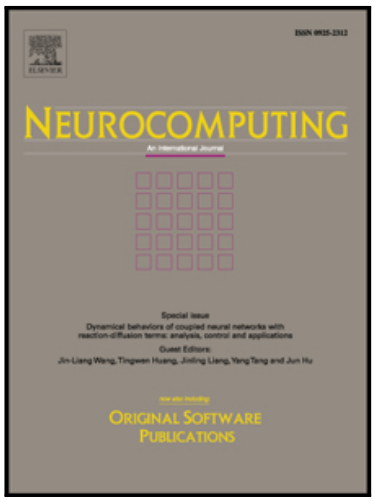

Bidirectional Handshaking LSTM for Remaining Useful Life Prediction

Ahmed Elsheikh , Soumaya Yacout , Mohamed-Salah Ouali

PII: S0925-2312(18)31157-3
DOI: https://doi.org/10.1016/j.neucom.2018.09.076
Reference: NEUCOM 20009

To appear in: *Neurocomputing*

Received date: 1 March 2018
Revised date: 20 July 2018
Accepted date: 27 September 2018

Please cite this article as: Ahmed Elsheikh , Soumaya Yacout , Mohamed-Salah Ouali , Bidirectional Handshaking LSTM for Remaining Useful Life Prediction , *Neurocomputing* (2018), doi: https://doi.org/10.1016/j.neucom.2018.09.076

Highlights

- Novel bidirectional LSTM architecture.
- Novel asymmetric objective function for making safe predictions.
- Novel approach for generating remaining useful life targets for training.
- Improved performance for short sequences with random starts.
- The turbofan engine dataset from NASA's repository is used for comparison.

Bidirectional Handshaking LSTM for Remaining Useful Life Prediction

Abstract

Unpredictable failures and unscheduled maintenance of physical systems increases production resources, produces more harmful waste for the environment, and increases system life cycle costs. Efficient Remaining Useful Life (RUL) estimation can alleviate such an issue. The RUL is predicted by making use of the data collected from several types of sensors that continuously record different indicators about a working asset, such as vibration intensity or exerted pressure. This type of continuous monitoring data is sequential in time, as it is collected at a certain rate from the sensors during the asset's work. Long Short-Term Memory (LSTM) neural network models have been demonstrated to be efficient throughout the literature when dealing with sequential data because of their ability to retain a lot of information over time about previous states of the system. This paper proposes using a new LSTM architecture for predicting the RUL when given short sequences of monitored observations with random initial wear. By using LSTM, this paper proposes a new objective function that is suitable for the RUL estimation problem, as well as a new target generation approach for training LSTM networks, which requires making lesser assumptions about the actual degradation of the system.

Keywords

Remaining useful life prediction, bidirectional handshaking, long short-term memory, asymmetric objective function, target generation.

1 Introduction

Remaining Useful Life (RUL) prediction is the attempt to predict the remaining period of normal operation, at a certain level of performance, for a physical system [1]. There has been numerous research on RUL production in the literature. Nevertheless, there is no one approach that is universal because of the variability in the physics of different systems, their surrounding conditions, initial working conditions, and the physics of the acquisition devices [2]–[4]. The initial condition of the working asset or its subcomponents affect the asset's RUL. Poor initial conditions put the asset at a higher risk of earlier failure, which in turn means shorter RUL [3].

Previous research can be mainly classified into physical modelling approaches and data-driven approaches [2], [5]. Physical modelling achieves the best performance if it can be accurately formulated. This is an almost impossible task as a result of the large amount of interacting variables that can affect the physical system directly and indirectly, which are mostly unknown [1]. Moreover, these variables interact with each other in highly complex, non-linear ways that cannot be anticipated [2].

Accurate RUL prediction allows users of a physical system to benefit the most from its working lifetime, and to make efficient decisions regarding maintenance and replacement [6]. In turn, this means more profit and fewer losses due to unexpected faults or failures [5]. Due to the random nature of the system behavior, there is always the possibility that the predicted RUL will either be earlier or later than the actual lifetime. In this case, it is safer to have earlier rather than later predictions to avoid catastrophic failures [7].

Data-driven approaches offer good approximations for a system's failure mechanism based on historical data. These approaches vary in their capacity to learn complex systems [1]. The monitored signals captured by sensors are acquired at a certain rate and in a chronological order from the working system. Therefore, processing sequential data such as this and continuously predicting the RUL is a problem that is suitable for sequential modelling. Data-driven approaches handle sequential data differently; either

intrinsically, such as Hidden Markov Models (HMM) [8], by windowing such as Convolutional Neural Networks [9], [10] and most machine learning techniques, or by transformations to compress variable length sequences in a set of informative features [11]. The latter two approaches do not harness much of the sequential information as a result of their limited scope of observation in time and compression of information. On the other hand, models that are sequential in nature capture sequential dependencies between various observations in time in a more compact manner.

There is also another type of non-sequential data-driven modelling which depend on Key Performance Indicators (KPIs) which try to focus on some important aspects of the production system that is crucial to the system such as fuel consumption or the amount of produced power for engine systems [12], [13]. Although these approaches are meant for diagnosis, they can yet be used for prognosis [14]. Although these approaches are statistically well formulated, they are either computationally demanding since they require matrix inversion or memory demanding for kernel calculations.

One of the most common sequential modelling techniques is the Recurrent Neural Networks (RNN) [15]. The main advantage of RNNs over HMMs is that HMMs have a finite, discrete set of states to represent the system, while RNN theoretically has no such limitations [16]. RNN and its more recent variations such as Long Short-Term Memory (LSTM) networks [17] and the Gated Recurrent Unit networks [18] have shown some success in various domains that have a sequential nature [19], or that can be processed sequentially [20].

There is very limited work in the literature using RNN and its variants for RUL prediction [1], [21], and even fewer publications that actually report performance and scores on real test scenarios. Very recent publications, as in [22], use LSTM and discuss how the data is prepared for the RUL prediction task, but they use a subset of the training set as the test data, which means they know the full testing sequence to allow it to produce detailed results about the performance of the LSTM over time. Nevertheless, they did not report or prepare the network to work with short sequences of sensor observations. Also, another very recent attempt was the use of an ensemble of Echo State Networks (ESN) [23]. However, none of the

previous publications report results using bidirectional RNNs [24], which process input sequences in both directions. This architecture has shown improvements in different recognition applications that are sequential in nature [25]. Yet, this type of architecture requires certain modifications to suit the RUL prediction problem, which is different than recognition.

