The Aeronautical Journal (2023), **127**, pp. 2143–2153
doi:10.1017/aer.2023.84

REGULAR PAPER

Introducing CNN-LSTM network adaptations to improve remaining useful life prediction of complex systems

N. Borst[1] and W.J.C. Verhagen[2]

[1]Air Transport & Operations, Faculty of Aerospace Engineering, Technical University of Delft, Delft, The Netherlands and [2]Aerospace Engineering & Aviation, School of Engineering, Royal Melbourne Institute of Technology, Carlton, VIC, Australia
Corresponding authors: W.J.C. Verhagen; Email: wim.verhagen@rmit.edu.au

Received: 30 April 2023; **Revised:** 1 August 2023; **Accepted:** 22 August 2023

Keywords: Predictive maintenance; Prognostics; Remaining useful life; Neural networks

Abstract
Prognostics and Health Management (PHM) models aim to estimate remaining useful life (RUL) of complex systems, enabling lower maintenance costs and increased availability. A substantial body of work considers the development and testing of new models using the NASA C-MAPSS dataset as a benchmark. In recent work, the use of ensemble methods has been prevalent. This paper proposes two adaptations to one of the best-performing ensemble methods, namely the Convolutional Neural Network – Long Short-Term Memory (CNN-LSTM) network developed by Li et al. (*IEEE Access*, 2019, **7**, pp 75464–75475)). The first adaptation (adaptable time window, or ATW) increases accuracy of RUL estimates, with performance surpassing that of the state of the art, whereas the second (sub-network learning) does not improve performance. The results give greater insight into further development of innovative methods for prognostics, with future work focusing on translating the ATW approach to real-life industrial datasets and leveraging findings towards practical uptake for industrial applications.

Nomenclature

AF	activation function
ATW	adaptable time window
C-MAPSS	commercial modular aero-propulsion system simulation
CNN	convolutional neural network
DAG	directed acyclic graph
LB	lower boundary
LR	learning rate
LSTM	long-short term memory
PHM	Prognostics and Health Management
ReLu	rectified linear unit
RMSE	root mean square error
RUL	remaining useful life
SGD	standard gradient descent
TW	time window
UB	upper boundary

1.0 Introduction

The research area of PHM focuses, amongst others, on developing accurate models and methods to estimate RUL of complex systems and components. Accurate estimation gives rise to several potential benefits, including lower maintenance costs and increased availability. However, while academic and industrial examples of successful PHM applications are on the rise, several challenges are currently present [2], including (1) a small amount of failure events, which complicates the development of data-driven PHM models in particular; (2) the use of sensors which are not specifically targeted to support PHM; (3) a lack of publicly available data to generate new and/or better-performing models; (4) a comparative lack of model validation on real-life datasets and across multiple components, with many available models developed for specific applications but not tested more broadly; and (5) consistent interpretation, explainability and reliability of PHM models and their output in a safety-oriented industry where mistakes may lead to major accidents.

Historically, one way to address the first three challenges identified above has been to use synthetic datasets. The academic state of the art on PHM has focused to a substantial degree on various prognostic datasets provided by NASA, with the NASA Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset on engine failures being the most notable and widely used dataset for PHM model development and testing. While various categories of approaches have been investigated [3], including data-driven, hybrid and physical model-based methods [4], in recent years deep learning methods have become increasingly popular. For instance, Babu et al. developed and applied a Convolutional Neural Network (CNN) towards the C-MAPSS data [5]. The main strength of a CNN is that features are extracted from the data, which cannot be defined by using standard pre-processing methods, such as statistics. It is especially practical when the data can be spatially ordered (in time or position) [6]. As an alternative, Long-Short Term Memory (LSTM) algorithms have been applied in literature [7, 8]. An LSTM network is able to maintain information from previous input values by combining a short-term and long-term cell state. This technique allows for the network to use the complete data flow and alter predictions based on earlier inputs. The main strength of this technique is that later predictions become increasingly more accurate as time increases. Improved RUL prediction can be achieved since aircraft component degradation is a continuous process. A LSTM network is highly applicable in situations where sequential prediction is required. Currently, a trend is showing where different neural network types are combined to achieve higher prediction accuracy by compensating for disadvantages of the contributing types [9]; such approaches are known as Ensemble Learning Methods (ELM). In the context of the C-MAPSS dataset, the combination of CNN and LSTMs in ensemble approaches has shown better accuracy than earlier techniques [1, 8, 9], with the work by Li et al. [1] showing notable prediction performance. A CNN-LSTM is best used in a situation where a spatial and sequential input is available. This is the case of the C-MAPSS dataset when certain pre-processing measures are used. However, the research in this area has some limitations, for instance in terms of reproducibility, but also in terms of more detailed consideration of degradation evolution over time [10]. In terms of the latter, the current state of the art lacks methods or adaptations which enable RUL predictions at early stages of the lifecycle. Furthermore, most methods are trained to predict entire degradation trajectories, rather than using information on degradation states along the way.

