ScienceDirect
IFAC PapersOnLine 53-3 (2020) 137–142

Recurrent Neural Networks and its variants in Remaining Useful Life prediction

YoudaoWang [a], Sri Addepalli [a], Yifan Zhao [a,*]

[a] Through-Life Engineering Services Institute, Cranfield University, MK43 0AL, Cranfield, United Kingdom
[*]Corresponding author: Tel.: +44 (0)1234 754729, E-mail yifan.zhao@cranfield.ac.uk

Abstract:
Data-driven techniques, especially artificial intelligence (AI) based deep learning (DL) techniques, have attracted more and more attention in the manufacturing sector because of the rapid growth of the industrial Internet of Things (IoT) and Big Data. Tremendous researches of DL techniques have been applied in machine health monitoring, but still very limited works focus on the application of DL on the Remaining Useful Life (RUL) prediction. Precise RUL prediction can significantly improve the reliability and operational safety of industrial components or systems, avoid fatal breakdown and reduce the maintenance costs. This paper reviews and compares the state-of-the-art DL approaches for RUL prediction focusing on Recurrent Neural Networks (RNN) and its variants. It has been observed from the results for a publicly available dataset that Long Short-Term Memory (LSTM) networks and Gated Recurrent Unit (GRU) networks outperform the basic RNNs, and the number of the network layers affects the performance of the prediction.

Keywords: Remaining useful life, Prognostics, asset lifecycle management, Deep Learning; Recurrent Neural Networks, Long Short-Term Memory, Gated Recurrent Unit

1. INTRODUCTION

Remaining useful life (RUL) prediction is an engineering discipline that works on the prediction of the future state or response of a given system based on the synthesis observations, calibrated mathematical models, and simulation (Leser, 2017). It generally refers to the study of predicting the specific time at which the system or the component will no longer be able to have its intended functional performance. Salunkhe *et al.* (2014) regard RUL as the time left before observing a failure. RUL is also called as remaining service life or remnant life referring to the time left before observing a failure given the current machine age, condition and the past operation profile. Okoh *et al.* (2014) define RUL as the time remaining for a component to perform its functional capabilities before failure. In recent years, RUL prediction has attracted vast attention from both academic researchers and industrial operators. There is no universal approach to predict RUL for all assets because of the variability in their surrounding conditions, initial working conditions and physics of different acquisition systems. Although many papers reviewed the methods of RUL prediction, there are very limited papers with a specific focus on the recent development of AI solutions for this topic. This paper gives a brief introduction of RUL prediction approaches and reviews the start-of-the-art DL approaches, particularly focusing on Recurrent Neural Networks (RNN) and its variants. The selected methods are then tested on a publicly available dataset and the performance is compared.

2. RELATED WORKS

In early years, RUL prediction approaches were simply catalogued into model-based (physics-based) approaches, data-driven based approaches and hybrid-based approaches. With the increasing study in this area, more detailed classifications have been proposed. The data-driven approaches were further classified into AI approaches and statistical approaches by Dawn *et al.* (2015). Okoh *et al.* (2014) divided the approaches into model-based, analytical-based, knowledge-based and hybrid-based simulation algorithms and tools. The model-based RUL prediction is applicable to statistics and computational intelligence approaches. The analytical-based methods refer to the physical failure technique. Since the system degradation modelling depends on the laws of nature, these approaches are generally quite efficient and descriptive. The knowledge-based RUL prediction is a combination of computational intelligence and experience. The hybrid approach is a collection methodology and technique, which can consist of any of the former approaches. A hybrid approach can often be the better option since it attempts to integrate advantages of both physics-based and data-driven based approaches. Meanwhile, based on a certain amount of data and relatively high fidelity models, a hybrid approach can usually achieve a higher accuracy for RUL prediction (Liao and Köttig, 2016). Nevertheless, the drawback of the hybrid approaches is that they also carry the shortcomings of both approaches and the increased complexity in achieving the solution.

