11-3-2021

A Framework of Dynamic Data Driven Digital Twin for Complex Engineering Products: the Example of Aircraft Engine Health Management

Zhenhua Wu

Jianzhi Li
The University of Texas Rio Grande Valley

Recommended Citation

Wu, Zhenhua, and Jianzhi Li. "A Framework of Dynamic Data Driven Digital Twin for Complex Engineering Products: the Example of Aircraft Engine Health Management." Procedia Manufacturing 55 (2021): 139-146. https://doi.org/10.1016/j.promfg.2021.10.020

Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 55 (2021) 139–146

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021)
7-10 September 2021, Athens, Greece.

A Framework of Dynamic Data Driven Digital Twin for Complex Engineering Products: the Example of Aircraft Engine Health Management

Zhenhua Wu[a]*, Jianzhi Li[b]

[a]Department of Engineering, Virginia State University, Petersburg, VA 23806, USA
[b]Department of Manufacturing and Industrial Engineering, the University of Texas-Rio Grande Valley, Edinburg, TX 78539, USA

* Corresponding author. Tel.: 1-804-524-1079. *E-mail address:* zwu@vsu.edu

Abstract

Digital twin is a vital enabling technology for smart manufacturing in the era of Industry 4.0. Digital twin effectively replicates its physical asset enabling easy visualization, smart decision-making and cognitive capability in the system. In this paper, a framework of dynamic data driven digital twin for complex engineering products was proposed. To illustrate the proposed framework, an example of health management on aircraft engines was studied. This framework models the digital twin by extracting information from the various sensors and Industry Internet of Things (IIoT) monitoring the remaining useful life (RUL) of an engine in both cyber and physical domains. Then, with sensor measurements selected from linear degradation models, a long short-term memory (LSTM) neural network is proposed to dynamically update the digital twin, which can estimate the most up-to-date RUL of the physical aircraft engine. Through comparison with other machine learning algorithms, including similarity based linear regression and feed forward neural network, on RUL modelling, this LSTM based dynamical data driven digital twin provides a promising tool to accurately replicate the health status of aircraft engines. This digital twin based RUL technique can also be extended for health management and remote operation of manufacturing systems.

Keywords: Digital Twin, Dynamic Data Driven, Complex Engineering Products, Aircraft Engine, Health Management

1. Introduction

The concept of the digital twin was coined around 2002 [1]. Digital twin uses Industry Internet of Things (IIoT) as a framework to connect equipment and processes with high fidelity simulation models which replicate the physical asset in cyberspace. It predicts the nominal behaviour of the system of interest, with aims of identifying potential issues of the real machine counterpart and suggesting corrective actions based on data, historical records, and other information collected from various sensors deployed on the physical equipment.

Digital twin is a type of computer simulation model, but with distinction [2]:

- A digital twin is the virtual model of a real 'thing'.
- A digital twin simulates both the physical state and behaviour of the thing.
- A digital twin is unique, associated with a single, specific instance of the thing.
- A digital twin is connected to the thing, updating itself in response to known changes to the thing's state, condition, or context.
- A digital twin provides value through visualization, analysis, prediction, or optimization.

Aircraft vehicle operations have been a particular area of interest for digital twin applications because these vehicles are operated in data-rich environments. NASA was the first to apply pairing a technology, the precursor to today's digital twin, as far back as the early days of space exploration [3]. Now, digital twin technology is being envisioned to be the paradigm shift for ship/aircraft fleet maintenance by NASA and the Air Force [4, 5]. Digital twin technology is spanning wide application areas ranging from rapid requirement development and trade-space decisions, to design and

10.1016/j.promfg.2021.10.020

prototype efforts, to fielding, and to the operation and sustainment of mission-critical systems. We envision that the digital twin models, if designed properly, could also support a feedback loop based on condition sensors [6]. This can be further used to monitor actual deployed systems to update digital models for decisions regarding remaining useful life (RUL) and other lifecycle decisions. The digital twin for health management on mission-critical systems thus can play a vital role in the maintenance, repair, and overhaul (MRO) business to reduce the high costs associated with in-flight malfunctions, maintenance-related delays and cancellations, potential loss of human lives, etc. More importantly, the digital twin can also improve logistical responsiveness at the point of need thus shortening the logistics tail.

Consequently, in this paper, a framework of dynamic data driven digital twin for aircraft engine health management is proposed. The process to build a digital twin for engine health management encompasses the following steps:

1. Collect data. The data collection can either be based on historical records or real time data from IIoT devices and sensors. The data collected will then be stored using a cloud service.
2. Prepare data. In many cases, the raw data collected from devices and sensors involves irregularities such as signal noise, or missing data. Therefore, it will require appropriate preparation before the machine learning step that follows. The data preparation step may involve data de-noising, reformatting, and/or pre-processing to convert raw data to useful information, which then becomes the input to machine learning models in digital twin.
3. Build the digital twin using a machine learning model. Based on the prepared data, this step will train different machine learning models and the associated parameters, then compare the modelling results. Further, it will validate and test the selected model using testing data.
4. Deploy the digital twin. When a model that satisfies the selection criteria is selected, it is ready to be deployed. That involves wrapping the model into cyberspace using a web service app that can read data and return analysis results. The digital twin is then packed into a docker container, which in turn can be deployed either in the cloud or in an IoT Edge device.
5. Maintain and refine the model. The work is not done after model deployment. The researchers and engineers need to continue collecting new data to update and refine the model. The updated digital twin model will be re-deployed to cyberspace.

