Deep Learning Approaches for Condition Monitoring and Prognostic Analysis

by

Zunya Shi

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Systems Engineering)
in the University of Michigan - Dearborn
2021

Doctoral Committee:

Assistant Professor Abdallah Chehade, Chair
Assistant Professor Fred Feng
Assistant Professor Zhen Hu
Associate Professor Samir Rawashdeh

Zunya Shi

zunyas@umich.edu

ORCID iD: 0000-0003-1883-2617

Dedication

To my beloved Yongfeng, Olivia, and parents.

Acknowledgements

It would not have been possible to complete this doctoral thesis without the help and support of the kind people around me, to only some of whom it is possible to give mention here.

Above all, I would like to thank my advisor and mentor, Professor Abdallah Chehade, for his guidance and wisdom throughout my Ph.D. program, as well as his incredible and unwavering patience in supporting me over the past 4 years. I have continuously kept learning from his unsurpassed knowledge and insightful suggestions, which have made a significant impact in my research and life.

I also wish to express my warm and sincere thanks to my committee members, including Professor Zhen Hu, Professor Fred Feng, and Professor Samir Rawashdeh. This thesis would not have been possible finished without their valuable advice and productive discussion.

I would like to thank and acknowledge the Department of Industrial Manufacturing and Systems Engineering, University of Michigan Dearborn for their great support in finance and academy since I started my doctoral program in 2017. My gratitude goes for the chair of Department, Professor Armen Zakarian, the Ph.D. program director Professor Yubao Chen, Professor Xi Chen, Professor Jian Hu, Professor Bochen Jia, and Professor Sang-Hwan Kim. Their good advice and guidance have been invaluable on both academic and personal level, for which I am extremely grateful.

Personally, I would like to always thank my husband Mr. Yongfeng Li for his personal support and great patience. My parents Mr. Qingan Shi and Ms. Guangping Gong have also given me their continuous encouragement and unequivocal support throughout my study period. Also, special thanks to Olivia Ming Li, for being my daughter for my entire life.

Table of Contents

List of Tables

List of Figures

Abstract

Huge amount of sensor signals facilitates the condition-based maintenance (CBM). Prognostic analysis, one of the most important tasks of CBM, is to predict when and how likely a potential failure will occur based on sensor signals. Although many data-driven methods have been proposed and made significant achievements, there are still many limitations to overcome. Frist, most existing methods usually underperform to learn the complex patterns of sensor signals. Second, traditional methods have limited abilities of handling large-scale sensor signals. Last, most existing methods require expertise to extract high-level features from sensor signals. To address these challenges, many researchers have applied Deep Learning (DL) technologies for prognostic analysis. DL methods can extract more complicated patterns of sensor signals. Besides, they can also significantly improve the performance with large-scale sensor signals those traditional methods cannot handle.

Therefore, this thesis mainly focuses on developing high-performance DL approaches for condition monitoring and prognostic analysis. These developed DL approaches will (i) monitor and assess the health condition of the system accurately, (ii) model the degradation process of the system, and (iii) predict the behavior of system in the future, including the remaining useful life (RUL).

In the third Chapter of this thesis, we investigate how to fuse sensor signals as one-dimensional health index to accurately indicate the health condition and predict the RUL of the machine system. We propose that the health index should be separable enough to help differentiate between any distinct degradation states. Accordingly, we develop a novel sensor fusion framework to create the health index and check the separability of the health index for prognostic analysis through a series of statistical hypothesis tests. Based on the health index constructed, the remaining time to various degradation states can be accurately inferred.

In the fourth Chapter, we develop a coupled duration dependent hidden semi-Markov model (CDD-HSMM) for sensor fusion to better recognize the health condition of the system. CDD-HSMM contains two correlated semi-Markov chains. The state transition of each stochastic

process is not only dependent on itself but also dependent on the other a nonlinear way. And the nonlinear correlations between different sensor signals can be captured by the CDD-HSMM, which is a big advantage that most existing fusion methods don't have.