Data-driven approaches are most widely used in the field of RUL prediction, where RUL is computed through statistical and probabilistic methods by utilising historic information and

10.1016/j.ifacol.2020.11.022

routinely monitored data of the system (Kabir *et al.*, 2012). The precondition for setting up the data-driven models for RUL prediction is the availability of multivariate historical data about system behaviour, which must encompass all phases of the system operation and degradation scenarios under certain operating conditions. Recent years, AI techniques, particularly deep learning (DL) techniques are becoming more and more attractive because of the rapid growth in the industrial Internet of Things (IoT) and Big Data (Zhao *et al.*, 2019). Deep learning is one of the sub-branches of machine learning, which is featured with multiple nonlinear processing layers, and originated from Artificial Neural Network (ANN). With the rapid development of computational infrastructure, DL has become one of the main research topics in the field of prognostics, given its capability to capture the hierarchical relationship embedded in deep structures (Ma, Sun and Chen, 2017). The characteristic of DL is its deep network architecture where multiple layers are stacked in the network to fully capture the representative information from raw input data (Geoffrey Hinton and Ruslan Salakhutdinov, 2006).

The published literature on DL approaches of prediction RUL mainly focused on four representative deep architectures, including Auto-encoder (AE), Deep Belief Network (DBN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) (Zhao *et al.*, 2015). AE and DBN are often used for pre-training of networks. CNN and RNN are generally used as predictive models. Both CNN and RNN have proved to outperform traditional prognosis algorithms in RUL prediction, while CNN based approaches are used more in fault diagnosis and surface integration inspection (Wang *et al.*, 2018). RNN, on the other hand, gained much more attention and achievement because it can model time sequence data (Zheng *et al.*, 2017). Nevertheless, RNN suffers from long-term time dependency problems that the gradients would either vanish or explode when propagated over many stages. Long Short-Term Memory (LSTM) network, as a type of RNN network for sequence learning, gains great favour for solving the long-term time dependency problems by controlling information flow using input gate, forget gate and output gate. LSTMs are naturally suited for RUL prediction tasks using sensor data with the inherent sequential nature due to their capability of remembering information over long periods of time.

The original LSTM was developed by Hochreiter and Schmidhuber (Hochreiter and Schmidhuber, 1997), when researchers discovered a vanishing and exploding gradient issue in traditional RNNs. To cope with the difficult learning long-term dependencies that traditional RNNs had, the LSTM introduced a memory cell that regulated the information flow in and out of the cell. Yuan *et al.* (2016) proposed an LSTM approach for different types of fault, where C-MPASS dataset was used as the study case. Compared to the traditional RNN, Gated Recurrent Unit LSTM (GRU-LSTM) and AdaBoost-LSTM showed improved performance in all cases. They also developed a vanilla LSTM approach two years later which further improved the prediction performance (Yuan *et al.*, 2018). Zhao *et al.* (2017) presented an integrated approach of CNN and bi-directional LSTM for machining tool wear prediction named Convolutional Bi-directional Long Short-Term Memory (CBLSTM) networks. CNN was firstly used to extract local robust features from the sequential input. Then, LSTM was utilised to encode temporal information. The proposed CBLSTM's capability of predicting the RUL of actual tool wear based on raw sensory data was verified with a real-life tool wear test. A multi-layer LSTM approach was provided by Zheng *et al.* (2017). The research investigated the hidden patterns from sensors and operational data with multiple operating conditions, fault and degradation models through combining multiple layers of LSTM cells with standard feed-forward layers. The superiority of the LSTM model in RUL prediction was validated on three widely used data sets, C-MAPSS Data Set, PHM08 Challenge Data Set and Milling Data Set. Consequently, Zhang *et al.* (2018) presented a bi-directional LSTM network to discover the underlying patterns embedded in time series to track the system degradation. The bi-directional LSTM network was implemented to track the variation of health index, and the RUL was predicted by the recursive one-step ahead method. Elsheikh *et al.* (Elsheikh, Yacout and Ouali, 2019) built a new LSTM architecture for RUL prediction, called Bidirectional Handshaking LSTM (BHLSTM) network, when short sequences of monitored observations were given with random initial wear. This method was able to predict the RUL with random starts, which made it more suitable for real-world cases as the initial condition of physical systems is usually unknown especially in terms of its manufacturing deficiencies. A new, asymmetric objective function that penalises late predictions rather than earlier ones was also presented to ensure safer predictions.