To address these shortcomings, this paper aims to contribute to the academic state of the art by proposing and testing two adaptations to the CNN-LSTM network presented by Li et al. [1], which is one of the highest-performing ensemble methods in the state of the art. The two adaptations are:

- **Adaptable time window (ATW):** with current time window operations, a RUL prediction cannot be made at algorithm initiation. Therefore, an adaptable time window is applied to be able to predict a RUL in the early stages of the prediction. This method also aims to improve the prediction accuracy at later stages when more data is available.
- **Sub-network learning:** Most techniques aim to predict accurate results with the same network and settings for every data point. Others aim to predict the point where degradation is starting [9, 11]. In this paper a sub-network learning method is described that first identifies the stage of degradation and then uses a more specific trained network with the goal of improving overall prediction results.

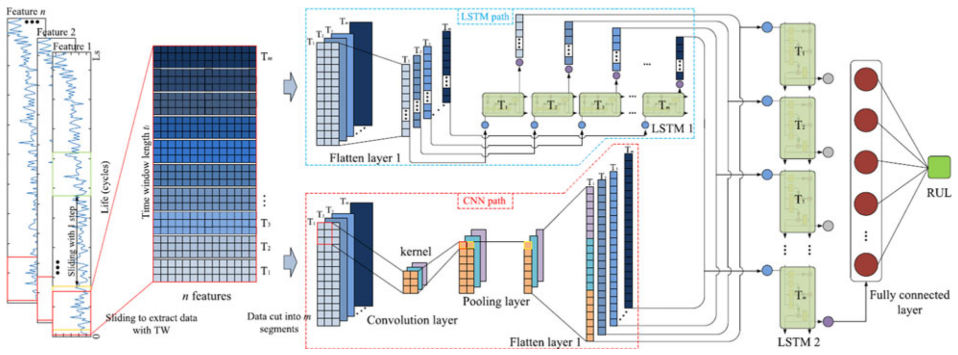

Figure 1. *CNN-LSTM network as applied by Li et al. [1].*

The structure of the paper is as follows. First, in the Method section a brief description is given of the 'baseline' model used by Li et al. [1], followed by a description of the two proposed adaptations. Subsequently, the Results section describes the C-MAPSS dataset and the pre-processing approach, while providing the general network hyperparameters, training and testing settings. The main results for the two provided adaptations are presented and discussed in detail. Finally, the Conclusions section wraps up the work while considering its main assumptions and limitations, pointing the way to future research.

2.0 Method

The model provided by Li et al. [1] is used as a baseline configuration, which is subsequently extended to include the two primary adaptations mentioned earlier, i.e., the adaptable time windows and sub-network adaptations. The baseline model comprises an ensemble network in the shape of a directed acyclic graph network (DAG) based on a simultaneous implementation of a 2D CNN network and a LSTM network. The outputs of both networks are combined afterwards in a second LSTM network. A visual representation of this network can be seen in Fig. 1 and is briefly explained below; for a more detailed description of the network architecture, the reader is referred to Li et al. [1].

To feed the DAG, feature input data is selected and pre-processed as described in Section 3.1, involving feature selection, normalisation and setting up a piece-wise linear target function. A sliding time window (TW) is passed over each feature (n in total) to extract training data for a certain TW length. This data is then implemented into two different parallel paths: the LSTM path and CNN path, respectively. The LSTM path involves flattening the input data since an LSTM network cannot directly use higher-order dimensional data. The resulting LSTM network contains u_1 nodes and one layer, where u_1 denotes the number of nodes adopted in this first LSTM network employed in the overall DAG network. The other path is based on a CNN algorithm. First a convolutional operation is implemented over the input data, followed by pooling and flattening operations to ensure both paths can be combined afterwards. The output size of the CNN is the same as the LSTM path. To obtain a final outcome, the CNN and LSTM paths are combined. First, the output of both paths are summed element-wise. This data is then further implemented in a second LSTM network. This network consists of u_2 nodes. Only the last output of this network will be input to a fully connected layer. This fully connected layer has a single output for each prediction. This output is the final estimated RUL.