In the fifth Chapter, we explore whether it is possible to improve the accuracy of RUL prediction by detecting the change point in advance, the time point when the system transits from the normal operation to degradation process. We try to improve the RUL prediction accuracy by filtering out the sensor signals irrelevant to the degradation process before the change point. Consequently, we propose a Dual-LSTM framework which can detect the change point and predict the RUL in a sequential manner with multi-sensor signals. The Dual-LSTM framework (i) characterizes the complex temporal patterns within each sensor, (ii) captures the correlations across different sensors, and (iii) allows for change point detection for better real-time RUL prediction.

In the sixth Chapter, we apply DL approaches to SOC estimation of the lithium-ion (Li-ion) batter cells. Although there are existing DL technologies available for estimating SOC of battery cells, most of them usually ignore the impact of previous charging and discharging cycles on the accuracy of SOC estimation. To overcome this limitation, we develop new RNN and LSTM networks which leverage historical information from previous cycles to achieve on-line high-precise SOC estimation.

In the seventh Chapter, we develop an integrated and accessible AutoML website-based application that allows users to conduct data analysis with Machine Learning (ML) and DL algorithms. The AutoML contains each step of the workflow and provides comprehensive built-in ML and DL algorithms in a single toolset. Most importantly, the user-friendly interface also makes it much easier for users to analyze dataset with much less effort.

In summary, this thesis contributes to the DL approaches for high-performance condition monitoring and prognostic analysis given large-scale multi-sensor signals. The developed methods can be applied to various applications to support smart maintenance scheduling, production, and recourse planning, etc.

Chapter 1 Introduction

1.1 Motivation and Overview

Reliability, productivity, and cost are major concerns for most industries. However, as time goes by, machine component or system will gradually deteriorate and even go through some unexpected failures. The unexpected failures will increase downtime and delay delivery during the production, which finally lead to severe economic losses [1]. Maintenance is an effective way to prevent failures and assure a satisfactory level of reliability during the useful life of the machine system [2]. Traditional maintenance strategies, such as corrective maintenance and time-based preventive maintenance, are performed either at breakdown or at a regular interval regardless of the health condition of the machine system. These techniques are easy to apply but may lead to unnecessary downtime and increase the cost. As the rapid development of the sensing technologies, a huge number of sensors are introduced to real-time monitor the health condition of the machine system. In this case, a new strategy, condition-based maintenance (CBM), is proposed to maximize the effectiveness of maintenance by making decisions based on the sensor signals collected through condition monitoring [3]. Figure 1-1 shows three key steps of CBM, which are data acquisition, data processing, and decision making. Data acquisition is to obtain the sensor signals monitoring the health condition of the machine system. Data processing is to analyze the sensor data to better understand the degradation process. Finally, based on the information acquired through data processing, decision makers are able to draw up an efficient maintenance schedule [2].As one of the most important tasks of CBM, prognostic analysis aims to predict when and how likely the

Figure 1-1 CBM working process

upcoming failure will occur, or the remaining useful life (RUL) of the machine system [4]. An accurate RUL prediction helps to raise early alarms to avoid unexpected failures, improve production efficiency, and reduce life-cycle costs. Therefore, it has received a great attention and developed rapidly under such a data-rich environment. Although there are many researches have been explored, prognostic analysis is still an emerging and challenging field [4]. This is because, first, most modern manufacturing systems are complex which may be composed of hundreds, even thousands of components or sub-systems. And the system may be conducted under different modes and operation conditions. All of these will increase the difficulty of prediction. Second, the massive sensor data also brings out lots of problems, such as how to remove the noise and extract real relevant information from the dataset, how to determine the complex correlations between sensors, how to model the degradation process based on the multiple sensor signals, etc. Therefore, we need more powerful prognostic models to improve the accuracy of RUL prediction, which is the major objective of this thesis.