All of the aforementioned sequential modelling techniques are supervised and therefore require training on targets. Since the RUL cannot be assumed to be degrading linearly at every working cycle, especially if the working asset starts at a new condition, some papers have made assumptions about the nature of the actual RUL by using the piece-wise degradation function, which starts constant and degrades according to a certain power function [22], [26]. The point of inflection from constant to degradation is determined by inspection [26] or by tuning a detector model [22]. Such assumptions restrict the model to making predictions given a full history of the working asset until failure. This implies that partial maintenance procedures causing the asset to start at an improved health condition are not taken into account, and such maintenance actions induce only minimal repair.

This paper presents four main contributions in order to predict the RUL:

1- Predicting the RUL from short observation sequences with random starts, since the initial condition of physical systems is usually unknown due to manufacturing deficiencies, replacements of parts of the system, and non-ideal maintenance. Hence, the network is trained to anticipate such requirements. Conventionally the initial condition is either considered to be given [27], or estimated as a separate model [28].
2- Proposing a new safety-oriented objective function for the LSTM network to train the network to favor safer, earlier prediction rather than later prediction, since all well-known accuracy measures are either symmetric, such as the Mean Squared Error (MSE) and Mean Absolute Error (MAE), or rectified such as hinge loss [29], and both types are not suitable for making safe predictions.
3- Proposing a new target RUL generation procedure for the training process of the network. Instead of relying on the manual examination of the data [26] or tuning other models [22], the proposed

procedure for generating predicted RUL uses the given sensor readings for defining an approximation for the actual RUL.

4- Proposing a new bidirectional LSTM network architecture that suits the RUL prediction problem. Since there are no intermediate predictions required for a given sequence of observations, and only the RUL needs to be predicted after the given sequence is observed, then there is no need to process the sequence simultaneously in both directions.

   Instead, the proposed architecture processes the observed sequence in both directions sequentially by processing the sequence in the forward direction, and then using the LSTM final states to initialize the backward processing cells. This architecture forces the network to obtain two different yet linked mappings of the observation sequences to the desired RUL. The reason is that each cell is a function of the current input and the previous state, and both are different for LSTMs processing in opposite directions, unlike all-forward LSTM architectures, where all LSTMs have the same input sequence but different previous states.

2 The Long Short-Term Memory Cell

An LSTM cell was proposed to overcome the limitations of training the classical RNN [17]. Instead of having the output of the RNN cell be a non-linear function of the weighted sum of the current inputs and previous output, the LSTM uses storage elements to pass information from the past outputs to current outputs.

The LSTM has three control signals, such that each is a non-linear function σ activated by a weighted sum of the current input observation x_t and previous hidden state h_{t-1} as shown in equations 1-6. The forget gate f_t decides whether to retain or forget the previous state c_{t-1} of the LSTM. The input gate i_t decides whether to update the state of the LSTM using the current input or not, and the output gate o_t decides whether to pass on the hidden state h_t to the next iteration or not. The new state c_t stored in the

LSTM is the sum of the new gated input a_t and the gated previous state c_{t-1} as shown in equation 5. Figure 1 illustrates how the gates and inputs interact.

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

$$o_t = \sigma(W_o x_t + H_o h_{t-1} + b_o) \quad (2)$$

$$f_t = \sigma\left(W_f x_t + H_f h_{t-1} + b_f\right) \quad (3)$$

$$a_t = \tanh(W_a x_t + H_a h_{t-1} + b_a) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot a_t \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

Where W_*, H_*, and b_* are the trainable weights and biases, respectively, for each gating signal indicated by $*$, h_{t-1} is the hidden layer activation of the previous iteration, while h_t is the current hidden layer activation. Similar to the hidden states, the cell states c_t, c_{t-1} are defined. The current input is x_t, and the gate activations are f_t, a_t, i_t as described previously. Finally, $\odot$ is the elementwise multiplication operator.

The training of the LSTM using backpropagation is much more stable than the classical RNN and can theoretically retain information for prolonged periods of time [17]. This is beneficial when attempting to make forecasts about the future by learning from long sequences of historical data.

Figure 1 LSTM cell diagram.

2.1 Bidirectional LSTM

Bidirectional RNNs are a modification to the conventional RNNs to process sequences of observations in both directions, starting from the first input observation to the last, and starting from the last observation back to the first [24]. This requires the processed sequence to be buffered into windows of observations to allow processing in both directions. At any point in time, the network makes use of the earlier observations processed by the forward LSTM cells until the point of prediction, as well as the upcoming observations processed by the backward LSTM cells to make the prediction. This approach is beneficial when making intermediate predictions, like recognizing phonemes in speech recognition [25]. This is

different from the prediction task, where predictions are required ahead of the whole given sequence. This requirement is the trigger for the new proposed architecture for bidirectional LSTMs.

3 The Proposed Methodology

3.1 Bidirectional Handshaking LSTM (BHSLSTM)

The aim of the proposed approach is to extract as much information as possible from a given subsequence of observations to make predictions about the RUL of the system. Processing the given sequence in the forward direction, which means in the same sequence as they appeared in the system through several LSTM cells, produces a summary vector, which contains the final output after passing the given sequence through the cells.

The proposed handshaking approach initializes another set of LSTM cells that process the given sequence in reverse order, starting with the last observation and ending up with another summary vector at the first observation. This allows the LSTM network to have more insights when identifying the trend of the sequence in both directions. Moreover, the handshaking procedure allows the learning process to be collaborative between the forward and backward units, and therefore provides better results.

Figure 2 illustrates the architecture when using a single LSTM cell in the forward direction and another for the backward direction. The figure shows how the final state of the forward processing cell initializes the state of the backward processing cell.

Figure 2 BHLSTM diagram for a single forward and a single backward LSTM cells.

3.2 Safety-Oriented Objective Function

The objective function is the error function at the output of the network that needs to be minimized. The networks are trained by minimizing the error between the network predictions and the actual RUL in the training data. As mentioned earlier, the main goal of RUL estimation is to anticipate upcoming failures before they occur to avoid unnecessary downtime, as well as additional maintenance costs. That is why

the Scoring Function (SF) proposed by [30] favors early estimations rather than late ones, as shown in the following equation, by assuming that $d = RUL_{pred} - RUL_{actual}$:

$$SF = \begin{cases} e^{-\frac{d}{\alpha_1}}, & d < 0 \\ e^{\frac{d}{\alpha_2}}, & d \geq 0 \end{cases}, \quad \alpha_1, \alpha_2 \in \mathbb{R}^+ \tag{7}$$

Where α_1 and α_2 are the penalty factors for the SF, such that $\alpha_1 > \alpha_2$, thus the penalty increases when the factors increase. In order to comply with the requirements of the scoring function, and in general any safety measure for prediction, the network is trained with an objective function that penalizes unsafe predictions, where $RUL_{pred} > RUL_{actual}$, at a much higher cost. The SF is not a suitable objective function, as raising the error measure to an exponential is not a well chain-rule-differentiable function because the exponential term must be calculated for every weight update, which makes it computationally expensive. It also can either overflow, resulting in very large updates, or vanish, resulting in no updates at all due to the multiple applications of the exponential term.

