Predicting Remaining Useful Life with Bidirectional Long-Short Term Memory Network

Anonymous Authors

***Abstract*—Asset failures are hard to foresee and can incur significant losses in the manufacturing industry. The present work explores the application of a Bidirectional Long Short-Term Memory network to predict the Remaining Useful Life, in the context of predictive maintenance. A successful predictive maintenance strategy can not only reduce breakdowns and downtime of assets, but also maximize production, improve product quality, and safety of workers. Therefore, the aim of this work is to propose a framework with the goal of accurately predicting the Remaining Useful Life of assets from monitoring sensing data. The model is applied to the N-CMAPSS dataset, which contains full run-to-failure curves of turbofan engines, published by the Prognostics Center of Excellence at NASA Ames Research Center, in 2021. The proposed Bidirectional Long-Short Term Memory model is compared against three benchmark models using Random Forest, Extreme Gradient Boosted Trees and Long-Short Term Memory Network. Simulation results with this novel dataset indicate that the implementation of the Bidirectional Long-Short Term Memory, with the proper fine-tuning and feature selection, outperforms the other models, being a promising solution for providing critical information to support decision making for predictive maintenance strategies.**

***Index Terms*—Remaining Useful Life Prediction, Predictive Maintenance, Machine Learning, Bidirectional Long Short-Term Memory, N-CMAPSS**

I. Introduction

In the manufacturing industry, an asset failure can be described as an asset performing its intended function defectively, for example, a product is manufactured by the asset with quality standards below those predefined, or the asset completely halting its operation, known as breakdown. The latter of the situations has more severe consequences, and both require maintenance actions to be taken. The consequences of a failure can be low. However, in safety-critical systems, such as the aircraft industry, failures can have extremely high economic impacts and even cause loss of lives. To avoid failures altogether, effective maintenance of assets is required [1].

Maintenance is the management and repair of assets and the control of costs, however asset monitoring and management are arguably complex tasks. Maintenance costs represent a significant part of all the costs associated with manufacturing and production. Considering different industries, maintenance costs can account for 15% up to 70% of the total production costs [2]. Additionally, maintenance, often times, can be an ineffective process, with up to one third of all investment being lost in incorrect maintenance actions. The main cause for this is the lack of knowledge regarding the actual health condition of assets and the need for maintenance [1]. Depending on statistical data from manufacturers to predict failures will lead to ineffective maintenance and waiting for an asset to breakdown is also not an option in safety-critical situations [3].

In this context, Predictive Maintenance (PdM) is an essential maintenance method that, through the monitoring of the actual operating condition of an asset and other important indicators, is able to predict an impending failure and allow enough time for maintenance actions to be taken, preventing the failure [3]. The main advantages of this maintenance method are the increased reliability and availability of assets, improved environmental and worker safety, and reduced costs in parts inventory and maintenance labor.

PdM combined with the recent technological advancements in Internet of Things (IoT), specifically sensor technology, and in Artificial Intelligence (AI) has converged in new predictive maintenance techniques through the use of Machine Learning (ML) algorithms [4]–[10]. Therefore, the aim of this work is to propose a framework, which, through the use ML algorithms, achieves accurate Remaining Useful Life (RUL) predictions of airplane's turbofan engines from the N-CMAPSS dataset, published by the Prognostics Data Repository at NASA Ames Research Center, in 2021. In this context, a Bidirectional Long-Short Term Memory (BiLSTM) model is proposed and then the performance is compared with three other ML models using Random Forest (RF), Extreme Gradient Boosted Trees (XGBoost) and Long-Short-Term-Memory (LSTM). The simulation results highlight the benefits of using the BiLSTM model in terms of Root Mean Square Error (RMSE) and NASA score (proposed in [11]) metrics, achieving a best RMSE of 4.46 cycles and a NASA score of 1956. This accuracy is possible after the fine-tuning of model parameters and feature selection applied. This represents promising results in providing information to support predictive maintenance strategies.