2.1 Adaptable time windows

Most approaches that use a TW for prediction apply this at a cost of having no prediction during the first number of cycles [1, 8]. This has no implication when only predicting the last cycle of each unit in the

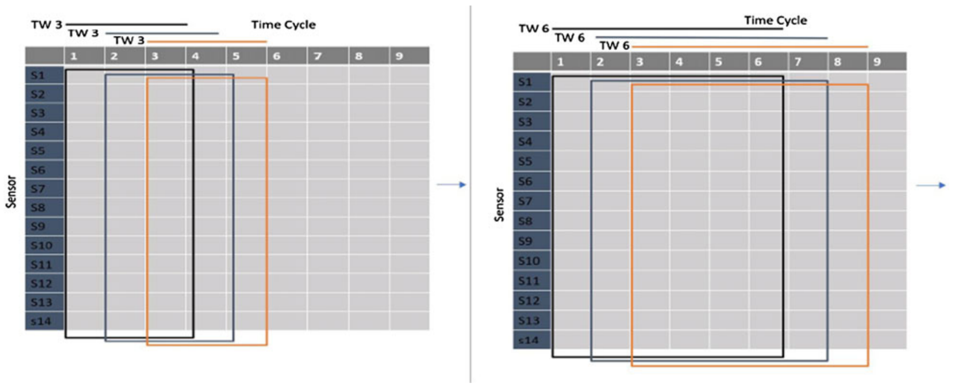

Figure 2. *Overview of the application of ATW for time window lengths of 3 and 6. On the left, a TW of 3 is applied for the time cycles 3, 4 and 5. Right a TW of 6 is applied for the time cycles 6, 7 and 8.*

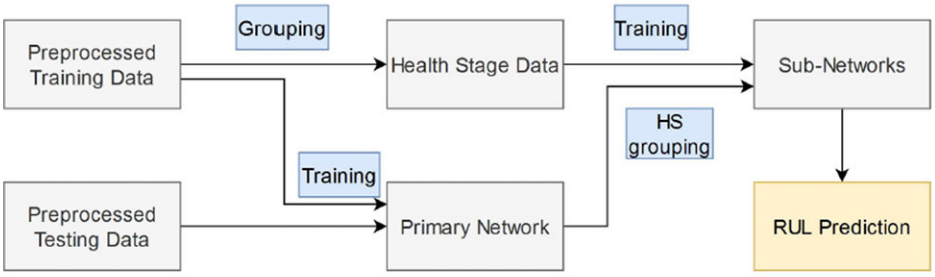

Figure 3. *Overview of the sub-network approach.*

testing set. However, this is undesirable for real-time application, hence an ATW approach is proposed here. In addition to the improved capability in generating early predictions, it is hypothsised that ATW may lead to better performance by including additional samples. In the ATW approach, the network is trained for different types of TW using all available samples. This is applied in increments of 3 (e.g., 3,6,9 etc.), since the convolutional operations are applicable for these increments due to the size of the slices used in this research. Other step increments could be achieved when the size of slices is altered. The weights of each training are stored and later used in the application of the model. Longer TW steps lead to less available samples, due to required data points to create a sample of this size. Output of the model can then be taken at any time cycle by taking the algorithm weights for largest possible TW (e.g., for time cycles 6,7 and 8 one can use a TW of 6). An example of ATW for different TW lengths leading to different sample sizes can be observed in Fig. 2. In this research, the ATW process is repeated for a TW length up to 45.

2.2 Sub-network learning

The second adaptation to the baseline network is based on sub-networks. First, a prediction is made by the original trained network (primary network). Afterwards a second network (sub-network) can be applied, which is only trained on the given health stage samples. The RUL prediction of sub-networks is used for the final prediction. This results in a double regression model. An overview of the approach is given in Fig. 3 and further discussed below.