Therefore, in this paper, we propose using an approximation of this function as shown in Figure 3, which also favors earlier predictions rather than later ones. This approximation is more suitable, unlike conventional objective functions, which are either symmetric, such as the MSE and MAE functions, or rectified, which means that they are one-side, favoring those such as the hinge functions.

When $d = RUL_{pred} - RUL_{actual}$, the proposed asymmetric objective functions are defined as follows, the Asymmetric Squared Error (ASE)

$$ASE = \begin{cases} \alpha_1 d^2, & d < 0 \\ \alpha_2 d^2, & d \geq 0 \end{cases}, \quad \alpha_1, \alpha_2 \in \mathbb{R}^+ \tag{8}$$

And the Asymmetric Absolute Error (AAE)

$$AAE = \begin{cases} \alpha_1 |d|, & d < 0 \\ \alpha_2 |d|, & d \geq 0 \end{cases}, \quad \alpha_1, \alpha_2 \in \mathbb{R}^+ \tag{9}$$

Where α_1 and α_2 are the penalty weights for early and late predictions respectively such that $\alpha_1 < \alpha_2$. It is recommended to add more penalty during training to force the network to comply to the desired safety constraints.

Figure 3 Comparison between the scoring function (left), the proposed asymmetric squared objective function (middle), and the asymmetric absolute objective function (right).

3.3 Target RUL Generation

The target RUL used for training the network is another challenge, since the actual state of health of a system is not given and is usually unknown. Hence, different suggestions about the nature of degradation were given by different authors [22], [26]. The core idea for their assumptions is that for healthy systems, the degradation is not noticeable, and all system behaviors look alike, making the RUL estimation an impractical task. Therefore, it is assumed that the RUL will be piecewise continuous, such that at the beginning of life, the RUL is constant and then starts decreasing.

The rate of decrease and the point where the system starts degrading are the two main concerns. The author in [26] made an assumption based upon a manual study of the data that the system will start degrading at 130 cycles and will degrade linearly afterwards. The authors of [22] suggested using an anomaly detector to identify the point where the system starts degrading, which signals the start of degradation after three triggers, and they tried several power law degradation functions. They had to tune the anomaly detector, the number of triggers, and the power of the degradation function parameters using a grid search until they achieved good validation results.

Our proposed target preparation is based on the behavior of the sensors' readings of the system. This is a real physical phenomenon that the previous researchers tried to approximate. We start the preparation of the target by taking the moving average smoothed version of all the sensor readings that show a trend, scale them down, and shift their values to start from 1 and end at 0. The sensor readings that show an upward trend are inverted before this step by subtracting all of the sensor readings from the maximum value.

Now, each sensor reading has a value ranging from 1 to 0, which is assumed to represent the ratio of the current health condition relative to the starting health condition at 1 and failure at 0. The lifespan of an asset is the working time, starting from a healthy condition until failure in a run-to-failure dataset. Hence, the RUL at any instant of the working time is a fraction of the lifespan, which is proportional to the health condition at that instant. To get the values to correspond with the remaining useful life, each of these ratios are then multiplied by the lifespan, which is the number of life cycles that the working asset took before failure starting from a healthy condition, as given in the training data. Next, for each sensor, there is the estimation of the actual RUL. The final RUL is chosen from the minimum RUL among all of the estimations to favor safer early predictions.

Finally, all RUL values above the minimum lifespan in the training set are truncated to that value. This truncation was shown to be useful in several previous works [26], [31], [32]. Figure 4 illustrates this procedure for one of the sensors' readings.

In summary, the point at which the system starts degrading and the degradation profile are dependent on the sensor readings and requires no tuning or any assumptions about the function of the degradation.

Figure 4 RUL target generation procedure.

3.4 Data Preparation

Since the network should be prepared to make predictions given a short sequence of observations, the training data must accommodate such requirements. That is why the training sequences are chunked into consecutive overlapping windows of observation sequences, each with a length equal to the minimum anticipated input sequence, which corresponds to the maximum desired number of cycles before the first RUL prediction is given. Each of these input sequences has an output corresponding to the RUL. These windows will allow the network to experience observation sequences from systems starting at different initial health states. The RUL predictions are made using only a single window of observation, so after each window with a certain number of observations, the RUL is predicted without making use of any other observation that came before or after the window. This approach relieves the RUL prediction model

from the requirement to have all observations start from full health until the moment of prediction, which is sometimes not available.

4 Experiments and Results

4.1 Benchmark Dataset Overview

The dataset used for the experiments is the NASA turbofan engine degradation simulation dataset, known as the Commercial Modular Aero-Propulsion System Simulation (CMAPSS) [30]. The dataset has four simulation settings. Two of them have single operating conditions and the other two have multiple operating conditions. The datasets are given in the form of training and testing subsets. The training sets have run-to-failure series so the tests stop some time prior to complete failure, thus it is required to forecast the RUL. The time-series contain 26 sensor readings, as well as operating condition indicators as described in [30].

The datasets used in the experiments are the 1st and 3rd, which contain only one operating condition. The 3rd dataset has multiple failure modes. These datasets are chosen since they can be generalized to multiple operating conditions if the operating modes are given by training a model for each operating condition.

Only the sensors that show trends were considered in the analysis, namely sensors number 4, 7, 8, 11, 12, 13, 15, 17, 20, and 21 as described in [22]. Another piece of information is added to the sensor readings, which is the forward difference of these readings. Since all sensor readings show a trend, such series cannot be considered stationary; hence, we use the forward difference as a detrending procedure to make the analysis of the time series more feasible [33]. The raw sensor readings, as well as the difference, are used as the input to the network. There is no feature extraction step as has been proposed by some of the works of literature such as [30], [34].

4.2 Performance Measures

In order to properly assess the performance of different network architectures, objective functions, and target RUL approximations, a set of performance measures should be defined.

