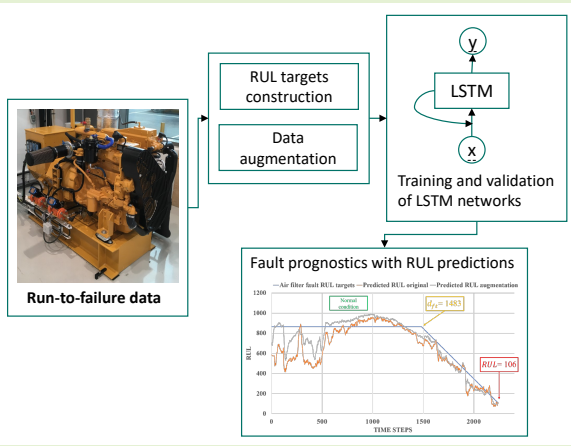
Fault Prognostics Using LSTM Networks: Application to Marine Diesel Engine

Peihua Han, *Student Member, IEEE*, André Listou Ellefsen, Guoyuan Li, *Senior Member, IEEE*, Vilmar Æsøy, and Houxiang Zhang, *Senior Member, IEEE*

***Abstract*— Maintenance is the key to ensuring the safe and efficient operation of marine vessels. Currently, reactive maintenance and preventive maintenance are two main approaches used onboard. These approaches are either cost-intensive or labor-intensive. Recently, Prognostics and Health Management has emerged as an optimal way to manage maintenance operations. In such a system, fault prognostics aims to predict the remaining useful life based on the sensor measurements. In this paper, the feasibility of applying data-driven fault prognostics to marine diesel engines is investigated. Real-world run-to-failure data of two independent fault-types in two different engine load profiles are collected from a hybrid power lab. The first profile is used for training and validation, while the second is used for testing. The LSTM networks are used to construct the fault prognostics model. Experiments and comparisons are performed to obtain the optimal structure of the networks. Results show that the proposed method generalizes well on the second profile and provides remaining useful life predictions with high accuracy.**



***Index Terms*— fault prognostics; remaining useful life; marine diesel engine; ship autonomy; prognostic and health management;**

I. Introduction

MAINTENANCE routines on ships today follow either a reactive maintenance (RM) or preventive maintenance (PvM) approach. RM can be regarded as post-failure repair, which might create large costs. PvM uses predetermined maintenance intervals, which often involves unnecessary maintenance. Recently, prognostics and health management (PHM) has emerged as a potential way to develop an ideal maintenance policy. A PHM system use sensor measurements to detect anomalies, isolate anomalous components, classify different fault-types, and predict the progression of faults [1]. Fault prognostics is the key action of a PHM system since the prognostics algorithm aims to estimate the available time before an anomalous component suffers an operational failure. Such estimations are normally referred to as the remaining useful life (RUL) and used to devise an ideal maintenance schedule.

Fault prognostics of real-world systems remains an area of active research and development nowadays [2]. Prognostics algorithms are usually divided into model-based [3] and data-driven [4] approaches. In the model-based approaches, a degradation model is formulated based on the underlying physics. Kalman filter, extended Kalman filter, or particle filter are usually used to track the degradation state through sensor measurements. In this way, the current degradation state is estimated, and the future degradation state can be calculated through the degradation model. The RUL is then the time when the degradation state reaches a threshold. Establishing an accurate degradation model is therefore of key importance in these approaches. However, such degradation models are usually unavailable for most marine systems and components. In addition, a specific marine component often involves different types of failures, leading to different degradation modes. Therefore, the use of these methods is limited for most marine components due to the unavailability of degradation models.

The data-driven approaches do not involve a physical degradation model. These methods rely on learning a mapping between sensor measurement and the remaining useful life from historical run-to-failure data. Sufficient run-to-failure

Peihua Han and André Listou Ellefsen are equal contribution.

Manuscript received xx xx, xxxx; accepted xx xx, xxxx. Date of publication xx xx, xxxx; date of current version xx xx, xxxx. *(Corresponding author: Houxiang Zhang.)*

This work was supported by the Norwegian University of Science and Technology within the Department of Ocean Operations and Civil Engineering under project no. 90329106. The authors would like to thank Digital Twins For Vessel Life Cycle Service and the Research Council of Norway, grant no. 280703.

Peihua Han, André Listou Ellefsen, Guoyuan Li, Vilmar Æsøy and Houxiang Zhang are with the Department of Ocean Operations and Civil Engineering, as part of the Mechatronics Laboratory, Norwegian University of Science and Technology, Aalesund, 6009, Norway, (e-mail: peihua.han@ntnu.no; andre.ellefsen@ntnu.no; guoyuan.li@ntnu.no; vilmar.aesoy@ntnu.no; hozh@ntnu.no).

data is then necessary for training and validation of the model. These methods can be applied to marine components if run-to-failure data can be collected. After collecting run-to-failure sensor data, the RUL targets are usually explicitly constructed using a piece-wise linear degradation model [5] or implicitly implied by health index [6]. Machine learning models are usually employed to learn this mapping. Recently, deep neural networks (DNN) [6]–[8], have emerged as extremely powerful to characterize the mapping without additional feature engineering. Convolutional neural network (CNN) and long short-term memory (LSTM) have been used for RUL prediction. For CNN, a sliding window is employed to provide continuous RUL predictions. The prediction is based on the sensor measurement value in the current window and therefore the previous value is ignored. For LSTM, the previous information is kept in its memory cell with gating manipulation. The temporal dependencies are modeled naturally and the RUL predictions are given continuously.