It has been identified from the review that RNN and its' variants dominate the state-of-the-art of DL-based RUL prediction. The next two sections present the typical RNN and its' variants in more details and compares their performance on a publicly available dataset.

3. RNN AND ITS' VARIANTS

3.1 RNN

In a traditional neural network, inputs are independent. While in RNN, the front neurons pass the information to the following neurons. As illustrated in Figure 1, an RNN can be regarded as numerous copies of the same neural network cell, in which each cell passes the message to the next. In other word, the output from a recurrent neuron is connected to the next one to characterise the current system state as a function of current sensing data and preceding system state.

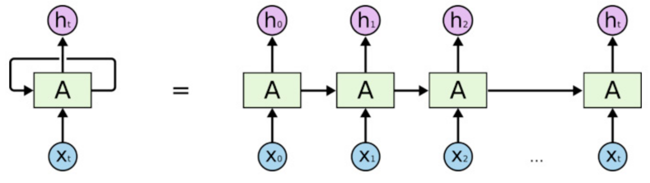

Fig. 1. An unrolled RNN(Retrieved from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

In an unrolled RNN, the sensing data (… x(t-1), x(t), x(t+1) …) are fed simultaneously into the corresponding neurons, which generate the corresponding neuron time series (… h(t-1), h(t), h(t+1) …). The output of a single recurrent neuron can be expressed as:

$$h_t = \sigma(W_x x_t + W_h h_{t-1} + b) \quad (1)$$

$$y_t = softmax(W_y h_t + c) \quad (2)$$

where W_x , W_h and W_y represent the weight vectors respectively. The symbol b and c denote the bias term and σ is the activation function, with the hyperbolic tangent or Relu being commonly used in RNN. y_t is the output of the recurrent neuron based on the output of the hidden state h_t. h_t is the hidden state at the time t which can be referred to a memory space containing the information of the current input and the former hidden state. It is worth mentioning that all the weight vectors are shared at every step, which means that the same task is repeated at every step with different inputs and the memory is renewed accordingly.

The main issue of the standard RNN is the gradient exploring and the gradient vanishing. These issues might happen when the network is too deep. In the other word, when the number of the time step is too large, the information carried in the front neuron will be lost because there is no structure in a standard recurrent layer that individually controls the flow of the memory itself. To solve this problem, the LSTM, a modified structure of the recurrent cell that incorporates the standard recurrent layer along with additional "memory" control gates, has been proposed.

3.2 Basic LSTM (Vanilla LSTM)

An LSTM cell was proposed to overcome the limitations of training the traditional RNN. LSTM uses storage elements to transfer information from the past output instead of having the output of the RNN cell to be a non-linear function of the weighted sum of the current inputs and previous output. Additionally, three gates are added to the model to control the information of the past hidden state and the current input. As demonstrated in Figure 2, these gates decide whether to forget the information or memorise it.