The measures used in this paper are the asymmetric SF, discussed in [30], which penalizes late predictions more than early ones. Assuming that $d = RUL_{pred} - RUL_{actual}$, then the SF has the same for as equation 7 with $\alpha_1 = 13$, and $\alpha_2 = 10$.

$$SF = \begin{cases} e^{-\frac{d}{13}}, & d < 0 \\ e^{\frac{d}{10}}, & d \geq 0 \end{cases} \quad 10$$

The RUL predictions are considered correct if d is in the range [-13, 10] as discussed in [30], [35]. The percentage of correct RUL is formulated as follows, when the correct indicator $I_c(d) = 1, -13 \leq d \leq 10$, and 0 otherwise

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n} I_c(d_i) * 100 \quad 11$$

Where n is the number of testing samples.

The Mean Absolute Error (MAE) is defined as follows

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |d| \quad 12$$

4.3 Performance Evaluation Using Different Network Architectures

Table 1 and Table 2 illustrate the performance measures of different network structures on the validation and training sets for datasets 1 and 3, respectively. The sequences are chosen to be of length 30, and the training set is windowed using this size. A random 20% subset is selected as the validation set. Each structure is run three times and the performance of the ones that perform the best run in the validation set is used to evaluate the performance of the testing set.

The number of LSTM cells in each architecture is selected to be the same: a total of 256 cells for dataset 1 and 512 for dataset 3. The number of LSTM cells is inspired by the results in [22]. Increasing the number of cells increases the representational capacity of the network to approximate complex functions, yet makes it prone to overfitting. On the other hand, decreasing the number of neurons can cause underfitting, and that is why the number of neurons is estimated by trial and error on a validation set. The number of neurons in the feedforward network on top of the LSTM cells has 1024 neurons for dataset 3, and 512 for dataset 1. A final linear neuron is placed at the output of the network to predict the RUL. Since the focus in this paper is to show the effect of the architecture of the network on the accuracy of the prediction, the number of LSTM cells and feedforward neurons remained constant across the experiments.

For the experiments in Table 1 and Table 2, the ASE objective function is used with $\alpha_1 = 0.25$, and $\alpha_2 = 9$, which were chosen by trial and error using the validation set performance of dataset 1. Early stopping is used with a tolerance of a maximum of 2 epochs without improvement in the training performance. The batch size used for training is 32. The optimizer used is the RMSprop algorithm [36].

In the testing phase only the last 30 observations, which is the minimum observation sequence required to be tested, are considered for the prediction task in order to assess the performance, given short sequences of random initial wear.

The results in Tables 1 and 2 illustrate that the BHSLTM outperforms the other structures on the test set performance, although it is not always the best on the training dataset, which means that this structure can generalize better than the other structure.

To assess the effectiveness of the state initialization of the backward processing cells by the final state of the forward processing cells used in the proposed architecture, an experiment is conducted on dataset 1 without the proposed handshake. The results are given in Table 3. They show an overall performance degradation, as well as the final performance on the test data.

Table 1 Dataset 1 performance measures for different network architectures.

Model	Train ASE	Train SF	Validation ASE	Validation SF	Test SF	Test Accuracy	Test MAE
BHSLSTM (128)	4554.56	218.77	3670.12	152.288			
	5589.12	287.31	5092.67	301.78			
	4593.86	235.17	4323.77	145.54	376.64	63	11.7
2 Layers LSTM (128)	4515.8	213.67	3961.57	224.97			
	5094.9	253.22	5056.98	335.82			
	4884.82	228.97	4032.45	151.4	1120.1	50	14.96
1 Layer LSTM (256)	5273.7	244.1	6186.3	378.1	464.8	57	13.7
	7543.53	512.6	7839.87	652.16			
	5129.47	237.5	6195.96	450.9			
BLSTM (128)	5182.67	243.47	5452.87	375.47			
	4926.5	221.93	4475.2	213.23	542.6	58	13
	5288.86	241.2	4377.4	243.95			

Table 2 Dataset 3 performance measures for different network architectures.

Model	Train ASE	Train SF	Validation ASE	Validation SF	Test SF	Test Accuracy	Test MAE
BHSLSTM (256)	6409	485	8747	1677			
	7914	766	6196	670			
	5767	464	5373	527	1422	52	15.79
2 Layers LSTM (256)	6550.89	507	5747	414.45	4931	45	18.36
	7584.9	1096	5828	519			
	5859	366.14	6668.7	530.9			
1 Layer LSTM (512)	8458.5	699	10213	502.4	8158	42	23
	10907.65	997.56	9238	1181.9			
	11435	1160.25	10633	1480			
BLSTM (256)	7947.42	581.52	7158.19	321.5	4715.3	43	19.855
	8400.29	756.75	7061.85	581.25			
	6962	538	7944	867.7306			

Table 3 Bidirectional LSTM performance without the proposed handshake procedure for dataset 1.

Train ASE	Train SF	Validation ASE	Validation SF	Test SF	Test Accuracy	Test MAE
5680.26	285.75	7161.5	596.55			
4745.56	216.18	4038.54	202.35	581.5	53	13.59
4827.67	219.8	4653.3	245.56			

4.4 Performance Evaluation Using the Mean Squared Error with BHSLTM

To assess the performance improvements when using the proposed asymmetric objective function, another set of experiments using the proposed BHLSTM is conducted on datasets 1 and 3 using the MSE objective function.

Table 4 BHLSTM performance using the MSE objective function for dataset 1.

Train ASE	Train SF	Validation ASE	Validation SF	Test SF	Test Accuracy	Test MAE
152.57	113	201.76	111.22	2407.8	26	20.4
145.36	106.16	165.58	144.156			
161.98	126.3	229.69	177.1			

Table 5 BHLSTM performance using the MSE objective function for dataset 3.

Train ASE	Train SF	Validation ASE	Validation SF	Test SF	Test Accuracy	Test MAE
181.8	230.4	224.05	389.39			
193.32	287.15	413.183	324.72			
142.17	140.68	200	273.54	6875	28	35.27

By comparing the performance of the BHLTSM in Table 1 and Table 2, to those in Table 4 and Table 5 a significant drop in performance on the test datasets is noted. As a means for a visual assessment of the effect of the objective function on the estimation of the RUL, Figure 5 illustrates the estimations of a

BHLSTM network trained with ASE and MSE along with the actual target RUL for two of the training engines. The predictions at each cycle use only the previous 30 sensor readings, not the whole sequence up to the point of forecast.

Figure 5 Comparison between RUL forecasts using ASE and MSE objective functions.