When developing a digital twin for physical systems or products, the following research questions have yet been fully considered [7]:

1) What variables in the physical domain can be extracted?
2) How should the number and deployment of sensors either during design time or during the usage time (for legacy systems) be selected?
3) How can a digital twin model be developed given the constraints of resources (such as sensors and costs)?
4) When should the digital twins (as lightweight as possible to meet the resource constraints) be updated to make sure that they can accurately predict system performance?

A dynamic data-driven digital twin is promising to address these questions. Dynamic data driven application systems (DDDAS) is an emerging powerful tool that allows more effective measurement processes to update the model [6]. DDDAS naturally couples with the digital twin to enhance the capability to fuse sensors, data, model, and decisions together. DDDAS incorporate additional data into an executing digital twin, and in reverse, enhance a digital twin to dynamically steer the decision on its physical asset. DDDAS bring challenges as well as opportunities for engineering applications on the aspects of mathematical algorithms, systems software, and data collection [6]. However, the utilization of the DDDAS approach for the digital twin is still in infancy. The only relevant work we are aware of comes from [7].

2. Literature review

The literature has been reviewed through two aspects: DDDAS based applications and RUL prediction for aircraft engines.

Since DDDAS's inception in 2000, the DDDAS concept has been successfully applied to broad application areas such as manufacturing [8], smart cities [9], health care [10], and security [11]. Common themes include agent simulation, model synchronization, user interaction, and data analytics. The emerging new technologies such as big data, Internet of Things, and cloud/edge computing are enabling DDDAS with larger-scale impacts [12]. Meanwhile, the DDDAS research is also facing new challenges in hardware and embedded system design, middleware, data analytics, and applications. Issues span theoretical, algorithmic, and computational aspects. However, when further research is investigated, DDDAS will support a broader range of application in areas where the opportunities afforded by the DDDAS paradigm in modelling, simulation, and run-time execution can be fully realized.

To address the shortcoming of traditional machine learning algorithms' deficiency in adapting to the complex and non-linear characteristics of manufacturing systems and processes, a deep learning Long Short-Term Memory (LSTM) network was proposed to track the system degradation and predict the RUL [13]. This research started from the conversion of the raw sensor data to an interpretable health index with the aim of better describing the system health condition and then tracked the historical system degradation for accurate prediction of its future health condition. Evaluation using NASA's C-MAPSS dataset verifies the effectiveness of the proposed method. Compared with other machine learning techniques, LSTM is more powerful and accurate in modeling degradation patterns, enabled by its time-dependent structure in nature.

An ensemble learning-based prognostic approach was studied to model degradation due to wear as well as to predict the RUL of aircraft engines [14]. The ensemble learning algorithm combined multiple base learners, including random forests (RFs), classification and regression tree (CART), recurrent neural networks (RNN), autoregressive (AR) model, adaptive network-based fuzzy inference system (ANFIS), relevance vector machine (RVM), and elastic net (EN), to achieve better predictive performance. The particle swarm optimization (PSO) and sequential quadratic optimization (SQP) methods were used to determine optimum weights to the base learners. The predictive model trained by the ensemble learning algorithm was demonstrated on the the C-MAPSS data. Experimental results have shown that the ensemble learning algorithm predicted the RUL of the aircraft engines with considerable robustness and also outperformed other prognostic methods reported in the literature.

A feature-representation based transfer learning (TL) method was proposed to predict RUL of equipment, under scenarios where samples with previously unseen conditions are presented in the target domain and the labels are available only for the source domain, but not the target domain [15]. This setting corresponds to generalizing from a limited number of run-to-failure experiments performed prior to deployment into making prognostics with data coming from deployed equipment that is being used under multiple new operating conditions and experiencing previously unseen faults. A deviation detection method, Consensus Self-Organizing Models (COSMO), was studied to create transferable features the RUL regression modeling. These features capture how different a particular equipment is in comparison to its peers. The efficiency of the proposed TL method was demonstrated using the NASA's C-MAPSS dataset. Models using the COSMO transferable features showed better performance than other methods on predicting RUL when the target domain is more complex than the source domain.

A transfer learning algorithm based on Bi-directional LSTM (BLSTM) neural networks was proposed for RUL estimation [16]. The authors first trained the BLSTM models on different but related datasets, and then fine-tuned the models by the target dataset. Their experimental results showed that transfer learning can, in general, improve the prediction models on the dataset with a smaller number of samples; however, when transferring from multi-type operating conditions to single operating conditions, transfer learning led to a worse result.

The LSTM Recurrent Neural Network (RNN) technique was investigated on RUL prediction within a digital twin framework as a means of synchronization with changing operational states [17]. LSTM encoder-decoder (LSTM-ED) was applied to train a multi-layered neural network and reconstruct the sensor data time series corresponding to a healthy state. The resulting reconstruction error can be used to capture patterns in input data time series and estimate health index (HI) of training and testing sets. Using a time lag to record similarity between the HI curves, a weighted average of the final RUL estimation is obtained. The described empirical approach is evaluated on publicly available engine degradation dataset with run-to-failure information. Results indicate a high RUL estimation accuracy with greater error reduction rate. This demonstrates wide applicability of the discussed methodology to various industries where event data is scarce for the application of only data-driven techniques.