In this paper, the feasibility of using data-driven fault prognostics for marine diesel engine is investigated. Real operational run-to-failure data is collected from a marine diesel engine. As in [9], two replicated autonomous ferry crossing operations are used as two different engine load profiles. Run-to-failure data of two independent fault-types are collected from both profiles. The RUL targets are constructed through a data-driven labeling approach as in [4]. Data augmentation technique is used to increase the generalization of the network. For the network structure, we follow recent RUL prediction research studies and include two long-short term memory (LSTM) layers, two feed-forward neural network (FNN) layers, one dropout layer, and a fully connected output layer in the DNN structure [7], [8]. Experimental results indicates that the proposed method predicts the RUL of two different fault-types in an engine load profile with high accuracy. Key contributions of this paper include the construction of a data-driven model for fault prognostics of the marine diesel engine and the use of data sampled from the hybrid power lab for the training of the data-driven model, as well as verification of the prediction performance.

The rest of this paper is organized as follows. Section II introduces the latest studies on data-driven fault prognostics. Section III introduces the methodology. The experimental setup is described in section IV, while the experimental results and discussions are presented in section V. Finally, section VI concludes the paper and indicates objectives for further work.

II. Related works

Data-driven prognostics using DNNs has been a topic of interest in the literature. Ellefsen et al. [7] proposed a semi-supervised DNN for RUL prediction purposes. The DNN consisted of one restricted Boltzmann machine layer, two LSTM layers, one FNN layer, and an output layer. A genetic algorithm was also proposed to tune a chosen search space of hyper-parameters, including learning rate, the number of hidden units, activation functions, etc. Thus, the number of hidden layers was predetermined before any tuning was performed. Miao et al. [8] proposed a dual-task DNN for joint

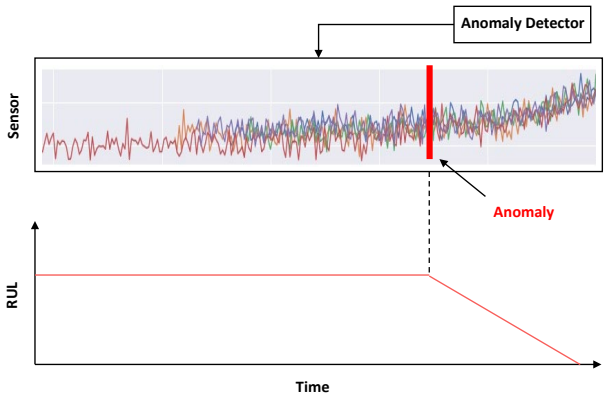


Fig. 1. Illustration of the improved piece-wise linear degradation model.

learning of degradation assessment and RUL prediction. The proposed DNN included two LSTM layers, one FNN layer as the classification sub-network, and three FNN layers as the regression sub-network. A 5-fold cross-validation procedure was performed to optimize the number of hidden units in each layer. Hence, as in [7], the number of hidden layers was predetermined in this study. Zhang et al. [6] proposed a novel health index (HI) construction method based on a deep multilayer perceptron (MLP) convolution neural network to predict the RUL of roller bearings. First, a one dimensional MLP convolutional block is applied to learn features from vibration data. Then, the learned features are mapped into HIs using a global pooling layer and a logistic regression layer to perform RUL predictions. Also, an outlier-region correction method is proposed to improve the interpretation of the established HIs significantly. In this study, the number of MLP convolutional blocks was tuned based on the experiment data.

Consequently, in [6]–[8], the trained DNNs assume that new field data will have similar distributions and complexity as the training data. When this assumption does not hold, one approach to address this issue is transfer learning. Sun et al. [10] proposed a deep network that included three transfer strategies. Weights, weights updates, and hidden features were used to transfer a sparse autoencoder (SAE) to a new domain. The SAE showed improved RUL prediction performance. Mao et al. [11] used deep feature representations from a contractive denoising autoencoder and transfer learning to show significant performance improvement for RUL predictions of roller bearings. Transfer component analysis was introduced to modify features sequentially before the RUL was predicted in the new domain.

III. Methodology

After the run-to-failure data is collected, the RUL targets can be constructed and then aligned with the sensor measurements. Then data normalization and data augmentation is performed. Finally the fault prognostics model is trained. This section first briefly introduces the RUL target construction as well as the data augmentation technique. Then the used network structure for fault prognostics of marine diesel engine is introduced.