Figure 5 shows two merits for the ASE objective function trained network. First, its forecasts are mostly safe and come before the actual RUL. Even the forecasts that come after the actual RUL, like around cycle 80 for engine number 45 and cycle 60 for engine number 57, are not too late. The second merit is that the network trained using the ASE objective approximates the actual RUL curve much better than the other network using the MSE objective function.

4.5 Performance Evaluation Using the Piecewise Linear Target RUL

Finally, to assess the performance improvement when using the proposed target RUL generation procedure with the fixed piecewise RUL proposed by [26], the BHLSTM is trained with such RUL for dataset 1 and the results are given in Table 6.

Table 6 BHLSTM performance on dataset 1 using the piecewise linear RUL.

Train ASE	Train SF	Validation ASE	Validation SF	Test SF	Test Accuracy	Test MAE
11668.16	1672.75	10300.94	1806.76			
10044.38	1478.14	10413.44	1384.8			
11344.18	1720.16	10288.53	2072.7	616.1	54	14.03

Again, by comparing the results of the BHLSTM in Table 1 with Table 6 we can see that the performance of the test set deteriorates.

4.6 Performance Comparison with Other Techniques

In this sub-section we present the results that are published by other researchers who solved the RUL prediction problem for the CMAPSS datasets. Although not all of them report the same metrics on all of the datasets, the scoring function represents a common ground for comparison. Table 7 shows the performance comparison between some of the techniques mentioned in literature.

In addition to the results reported in literature, experiments using the DB-KIT [13] on the same data used for training the neural network in terms of windowed input and targets generated by the proposed approach for RUL generation as the KPI as in [14]. The techniques experimented are the Total Partial Least Squares (TPLS), the Modified Partial Least Squares (MPLS), and the Locally Weighted Projection Regression (LWPR). The number of latent variables required for the TPLS and MPLS were selected using a 5-fold cross-validation to choose the best performing hyperparameter. The kernel-based techniques were not feasible on the computer used for the experiments due to the high dimensionality of the input data ($20 \text{ features} \times 30 \text{ samples per window}$) and the considerable number of training windows in the datasets (17731 for dataset 1 and 21820 for dataset 3).

Results in Table 7 are presented either as a range, or as a single value. N/A indicates that the information is not available. The proposed BHLSTM does not take into consideration the total previous history when making a prediction. Thus, its ability to make predictions for engines with random initial states. Moreover, the performance of the BHLSTM is very close to the best reported results in the literature on dataset 1.

Moreover, the proposed technique used all sensor readings that show trends, without trying to make any feature selection that largely impacts the performance [37], nor to optimize the networks' hyper-parameters. This means that the proposed network is able to properly learn the correct mapping from inputs to outputs requiring lesser effort from the analyst. The focus in this paper is to show the impact of the new proposed architecture, objective function, and target generation on performance of conventional

LTSM networks used for RUL prediction, and to decrease the number of assumptions required by the analyst.

Table 7 Comparison between different the performance of different techniques in literature

Technique	Dataset 1 Score	Dataset 1 Accuracy	Dataset 3 Score	Dataset 3 Accuracy
Multi-Objective DBN Ensemble [38]	334.23	N/A	421.91	N/A
DBN [38]	417.59	N/A	442.43	N/A
LASSO [38]	653.85	N/A	1058.36	N/A
ETR [38]	1667.86	N/A	2240.7	N/A
KNR [38]	729.32	N/A	1030.29	N/A
Gradient Boosting [38]	474.01	N/A	576.72	N/A
SVR [39]	[388, 538]	[54, 64]	N/A	N/A
RULCLIPPER [37]	[310, 440]	[56, 64]	[480, 632]	[55, 63]
Extreme Learning Machine and Fuzzy clustering [40]	1046	48	N/A	N/A
CNN [41]	1290	N/A	1600	N/A
Deep LSTM [42]	338	N/A	852	N/A
TPLS	3914.1	29	3463.8	21
MPLS	2203.1	36	3648.4	23
LWPR	1647.8	33	3848.6	23
BHLSTM	376.64	63	1422	52

As for dataset 3, the performance is still comparable but less efficient due to the lack of information about the operating conditions from the input. This is a more general assumption since the operating conditions of machines are not always explicitly known, nor readable as a continuous input [7], [43], [44].

The KPI-based techniques show almost comparable performance to each other on dataset 3, while the LWPR is the best performing on dataset 1. These models provide a simple representation for the problem but on the account of the performance. So, they can be used to give an approximation for the physics of the system or identification of the most important factors acquired by the monitoring system but cannot be well relied upon for direct RUL prediction.

5 Discussion and Conclusion

In this paper, a new Long Short-Term Memory (LSTM) network architecture, objective training function, and training target generation are proposed. These proposals aim at solving the problem of making Remaining Useful Life (RUL) predictions for physical systems using short sequences of observations with random starting health conditions more efficient.

The proposed Bidirectional Handshaking LSTM (BHLSTM) network architecture uses a bidirectional sequence processing approach in a sequential manner. The forward processing LSTM units pass their final states to the backward processing units instead of simultaneously partially processing the sequence up to the time of prediction, which is more suitable for making intermediate predictions rather than making a single prediction after observing a sequence, such as in the case of RUL prediction. The summary vectors of both directions of the BHLSTM are concatenated for the higher classification layers.

This paper also proposed a new, asymmetric objective function that penalizes late predictions more than earlier ones, thereby ensuring safer predictions. This is in contrast to the commonly used mean squared error objective function, which is a symmetric function.

Finally, the proposed target generation for the RUL training requires no assumptions about the degradation function part, nor the point at which the degradation starts. The proposed approach uses the sensor readings to estimate the health of the system, which is then mapped to the target RUL. This decreases the amount of parameter tuning required by previous works of literature.

The experiments conducted in this paper show that the proposed modifications outperform the conventional network architectures and objective functions when combined and when used alone. This means that each of the proposed approaches can be used along with other types of neural networks, as well as for training other machine learning models.

The results also indicate that the proposed modifications allow the network to make robust predictions when compared to other techniques in literature, even though it uses only small chunks of information

and assumes random initial conditions for the working machine, and it assumes that no information is provided about the operating conditions.

6 References

[1] J. Z. Sikorska, M. Hodkiewicz, and L. Ma, "Prognostic modelling options for remaining useful life estimation by industry," *Mech. Syst. Signal Process.*, vol. 25, no. 5, pp. 1803–1836, 2011.