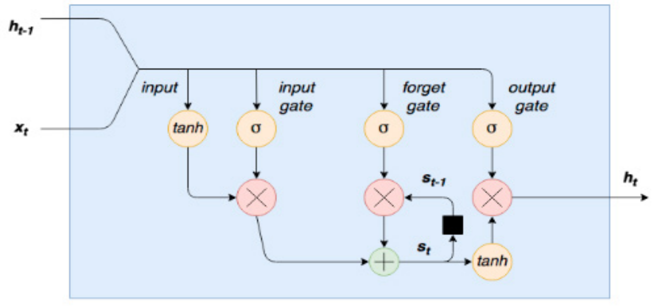

Fig. 2. A Basic LSTM cell (Retrieved from: https://adventuresinmachinelearning.com/keras-lstm-tutorial/)

The output of LSTM at step t is calculated using the following equations:

$$i = \sigma(U^i x_t + W^i s_{t-1} + b_i) \quad (3)$$

$$f = \sigma(U^f x_t + W^f s_{t-1} + b_f) \quad (4)$$

$$o = \sigma(U^o x_t + W^o s_{t-1} + b_o) \quad (5)$$

$$g = \tanh(U^g x_t + W^g s_{t-1} + b_g) \quad (6)$$

$$c_t = c_{t-1} \cdot f + g \cdot i \quad (7)$$

$$s_t = \tanh(c_t) \cdot o \quad (8)$$

where U , W and b are the trainable weights and biases, respectively, and i, f and o represent the input gate, forget gate and output gate respectively. These three gates have the same shape with different parameters U and W, which need to learn from the training process. The hidden state g cannot be used directly. It must pass through the input gate and then be used to calculate the internal storage c_t. While c_t is not only affected by the hidden state but also by c_{t-1} that is controlled by the forget gate. Based on c_t, a layer of $tanh$ function is applied to the output information s_t, which is constrained by the output door. The existence of the gates enables the LSTM to fulfil the long-term dependencies in the sequence, and by learning of the gate parameters, the network can find the appropriate internal storage behaviour.

3.3 Bi-directional LSTM

As shown in the basic LSTM, the hidden outputs between the LSTM layers are relayed to both the adjacent LSTM cells and the collected and used cells as the inputs for the LSTM next to it. A bi-directional LSTM structure is proposed with the information flowing back to the former LSTM cells. In the Bi-directional LSTM, the forward flow of information can discover the system variation, and it flows back to smooth the predictions. The outputs of the forward path and the backward path will then be concatenated. The governing equations of Bi-directional LSTM can be presented as :

$$h_i^1 = f(U^1 \cdot x_i + W^1 \cdot h_{i-1}) \quad (9)$$

$$h_i^2 = f(U^2 \cdot x_i + W^2 \cdot h_{i-1}) \quad (10)$$

$$y_i = softmax(V \cdot [h_i^1; h_i^2]) \quad (11)$$

where Equation (9) refers to the forward paths and Equation (10) refers to the backward path. y_i is the output of the Bi-directional LSTM obtained by fusing the results from both directional paths.

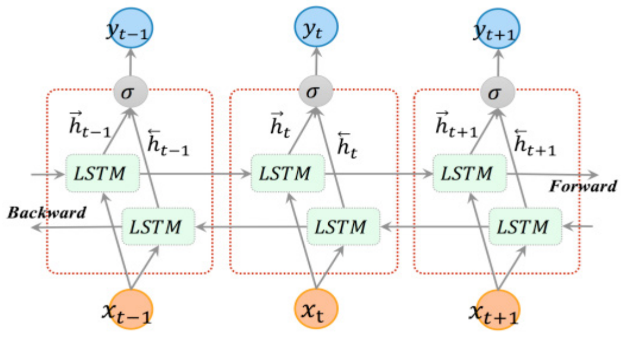

Fig. 3. A Bi-directional LSTM structure(Cui, Ke and Wang, 2018)

3.4 Gated Recurrent Unit (GRU)

GRU is the newer generation of RNNs and it looks very similar to LSTM as demonstrated in Figure 4. Instead of using the cell state, GRU uses the hidden state to transfer information. Moreover, it only has two gates (a reset gate and update gate) instead of three. Similar to the forget and input gate of LSTM, the function of the update gate is to decide what information to keep and what to throw away. The function of the reset gate is to decide what to keep from the past information.