In short, Fig. 3 highlights that pre-processed training data is used to first of all train a primary network. A primary network may provide less accurate predictions, since all different inputs should result in the

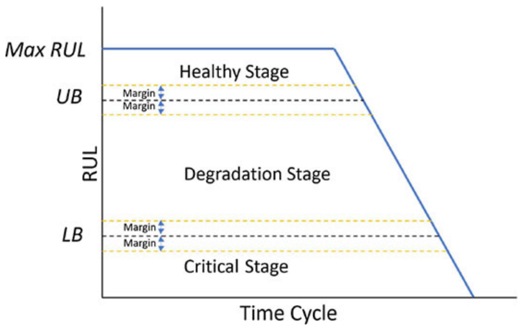

Figure 4. *Representation of the different health stages for sub-network training.*

correct outcome. To alleviate this, the primary network is used towards the identification of health states and subsequent grouping of the original pre-processed training data into these health states. This data, grouped according to health states, is subsequently used to train sub-networks for each health state. The final step in the sub-network is to combine individual sub-network predictions into an overall RUL prediction. The trained sub-networks and associated health states are applied in a separate testing dataset to generate RUL trajectories for the C-MAPSS dataset, a process further detailed in Section 3.1.1. Pre-processing of data is applied equally across the training and testing datasets and is further detailed in Section 3.1.2.

A critical aspect of the proposed approach is to identify health states. In the proposed approach, the original data samples are divided in three stages as shown in Fig. 4: a healthy state, degradation state and critical state. These states are constructed by setting two boundaries. The boundary that divides the healthy and the degradation state is named upper boundary (UB) and the boundary between the degradation state and the critical state is denoted the lower boundary (LB). The locations of these boundaries can be varied for optimal accuracy. To reflect the potential inaccuracy of the primary network predictions, a margin is introduced around the UB and the LB. When a prediction of the primary network is between the margins, no sub-network is applied. The primary prediction is used as the outcome for that sample. This method promises to reduce the effect of incorrect classification and different margin sizes can be applied to test for optimal accuracy.

By training in specific states the accuracy of the network can be increased. However, when the primary network specifies a data sample in the wrong sub-network (incorrect classification) an error is introduced. For the sub-network approach to lead to increased accuracy, this introduced error is required to be lower than the increase in accuracy given by the individual sub-networks.

3.0 Results

In the following section, the results of applying the adaptations to the baseline DAG network are provided. First, details are given as to how the DAG network and its adaptations have been implemented and prepared for application towards a case study involving the C-MAPSS FD001 dataset. Subsequently, results are provided and discussed for both adaptations.

3.1 Implementation towards C-MAPSS case study

3.1.1 Dataset characteristics

The baseline network and proposed adaptations are applied to the C-MAPSS dataset which comprises simulated engine degradation and failure data. The dataset has been created by simulating engine degradation and providing data for prognostic research [12]. The C-MAPSS dataset is divided in four different sub-datasets (FD001-004). Each dataset contains three different files: training file, testing file and a file

Table 1. *C-MAPSS dataset characteristics*

Parameter	FD001	FD002	FD003	FD004
Training units	100	260	100	249
Testing units	100	259	100	248
Operation conditions	1	6	1	6
Fault modes	1	1	2	2

with the actual RUL value for the final testing data points. The training and testing data both contain a number of engines. Each engine has a different initial health stage and different operation criteria thus provide a different number of operational cycles until failure. Each cycle of each engine contains operational settings and 21 sensor data values. The last cycle in the training set also indicates failure afterwards. The last value of the testing set has a RUL equal to the value given in the file with the actual RUL. Each sub-datasets has a different number of operating conditions and failure modes. The datasets are getting increasingly more complex. With FD001 having only one operating condition and fault mode, whilst FD004 has six operating conditions and two fault modes. The FD001 dataset is applied in this research, since application and optimisation for all datasets would be time consuming and not the main topic of this research. An overview of the different (sub)datasets is provided in Table 1.

3.1.2 Pre-processing

Before feeding the data to the network, several pre-processing steps are required.

First, the data needs to be arranged per engine and the correct input features need to be selected. The selected sensors for this research are obtained by the procedure mentioned by Zheng et al. and Li et al. [1, 13]. The sensors that show no positive or negative trend over time (irregular) are not in the model. This results in a total of 14 sensors being used.

The next step is to normalise the data, since neural networks require this to work optimally, without resulting in an exploding gradient [14]. The chosen methods of normalisation are Z-score normalisation and min-max normalisation. Z-score handles outliers well, in contrast to min-max normalisation. However, Z-score does not maintain its exact scale, while min-max does.