[2] K. Javed, R. Gouriveau, and N. Zerhouni, "State of the art and taxonomy of prognostics approaches , trends of prognostics applications and open issues towards maturity at different technology readiness levels," *Mech. Syst. Signal Process.*, vol. 94, pp. 214–236, 2017.

[3] A. Hess, G. Calvello, and P. Frith, "Challenges, issues, and lessons learned chasing the 'Big P': Real predictive prognostics part 1," in *IEEE Aerospace Conference Proceedings*, 2005, pp. 3610–3619.

[4] V. Venkatasubramanian, "Prognostic and diagnostic monitoring of complex systems for product lifecycle management: Challenges and opportunities," *Comput. Chem. Eng.*, vol. 29, no. 6 SPEC. ISS., pp. 1253–1263, 2005.

[5] D. An, N. H. Kim, and J. H. Choi, "Practical options for selecting data-driven or physics-based prognostics algorithms with reviews," *Reliab. Eng. Syst. Saf.*, vol. 133, pp. 223–236, 2015.

[6] X. S. Si, W. Wang, C. H. Hu, and D. H. Zhou, "Remaining useful life estimation - A review on the statistical data driven approaches," *Eur. J. Oper. Res.*, vol. 213, no. 1, pp. 1–14, 2011.

[7] A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel, "Metrics for Offline Evaluation of Prognostic Performance," *Int. J. Progn. Heal. Manag.*, no. 1, pp. 1–20, 2010.

[8] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Poceedings of theIEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[9] Y. LeCun and Y. Bengio, "Convolution Networks for Images, Speech, and Time-Series," *Handb. brain theory neural networks*, vol. 3361, no. 10, pp. 1–5, 1995.

[10] X. Li, Q. Ding, and J. Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliab. Eng. Syst. Saf.*, vol. 172, pp. 1–11, 2018.

[11] T. C. Fu, "A review on time series data mining," *Eng. Appl. Artif. Intell.*, vol. 24, no. 1, pp. 164–181, 2011.

[12] Y. Jiang and S. Yin, "Recent results on key performance indicator oriented fault detection using the DB-KIT toolbox," in *Proceedings IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 7103–7108.

[13] Y. Jiang, S. Yin, and Y. Yang, "Comparison of KPI related fault detection algorithms using a newly developed MATLAB toolbox : DB-KIT," in *Industrial Electronics Society, IECON 2016-42nd Annual Conference of the IEEE*, 2016, pp. 7149–7154.

[14] F. Yang, M. S. Habibullah, T. Zhang, Z. Xu, P. Lim, and S. Nadarajan, "Health index-based prognostics for remaining useful life predictions in electrical machines," *IEEE Trans. Ind. Electron.*, vol. 63, no. 4, pp. 2633–2644, 2016.

[15] S. Haykin, *Neural Networks and Learning Machines*, Third., vol. 40, no. 6. 2001.

[16] F. a Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning Precise Timing with LSTM Recurrent Networks," *J. Mach. Learn. Res.*, vol. 3, no. 1, pp. 115–143, 2002.

[17] S. Hochreiter and J. Urgen Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," 2014.

[19] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, 2017.

[20] Q. V. Le, N. Jaitly, and G. E. Hinton, "A Simple Way to Initialize Recurrent Networks of Rectified Linear Units," 2015.

[21] Q. Wu, K. Ding, and B. Huang, "Approach for fault prognosis using recurrent neural network," *J. Intell. Manuf.*, pp. 1–13, 2018.

[22] 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. 0, pp. 1–13, 2017.

[23] M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury, K. Goebel, and S. Poll, "Ensemble of optimized echo state networks for remaining useful life prediction," *Neurocomputing*, vol. 281, pp. 121–138, 2017.

[24] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997.

[25] A. Graves and N. Jaitly, "Towards End-To-End Speech Recognition with Recurrent Neural Networks," *JMLR Workshop Conf. Proc.*, vol. 32, no. 1, pp. 1764–1772, 2014.

[26] F. O. Heimes, "Recurrent Neural Networks for Remaining Useful Life Estimation," in *Prognostics and Health Management, 2008. PHM 2008. International Conference on*, 2008, pp. 1–6.

[27] A. Heng, A. C. C. Tan, J. Mathew, N. Montgomery, D. Banjevic, and A. K. S. Jardine, "Intelligent condition-based prediction of machinery reliability," *Mech. Syst. Signal Process.*, vol. 23, no. 5, pp. 1600–1614, 2009.

[28] Q. Liu, M. Dong, W. Lv, X. Geng, and Y. Li, "A novel method using adaptive hidden semi-Markov model for multi-sensor monitoring equipment health prognosis," *Mech. Syst. Signal Process.*, vol. 64, pp. 217–232, 2015.

[29] K. Crammer and Y. Singer, "On The Algorithmic Implementation of Multiclass Kernel-based Vector Machines," *J. Mach. Learn. Res.*, vol. 2, pp. 265–292, 2001.

[30] A. Saxena, M. Ieee, K. Goebel, D. Simon, and N. Eklund, "Damage Propagation Modeling for Aircraft Engine Prognostics," in *Proceedings of IEEE International Conference on Prognostics and Health Management*, 2008, pp. 1–9.

[31] T. Wang, J. Yu, D. Siegel, and J. Lee, "A Similarity-Based Prognostics Approach for Engineered Systems," in *International Conference on Prognostics and Health Management*, 2008, pp. 1–6.

[32] A. M. Riad, H. K. Elminir, and H. M. Elattar, "Evaluation of neural networks in the subject of prognostics as compared to linear regression model," *Int. J. Eng. Technol.*, vol. 10, no. 06, pp. 52–58, 2010.

[33] M. Qi and G. P. Zhang, "Trend time-series modeling and forecasting with neural networks," *IEEE Trans. Neural Networks*, vol. 19, no. 5, pp. 808–816, 2008.

[34] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, 2017.

[35] A. Saxena *et al.*, "Metrics for evaluating performance of prognostic techniques," in *International Conference on Prognostics and Health Management*, 2008, pp. 1–7.

[36] M. C. Mukkamala and M. Hein, "Variants of RMSProp and Adagrad with Logarithmic Regret Bounds," 2017.

[37] E. Ramasso, "Investigating computational geometry for failure prognostics in presence of imprecise health indicator: Results and comparisons on C-MAPPS datasets," in *2nd Europen confernce of the prognostics and health management society.*, 2014, vol. 5, pp. 1–13.

[38] 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 Networks Learn. Syst.*,

vol. PP, no. 99, pp. 1–13, 2016.