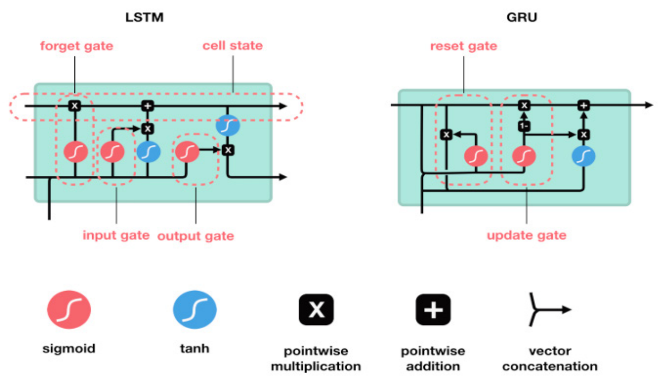

Fig.4. comparison of LSTM and GRU (Retrieved from https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21)

Since there are fewer tensor operations in GRU, it runs a little faster when training the structure than LSTM. However, as there is one gate less, it ranks behind the LSTM network in terms of performance. Thus, when the computational resource is limited, or a faster training is required, GRU could be a good option.

4. CASE STUDY ON RUL PREDICTION

4.1 Benchmark dataset overview

The case study focuses on adopting RNN algorithms on RUL prediction using NASA's C-MAPSS dataset. The dataset was collected from a Commercial Modular Aero-Propulsion System Simulation which could model the damage propagation of aircraft gas turbine engines.

The turbine engine includes five modules: fan, low-pressure compressor (LPC), high-pressure compressor (HPC), low-pressure turbine (LPT) and high-pressure turbine (HPT). To monitor the degradation process of the turbine engine, 58 on-board sensors were set on the turbine, recording the measurements of speed, temperature, and pressure at different locations. In this engine simulator, four datasets of different issues are presented, and consisted of three operational condition indicators and 21/58 sensor measurements.

In the dataset, engine profiles were simulated, with different initial degradation conditions. In addition, the maintenance was not considered during the simulation. The dataset includes one training set and one testing set for each engine. The training set consists of the historical run-to-failure measurement records of the engines from 21 on-board sensors. The testing dataset consists of the sensor measurements of engines which stopped at a certain point before failure. The objective is to predict the RUL of each engine based on the given sensor measurements. The information of the four sub-datasets is listed in Table 1.

Dataset FD001 refers to the engine failure arising from the high-pressure compressor under a single operation condition. Dataset FD002 refers to the engine failure from the high-pressure compressor under multiple operation conditions. Dataset FD003 refers to the engine failure from both high-pressure compressor and fan under a single operation condition. Dataset FD004 refers to the engine failure from both high-pressure compressor and fan under multiple operation conditions.

Table 1 C-MAPSS dataset

Dataset	FD001	FD002	FD003	FD004
Data for training	100	260	100	249
Data for test	100	259	100	248
Operating conditions	single	multiple	single	multiple
Fault conditions	compre ssor	compres sor	compress or & fan	compress or & fan

4.2 Data pre-processing

The raw sensor data were normalised to [0, 1]. Some of the operation conditions and sensor readings are constant, so the related data after normalisation are zeroes as indicated in 's6' in Figure 5. No feature extraction has been taken place in this case study and the entire sensor data stack was used as inputs for training. In addition, since there is no target output in raw datasets, the RUL has to be labelled at every cycle for each sample before training the models.