When a digital twin is deployed to an aircraft engine, this digital twin is subject to continuous updating from the data received from the engine by sensors or IIoT. There are many scenarios that may challenge the process to update the digital twin. For example, the engines will go through deterioration, or the operation environment may have to be changed or disrupted. Both scenarios will cause pattern changes in the collected data. How to update the model subject to changed data is one of the challenges in the digital twin research. Through a review of literature, we want to propose a dynamical data driven digital twin based on LSTM to address this research question. The proposed methodology is as follows in Section 3.

3. Methodology

3.1 Overview of the proposed digital twin

The proposed digital twin of "sensing, monitoring, analysis, and control" for the aircraft engine health management set up is shown in Fig. 1. The digital twin is built using IIoT, sensors and data acquisition (DAQ), edge devices, and fog/cloud computing based on Deloitte University's suggestion [18]. This digital twin will be infused into the aircraft health management monitoring with five function blocks: 1) monitoring, 2) visualization, 3) data storage, 4) analysis, and 5) control. Fog and cloud computation are envisioned into the digital twin model. The fog nodes will perform some rudimentary computation on the data and make local decisions on the engine operation status. When intensive computation and global decision are needed, cloud computing will start. The cloud includes the local/public cloud. In the current vision, the digital twin will model and monitor two interesting areas in: 1) remaining useful life prediction and 2) diagnostics and prognostics. But it can be extended to other areas such as maintenance scheduling, and feedback control of the physical asset.

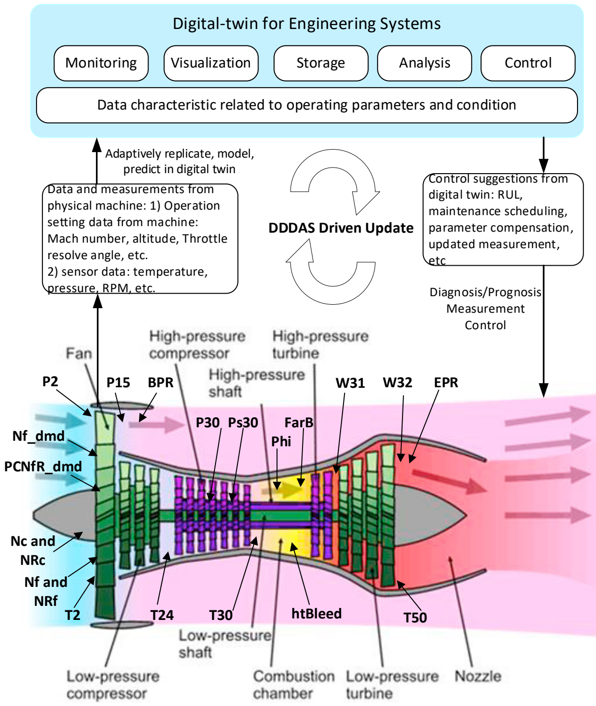


Fig. 1. Digital twin of "sensing, monitoring, analysis, and control" for the aircraft engine health management

The framework of proposed dynamic data driven approaches is shown in Fig. 2. A simulation model is set up to gain insight about the engine operation status. Then, this insight is used to determine what new observations should be collected, and the simulation is adapted to reflect these observations. Through measurements from sensors, computing systems identify the data pattern, and update the simulation and machine-learning model in real time. The collected data go through selection and normalization for pre-processing. Selection on the collected data is based on the linear degradation model which examines the trend of sensor measurement towards the degradation. The selected and processed data is fed into machine learning models for RUL estimation. According to literature, the LSTM neural network is selected as the RUL estimation model. The challenges for this digital twin include the development of interfaces to physical devices and the creation of an infrastructure to support the communication and data requirements.

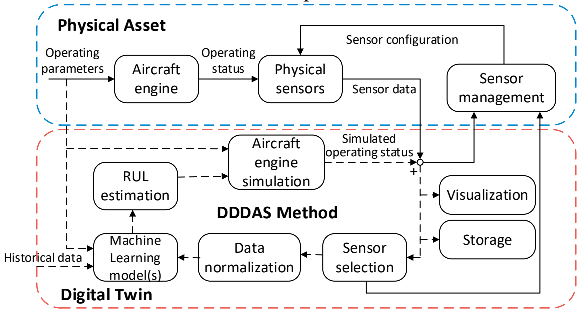


Fig. 2. Framework of DDDAS based digital twin for aircraft engine

3.2 Long short-term memory (LSTM) networks

A long short-term memory (LSTM) network is a deep learning type of recurrent neural network (RNN) [19]. A disadvantage of the RNN is that the vanishing gradients often causes the parameters to capture short-term dependencies while the information from earlier time steps decays. The reverse issue, exploding gradients, may also occur, causing the error to grow drastically with each time step. LSTM networks aim to overcome the issue of the vanishing gradients by using the gates to selectively retain information that is relevant and forget information that is not relevant. Lower sensitivity to the time gap makes LSTM networks better for analysis of sequential data than simple RNNs. LSTMs excel in learning, processing, and classifying sequential data. The typical architecture for an LSTM based deep learning neural network is shown in Fig. 3. The architecture includes: 1) an input layer, 2) an LSTM layer, 3) a fully-connected layer 1, 4) a drop out layer, 5) a fully connected layer 2, and 6) a response layer.