Thirdly, an often-used pre-processing technique is to apply a piece-wise linear RUL function, instead of a linear RUL function. This was introduced by Heimes [15]. The maximum target RUL is limited with this technique. This allows the model to represent real life degradation more accurately. It is based on the principle that after a certain amount of time degradation is visible and during normal operation no failure is apparent.

The final pre-processing technique to be applied originates from Zhao et al. [16] and informs one of the contributions of this work. This involves the application of a TW over the data points. The window is created by sliding over the training data for a certain TW length, step size of one and the label being the last cycle of every window. This allows the use of CNN and the ability to implement a representation over time. A TW of 30 is applied for the FD001 dataset, as described by Li et al. for the baseline case [1]. An adaptable TW technique has been described in the Method section, which allows for early prediction and enhanced prediction for longer sustaining components; the results of the ATW application are described in Section 3.2.

An overview of the pre-processing parameters is given in Table 2, with several discussed above. The values of these parameters have been set in accordance with the baseline case [1], where additional detail on other parameter settings (such as the number of slices and training epochs) can also be found.

3.1.3 Training approach

The baseline DAG network and the introduced adaptations all require training to be able to predict an accurate RUL for a given input. This training is performed over a number of iterations, known as epochs. The required number of epochs is based on the type of dataset, learning rate and applied batch-size. The

Table 2. *Pre-processing and training parameters, selected values and settings*

Pre-processing		Training	
Parameter	Value	Parameter	Value
Time window	30	Epochs	10
Selected sensors	14	Training samples	17,731 (for TW30)
Number of slices	10	Mini-batch size	100
Normalisation	z-score; min-max	Optimiser	RMS-Prop
RUL target	Piece-wise linear	Learning rate	0.001
Max RUL	125	Loss function	Smooth L1 Loss
		Activation function	ReLu
		Deep learning software	Pytorch/Python

selection of these parameters is important for the prevention of model overfitting. Table 2 provides an overview of pre-processing and training characteristics, which are kept as close as possible to Li et al. [1] to enable comparison of performance. The training data is firstly divided in a mini-batch of 100, as also applied by Li et al. [1]. This mini-batch allows the network to be updated after a sample of 100 data frames. During each epoch, the network is fed with all the different mini-batches. After each mini-batch, the network is updated with the given learning rate. This process prevents overfitting by only providing the network with a small amount of frames and outperforms other training methods, such as updating after the complete set and after every single input (batch gradient descent and stochastic gradient descent) [17]. Mini-batch training allows the network to be trained with longer sets of data without overfitting on long term data relations. However, applying a mini-batch is computational more costly than batch gradient descent, since the weights need to be updated more often.

After each mini-batch the weights need to be tuned. For this the type of activation function (AF), loss function, learning rate and optimiser are important. The AF used is the rectified linear unit (ReLu) AF. This is a commonly used activation function, which is computationally faster than the original sigmoid AF and more consistent than leaky ReLU, parametric ReLU or swish. However, some might outperform ReLu in certain prediction scenarios [17]. The loss function applied is the smooth L1 loss function (Huber loss). This type of loss is mostly applied for regression problems and is suitable in most cases. Exploding of the gradient is prevented more often and it is less sensitive to outliers compared to mean squared error loss [18]. The learning rate (LR) indicates the quantity each weight is updated each mini-batch. A low LR can find an optimum more readily than a higher LR, however the training might converge to a local minimum. A higher LR convergences faster towards an optimum but might not be able to find the optimal point. The LR is varied after the suitable normalisation type and optimiser are chosen. After the learning rate is chosen, a suitable stopping point is also allocated (amount of epochs to train). When trained for too long, the network can only recognise the training data itself due to overfitting. When too few training cycles are applied, underfitting is applicable. An optimal amount of epochs is required.

A suitable optimiser is required for good training. An optimiser indicates the direction each weight is altered to after each update. The most commonly used optimisers are standard gradient descent (SGD) with momentum, RMS-Prop and Adam optimisers. SGD with momentum is able to find more flatter local minima, however tuning of the weights is more important and critical. RMS-prop adapts the LR automatically when converging to a minimum. Adam is a combination of both techniques and also possesses an adaptable LR. These techniques are therefore selected and both tested for the best accuracy [17].