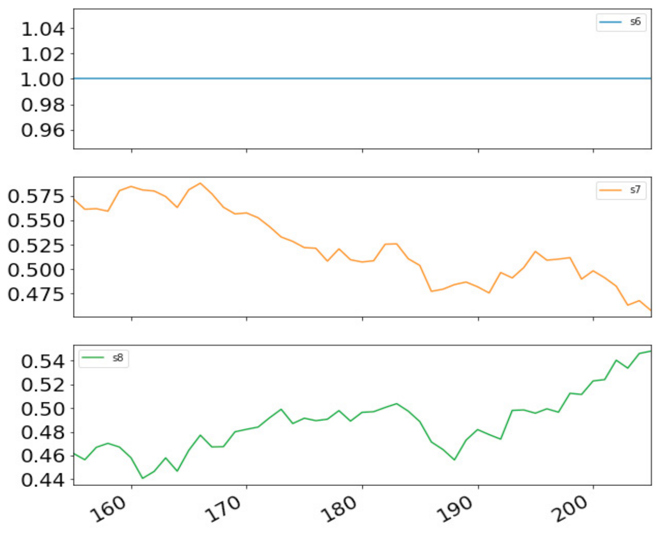

Fig. 5. Sensor data after normalisation (dataset FD001_engine id=3, windows of 50 cycles, sensor 6,7,8)

4.3 Performance evaluation

In this case study, the mean square error (MSE) are used to evaluate the performance of the trained neural networks. The mathematical expression is:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} d_i^2 \tag{12}$$

where n is the total number of true RUL targets in the related test set and d_i refers to the difference between the true RUL and the predicted RUL. Table 2 summarises the performance of different network structures on the training set and test set for FD001. The sequence length is set to be 50 indicating a window size of 50 cycles. All the structures were run for 5 times and the average value was used to evaluate the performance. The activation of hidden neurons was 'ReLU'. Probability of dropping out neurons at the output layer was set to 0.4. The quantity of the hidden layer neurons is 128. The optimiser used for this evaluation is RMSprop.

Table 2 Performance of selected methods for FD001

Structures	MSE(Train)	MSE(Test)
RNN	1,073.8±120.7	568.5±139.4
RNN_2LAYERS	943.2±236.2	494.0±178.1
Vanilla LSTM	616.6±88.5	566.0±147.3
LSTM_2LAYERS	415.3±71.1	397.7±123.3
LSTM_3LAYERS	459.9±98.7	411.4±74.2
Bi-directional LSTM	926.5±184.0	717.7±371.4
Bi-directional LSTM_2LAYERS	429.4±81.0	440.9±143.4
GRU	495.1±54.4	457.2±103.7
GRU_2LAYERS	497.8±52.9	416.8±94.1

As demonstrated in Table 2, LSTM and GRU perform much better than the basic RNN structure. In addition, the outcome of the same structure varies at every run with a relatively large standard deviation. The performance of GRU, LSTM and Bi-directional LSTM is quite similar, and as for the dataset FD001, a 2-layer LSTM structure has the best performance. Moreover, the number of structure hidden layers affects the prediction performance, but there is no clear monotonous relationship.

The performance of different structures on the other three datasets is listed in Table 3, 4 and 5 with the same parameters for the structure. Generally, the observations are similar to that of the dataset FD001. The highlighted values refer to the gradient vanishing or explosion problem which means that structure is not suitable for the corresponding dataset or some parameters need to be changed. For instance, the quantity of the hidden layer neurons, the activation function, the dropping out neurons at the output layer and the optimiser can all affect the training performance.

Table 3 Performance of selected methods for FD002

Structures	MSE(Train)	MSE(Test)
RNN	**3303.6**	**2820.76**
RNN_2LAYERS	**10499.13**	**8430.26**
Vanilla LSTM	1,351.8±196.9	970.2±166.6
LSTM_2LAYERS	664.6±161.0	768.7±100.7
LSTM_3LAYERS	820.1±103.7	722.6±43.3
Bi-directional LSTM	1,314.5±167.3	955.1±141.1
Bi-directional LSTM_2LAYERS	821.7±181.5	821.2±166.1
GRU	**3304**	**3051**
GRU_2LAYERS	518.1±176.1	721.6±81.4