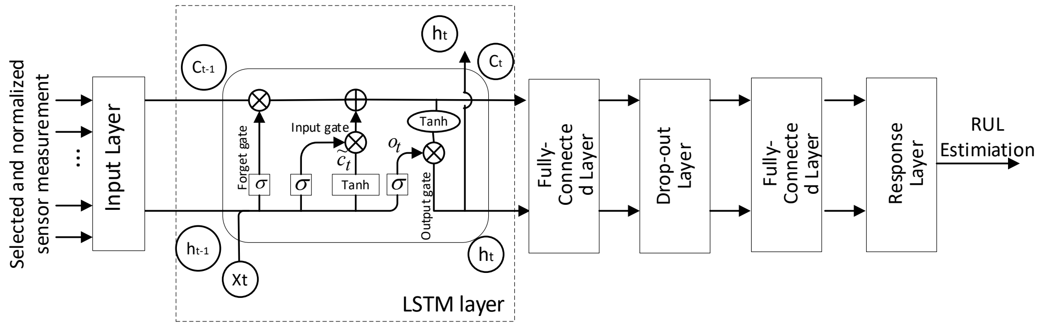


Fig. 3. Architecture of the LSTM based neural network for RUL estimation

The **input layer** contains a sequence of features that is extracted from the time series of sensor measurement.

The **LSTM layer** has a memory cell with two outputs: the long-term state C_t and the short-term state h_t, three control gates: a forget gate f_t, input gate i_t, and output gate o_t, on the state path to control the output [19].

The forget gate f_t controls the information removal from the previous long-term state C_{t-1}:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \quad (1)$$

The input gate i_t controls which values to be updated. Next, a tanh function creates a vector of new candidate values, $\tilde{C}_t$, that could be added to the state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \sigma(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

$$C_t = f_t \otimes C_{t-1} \oplus i_t \otimes \tilde{C}_t \quad (4)$$

The output gate o_t controls the formation of the current short-term state h_t using the information from the current long-term state C_t. The output o_t is computed as:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = O_t \otimes \tanh(C_t) \quad (6)$$

In equations (4) and (6), $\otimes$ and $\oplus$ are element-wise multiplication and addition respectively.

Training of the LSTM network is using the backpropagation through time (BPTT) procedure [19], which is similar to train an RNN.

Fully Connected layers connect the output of the previous layers, "flattens" them and turns them into a single vector that can be an input for the next layer. A fully connected layer finishes the high-level reasoning in the neural network. Neurons in a fully connected layer have connections to all activations in the previous layer. Their activations can thus be computed as an affine transformation, with matrix multiplication followed by a bias offset (vector addition of a learned or fixed bias term).

The **Dropout layer** helps reduce over fitting and improve training speed. At each training stage, individual nodes are either "dropped out" of the net (ignored) with the probability 1-p or kept with the probability p, so that a reduced network remains; incoming and outgoing edges to a dropped-out node are also removed. Only the reduced network is trained on the data in that stage. The removed nodes are then reinserted into the network with their original weights. p usually takes the value of 0.5 for input nodes.

The **Response layer** can return either sequence or state. In this case, it returns sequence, which is the piece-wise RUL of an aircraft engine. The entire degradation can be classified into two stages: normal performance stage showing relative flat part at the first part of the cycles and performance degradation stage showing an exponential drop trend. It is difficult to predict the RUL at the first stage, hence in some literatures such as [13], the RUL is assumed to be constant until it crosses certain cut off limit. Thus, the RUL curve over cycle is then modelled as two piece-wise linear functions. The difference on true and piece-wise RUL is shown in Fig. 4.

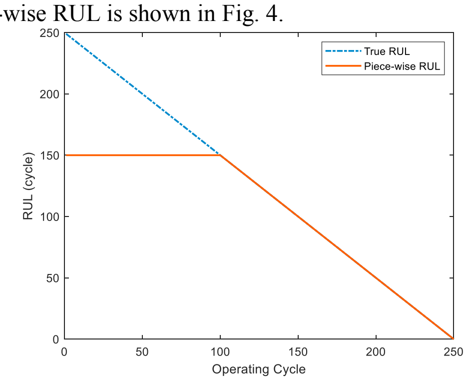


Fig. 4. Piece-wise RUL (Piece-wise threshold RUL is 150 flight cycles)

4. Case study

In order to illustrate the proposed approach on dynamic data driven digital twin on aircraft engine health management, a case study was developed as below. This case study is based on the famous NASA C-MAPSS dataset, which tracks performance delegation and predicts RUL of a turbofan jet engine [20].

In the proposed digital twin, 21 sensors were deployed on a jet engine at different parts, where at fan, low-pressure compressor (HPC), high-pressure compressor (HPC), low-pressure turbine (LPT), and high-pressure turbine (HPT). The sensor deployment is tabulated in Table 1 and shown in Fig. 1. These sensors collect time series of measurement data under a number of operating conditions by varying operate settings on altitude, Mach number, and throttle resolver angle. The dataset FD004 in C-MAPSS is used as an example to illustrate the proposed method. In the FD004 dataset, there are total 249 trajectories (units) of data on the engine operation. The engine was operated under six different operating conditions. When collecting FD004, the engine suffered from two fault modes of HPC degradation and/or fan degradation.