II. Proposed Framework

A. Proposed Machine Learning models

For this work, we propose optimizing the BiLSTM model, due to its forecasting capabilities. The RF, XGBoost and LSTM models will serve to obtain baseline results, from which we can then compare with the results obtained from the BiLSTM model. The RF and the XGBoost models were chosen due to the ease of use of these models as well as their good performance in both RUL prediction and computation time. The LSTM neural network is used as it often shows better performance than the RF and the XGBoost models, when handling time-series data. The LSTM model parameters are

chosen based on previous relevant literature [10]. Therefore, the LSTM model has one input layer and one hidden layer with 64 neurons each, the output layer is a linear regression layer with one neuron. The remaining parameters are a dropout of 0.5, a learning rate of 0.01 with exponential decay, a batch size of 512, the optimizer function used is the RMSprop function, an early stopping technique with a patience of 10 epochs, a training/validation split of $80\%/20\%$, respectively, and a sliding time window size of 50.

Finally, we propose a BiLSTM network, due to its ability to capture information from past and future information in the dataset. We expect results to outperform the benchmark models after appropriate parameter and feature selection. The parameter optimization process will follow the framework seen in Fig. 1. The BiLSTM model proposed has one input layer and two hidden layer with 256, 256 and 64 neurons each, the output layer is a linear regression layer with one neuron. The remaining parameters are a dropout of 0.5, a learning rate of 0.001, batch size of 512, the optimizer function used is the RMSprop function, an early stopping technique with a patience of 10 epochs, a training/validation split of $80\%/20\%$, respectively, and a sliding time window size of 50. As a drawback to using the BiLSTM model training and testing takes a considerably longer time, which limits the optimization process. To counter this limitation the parameter optimization process used is a manual process, this is less time consuming than an automatic method, but is also less precise and does not guarantee that a local optimal solution is found. More important is the use of cloud services that is employed, through the Kaggle platform[1] for the training and testing of all the models. This platform is intuitive to use and provides a graphics processing unit (GPU) necessary for the intensive computations performed in training and testing complex models, such as the BiLSTM model, which reduced the computation time considerably. Each training and testing of the BiLSTM model took approximately . Without this feature, achieving these results would not have been possible, due to time constraints.

One of the limitations of ML prediction algorithms is the generalization and transferability to new, unseen data. To measure this, the BiLSTM model was fine-tuned, trained and tested with a subset of the full dataset available. Afterwards the remaining subsets of data are trained and evaluated with the BiLSTM model obtained to further compare results.