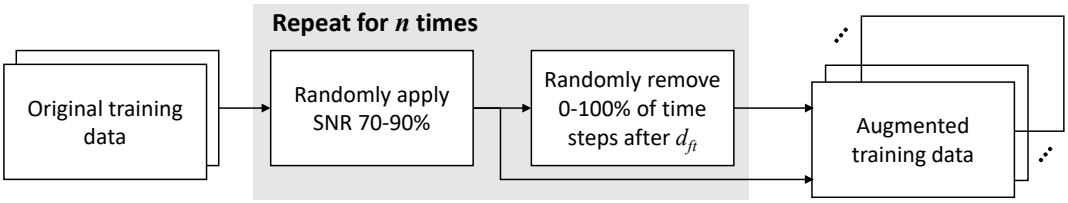


Fig. 2. Data augmentation technique for run-to-failure time-series data.

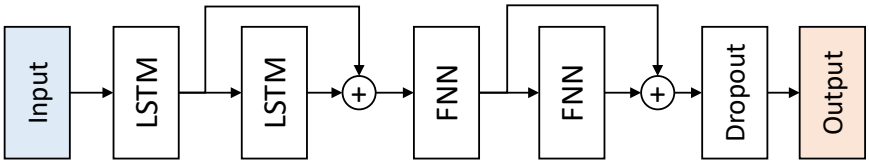


Fig. 3. Network architecture.

A. RUL targets construction

The construction of RUL targets is an essential part of data-driven fault prognostics since the model is trained in a supervised way. The piece-wise linear (PwL) degradation model is widely used to label the RUL targets. In the original PwL model, all engines utilize the same initial RUL values. Since the engines might follow different degradation pattern, an improved PwL model as presented in [4] is used.

Fig. 1 depicts the improved PwL model. An anomaly detector is used to detect the time step where the fault is initially provoked. Then the RUL targets are constructed based on the PwL model: constant RUL targets before the fault time step and linearly degraded RUL targets after the fault time step. Consequently, the last RUL target = 0. In this paper, the anomaly detector proposed in [9] is used. This anomaly detector only uses normal operating data for training, and performs well in predicting the fault time step of marine diesel engine. Note that any type of fault detection model can be used to replace the anomaly detector in this paper for the improved PwL model.

B. Data normalization and augmentation

Each sensor measurement is scaled with zero mean and unit variance (z-score) normalization:

$$\hat{x}_n = \frac{x_n - \mu}{\sigma} \tag{1}$$

where x_n is the sensor measurement, $n = 1, 2, \ldots, 47$, and μ and σ is the mean and standard deviation of that sensor measurement, respectively.

DNNs usually require extensive amount of data to train and validate the performance [12]. This fact also holds for DNNs constructed to provide RUL predictions. In real-world applications, however, run-to-failure data might be more time-consuming and difficult to acquire. In such a context, data augmentation techniques could be used to alleviate this problem. The data augmentation procedure is illustrated in Fig. 2. First, random white Gaussian noise, g, is added to each $\hat{x}_n$ in the original training set with a random signal-to-noise ratio (SNR) between 70 and 90%:

$$SNR(\%) = \frac{P_{signal}}{P_{noise}} \cdot 100 \tag{2}$$

where P_{signal} and P_{noise} are the average power of the signal and the noise, respectively, and calculated as follows:

$$P_{signal} = \frac{1}{T_t} \sum_{t=1}^{T_t} \left(\sqrt{\frac{1}{n}\left(\hat{x}_1^2 + \cdots + \hat{x}_n^2\right)} \right) \tag{3}$$

$$P_{noise} = \frac{1}{T_t} \sum_{t=1}^{T_t} \left(\sqrt{\frac{1}{n}\left((\hat{x}_1 + g)^2 + \cdots + (\hat{x}_n + g)^2\right)} \right) \tag{4}$$

where T_t is the total time step length and n is the number of input features. The resulting noisy data set will exhibit similar statistics to the original data set, but differ based on the SNR. The aim is to increase the range of statistics that the model will learn in the training procedure. Next, following [13], a random interval of time steps are removed after d_{ft} to also include some time-series that will end some time prior to failure. Thus, the model is forced to learn distributions that are more similar to a real-time PHM system, where the main goal is to accurately predict the available time before operational failure.

C. Network architectures

Following the recent RUL prediction research [7], [8], LSTMs and FNNs are used as the main building blocks in this paper. The LSTM layers are used to learn temporal and long-term dependencies within the features of degradation data. The FNN layers are then used to map all extracted features before a dropout layer is used to reduce overfitting. Dropout [14] randomly drops units of the dropout layer during training. This forces the model to learn to construct generalized representations of the input data. The last layer consists of a fully connected output layer with one unit. This

layer handles error calculations and perform RUL predictions. The mean squared error is utilized as the loss function.