Table 4 Performance of selected methods for FD003

Structures	MSE(Train)	MSE(Test)
RNN	**2823.27**	**2102.9**
RNN_2LAYERS	**21754.34**	**7036.18**
Vanilla LSTM	1,709.3±119.8	1,065.4±357.4
LSTM_2LAYERS	1,218.8±170.1	778.2±311.5
LSTM_3LAYERS	1,383.3±146.9	522.4±98.5
Bi-directional LSTM	1,536.1±252.4	967.9±358.4
Bi-directional LSTM_2LAYERS	1,208.1±219.9	799.5±245.0
GRU	**8095.06**	**3479.04**
GRU_2LAYERS	1,265.9±208.3	446.8±66.5

Table 5 Performance of selected methods for FD004

Structures	MSE(Train)	MSE(Test)
RNN	**6273.05**	**3645.16**
RNN_2LAYERS	**16861.05**	**9171.44**
Vanilla LSTM	**6273.4**	**3663.36**
LSTM_2LAYERS	1,486.5±85.3	1,024.4±130.7
LSTM_3LAYERS	1,793.6±195.9	1,142.5±276.5
Bi-directional LSTM	2,825.0±264.3	2,062.8±347.0
Bi-directional LSTM_2LAYERS	2,250.5±364.4	1,582.0±149.5
GRU	**6273**	**3663.15**
GRU_2LAYERS	1,602.2±279.4	944.9±354.4

5. CONCLUSIONS

Compared with traditional physics-based models, data-driven models gain more attention due to the significant development of sensors, sensor networks and computing systems. Machine learning techniques, especially, the DL techniques are regarded as a powerful solution due to their ability to provide more agility to process data associated with highly nonlinear and complex feature abstraction through a cascade of multiple layers. DL provides the decision-makers new visibility into their operations, as well as real-time performance measures and costs. This paper reviewed one of the most popular DL algorithms, RNN and its variants, on RUL prediction. RNNs are good for processing sequence data for predictions but suffer from short-term memory issue. LSTMs and GRUs were designed as the solution to address this issue by adding some gates to the RNN structure. These gates are used to control the information flow through the sequence chain. A case study on RUL prediction using the C-MAPSS dataset was carried out to validate all these approaches. Some of the RNN structures were adopted in this study including LSTM, Bi-directional LSTM and GRU. When the size and complexity of dataset are relatively small, the results obtained by various algorithms are relatively similar. With the increase of the size and complexity, the performance difference of the selected methods started to show up. For instance, gradient vanishing or explosion problem of basic RNN has been observed in all the datasets except FD001. In addition, the performance of GRU and LSTM are relatively close in every tested dataset. A vanilla LSTM or GRU may also be affected by some gradient explosion problems. It also has been observed that the number of the layers affects the performance of the structure. As in this case study, a 2-layer LSTM network performs better than a 3-layer LSTM network in most of the datasets. Thus, the choice of the optimum network structure is often based on the volumn of the dataset.

REFERENCES

An, D., Kim, N. H. and Choi, J. H. (2015) 'Practical options for selecting data-driven or physics-based prognostics algorithms with reviews', *Reliability Engineering and System Safety*. Elsevier, 133, pp. 223–236. doi: 10.1016/j.ress.2014.09.014.

Cui, Z., Ke, R. and Wang, Y. (2018) 'Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction', pp. 1–11. Available at: http://arxiv.org/abs/1801.02143.

Elsheikh, A., Yacout, S. and Ouali, M. S. (2019) 'Bidirectional handshaking LSTM for remaining useful life prediction', *Neurocomputing*. Elsevier B.V., 323, pp. 148–156. doi: 10.1016/j.neucom.2018.09.076.