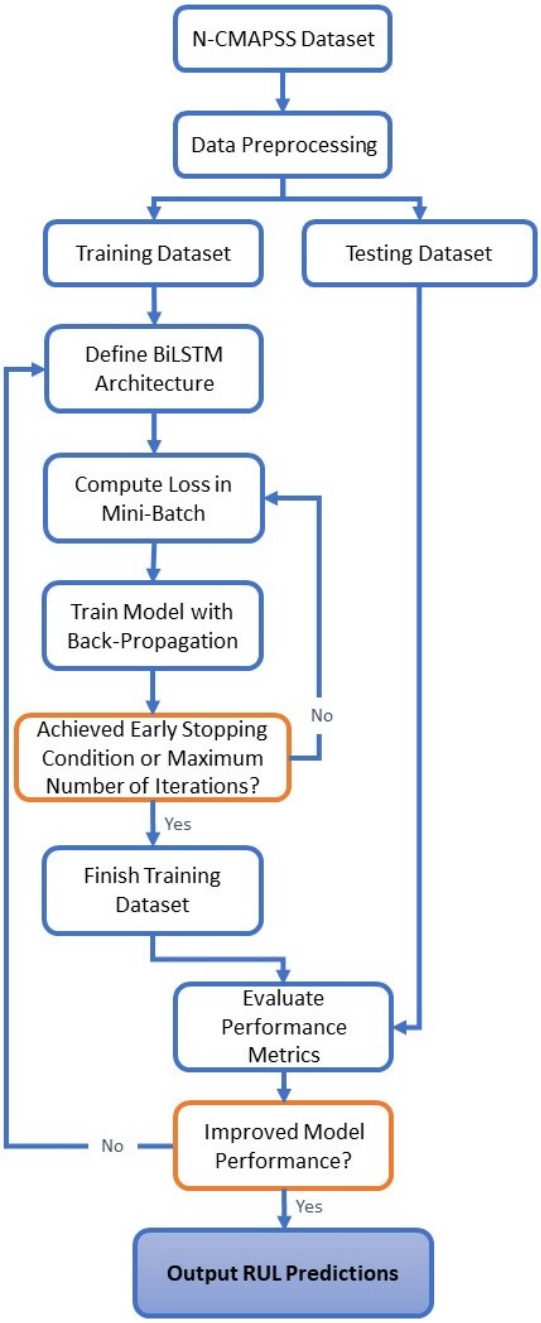

Fig. 1. Framework applied to achieve RUL predictions

B. Evaluation metrics

The evaluation metrics used are the ones suggested in [12]. This is the RMSE and NASA's scoring function, denominated $Score$. These metrics have been used extensively by researchers in the context of RUL prediction and make it easier for comparison of results between models. The formula for each performance metric can be seen in (1), (2), and (3).

$$\Delta^{(j)} = y_j - \hat{y}_j \tag{1}$$

$$RMSE = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(\Delta^{(j)})^2} \tag{2}$$

$$Score = \sum_{j=1}^{m} \exp\left(\alpha \left|\Delta^{(j)}\right|\right) \tag{3}$$

$$\alpha = \begin{cases} \frac{1}{13} & \text{if} \quad y_j - \hat{y}_j \leqslant 0 \\ \frac{1}{10} & \text{if} \quad y_j - \hat{y}_j > 0 \end{cases} \tag{4}$$

[1]https://www.kaggle.com/

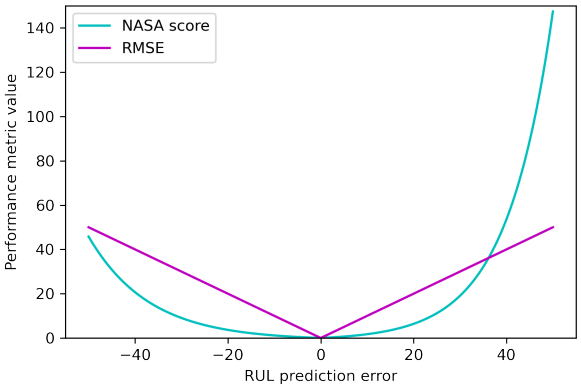

Fig. 2. Behaviour of the performance metrics used in evaluation

For (1) and (4) y_j is the output of the model of test sample j, and $\hat{y}_j$ is the real value of the target data of test sample j. In (2) and (3), m is the total number of data samples, α has two different values whether the model prediction is under estimated or over-estimated, these values are recommended in [12]. The reasoning for these values is that, in maintenance, an under-estimation of RUL would lead to sub-optimal maintenance, but an over-estimation would lead to asset failure, which is a situation much more costly than the former. The RMSE is a performance metric that has been commonly used in RUL estimation research and accurately represents the overall error of estimation.

The behaviour of both performance metrics are represented in Fig. 2, which clearly shows the symmetric and asymmetric behaviour of each metric.

C. Model parameter selection

During the parameter optimization, the parameter being tested changes while the remaining parameters are fixed. At each iteration the loss function and the RUL prediction graphs are analysed to help guide the choice of parameters. The loss function chosen is the Mean Squared Error function, a common function for regression problems. While this does not guarantee a local optimal solution, this process is not too time consuming and improves the performance of the model.

The parameters tested include the layer topology, the Dropout value, the Early stopping value, the mini batch size, the learning rate, the training/validation split and the sliding time window size.