The proposed network architecture can be seen in Fig. 3. Two layers of LSTM and two layers of FNN are used, with skip-connection on last layer of LSTM and FNN, respectively. The skip connections in the model use element-wise addition to combine the activations of two layers. The details of the FNN and LSTM are illustrated as follows:

1) Feed-forward neural network (FNN): An FNN was the first and most basic type of artificial neural network developed. An FNN is fully connected such that each unit in one layer has a direct weight connection to all units of the subsequent layer. Normally, FNNs learn in a supervised manner by mapping an input x to a target y. Thus, an FNN describes a mapping $y = f(x;\theta)$ and uses the back-propagation algorithm [15] to learn the parameters θ which consist of biases and weights. The output of unit k of layer l is:

$$a_k^l = \phi(z_k^l)\,, \tag{5}$$

where ϕ is a non-linear activation function and the function argument is:

$$z_k^l = b_k^l + \sum_j w_{jk}^l a_j^{l-1} \tag{6}$$

where a_j^{l-1} is the output from unit j in the previous layer $l-1$, w_{jk}^l are weight factors, and b_k^l is the bias. In the first hidden layer $l = 1$, however, the input is $a_i^0 = x_i$, where x_i, $i = 1 \ldots n$ are the inputs to the FNN.

2) long-short term memory (LSTM): LSTM is a type of recurrent neural networks (RNN). As opposed to traditional RNN, the LSTM introduces a memory cell that regulates the information flow in and out of the cell. The memory cell consists of three non-linear gating units that protect and regulate the cell state. The introduction of these gating units enable easy information flow along the entire chain, therefore, the gradient vanish problem can be eliminated and it is able to learn long term dependencies. For each element in the input sequence, the LSTM computes the following function:

$$\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \\
f_t &= \sigma(W_{if}x_i + b_{if} + W_{hf}h_{t-1} + b_{hf}) \\
g_t &= \tanh(W_{ig}x_i + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \\
o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned} \tag{7}$$

Where h_t is the hidden state at time t, c_t is the cell state at time t, x_t is the input at time t, h_{t-1} and c_{t-1} is the hidden state and cell state at time $t-1$, respectively. i_t, f_t, g_t, o_t are the input, forget, cell and output gates, respectively. σ is the sigmoid function, where $\sigma(x) = 1/(1+e^{-x})$. $\odot$ is the Hadamard product. W and b are the weights and bias in the LSTM cell.

IV. Experimental Setup

A. Data collection

A hybrid power lab, which is designed to research ship autonomy, is used to collect the data sets. The lab is established by the Department of Ocean Operations and Civil Engineering at the Norwegian University of Science and Technology in Aalesund. The lab includes a marine diesel engine, a marine battery system, and a marine automation system to control the process. To simulate load alterations in the system, the produced power is supplied back to the power grid. The diesel engine is shown in Fig. 4.

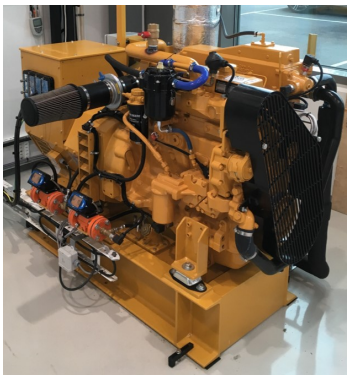


Fig. 4. The marine diesel engine included in the hybrid power lab.

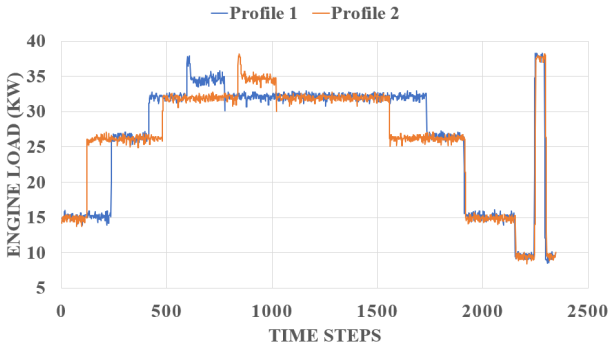


Fig. 5. Profile 1 vs. profile 2.

TABLE I
THE FOUR ORIGINAL RUN-TO-FAILURE DATA SETS COLLECTED FROM THE MARINE DIESEL ENGINE.

Data set	Profile	Usage	d_{ft}	Last RUL target	Time steps
Air filter	1	Train/val	1,660	0	2,346
Turbo	1	Train/val	1,347	0	2,346
Air filter	2	Test	1,483	106	2,240
Turbo	2	Test	1,347	490	1,856

Two engine load profiles have been used during the data collection process. As seen in Fig. 5, the profiles aim to replicate two different environmental conditions the autonomous ferry may encounter during a ferry crossing on the west coast of Norway. First, the engine runs in normal condition. Then, two typical and independent fault-types associated with the marines diesel engine were provoked gradually in both profiles to obtain degradation data. The first fault-type is clogging of the air filter, while the second fault-type is a malfunction of the turbocharger. The data collection process was terminated when the engine reached operational failure. That is, a time after the fault-types were 100% provoked and the engine loses its operational ability.

Profile 2 is subjected to different engine loads, and hence, it will be used as the test set. However, the degradation in the test

set should end sometime before failure in order to verify that the model is able to predict the RUL. Accordingly, a random interval of time steps is removed in both the air filter fault and the turbo fault in profile 2. Table I summarizes the data sets used. All data sets include 47 input features, e.g., engine load, engine speed, flow, pressure, and temperature measurements. See [9] for a detailed description of the two fault-types and see [16] for analysis of the input features.