Table 1. Deployed sensors and measurements [20]

Sensors	Measurements	Sensors	Measurements
T2	Total temperature at fan inlet (◦R)	Ps30	Static pressure at HPC outlet (psia)
T24	Total temperature at LPC outle(◦R)	Phi	Ratio of fuel flow to Ps30 (pps/psi)
T30	Total temperature at HPC outlet(◦R)	NRf	Corrected fan speed (rpm)
T50	Total temperature at LPT outlet(◦R)	NRc	Corrected core speed (rpm)
P2	Pressure at fan inlet (psia)	BPR	Bypass ratio
P15	Total pressure in bypass-duct (psia)	farB	Burner fuel-air ratio
P30	Total pressure at HPC outlet (psia)	htBleed	Bleed Enthalpy
Nf	Physical fan speed (rpm)	Nf_dmd	Demanded fan speed (rpm)
Nc	Physical core speed (rpm)	PCNfR_dmd	Demanded corrected fan speed (rpm)
EPR	Engine pressure ratio (P50/P2)	W31	HPT coolant bleed (lbm/s)
		W32	LPT coolant bleed (lbm/s)

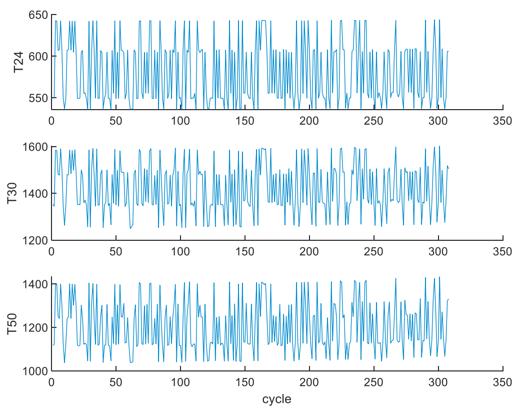
4.1 Data collection and selection

Among the 21 sensor measurements, features have to be extracted to reduce the dimensionality of the raw data collected and to improve the performance of the machine learning models used for creating the digital twin. The feature selection and reduction reduces the computational complexity and avoid the "curse of dimensionality", which is a phenomenon that the amount of data needed to support the result often grows exponentially when the dimensionality increases, for the machine-learning algorithm. Instead of using the conventional features reduction and selection approach such as principle component analysis (PCA), a linear degradation model as equation (7) [21] was applied for sensor selection. The linear degradation selection approach estimates and ranks the slope of each sensor measurement's trend towards the degradation.

$$S(t) = \phi + \theta(t) \cdot t + \varepsilon(t) \quad (7)$$

In this equation, t is the time, ϕ is the model intercept, and $\theta(t)$ is the model slope, which are estimated using sensor data regarding the health of an ensemble of similar components such as multiple engines with the same specification. $\varepsilon(t)$ is the model additive noise.

Through the linear degradation models, eight (8) sensor measurements with the most trendable $\theta(t)$ towards degradation were selected. These sensor measurements are: [T24, T30, T50, Nf, Nc, NRf, Ps30, htBleed]. The data acquired from the selected sensors on the engine operation is shown in Fig. 5.



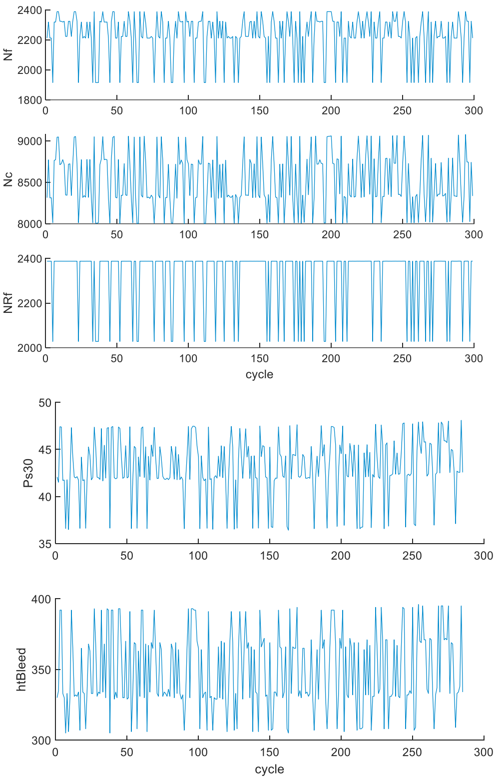


Fig. 5. Acquired raw signal from selected sensors

4.2 Data normalization

Note that the FD004 was collected under six (6) different operating conditions, which would cause data pattern changes. Data normalization is a way to eliminate the effect from different operating conditions. First, the data was grouped into six regimes using the k-means clustering with the squared Euclidean distance metric. The clusters and centroids of the operations are illustrated in Fig. 6.

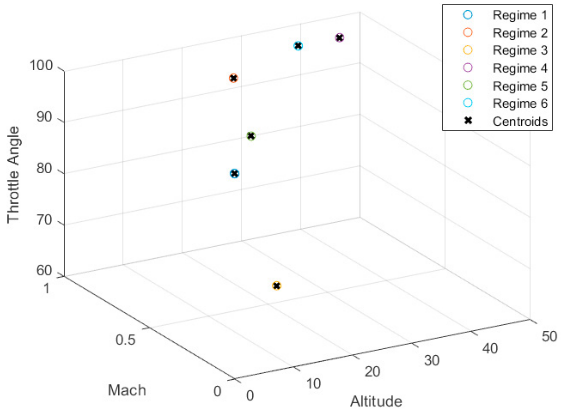


Fig. 6. Clusters of the trajectory data

Then, z-score normalization was performed to normalize the acquired sensor data for each engine according to the regime that it is clustered to. For the i-th engine, the z-score normalization for the sensing data is expressed as equation (8).

$$x_{normed}(i,j,l)=\frac{x(i,j,l)-mean(x(l,r))}{std(x(l,r))} \qquad (8)$$

where $x(i,j,l)$ represents the j-th data point from the l-th sensor in the i-th engine, and the expressions $mean(x(l,r))$ and $std(x(l,r))$ are the mean and standard deviation values on all the data points from the l-th sensor in r-th regime (r = 1,2…6), where the i-th engine is clustered to.