[39] 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 Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2276–2285, 2017.

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

[41] G. Sateesh 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.

[42] 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.

[43] D. Galar, U. Kumar, and Y. Fuqing, "RUL prediction using moving trajectories between SVM hyper planes," in *Proceedings - Annual Reliability and Maintainability Symposium*, 2012, pp. 1–6.

[44] J. Lee, M. Ghaffari, and S. Elmeligy, "Self-maintenance and engineering immune systems: Towards smarter machines and manufacturing systems," *Annu. Rev. Control*, vol. 35, no. 1, pp. 111–122, 2011.

Author Biographies

Ahmed Elsheikh, is a Ph.D. student at the Department of Mathematical and Industrial Engineering, University of Polytechnique de Montréal. He holds a B.Sc. degree in Electronics and Communication Engineering, and a M.Sc. degree in Mathematical and Engineering Physics from Cairo University, Egypt. His research interest is focused on machine learning techniques and applications.

Soumaya Yacout, Soumaya Yacout is Professor of Industrial Engineering and Operations Research in the Department of Mathematics and Industrial Engineering at École Polytechnique de Montréal in Canada since 1999. She received a D.Sc. in Operations Research in 1985, a B.Sc. in Mechanical Engineering in 1975 and a M.Sc. in Industrial Engineering in 1979. She designed and taught courses on quality engineering, reliability and maintenance for undergraduate, graduate, and professional engineers in Canada and internationally. Her research interests include Condition Based Maintenance and optimization of decision making for product quality. She has publications in journals including Quality Engineering, International Journal of Production Research, Computers and Industrial Engineering, IEEE Transactions, Journal of Intelligent Manufacturing and papers in international conferences, some of which received the

best paper award. She is the co-editor and the co-writer of a book on minimal repair, and the book: Current Themes in Engineering Technologies. She is a senior member of the American Society for Quality ASQ and the Canadian Operations Research Society CORS.

Mohamed-Salah Ouali, Mohamed-Salah Ouali is a Professor of Industrial Engineering at the École Polytechnique de Montréal, Quebec, Canada, since 2000. His research interests focus on reliability of deteriorating system, statistical and Bayesian learning, machine learning and knowledge discovery in database, failure mode analysis and residual lifetime modeling, long-term asset Maintenance strategies and availability models, and safety of maintenance activities. He obtained his Doctorate degree from the Institut National Polytechnique de Grenoble, France, in 1996, and worked as assistant professor at Moncton University, New-Brunswich, Canada, from 1998 to 2000. He is member of the Interuniversity Research Centre on Enterprise networks, Logistics and Transport (CIRRELT) and Institue for data valorization (IVADO). Current information about publications and training of high-qualified student are available at http://www.polymtl.ca/expertises/en/ouali-mohamed-salah.

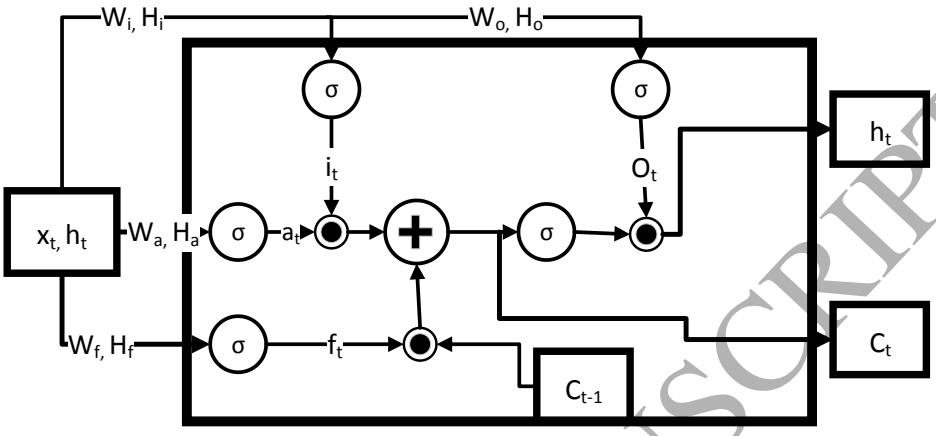

Figure 6 LSTM cell diagram.

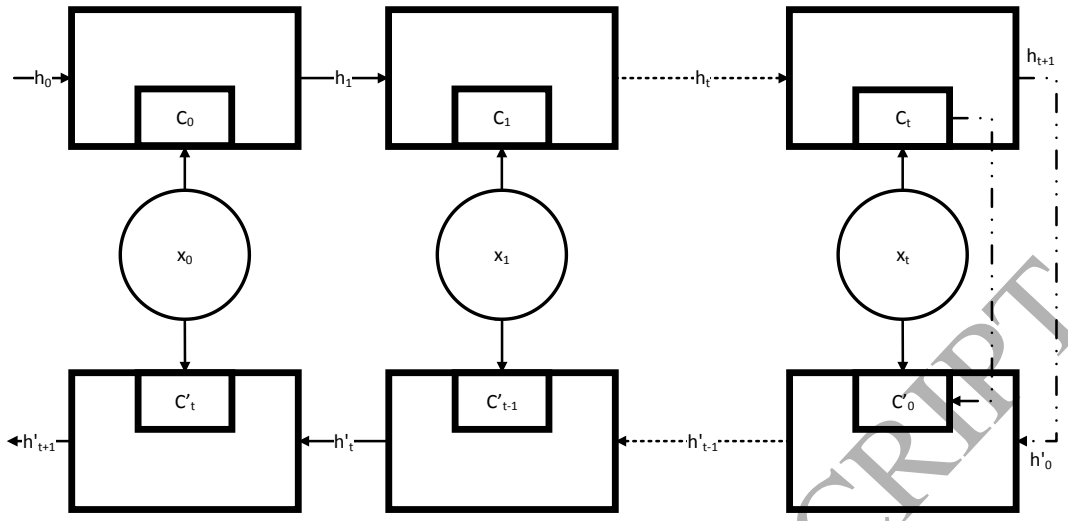

Figure 7 BHLSTM diagram for a single forward and a single backward LSTM cells.