It is worth noting that run-to-failure data sets are normally accumulated and collected through months, or perhaps even years. In this study, however, the run-to-failure data sets are collected more rapidly due to time constraints. Even though the collected run-to-failure data sets only consist of 2,346 time steps, the real degradation patterns are assumed to remain. One time step equals 0.5 seconds.

B. Performance indicators

In this case study, the root mean square error (RMSE) is used as one of the performance indicators. The training procedure can be considered as a regression task since the goal is to train the model to predict the constructed RUL targets with high accuracy at each time step. The RMSE is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{8}$$

where n is the total number of constructed RUL targets and $d_i = RUL_{predicted,i} - RUL_{target,i}$. Additionally, the maintenance scoring function (SCORE), provided in [17], is used as a performance indicator for the last RUL targets in the test set:

$$SCORE = \begin{cases} \sum_{j=1}^{m} e^{-(\frac{d_j}{13})} - 1, & \text{for} \quad d_j < 0 \\ \sum_{j=1}^{m} e^{(\frac{d_j}{10})} - 1, & \text{for} \quad d_j \geq 0 \end{cases} \tag{9}$$

where m is the total number of last RUL targets in the test set and $d_j = RUL_{predicted,j} - RUL_{target\,last,j}$. In SCORE, the punishment for late RUL predictions, when $d_j > 0$, is greater than early RUL predictions, when $d_j < 0$. This indication is especially suited for the model as late RUL predictions could lead to a potential disaster for autonomous ferries. Late RUL predictions are vulnerable to engine failure since maintenance operations would be scheduled too late. Early RUL predictions, however, pose less risk of engine failure as maintenance operations would be scheduled too early. Additionally, SCORE is a suitable metric for real-time predictions since it only involves the latest input data and corresponding RUL prediction.

The main objective for both performance indicators is to reach the lowest value possible, namely, when $d_i = 0$ and $d_j = 0$. The RMSE and the SCORE are illustrated in Fig. 6.

C. Network configuration and training

All experiments use Microsoft Windows 10, Java 8, DL4J version 1.0.0-beta4 [18] as the DL library, and NVIDIA GeForce GTX 1060 6 GB as the GPU.

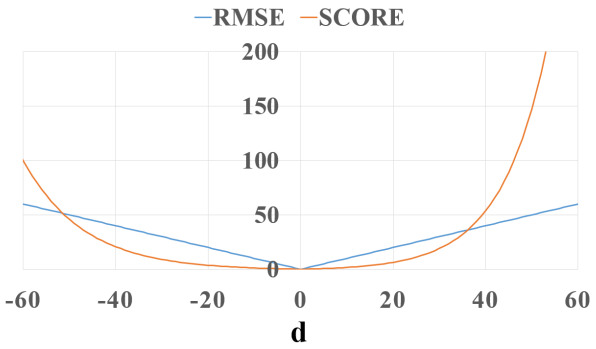


Fig. 6. RMSE vs. SCORE

TABLE II
HYPER-PARAMETERS.

Hyper-parameter	Method/Value/Values
Activation function in FNN layers	ReLU
Activation function in LSTM layers	tanh
Dropout	0.5
Hidden units in all hidden layers	64, 128, 192
Learning rate first LSTM layer	$5 \cdot 10^{-4}$
Learning rate remaining layers	$1 \cdot 10^{-4}$
$l2$ regularization	$1 \cdot 10^{-5}$
Mini-batch size	5
Optimizer	Adam
Weight initialization	Xavier

DNNs introduce a big search space of hyper-parameters, which can be laborious to optimize in the training process. Recent RUL prediction research studies [4], [7], [8] can be used as a guide to determine which hyper-parameters to include and which to omit. The rectified linear unit (ReLU) activation function [19] is used in FNN layers and the tanh activation function is used in LSTM layers. Additionally, to better maintain low-level degradation features, the learning rate in the first LSTM layer is a half order of magnitude higher than the learning rate in the remaining layers. The l_2 regularization coefficient is $1 \cdot 10^{-5}$. As recommended in [14], the dropout retaining probability is 0.5. In DL4J, time-series have three dimensions: [numExamples, inputSize, timeSeriesLength], where numExamples is the number of time-series included in a mini-batch, inputSize is the number of input features, and timeSeriesLength is the total time step length in a mini-batch. The mini-batch size is selected to be five run-to-failure data sets for all augmented data set scenarios, except the first scenario. There are only two run-to-failure data sets in the original training data, so no mini-batch is used in the first scenario. As in [4], [7], [8], stochastic gradient descent (SGD) is used as the optimization algorithm together with Adam as the learning rate method [20]. Xavier weight initialization [21] is used in all layers. To ensure the training results are reproducible, the seed is fixed to a value in all experiments. The hyper-parameters are given in Table II.