The normalized signal from the selected sensors are shown in Fig. 7.

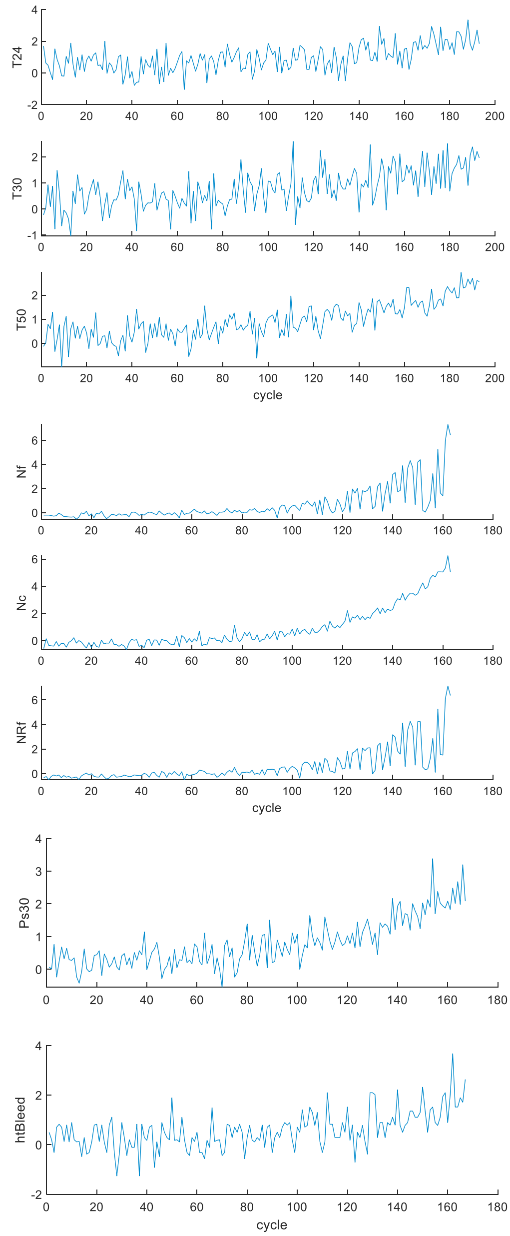


Fig. 7. Normalized signal from selected sensors

4.3 RUL prediction with selected normalized signals

The normalized selected signals are input to the LSTM model for RUL estimation. Parameters used in the LSTM model are shown below in Table 2. The LSTM was implemented on a laptop with i7 CPU at 2.60 GHz, and 8.00 GB of memory. The input was randomly divided into 80% vs 20% for training and validation. The training and validation progress and performance on root mean square error (RMSE, as defined in equation (9)) and loss are shown in Fig. 10. The error histogram in test samples is shown in Fig. 8. This test case has a very low RMSE of 7.17.The RUL estimation on four randomly selected test units are illustrated in Fig. 9, which shows the RUL estimation closely follow the test data degradation.

Table 2. Parameters used in the LSTM model

No. of feature	Hidden units	Learning rate	Max epochs	Batch size	Units in fully connected layer
8	200	0.01	60	20	50

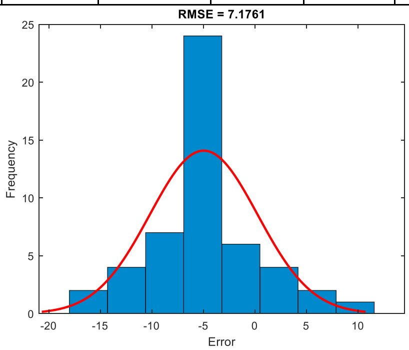


Fig. 8. RMSE for LSTM modelling of engine RUL

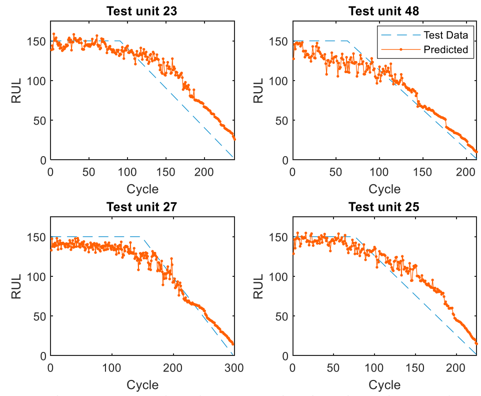


Fig. 9. RUL estimation on randomly selected test units

$$RSME = \sqrt{\frac{\sum_{i=1}^{n}\left(\hat{RUL_i} - RUL_i\right)}{n}} \quad (9)$$

In this equation, $\hat{RUL_i}$ is the estimated RUL for the true RUL on the i-th engine.

