,832 | 16.90 | 24.36 | 0.89 | 75,231 |
| *100* | 17.78 | 23.69 | 0.77 | 3,448 | *15.71* | *23.05* | *0.89* | *36,726* |
| 110 | 16.11 | 21.97 | 0.81 | 2,726 | 16.95 | 24.98 | 0.87 | 96,205 |
| *120* | *16.29* | *21.92* | *0.82* | *2,260* | 16.75 | 25.09 | 0.88 | 122,440 |

discrete Weibull distribution. Our works can not only estimate RUL but also learn the failure probability. We can provide reference for making a decision about when and how often the replacement should be implemented. In particular, our approach does well on the harder task under the complex conditions with a subtle drop on the error and scoring function compared with other existing state-of-the-art methods. The improved results have proven that our proposed model can capture the degradation trend of a fault under complex conditions and avoid failure with the early prediction. The limitation of our approach is that our model relies on the specific probability distributions corresponding to a mixture of the two-parameter discrete Weibull distributions that may not be suitable for every degradation process. There are still some available distributions that can be implemented in the survival analysis approach. The data-driven deep learning approach depends on the quality of the data and strongly requires large labeled training datasets in the supervised learning. But getting sufficient run-to-failure data for training process is very difficult, especially for new systems. For further improvements, it may be possible to use a generative adversarial network for data augmentation or generation for the future research. Since the data condition and fault mode are different between subdatasets, further optimization via transfer learning method is still necessary to improve the stability of the method and then efficiently apply to solve other problems.

## Abbreviations

$X(n)$: Matrix including cycles and sensor measurements of $n$th engine
$k$: Types of sensor measurements (features)
$L_s$: Last observed cycle or the cycle that fault occurs
$m$: Time window (TW) length
$x^t$: $k$-dimensional array at cycle $t$ of an engine
$X_i(n)$: Sequential data extracted from $X(n)$ with size $m \times k$
$x_k^t$: The value at cycle $t$ and sensor measurement $k$ in $x^t$
$d$: Kinds of the sliding windows (feature detector) in CNN
$X_{dw}$: Set of feature detector regions with w kernel size $w$ in CNN
$C^{(l)}$: Convolutional $l$th layer
$f_l$: Forget gate's activation vector of the layer $l$ in LSTM
$\widetilde{Z_l}$: Cell input activation vector of the layer $l$ in LSTM
$Z_l$: Cell state vector of the layer $l$ in LSTM.

## Data Availability

The dataset used in this paper is the Turbofan Engine Degradation Simulation Dataset, provided by NASA, retrieved from https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## References

[1] A. K. S. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical Systems and Signal Processing*, vol. 20, no. 7, pp. 1483–1510, 2006.

[2] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, "Machinery health prognostics: a systematic review from data acquisition to RUL prediction," *Mechanical Systems and Signal Processing*, vol. 104, pp. 799–834, 2018.

[3] R. Zhao, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring: A surve," 2016, https://arxiv.org/abs/1612.07640.

[4] D. Wang, K.-L. Tsui, and Q. Miao, "Prognostics and health management: a review of vibration based bearing and gear health indicators," *IEEE Access*, vol. 6, pp. 665–676, 2018.

[5] P. C. Paris, M. P. Gomez, and W. E. Anderson, "A rational analytic theory of fatigue," *Trend Eng*, vol. 13, pp. 9–14, 1961.

[6] W. K. Yu and T. A. Harris, "A new stress-based fatigue life model for ball bearings," *Tribology Transactions*, vol. 44, no. 1, pp. 11–18, 2001.

[7] J. Luo, K. R. Pattipati, Liu Qiao, and S. Chigusa, "Model-based prognostic techniques applied to a suspension system," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 38, no. 5, pp. 1156–1168, 2008.

[8] A. Mosallam, K. Medjaher, and N. Zerhouni, "Data-driven prognostic method based on Bayesian approaches for direct remaining useful life prediction," *Journal of Intelligent Manufacturing*, vol. 27, no. 5, pp. 1037–1048, 2016.

[9] N. Li, Y. Lei, T. Yan, N. Li, and T. Han, "A wiener-process-model-based method for remaining useful life prediction considering unit-to-unit variability," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 3, pp. 2092–2101, 2019.

[10] K. Javed, R. Gouriveau, and N. Zerhouni, "A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering," *IEEE Transactions on Cybernetics*, vol. 45, no. 12, pp. 2626–2639, 2015.

[11] J. Liu and Z. Chen, "Remaining useful life prediction of lithium-ion batteries based on health Indicator and Gaussian process regression model," *IEEE Access*, vol. 7, pp. 39474–39484, 2019.