D. Feature selection and analysis

1) Data preprocessing: The preprocessing of data is an important step to improve the quality of the data, therefore, before training the models the dataset is analysed to find features that have missing values and features that have only constant values. The features with these characteristics are then removed from the dataset.

2) Feature analysis: Feature analysis was performed after the parameter selection on the proposed BiLSTM model, and is based on the results of two different techniques. The Pearson correlation coefficients [13] and the variable importance analysis from the RF model [14]. The performance of the models is evaluated for three different feature filters. One model has all the features, the second model has features based on the Pearson correlation coefficients and the last model has features based on the variable importance analysis from the RF base model. These feature filters are applied to reduce the number of features used, with the aim of improving computation time without hurting performance or even improve the performance of the model.

To select features based on the Pearson correlation coefficients, features with a correlation coefficient lower than 0.01 with the target feature RUL are removed. Also, from features with a correlation coefficient higher than 0.95 between each other, only one of such features is kept. To select features based on the variable importance analysis, one third of the total number of features are removed, from those with lowest variable importance value. The results are then compared to determine the effects on performance based on the chosen techniques.

III. Dataset description and analysis

The dataset chosen is the Turbofan Engine Degradation Data Set-2 obtained from the Prognostics Data Repository[2], published in early 2021 by the PCoE at NASA Ames.

The main advantages in using this dataset is not only its quality, but its quantity as well. The N-CMAPSS dataset provides full run-to-failure trajectories of a fleet of turbofan engines under real conditions. Having the time-to-failure is very important and not typically available from real life applications. This is both due to the rarity in failures occurring from excessive maintenance, as well as from the inherent sensitive nature of failures that inhibit companies from publicly sharing their assets' data [11].

A. Data description

The N-CMAPSS dataset is divided into 10 sub datasets with differences between each other, mainly in regards to the number of engines and the failure modes in each sub dataset. Each data file is divided into a development dataset and a test dataset. This is the splitting of data used when training and testing the models. Both datasets contain 6 types of variables: , the operative conditions, the measured sensors signals, the virtual sensors, the engine health parameters, and the RUL label. From these variables, the auxiliary data, the operative conditions, the measured sensor signals, and the virtual sensors compose the total number of features used in the models, amounting to a total of 36 features, and the RUL label is the target output data. We denote x^n as a vector of all the features, n being the number of features with a range of values of [1,36].

Tab. I details all the features present in the data, these include the categorical data from the auxiliary data variables

[2]https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository

and monitoring measurements from the the operative conditions, the measured sensor signals, and the virtual sensors variables. In these features are physical properties such as flow, speed, temperature and pressure at different points in the turbofan engine. Note that LPC, HPC, HPT, LPT stand for low pressure compressor, high pressure compressor, high pressure turbine and low pressure turbine, respectively. These 36 features simulate the condition monitoring measurements that real life applications perform.

TABLE I
FEATURES - x^1 TO x^{36}

Id	Description	Unit
x^1	Unit number	-
x^2	Flight cycle number	-
x^3	Flight class	-
x^4	health state	-
x^5	Altitude	ft
x^6	Flight Mach Number	-
x^7	Throttle-resolver angle	%
x^8	Total temperature at fan inlet	°R
x^9	Fuel flow	pps
x^{10}	Physical fan speed	rpm
x^{11}	Physical core speed	rpm
x^{12}	Total temperature at LPC outlet	°R
x^{13}	Total temperature at HPC outlet	°R
x^{14}	Total temperature at HPT outlet	°R
x^{15}	Total temperature at LPT outlet	°R
x^{16}	Total pressure in bypass-product	psia
x^{17}	Total pressure at fan inlet	psia
x^{18}	Total pressure at fan outlet	psia
x^{19}	Total pressure at LPC outlet	psia
x^{20}	Static pressure at HPC outlet	psia
x^{21}	Total pressure at burner outlet	psia
x^{22}	Total pressure at LPT outlet	psia
x^{23}	Total temperature at burner outlet	°R
x^{24}	Total pressure at HPC outlet	psia
x^{25}	Total pressure at HPT outlet	psia
x^{26}	Fan flow	lbm/s
x^{27}	Flow out of LPC	lbm/s
x^{28}	Flow into HPC	lbm/s
x^{29}	HPT coolant bleed	lbm/s
x^{30}	HPT coolant bleed	lbm/s
x^{31}	Flow out of HPT	lbm/s
x^{32}	Flow out of LPT	lbm/s
x^{33}	Fan stall margin	-
x^{34}	LPC stall margin	-
x^{35}	HPC stall margin	-
x^{36}	Ratio of fuel flow to x^{20}	pps/psi