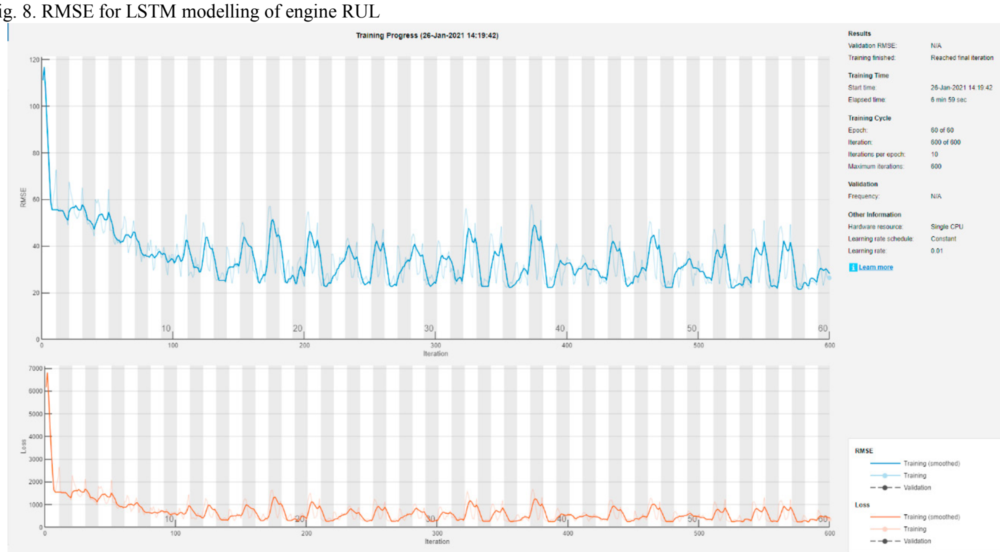


Figure 10. LSTM training and validation progress on the RMSE and loss

4.3 Performance evaluation and comparison

Goodness-of-fit statistics for the RUL estimation in arithmetic scale were performed using K-fold cross validation (specifically 5-fold) on two performance metrics: 1) mean Root Mean Square Error (RMSE, as equation 9), 2) mean s-score for degradation applications.

k-fold cross-validation randomly partition the original sample into k equal sized sub-samples. Over the *k* subsamples, a single sub-sample is used as the validation data for testing the model, and the remaining (*k*-1) sub-samples are used for model training. The cross-validation process is then repeated *k* times, with each of the *k* subsamples used exactly once as the validation data. The *k* results can then be averaged to produce a single estimation. Usually *k* takes the values of 5 or 10.

Further, for the degradation scenario, an early prediction is preferred over late predictions. Therefore, the s-score as defined in equation (10), which is asymmetric around the true time of failure such that late predictions were more heavily penalized than early predictions [20], is selected as the second evaluation metric.

s-score [14]:

$$s = \begin{cases} \exp\left(-\frac{d_i}{13}\right) - 1, & for d_i < 0 \\ \exp\left(\frac{d_i}{10}\right) - 1, & for d_i \geq 0 \end{cases} \quad (10)$$

Here, $d_i = \hat{RUL_i} - RUL_i$ (estimated RUL-true RUL).

Using the 5-fold cross validation, the proposed LSTM model was compared using other approaches including similarity based linear regression RUL model and feedforward neural network (FFNN). The similarity linear regression RUL model was built according to [22] and [23]. The FFNN has a structure of 8-15-1 on the input layer, hidden layer, and output layer. The comparison results are summarized in Table 3. The LSTM model does show better performance than the other two approaches. This is due to LSTM's specialty in discovering the underlying patterns embedded in time sequences.

Table 3. Performance comparison RUL models on LSTM, similarity based linear regression, and FFNN

	LSTM	Similarity based linear regression	FFNN
Mean RMSE	20.22	190.38	23.30
Mean s-score	25.27	1.37e12	59.42

5. Conclusion and future work

A dynamic data driven digital twin was proposed to model the relationship between engine remaining useful life (RUL) with simulation model and sensor signals for engine operation status monitoring. This approach starts by deploying various sensors to monitor the running condition of both cyber and physical domains. Then, with sensor measurements selected from linear degradation models, a LSTM neural network is presented to dynamically update the digital twin, which is able to check the most up-to-date RUL of the physical aircraft engine. On the modelling performance, it was seen that LSTM, indeed, has better performance than other RUL models including similarity based linear regression and FFNN on performance metrics of RMSE and s-score. The proposed dynamic data driven digital twin framework is promising for complex engineering products. This digital twin based RUL technique is also potential for manufacturing equipment health management.

For future research, further investigation of performance degradation is suggested. When working conditions change, the distribution of the source domain data (on which the model is trained) is different from the distribution of the target domain data (where the learned model is actually deployed), which leads to performance degradation. Adapting the machine learning model trained in a source domain for use in a different but related target domain also can be addressed in the future.

Another future investigation that deserves exploration is the implementation of this digital twin framework to replicate manufacturing systems such as CNC machines and assembly lines etc., when remote operations are preferred. As the impact the current COVID-19 pandemic continues to grow globally, the proposed digital twin system could be helpful for scenarios where direct machine interaction from operators is limited. The sensors and IIoT can be deployed to acquire data, visualize system operations, extract key information from the data, and then make decisions on operation status of manufacturing systems. That will be a key enabling technology for smart manufacturing with the aim to improve the response and resilience of manufacturing systems.