3.1.4 Performance metrics

To enable the evaluation of the results, it is necessary to apply performance metrics. The selected metrics are the root mean square error (RMSE) and a scoring function, first introduced at the 8th

Table 3. *Overview of ATW accuracy results (10 iterations)*

Learning rate	RMSE	Final RMSE	Score	Final score
Reference	13.34	13.73	351.07	369.24
ATW	11.17	12.77	179.23	268.26
ATW 11	11.09	12.75	176.69	267.17
ATW 14	12.75	11.09	177.38	267.14

international prognostics and health management conference. These are considered benchmark metrics for comparison of prognostics methods, though these functions do not necessarily align with operational considerations for end users.

Performance can be calculated based on the complete test set, as well as based on solely the final value, which is given for the CMAPSS test dataset. The final value is the RUL prediction of the last available data point for each individual engine unit. Most authors use the latter to evaluate the effectiveness of their algorithm, since this was the main goal of the PHM08 challenge [1, 9, 19]. Nonetheless, prediction accuracy over the whole dataset can be applied to see its overall effectiveness and is more realistic when real-time application is available.

In the following section, both RMSE and scoring function values are provided. Also, a distinction is made between performance evaluated over the complete set versus the final RUL. This results in a total of four different accuracy metrics, being referred to in the remainder as the RMSE, final RMSE, score and final score.

3.2 Results for adaptable time window and sub-network learning adaptations

The results provided are based on 10 iterations to reduce training and testing variability for network instances. The accuracy of median iterations is presented based on the testing set. The results of the baseline DAG network application are given as part of the discussion of the two adaptations and are referred to as the ‘reference’ case. It must be noted that the baseline network results are in line with and close to those provided by Li et al. [1], though minor deviations exist as not all information was present in Ref. (1) to fully reproduce the research outcomes.

3.2.1 Adaptable time window results

First, the results of the ATW extension are discussed. In Table 3 estimation performance results are shown for three different types of ATW.

It should be noted that the given numbers are based on a total of 10 iterations for each TW length tested to reduce training and testing variability for network instances. The network used to represent results for each different TW length is the one where the given accuracy metric is the median of all training iterations. This takes into account the relative randomness of neural network training. In terms of the provided results in Table 3, three different setups are compared to the baseline DAG results, referred to as ‘reference’ in the table. The first (simply labelled ATW) takes the highest possible TW length for each time cycle (e.g., time cycle 12, 13 and 14 use a TW length of 12). This results in an improvement across all accuracy metrics. Two other types are introduced, which are based on the effect that for the time cycles where the RUL prediction is low a TW of 3 is optimal. ATW 11 and ATW 14 are predicting time cycles up to 11 and 14, respectively. What can be seen is that the accuracy increases even further, since early time cycles are more accurate to predict with a TW length of 3. Overall, the ATW approach provides more accurate RUL estimations for the C-MAPSS FD001 dataset. This is put into further context by considering results from prior research, where it is evident that the ATW variant of the DAG proposed in this work outperforms prior work (see Table 4, where the final entries (ATW DAG) represents the outputs of the current study for different time window lengths, as discussed previously and presented in Table 3).

Table 4. *Prediction RMSE of other models (See Ref. (1) for references to the given models)*

Methods	Years	RMSE
MLP	2016	37.56
SVR	2016	20.96
RVR	2016	23.80
CNN	2016	18.45
LSTM	2017	16.14
ELM	2017	17.27
DBN	2017	15.21
MODBNE	2017	15.04
BLSTM	2018	14.26
RNN	2018	13.44
DCNN	2018	12.61
BiLSTM	2018	13.65
DAG	2019	11.96
ATW DAG (highest TW setting)	2023	11.17
ATW DAG (TW 11)	2023	11.09
ATW DAG (TW 14)	2023	12.75

Table 5. *Overview of the median prediction accuracy regarding sub-network training with a LB of 30 and an UB of 100 (10 iterations)*

Network	RMSE	Final RMSE	Score	Final score
Reference (ref.)	13.34	13.73	351.07	369.24
All health stages	13.31	13.85	449.13	562.21
Critical	5.34	4.80	49.08	38.94
Critical ref.	5.27	4.46	53.78	46.03
Degradation	17.90	13.85	600.26	314.65
Degradation ref.	18.80	20.23	818.10	792.84
Healthy	11.31	14.46	281.57	393.72
Healthy ref.	10.64	9.87	254.7	123.78

3.2.2 Sub-network learning results

Next, the results for the sub-network learning adaptation are given and discussed. Table 5 gives the results for sub-network learning using a lower bound of 30 cycles and an upper bound of 100 cycles. The accuracy of each different health stage is shown for two different models. The baseline (reference) DAG network is applied on each different health stage, as well as the sub-network training. The reference case uses the results of the primary network and does not calculate a new value for the sub-network. The accuracies of the sub-network learning are based on the location where each sample is indicated by the primary network.