B. Data sampling

The raw data is sampled in seconds. This means that the model has a very fine degree of information for training. However, a common problem, specially with complex models, is that the bigger the size of the data file, the longer it takes to train and test the models. Observing our specific dataset and the behaviour of the features over time, we can conclude that increasing the sampling size of the data is beneficial to reduce the size of data while still capturing the features of the data. For this purpose, a sampling size of ten minutes was chosen.

1) Feature normalization: Normalization is a transformation of data from its original range of values to a specific desired range of values. The method applied in this step is the min-max normalization. This method re-scales the range of the data to the range of values of [0, 1], the formula is shown in (5) . Min-max normalization was chosen as it is a simple method of feature scaling that has been previously used to great success in the literature [15] [7].

$$x_i^{n\prime} = \frac{x_i^n - \min(x^n)}{\max(x^n) - \min(x^n)} \tag{5}$$

Where x_i is the i-th value of a given feature x^n and $x_i{}'$ is the normalized data of feature **x** at the i-th value.

IV. Results and Discussion

The results of the optimization of the BiLSTM parameters is shown first. Next, a comparison between the benchmark models and the proposed BiLSTM models, as well as the results of the different feature filters proposed. Then, the BiLSTM model is applied to the remaining sub datasets and the evaluation metrics compared between each sub dataset.

A. BiLSTM model parameter optimization

The parameter optimization was performed with sub dataset DS01, which makes the proposed BiLSTM model optimized for this dataset, but not necessarily for the remaining sub datasets, given that the datasets are different, most significantly in the number of failure modes. The optimization of the BiLSTM model parameters for each dataset was not performed as it is too time consuming, given the time constraints. This is something that can be improved upon to obtain an optimized model for each subset of the data available.

The parameters optimized and respective values can be seen in Tab. II. Following the proposed framework we obtained the optimal parameters of a network topology with the shape (256,256,64,1), in respect to the number of neurons of the input layer, the two hidden layers and the output layer, respectively. The same notation is used in Tab. II when describing the different topologies tested. This is a contracting form, as the initial layers have a larger number of neurons and the last layers have less neurons. This structure seems to work better for our situation than a constant form or an expanding one, as the results indicate. It's unclear why this happens, but it may be that a large number of neurons are required in the initial and middle layers to represent the complex interactions between the input data. The remaining parameters are a Dropout of 0.5, a learning rate of 0.001, a mini-batch size of 512, a Training/Validation split of 10%, this means that from the total training data, 10% are reserved for validation when training the model. An Early stopping condition of 20, this means the model trains for 20 consecutive epochs with no improvement of the loss function value before stopping the training. Finally, a sliding time-window size of 50 samples.

B. Models results with the different feature filters

Tab. III shows the results of the benchmark models and of the proposed BiLSTM model for both evaluation metrics, RMSE and the NASA Score, applied to sub dataset DS01, with

TABLE II
BiLSTM MODEL PARAMETERS OPTIMIZATION APPLIED TO SUB DATASET DS01

Model parameter	Values tested	Best value
Network topology	(64,32,16,8), (64,64,64,64), (256,256,256,256), (64,256,32,32), (128,128,64,64), (256,256,64,64), (256,256,64), (128,128,64)	(256,256,64)
Dropout	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8	0.5
Learning rate	0.0001, 0.0005, 0.001, 0.01, 0.1	0.001
Mini-batch size	128, 256, 512, 768, 1024	512
Training / Validation split	10%, 20%, 30%	20%
Early Stopping Condition	10, 20, 30	10
Sliding Time-Window size	25, 50, 100	50

the different feature filters from the feature selection analysis. Three different feature filters were tested on the DS01 dataset, one uses all the features, another reduced features based on the Pearson correlation coefficients as explained during the , the last model was trained with the reduced features based on the Importance analysis from the RF model.