CrediT author statement

ZW: Conceptualization, Methodology, Software, Writing-draft and revision. JL: Writing-reviewing and editing.

Acknowledgements

The author ZW would like to acknowledge support from NASA (award number: 80NSSC20M0015) and NSF (award number: 1818655). The author JL would like to acknowledge support from ONR (award number: N00014-19-1-2728). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NASA, NSF, and ONR.

References

[1] Grieves, M. J. Vickers, Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems, in Trans-Disciplinary Perspectives on System Complexity, F.-J. Kahlen, S. Flumerfelt, and A. Alves, Editors. 2016, Springer: Switzerland. p. 85-114

[2] DHL Trend Research, Digital twins in logistics-a DHL perspective on the impact of digital twin on the logistics industry, 2019

[3] Marr, B. (2017). What is digital twin technology–and why is it so important. https://www.forbes.com/sites/bernardmarr/2017/03/06/what-is-digital-twin-technology-and-why-is-it-so-important/?sh=7d7cd7572e2a

[4] E. J. Tuegel, A. R. Ingraffea, T. G. Eason, S. M. Spottswood, Reengineering aircraft structural life prediction using a digital twin, International Journal of Aerospace Engineering, Volume 2011, 154798, 14 pages

[5] E. H. Glaessgen, D.S. Stargel, The digital twin paradigm for future NASA and U.S. Air Force vehicles, the 53rd Structures, Structural Dynamics, and Materials Conference: Special Session on the Digital Twin, 2012, page 1-14

[6] F. Darema, InforSymbioticSystems/DDDAS-Large-Scale Dynamic Data and Large-Scale Big Computing for Smart Systems, 2015 IEEE 22nd International Conference on High Performance Computing Workshops

[7] S. R. Chhetri, M. A. Al Faruque, Dynamic data-driven digital twin, Chapter 7 in "ModelingData-Driven Modeling of Cyber-Physical Systems using Side-Channel Analysis", Springer, 2020

[8] Z. Wu, Cutting tool condition monitoring and prediction based on dynamic data driven approaches, ASME 10th International Manufacturing Science and Engineering Conference, Charlotte, North Carolina, 2015

[9] R. Fujimoto, N. Celik, Haluk, Damgacioglu, M. Hunter, D. Jin, Y-J Son, J. Xu, Dynamic data driven application systems for smart cities and urban infrascture, Proceedings of the 2016 Winter Simulation Conference

[10] Mark Gaynor, Margo Seltzer, Steve Moulton, Jim Freedman, A dynamic data driven decision support system for emergency medical services, ICCS 2005, LNCS 3515, pp. 703 – 711, 2005

[11] Y. Badr, S. Hariri, Y. AL-Nashif, E. Blasch, Resilient and trustworthy dynamic data-driven application systems (DDDAS) services for crisis management environments, Procedia Computer Science, 51, 2015, 2623-2637

[12] R. Fujimoto, J. Barjis, E. Blasch, W. Cai, D. jin, S. Lee, Y-J Son, Dynamical data driven application sysytems: research challenges and opportunities. Proceedings of the 2018 Winter Simulation Conference

[13] J. Zhang, P. Wang, R. Yan, R. X. Gao, Long short-term memory for machine remaining life prediction, Journal of Manufacturing Systems, 48 (2018) 78–86

[14] Z. Li, K. Goebel, D. Wu, Degradation modeling and remaining useful life prediction of aircraft engines using ensemble learning, ASME Journal of Engineering for Gas Turbines and Power, 2019, Vol. 141 / 041008 1-10

[15] Y. Fan, S. Nowaczyk, T. Rognvaldsson, Transfer learning for remaining useful life prediction based on consensus self-organizing models, Reliability Engineering & System Safety, 203, 2020, 107098

[16] A. Zhang, H. Wang, S. Li, Y. Cui, Z. Liu, G. Yang, and J. Hu, Transfer learning with deep recurrent neural networks for remaining useful life estimation, Applied Science 2018, 8, 2416; doi:10.3390/app8122416

[17] Mohamad Danish Anis, Sharareh Taghipour, hi-Guhn Lee, Optimal RUL estimation: a state-of-art digital twin application, 2020 Annual Reliability and Maintainability Symposium (RAMS)

[18] A. Parrott, L. Warshaw, Industry 4.0 and the digital twin: manufacturing meets its match, Deloitte University Press, May 2017.

[19] Greff K, Srivastava RK, Koutnik J, Steunebrink BR, Schmidhuber J. LSTM: a search space odyssey. IEEE Trans Neural Networks Learn Syst 2017; 28: 2222–32.

[20] A. Saxena and K. Goebel (2008). Turbofan engine degradation simulation data set, NASA Ames Prognostics Data Repository, (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA

[21] Chakraborty, S., N. Gebraeel, M. Lawley, and H. Wan. Residual-life estimation for components with non-symmetric priors. IIE Transactions. Vol. 41, Number 4, 2009, pp. 372–387

[22] T. Wang, J. Yu, D. Siegel, and J. Lee, A similarity-based prognostics approach for remaining useful life estimation of engineered systems, 2008 International Conference on Prognostics and Health Management

[23] https://www.mathworks.com/help/predmaint/ug/similarity-based-remaining-useful-life-estimation.html