For the used network, the most important hyper-parameter is considered to be the total number of parameters, which relates to the number of hidden units in each layer. Thus, the number of hidden units will be tuned through a 5-fold cross-validation procedure for each augmented data set scenario. Each scenario is divided into 80% training and 20% cross-

validation, randomly. To prevent overfitting, early stopping is used to monitor the RMSE performance on the cross-validation set for each fold. If the number of epochs with no reduction on the RMSE gets greater than four, the training process is aborted. Then, the model, in the epoch with the lowest RMSE on the cross-validation set, is saved. In the end, the average cross-validation RMSE of all folds for each augmented data set scenario is calculated.

V. Performance Evaluation

In this section, the average RMSE performance for all augmented data set scenarios was compared, with different numbers of total parameters. Then, the performance of the model on the test set is evaluated with different variants. The proposed data augmentation technique is repeated for 0, 10, 20, 30, 40, and 50 times for each fault-type in the training set. This results in six different scenarios of 0, 20, 40, 60, 80, and 100 augmented training data sets.

A. Cross validation results

The cross validation results are evaluated with different number of hidden units. The optimal number of hidden units can be found with the least cross validation error. The first scenario includes zero augmented data sets, and hence, the model is only trained and validated on the original training set. The rest scenarios use 20, 40, 60, 80, 100 augmented dataset, respectively.

As present in Table III and Fig. 7, the model with 64 hidden units in all hidden layers includes too few parameters to model the augmented data set scenarios. Instead, it provides the lowest cross-validation RMSE on the original training set. When it comes to the model with 192 hidden units, the total number of parameters is increased. This number of parameters is clearly too big compared to the number of examples in the original training set. That said, it provides a major decrease in cross-validation RMSE in all augmented data set scenarios. As seen in Table III, the model with 128 hidden units provides the lowest average RMSE for all augmented data sets scenarios. Therefore, this model is considered as the most robust configuration and will be further used in the test on profile 2.

The training time is also taken into consideration. As shown in Table III, the average training time per epoch, which is stated in seconds, is very similar between all three configurations of the model. As a consequence, the comparison in real-time test neglects the training time.

B. Real-time test

The marine diesel engine is prone to various operating conditions. The profile 2 is used as the test set to evaluate the generalization of the model. In profile 2, two different fault types, air filter fault and turbo fault, are considered.

1) Variants comparison: In this part, 4 variants of the original model is used as the baselines. The variants are implemented with different number of LSTM layers and FNN layers following the same architectures as Fig. 3. For

TABLE III
THE AVERAGE RMSE OF CROSS-VALIDATION PROCEDURE FOR EACH AUGMENTED DATA SET SCENARIO.

Number of hidden units	Total params	Augmented data sets	RMSE train	RMSE cross	Avg. epoch time
64	73,985	0	29.47	29.47	0.64
		20	31.45	34.64	2.86
		40	33.37	37.20	5.28
		60	34.84	33.58	7.59
		80	34.52	31.97	10.21
		100	41.84	31.01	12.53
		Avg.	34.25	32.98	**6.52**
128	270,849	0	31.26	31.26	0.69
		20	26.75	28.83	2.94
		40	29.46	29.13	5.48
		60	31.66	27.22	7.85
		80	34.42	27.70	10.66
		100	39.19	31.55	13.02
		Avg.	**32.12**	**29.28**	6.77
192	590,593	0	57.10	57.10	0.75
		20	30.00	30.20	3.11
		40	35.63	27.73	5.92
		60	33.81	31.42	8.54
		80	52.31	32.85	11.62
		100	43.70	37.10	14.32
		Avg.	42.09	36.07	7.38

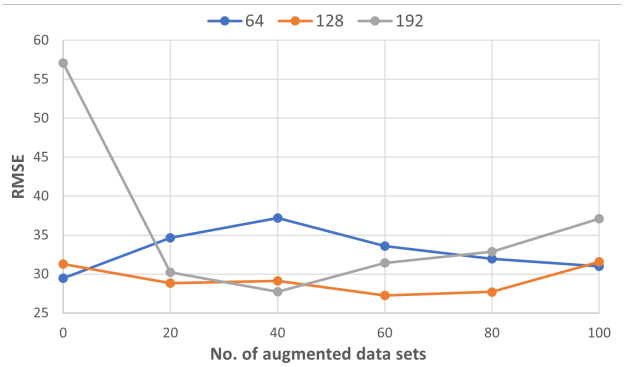


Fig. 7. The average cross-validation RMSE accuracy of all folds for each augmented data set scenario.

abbreviation, the notation "LmFm"" is used to represent the different variants, where m is the number of layers. For instance, L1F1 represents the network with 1 layer of LSTM and 1 layer of FNN, L1F2 represents the network with 1 layer of LSTM and 2 layer of FNN. The L2F2 model with skip connection is then denoted as L2F2-SC, which is the proposed model in Fig. 3. The trained model and its different variants in each augmented data set scenario are employed to predict the RUL at each time step on the test set. The evaliation test can be considered as a real-time test because this is how the networks would potentially be employed in a real-life data-driven PHM system. Both RMSE and the SCORE are used as the performance indicators.