It can be observed from the results that the network using all the health stage sub-networks hardly improves results in terms of RMSE. The other accuracy metrics are showing inferior results. This likely is associated with the effect that the primary network classifies samples towards the incorrect health stage.

In Table 6 an overview can be seen of four different types of misclassification. A represents the samples that should be placed in the healthy life stage but are placed in the degradation stage. B and D should be placed in the healthy or critical stage, respectively, but are placed in the degradation stage.

Table 6. *Overview of the prediction accuracy regarding sub-networks error types*

Bounds	A (%)	B (%)	C (%)	D (%)
LB30 & UB100	5.1	34.3	1.6	25.5
LB50 & UB110	10.5	33.4	2.5	69.0

Table 7. *RMSE prediction accuracy with a given margin*

Margin (%)	RMSE
0	13.31
2	13.71
5	13.98
10	14.47

Finally C represents samples that should be placed in the critical stage, but are placed in the degradation stage.

The different learned sub-networks based on the different health stages show that degradation stage prediction is increased in accuracy. However, the other health stages are performing less accurately with respect to the reference case. The number of different errors show that most of the errors occur in B and D. This would indicate that increasing the UB and LB should reduce these errors. However, when these bounds are increased, accuracy metrics remain worse compared to the reference case. For example, an UB and a LB of 50 and 110 provides a testing RMSE and score of 13.45 and 571.56, respectively. Another approach which has been applied is to create a margin around each boundary. The effect of different margins is shown in Table 7. However, an increase in the margin does not improve the prediction accuracy.

4.0 Conclusion and future work

This paper reproduced the CNN-LSTM ensemble method proposed by Li et al. [1], including its application to the C-MAPSS FD001 dataset. Two adaptations were proposed: (ATW and sub-network learning.

- Application of the ATW adaptation has led to an improvement in RUL estimation accuracy for the evaluated dataset, surpassing the performance of other ensemble methods applied in the state of the art. When applying the ATW technique the prediction accuracy increases to a RMSE of 11.09 and a score of 176.69. This is due to the effect that later predictions are predicted more accurately and early predictions are added, which are naturally better predicted. This technique allows RUL predictions already from the third time cycle. This method could therefore be applied for real-life applications. A RUL prediction is already available from the third time cycle of a unit/component. Accuracy is increasing when more time cycles are available. The results of this model could be used for planning maintenance in advance.
- The sub-network learning approach has been applied but its results do not (yet) show improvement over the reference DAG network. This is likely due to incorrect health state classification. In addition, applying a margin around the boundaries of each sub-network did not improve the accuracy for this approach.

In terms of future work, the ATW approach needs to be applied on real-life datasets rather than solely on synthetic data, where data is available for the complete operational profile (not only a single value per sensor per cycle). For the sub-network learning approach, the data could be re-normalised after the samples are divided by the primary network. This results in a larger difference and might yield better results. An optimisation of different bounds can be applied, potentially involving definition and testing of one or more technique(s) to detect incorrectly assigned samples and relocate them to the correct health stage sub-network. A combination of these techniques might result in an improvement with respect to the original DAG network. beyond these particular points, real-life datasets have some of the challenging characteristics set out in the introduction, in particular regarding limited failure event data, limited sensor data 'fitness for purpose', and may furthermore reveal issues with generalisability, interpretation and application of approaches such as those proposed in this work. Therefore, the ensemble method extended in this work by the proposed adaptations should be applied to real-life complex datasets, setting the stage for evaluation of RUL prognostics in real-life applications.

References

[1] Li, J., Li, X. and He, D. A directed acyclic graph network combined with CNN and LSTM for remaining useful life prediction, *IEEE Access*, 2019, **7**, pp 75464–75475. https://doi.org/10.1109/ACCESS.2019.2919566

[2] Scott, M.J., Verhagen, W.J.C., Bieber, M.T. and Marzocca, P. A systematic literature review of predictive maintenance for defence fixed-wing aircraft sustainment and operations, *Sensors*, 2022, **22**, p 7070. https://doi.org/10.3390/s22187070

[3] Ramasso, E. and Saxena, A. Review and analysis of algorithmic approaches developed for prognostics on CMAPSS dataset, PHM 2014 - Proceedings of the Annual Conference of the Prognostics and Health Management Society 2014, 2014, pp 612–622.