Applying the Pearson correlation coefficients thresholds reduced the number of features by 60%, which amounts to using only 14 of the 36 features available. The RF Importance value analysis showed that only three features have significant Importance values, the cycle, the flight class and the unit. The remaining features all have extremely low values, the features with lowest Importance value were removed, reducing the total number of features by one third, which amounts to 24 of the total 36 features. The results are presented in Tab. III. The best performance was given by the BiLSMT model with the reduced features based on the RF Importance value analysis, and the RUL predictions of the BiLSTM model can be seen in Fig. 3. Also, the network with reduced features based on the Pearson correlation coefficients performed worse than using all the features. This means that the dependencies of the dataset cannot be well represented through linear correlation.

C. Performance of the BiLSTM model on all sub datasets

The BiLSTM network was applied to the remaining sub datasets and the results are shown in Tab. IV. For these results all features are used to fairly compare the results between sub datasets, as the feature analysis was performed only for sub dataset DS01. Since the model was optimized using the DS01 dataset, it has a poorer performance on the remaining sub datasets, as was expected. It is not fair to directly compare the Score metric between sub datasets as they have different number of samples. This is apparent when looking at the results of DS02 and DS03, where RMSE is higher for DS02 but Score is lower. This is a consequence of the number of samples of DS03 being higher than in DS02. The highest RMSE is 23.58, in dataset DS04, and the lowest is 5.015, in dataset DS01.

TABLE III
BENCHMARK AND BiLSTM MODELS RESULTS ON TEST DATA DS01 FOR THE DIFFERENT FEATURE FILTERS APPLIED: RF, XGBOOST, LSTM, BiLSTM

All features - 36		
Models	Score	RMSE
RF	5170	7.66
XGBoost	5004	7.50
LSTM	2441	5.11
BiLSTM	2487	5.15
Pearson correlation coefficients feature filter - 14		
Models	Score	RMSE
RF	5173	7.66
XGBoost	4986	7.46
LSTM	3723	6.60
BiLSTM	4445	7.34
RF Importance value feature filter - 24		
Models	Score	RMSE
RF	5168	7.66
XGBoost	5001	7.50
LSTM	2434	5.05
BiLSTM	1956	4.46

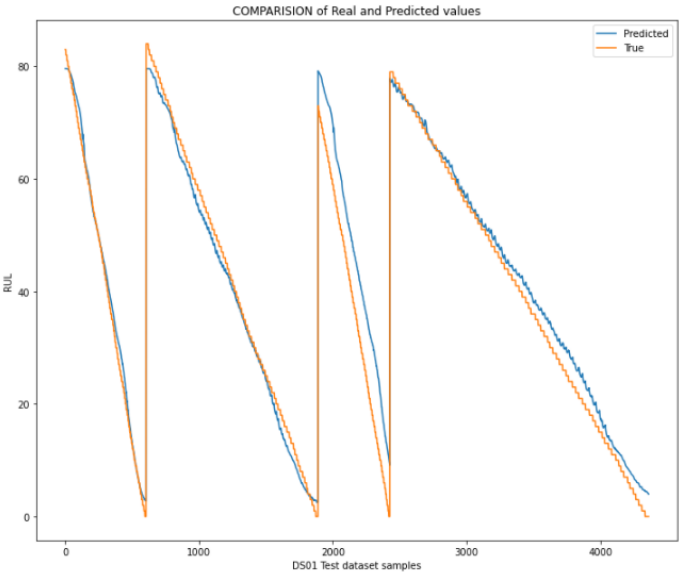

Fig. 3. RUL prediction of the BiLSTM model applied to test data DS01

It is clear that the BiLSTM network is capable of providing very good RUL predicitions, as seen from the RUL prediction of DS01, but the model is not directly transferable to the remaining datasets. A new hyper parameter optimization would need to be performed to achieve better results on the remaining datasets.