Table IV and Fig. 8 shows the average RMSE and SCORE of the model in each augmented data set scenario. The L2F2-SC provides the lowest RMSE for each augmented data set scenario. As expected, L1F1 provides the worst overall RMSE due to having the lowest number of parameters. L2F2 provides worse overall RMSE compared to the L2F2-SC, even though L2F2 has the same number of parameters, which reveals that including skip connection is beneficial in this study. The

TABLE IV
THE AVERAGE RMSE AND SCORE PERFORMANCE ON THE TEST SET.

Network	Total params	Augmented data sets	RMSE	SCORE
L1F1	122,753	0	315.95	8,192.14
		20	285.09	67.63
		40	245.09	218.73
		60	254.61	731.32
		80	279.81	600.80
		100	256.86	226.77
L1F2	139,265	0	213.26	1,796.67
		20	194.51	18,086.95
		40	207.36	5,444.71
		60	194.55	3,023.56
		80	189.71	6,939.86
		100	181.01	177.45
L2F1	254,337	0	139.86	5.1
		20	122.14	320.60
		40	132.82	2,845.32
		60	156.84	69.38
		80	145.94	72.31
		100	135.03	88.35
L2F2	270,849	0	182.37	420.17
		20	218.03	34.75
		40	197.05	201.29
		60	177.41	16.90
		80	170.60	205.12
		100	158.41	66.65
L2F2-SC	270,849	0	137.89	8.59
		20	**85.76**	**9.92**
		40	106.11	17.23
		60	72.98	70.28
		80	91.85	59.50
		100	97.86	397.26

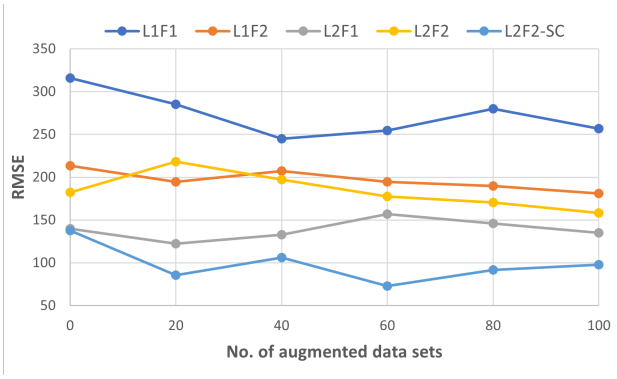


Fig. 8. The average RMSE on the test set for each augmented data set scenario.

TABLE V
COMPARISON WITH DIFFERENT APPROACHES.

Approach	RMSE	SCORE
CNN+FNN [22]	96.26	40.32
LSTM+FNN [23]	285.09	67.63
RBM+LSTM+FNN [7]	120.54	128.70
LSTM+FNN+SC	**85.76**	**9.92**

SCORE is also important to consider in real-life data-driven PHM systems suitable for autonomous ferries. A reliable and low SCORE performance close to the end of the marine diesel engine's lifetime has great significance, as this period is critical in order to schedule maintenance operations. The L1F1 without augmented dataset gives the lowest SCORE and relatively higher RMSE, which suggests that the model tends to provide early RUL prediction with a certain degree of error. The L2F2-SC provides satisfactory SCORE performance

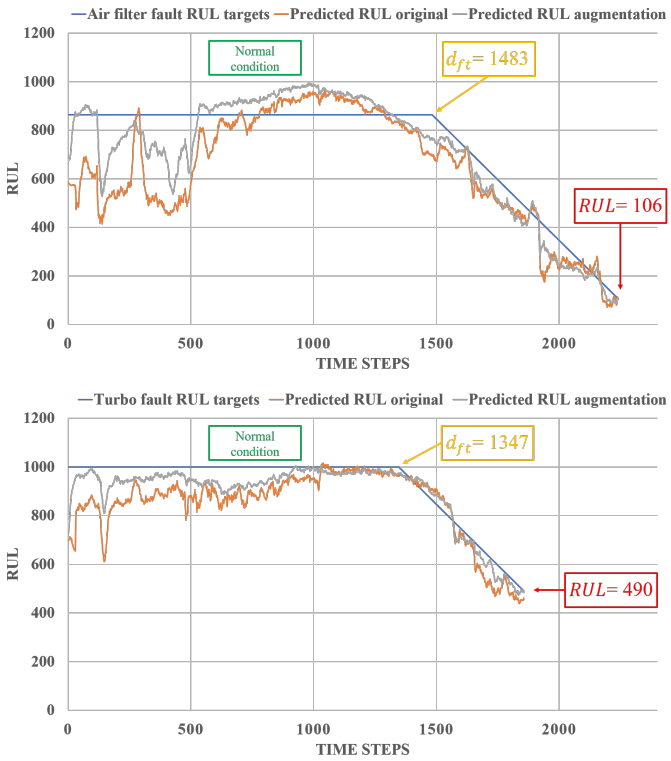


Fig. 9. The prediction results on the air filter fault and the turbo fault in the test set when the model is trained on 20 augmented data sets and the original training set.

on the test set when trained on 0, 20, and 40 augmented data sets. Besides, the L2F2-SC provides the lowest RMSE on the test set when trained on 20 and 60 augmented data sets and it is more stable in terms of SCORE with different augmented datasets. Among them, the L2F2-SC provides the best overall RUL performance on the test set when trained on 20 augmented data sets.