[4] Sikorska, J.Z., Hodkiewicz, M. and Ma, L. Prognostic modelling options for remaining useful life estimation by industry, *Mech. Syst. Signal Process.*, 2011, **25**, (5), pp 1803–1836. https://doi.org/10.1016/j.ymssp.2010.11.018

[5] Babu, G.S., Zhao, P. and Li, X.-L. Deep convolutional neural network based regression approach for estimation of remaining useful life, in Navathe, S.B., Wu, W., Shehkar, X., Du, X., Wang, S. and Xiong, H. (Eds), *Database Systems for Advanced Applications*, Springer International Publishing, 2016, Cham, Switzerland, pp 214–228.

[6] Jiao, J., Zhao, M., Lin, J. and Liang, K. A comprehensive review on convolutional neural network in machine fault diagnosis, *Neurocomputing*, 2020, **417**, pp 36–63. https://doi.org/10.1016/j.neucom.2020.07.088

[7] Shi, Z. and Chehade, A. A dual-LSTM framework combining change point detection and remaining useful life prediction, *Reliab. Eng. Syst. Safety*, 2021, **205**, p 107257, https://doi.org/10.1016/j.ress.2020.107257

[8] Zheng, S., Ristovski, K., Farahat, A. and Gupta, C. Long short-term memory network for remaining useful life estimation, IEEE International Conference on Prognostics and Health Management (ICPHM 2017), 2017, pp 88–95.

[9] Al-Dulaimi, A., Zabihi, S., Asif, A. and Mohammadi, A. Hybrid deep neural network model for remaining useful life estimation, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp 3872–3876.

[10] Zio, E. Prognostics and Health Management (PHM): Where are we and where do we (need to) go in theory and practice, *Reliab. Eng. Syst. Safety*, 2022, **218**, *Part A,* p 108119. https://doi.org/10.1016/j.ress.2021.108119

[11] Jayasinghe, L., Samarasinghe, T., Yeunv, C., Ni Low, J.C. and Sam Ge, S. Temporal convolutional memory networks for remaining useful life estimation of industrial machinery, Proceedings of the IEEE International Conference on Industrial Technology, 2019, pp 915–920.

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

[13] Zheng, C., Liu, W., Chen, B., Gao, D., Cheng, Y., Yang, Y., Zhang, X., Li, S., Huang, Z. and Peng, J. A data-driven approach for remaining useful life prediction of aircraft engines, IEEE Conference on Intelligent Transportation Systems, Proceedings ITSC, 2018, pp 184–189.

[14] Bengio, Y., Goodfellow, I. and Courville, A. *Deep Learning*, Vol. **1**, MIT Press, 2017, Cambridge, MA, USA.

[15] Heimes, F.O. Recurrent neural networks for remaining useful life estimation, 2008 International Conference on Prognostics and Health Management, 2008, pp 1–6, doi: 10.1109/PHM.2008.4711422

[16] Zhao, Z., Liang, B., Wang, X. and Lu, W. Remaining useful life prediction of aircraft engine based on degradation pattern learning, *Reliab. Eng. Syst. Safety*, 2017, **164**, 457, 74–83.

[17] Ruder, S. An overview of gradient descent optimization algorithms, 2016, arXiv preprint arXiv:1609.04747.

[18] Jha, P. A brief overview of loss functions in Pytorch, *Medium*, 2019. Accessed online 03-04-2022: https://medium.com/udacity-pytorch-challengers/a-brief-overview-of-loss-functions-in-pytorch-c0ddb78068f7

[19] Mathew, V., Toby, T., Singh, V., Maheswar Rao, B. and Goutham Kumar, M. Prediction of Remaining Useful Lifetime (RUL) of turbofan engine using machine learning, Proceedings of 2017 IEEE International Conference on Circuits and Systems (ICCS 2017), 2017.

Cite this article: Borst N. and Verhagen W.J.C. (2023). Introducing CNN-LSTM network adaptations to improve remaining useful life prediction of complex systems. *The Aeronautical Journal*, **127**, 2143–2153. https://doi.org/10.1017/aer.2023.84