Geoffrey Hinton and Ruslan Salakhutdinov, C. (2006) 'Reducing the dimensionality of data with neural networks', *Science*, 28(5786), pp. 502–504. Available at: http://www.sciencemag.org/content/313/5786/502.short %5Cnhttp://www.cs.toronto.edu/~hinton/science.pdf.

Hochreiter, S. and Schmidhuber, J. (1997) 'Long Short-Term Memory', *Neural Computation*, 9(8), pp. 1735–1780. doi: 10.1162/neco.1997.9.8.1735.

Kabir, A. *et al.* (2012) 'A review of data-driven prognostics in power electronics', *Proceedings of the International Spring Seminar on Electronics Technology*. IEEE, pp. 189–192. doi: 10.1109/ISSE.2012.6273136.

Leser, P. E. (2017) *Probabilistic Prognostics and Health Management for Fatigue-critical Components using High-fidelity Models*. North Carolina. Available at: https://repository.lib.ncsu.edu/bitstream/handle/1840.20/34868/etd.pdf?sequence=1&isAllowed=y.

Liao, L. and Köttig, F. (2016) 'A hybrid framework combining data-driven and model-based methods for system remaining useful life prediction', *Applied Soft Computing Journal*. Elsevier B.V., 44, pp. 191–199. doi: 10.1016/j.asoc.2016.03.013.

Ma, M., Sun, C. and Chen, X. (2017) 'Discriminative Deep Belief Networks with Ant Colony Optimization for Health Status Assessment of Machine', *IEEE Transactions on Instrumentation and Measurement*, 66(12), pp. 3115–3125. doi: 10.1109/TIM.2017.2735661.

Okoh, C. *et al.* (2014) 'Overview of Remaining Useful Life prediction techniques in Through-life Engineering Services', *Procedia CIRP*. Elsevier B.V., 16, pp. 158–163. doi: 10.1016/j.procir.2014.02.006.

Salunkhe, T., Jamadar, N. I. and Kivade, S. B. (2014) 'Prediction of Remaining Useful Life of Mechanical Components-A Review', 3(6), pp. 125–135.

Wang, J. *et al.* (2018) 'Deep learning for smart manufacturing: Methods and applications', *Journal of Manufacturing Systems*. The Society of Manufacturing Engineers, 48, pp. 144–156. doi: 10.1016/j.jmsy.2018.01.003.

Yuan, M. *et al.* (2018) 'Remaining useful life estimation of engineered systems using vanilla LSTM neural networks', *Neurocomputing*. Elsevier B.V., 275, pp. 167–179. doi: 10.1016/j.neucom.2017.05.063.

Yuan, M., Wu, Y. and Lin, L. (2016) 'Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network', *AUS 2016 - 2016 IEEE/CSAA International Conference on Aircraft Utility Systems*, pp. 135–140. doi: 10.1109/AUS.2016.7748035.

Zhang, J. *et al.* (2018) 'Long short-term memory for machine remaining life prediction', *Journal of Manufacturing Systems*. Elsevier, 48(December 2017), pp. 78–86. doi: 10.1016/j.jmsy.2018.05.011.

Zhao, R. *et al.* (2015) 'Deep Learning and Its Applications to Machine Health Monitoring : A Survey', 14(8), pp. 1–14.

Zhao, R. *et al.* (2017) 'Learning to monitor machine health with convolutional Bi-directional LSTM networks', *Sensors (Switzerland)*, 17(2), pp. 1–18. doi: 10.3390/s17020273.

Zhao, R. *et al.* (2019) 'Deep learning and its applications to machine health monitoring', *Mechanical Systems and Signal Processing*. Elsevier Ltd, 115, pp. 213–237. doi: 10.1016/j.ymssp.2018.05.050.

Zheng, S. *et al.* (2017) 'Long Short-Term Memory Network for Remaining Useful Life estimation', *2017 IEEE International Conference on Prognostics and Health Management, ICPHM 2017*. IEEE, pp. 88–95. doi: 10.1109/ICPHM.2017.7998311.