2) Baseline comparison: The model used in this paper is compared with the model proposed in the literature to further validate the approach. Table V presents the RMSE and SCORE with different approaches. For a fair comparison, the same RUL targets construction procedure is used. All the approaches are tested when trained on 20 augmented data sets. For the approach with CNN, a sliding window is necessary. The window size from 20 to 80 is tested, Table V provides the best results within the window size. It is shown that the model used in this paper provides the smallest RMSE and SCORE when compared with other approaches.

3) Effect of data augmentation: Fig. 9 compares the RUL predictions on both the air filter fault and the turbo fault in the test set when the L2F2-SC is trained on 20 augmented data sets and the original training set. It increases the generalization power of the model. The RUL predictions are more noisy when no augmented data is used. In addition, when L2F2-SC is trained on the augmented data set, the RUL prediction is close to the RUL target not only at the end of the engine's life, but also at the early stage of degradation. Taking into account the RMSE and SCORE on the test set in Table V, L2F2-SC trained on the 20 augmented data sets provides better overall RUL

performance than training on the original data set. It proves the advantage of using the data augmentation technique.

VI. Conclusion and future work

In this paper, a LSTM network is used for fault prognostics of the marine diesel engine. Run-to-failure data of two fault-types in two different engine load profiles is collected. Data in profile 1 is used for training and validation, while data in profile 2 is used for real-time testing. Data augmentation technique is used to augment the data in profile 1. The detailed network architectures are discussed and the optimal network architecture is obtained. The advantage of using data augmentation technique is also shown. Experimental results show that the model provides accurate remaining useful life predictions for two different fault types in the profile 2, which suggests that the proposed model has high generalization power towards different engine load profiles.

For real-life fault prognostics system, confidence bounds in the remaining useful life should be included. Maintenance decisions should be anchored in a remaining useful life with confidence bounds rather than a particular remaining useful life values. Future works will focus on data-driven remaining useful predictions with confidence bounds, as well as the effectiveness of different data augmentation methods and deploying the system to real-world operations.

References

[1] P. W. Kalgren, C. S. Byington, M. J. Roemer, and M. J. Watson, "Defining phm, a lexical evolution of maintenance and logistics," in *2006 IEEE Autotestcon*, Sept 2006, pp. 353–358.

[2] O. Bektas, J. A. Jones, S. Sankararaman, I. Roychoudhury, and K. Goebel, "A neural network filtering approach for similarity-based remaining useful life estimation," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 1-4, pp. 87–103, 2019.

[3] Y. Lei, N. Li, and J. Lin, "A new method based on stochastic process models for machine remaining useful life prediction," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 12, pp. 2671–2684, 2016.

[4] A. L. Ellefsen, S. Ushakov, V. Æsøy, and H. Zhang, "Validation of data-driven labeling approaches using a novel deep network structure for remaining useful life predictions," *IEEE Access*, vol. 7, pp. 71 563–71 575, 2019.

[5] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–6.

[6] D. Zhang, E. Stewart, J. Ye, M. Entezami, and C. Roberts, "Roller bearing degradation assessment based on a deep mlp convolution neural network considering outlier regions," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 6, pp. 2996–3004, 2020.

[7] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliability Engineering & System Safety*, vol. 183, pp. 240 – 251, 2019.

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

[9] A. L. Ellefsen, P. Han, X. Cheng, F. T. Holmeset, V. Æsøy, and H. Zhang, "Online fault detection in autonomous ferries: Using fault-type independent spectral anomaly detection," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 8216–8225, 2020.

[10] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, "Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2416–2425, April 2019.

[11] W. Mao, J. He, and M. J. Zuo, "Predicting remaining useful life of rolling bearings based on deep feature representation and transfer learning," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 4, pp. 1594–1608, 2020.

[12] N. Jones, "Computer science: The learning machines," *Nature*, vol. 505, pp. 146–148, 2014.

[13] L. Jayasinghe, T. Samarasinghe, C. Yuen, J. C. N. Low, and S. S. Ge, "Temporal convolutional memory networks for remaining useful life estimation of industrial machinery," *arXiv preprint arXiv:1810.05644*, 2018.

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[15] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

[16] X. Cheng, A. L. Ellefsen, F. T. Holmeset, G. Li, H. Zhang, and S. Chen, "A step-wise feature selection scheme for a prognostics and health management system in autonomous ferry crossing operation," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, Aug 2019, pp. 1877–1882.

[17] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 International Conference on Prognostics and Health Management*. IEEE, Oct 2008, pp. 1–9.

[18] "Eclipse deeplearning4j development team, deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Software Foundation License 2.0, http://deeplearning4j.org*, 2020.

[19] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, vol. 15, 2011, pp. 315–323.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[22] 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.

[23] 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)*. IEEE, 2017, pp. 88–95.