V. CONCLUSIONS

The present work explores Remaining Useful Life estimation using Machine Learning algorithms in the context of Predictive Maintenance. The proposed framework comprises the use of BiLSTM model with the aim of achieving high performance metrics.

The simulation results highlight that the use of the proposed framework enables to achieve good results in terms of the

TABLE IV
RESULTS OF THE BILSTM NETWORK ON ALL SUB DATASETS

Dataset	DS01	DS02	DS03	DS04	DS05	DS06	DS07	DS08a	DS08c	Mean
Score	**2487**	5657	7369	89725	24988	19317	50292	4356	3665	23075
RMSE	**5.15**	13.63	8.202	23.58	17.77	17.64	21.19	5.999	10	13.67

evaluated metrics (RMSE and Score). Moreover, the use of parameter optimization and feature selection filters allows to enhance the results, achieving an overall RMSE of 4.46 cycles. The results obtained show promise for further research and in providing critical information to support decision making for predictive maintenance strategies.

A. Future work

Regarding the framework, the future work proposed focuses on two aspects. First, applying an automatic parameter selection method in order to confidently achieve local optimal solutions when selecting the parameters of the BiLSTM model. Second, perform the training and testing several times in order to calculate average, median, maximum and minimum, as well as the frequency distribution of the performance results. This way we can reduce the effect of variability on the results presented. With these two improvements we could achieve better a predicting model and a better degree of confidence on the results obtained.

ACKNOWLEDGMENT

This work was partially supported by Axians Digital Consulting.

REFERENCES

[1] R Keith Mobley. *An introduction to predictive maintenance*. Elsevier, 2002.

[2] Ming-Yi You, Fang Liu, Wen Wang, and Guang Meng. Statistically planned and individually improved predictive maintenance management for continuously monitored degrading systems. *IEEE Transactions on Reliability*, 59(4):744–753, 2010.

[3] Sule Selcuk. Predictive maintenance, its implementation and latest trends. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 231(9):1670–1679, 2017.

[4] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International conference on database systems for advanced applications*, pages 214–228. Springer, 2016.

[5] Chong Zhang, Pin Lim, A Kai Qin, and Kay Chen Tan. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE transactions on neural networks and learning systems*, 28(10):2306–2318, 2016.

[6] Jialin Li, Xueyi Li, and David He. A directed acyclic graph network combined with cnn and lstm for remaining useful life prediction. *IEEE Access*, 7:75464–75475, 2019.

[7] Chengying Zhao, Xianzhen Huang, Yuxiong Li, and Muhammad Yousaf Iqbal. A double-channel hybrid deep neural network based on cnn and bilstm for remaining useful life prediction. *Sensors*, 20(24), 2020.

[8] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering System Safety*, 172:1–11, 2018.

[9] Jiujian Wang, Guilin Wen, Shaopu Yang, and Yongqiang Liu. Remaining useful life estimation in prognostics using deep bidirectional lstm neural network. In *2018 Prognostics and System Health Management Conference (PHM-Chongqing)*, pages 1037–1042, 2018.

[10] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. In *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 88–95, 2017.

[11] Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data*, 6(1), 2021.

[12] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, pages 1–9, 2008.

[13] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

[14] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[15] Tangbin Xia, Ya Song, Yu Zheng, Ershun Pan, and Lifeng Xi. An ensemble framework based on convolutional bi-directional lstm with multiple time windows for remaining useful life estimation. *Computers in Industry*, 115:103182, 2020.