[12] Y. Zhang, R. Xiong, H. He, and M. G. Pecht, "Lithium-ion battery remaining useful life prediction with box–cox transformation and Monte Carlo simulation," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1585–1597, 2019.

[13] 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.

[14] S. Hong, E. Z. Zhou, K. Zio, and K. Hong, "Condition assessment for the performance degradation of bearing based on a combinatorial feature extraction method," *Digital Signal Processing*, vol. 27, pp. 159–166, 2014.

[15] Y. Lei, *Intelligent Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery*, Butterworth-Heinemann, 2016.

[16] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2306–2318, 2017.

[17] S. J. Wu, N. Gebraeel, M. A. Lawley, and Y. Yih, "A neural network integrated decision support system for condition-based optimal predictive maintenance policy," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 37, no. 2, pp. 226–236, 2007.

[18] P. Baruah and R. B. Chinnam, "HMMs for diagnostics and prognostics in machining processes," *International Journal of Production Research*, vol. 43, no. 6, pp. 1275–1293, 2005.

[19] A. Malhi, R. Yan, and R. X. Gao, "Prognosis of defect propagation based on recurrent neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 3, pp. 703–711, 2011.

[20] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *2008 International Conference on Prognostics and Health Management*, Denver, CO, USA, October 2008.

[21] 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.

[22] 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)*, Dallas, TX, USA, June 2017.

[23] J. Zhang, P. Wang, R. Yan, and R. X. Gao, "Long short-term memory for machine remaining life prediction," *Journal of Manufacturing Systems*, vol. 48, pp. 78–86, 2018.

[24] A. Elsheikh, S. Yacout, and M.-S. Ouali, "Bidirectional handshaking LSTM for remaining useful life prediction," *Neurocomputing*, vol. 323, pp. 148–156, 2019.

[25] 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.

[26] J. Wang, G. Wen, S. Yang, and Y. Liu, "Remaining useful life estimation in prognostics using deep bidirectional LSTM neural network," in *2018 Prognostics and System Health Management Conference (PHM-Chongqing)*, pp. 1037–1042, Chongqing, China, October 2018.

[27] T. A. Shifat and H. J. Wook, "Remaining useful life estimation of BLDC motor considering voltage degradation and attention-based neural network," *IEEE Access*, vol. 8, pp. 168414–168428, 2020.

[28] W. Bao, X. Miao, H. Wang, G. Yang, and H. Zhang, "Remaining useful life assessment of slewing bearing based on spatial-temporal sequence," *IEEE Access*, vol. 8, pp. 9739–9750, 2020.

[29] X. Li, W. Zhang, H. Ma, Z. Luo, and X. Li, "Data alignments in machinery remaining useful life prediction using deep adversarial neural networks," *Knowledge-Based Systems*, vol. 197, article 105843, 2020.

[30] L. Jayasinghe, T. Samarasinghe, C. Yuenv, J. C. N. Low, and S. S. Ge, "Temporal convolutional memory networks for remaining useful life estimation of industrial machinery," in *2019 IEEE International Conference on Industrial Technology (ICIT)*, pp. 915–920, Melbourne, Australia, February 2019.

[31] 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*, Springer, 2016.

[32] 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.

[33] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, https://arxiv.org/abs/1803.01271.

[34] K. Aggarwal, O. Atan, A. K. Farahat, C. Zhang, K. Ristovski, and C. Gupta, "Two birds with one network: unifying failure event prediction and time-to-failure modeling," in *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, December 2018.

[35] J. Li, X. Li, and D. He, "A directed acyclic graph network combined with CNN and LSTM for remaining useful life prediction," *IEEE Access*, vol. 7, pp. 75464–75475, 2019.

[36] E. Martinsson, *Wtte-Rnn: Weibull Time to Event Recurrent Neural Network*, Chalmers University of Technology & University of Gothenburg, 2016.

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting, 7e," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[38] E. T. Lee and J. W. Wang, *Statistical Methods for Survival Data Analysis*, Wiley Series in Probability and Statistics Copyright © 2003 John Wiley & Sons, Inc., Third edition, 2003.

[39] P. Wang, Y. Li, and C. K. Reddy, "Machine learning for survival analysis: a survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, pp. 1–36, 2019.

[40] A. Saxena and K. Goebel, *Turbofan Engine Degradation Simulation Data Set*, NASA Ames Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, USA, 2008.

[41] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[42] C. C. Chang and C. J. Lin, "LIBSVM: a library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011.

[43] M. E. Tipping, "The relevance vector machine," in *NIPS'99: Proceedings of the 12th International Conference on Neural Information Processing Systems*, pp. 652–658, Denver, CO, USA, November 1999.