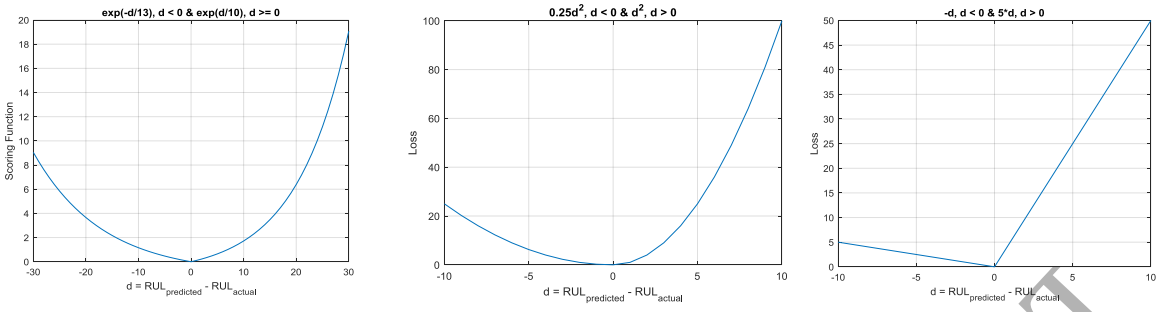

Figure 8 Comparison between the scoring function (left), the proposed asymmetric squared objective function (middle), and the asymmetric absolute objective function (right).

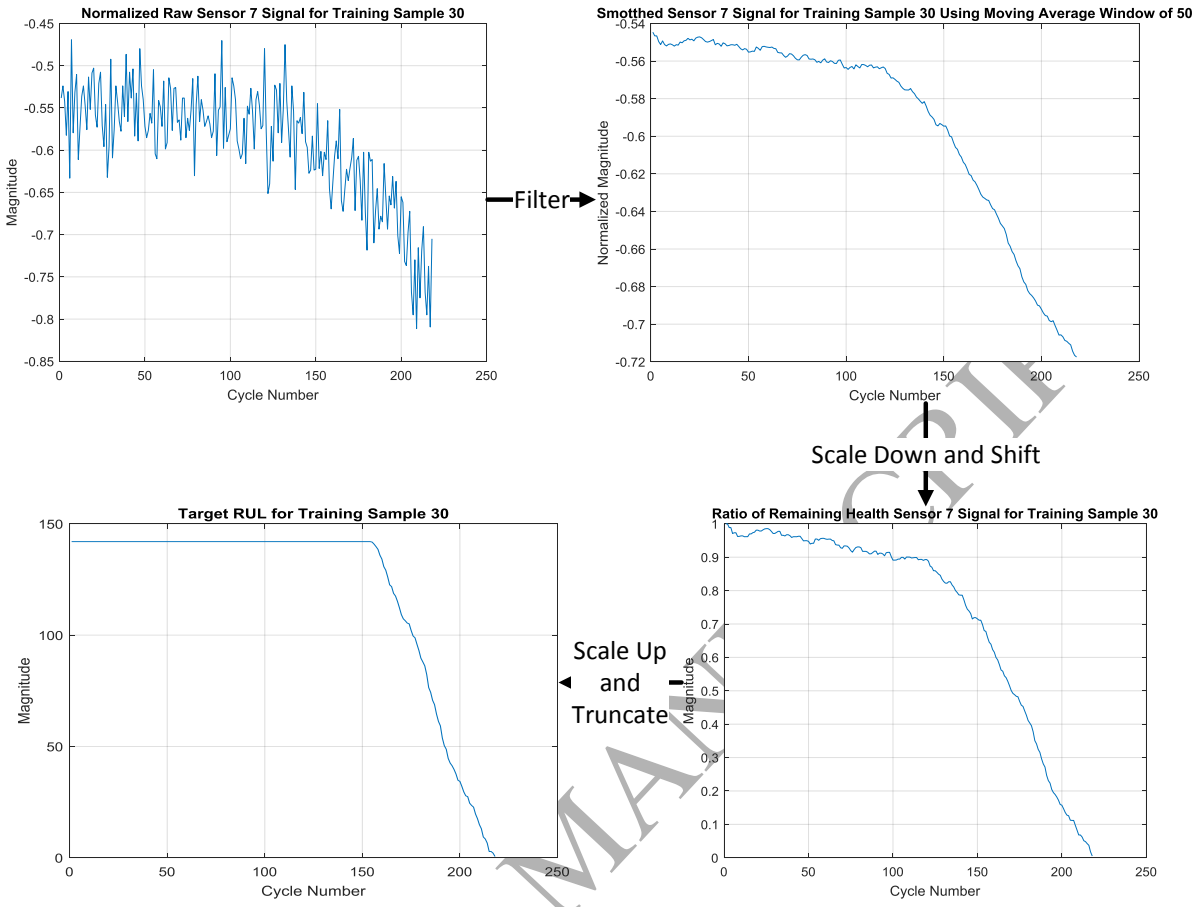

Figure 9 RUL target generation procedure.

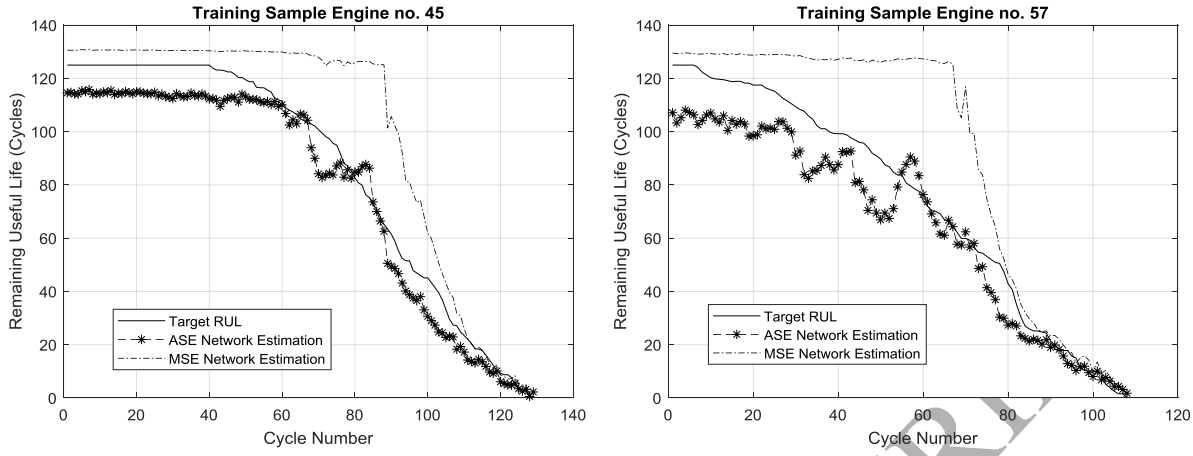

Figure 10 Comparison between RUL forecasts using ASE and MSE objective functions.