1.2 Traditional Data-Driven Prognostic Analysis

The massive sensor signals provide a huge opportunity to the CBM and data-driven prognostic analysis, but also bring out some challenging problems to be solved. In most practical cases, not all sensors are relevant to the underlying degradation mechanism. Some sensor signals show strong degradation trends while others do not [5]. In this case, it is necessary to determine which sensors are appropriate for prognostic analysis. Besides, there may exist complicated correlations between different sensors. It is highly possible in practice that some sensor signals share the same patterns or characteristics of the degradation process. Therefore, it's important to select and combine/fuse multiple sensors before we develop the prognostic model to achieve more accurate RUL prediction results. As a consequence, traditional data-driven approaches of prognostic analysis usually consist of two key parts, sensor selection/fusion, as well as model training [6], [7]. First, the sensor signals which contain information relevant to the degradation process will be selected or fused as high-level features (health index for example). Second, the selected/fused features are further provided to statistical or machine learning models to predict the RUL of the machine system.

With selected sensors or fused high-level features, researchers have proposed many statistical or machine learning methods for RUL prediction. These data-driven prognostic methods can be

roughly classified into four categories, regression-based models, Gamma process-based models, Wiener process-based models, and Markovian process-based models [8], [9]. For example, Peng Guo and Nan Bai [10] proposed a new condition monitoring method, which uses an auto-associative kernel regression (AKKR) to model the normal behavior of the gearbox and utilizes a moving window statistical method to detect the failure. Jian Liu *et al.* [11] proposed a method which combines indirect health indicator and multiple Gaussian process regression (GPR) model to predict the RUL of lithium-ion batteries under the situation when the capacity of the operating battery is not measurable. Racha Khelif *et al.* [12] developed a support vector regression model to predict the RUL of the equipment directly from sensor values without the need for degradation state estimation or a predetermined failure threshold. Shah Limon and Om Prakash Yadav [13] considered to use a multi-stress dependent gamma process to model the degradation process of the complex engineering system. T.B. Lien Nguyen *et al.* [14] presented a three-step data-driven prognostic method for batch manufacturing processes (BMP). This method constructed a health index to represent the operating state of the system at first, and then isolated the useful information of the health index from noises. Finally, the health index was modeled by Gamma process to predict the RUL of the system. Qingqing Zhai and Zhisheng Ye [15] proposed a new Wiener process model utilizing Brownian motion for the adaptive drift to predict the RUL of deteriorating products. Shengjin Tang *et al.* [16] developed a novel RUL prediction model for lithium-ion batteries based on the Wiener process with measurement error (WPME). Xiaopeng Xi *et al.* presented an improved non-Markovian degradation model with both long-term dependency and item-to-item uncertainty for high-precision RUL prediction. In addition to those four types of models, there are some other machine learning methods also used for RUL prediction. For example, R. Jegadeeshwaran and V. Sugumaran [17] uses decision tree to select statistical features at first, then detects faults with support vector machine (SVM). Selina S. Y. Ng *et al.* [18] proposed a naïve Bayes (NB) model to predict the state-of-health (SOH) of lithium-ion batteries under different usage conditions and ambient temperatures.

Although these traditional data-driven approaches have been proved to be powerful solutions for prognostic analysis and easy to apply, they are still limited in addressing large-scale sensor signals. Besides, they usually underperform when the machine system works under various operating conditions or multiple failure modes. Therefore, some researchers have started to utilize DL

techniques for prognostic analysis, trying to capture more complicated relationships between the sensor signals and the RUL to improve the performance and efficiency.

1.3 Deep Learning Techniques for Prognostic Analysis

During the last few years, DL technologies have grown rapidly and attracted great attention due to their excellent capacities of addressing large-scale data. Under such a data-rich environment, modern manufacturing system has started to embrace the big data revolution and applied DL techniques in the field of prognostic analysis. Until now, various deep neural networks have been proposed to achieve high-precision RUL prediction. Three most widely used networks are recurrent neural network (RNN), convolutional neural network (CNN), and autoencoder (AE). For example, Levent Eren *et al.* [19] developed an adaptive CNN classifier to detect different types of fault conditions of bearings. Heimes [20] utilized recurrent neural network (RNN) to predict the RUL of turbofan engines. Ghate *et al.* [21] applied multilayer perceptron (MLP) to detect the faults of induction motors. Lei Ren *et al.* [22] used CNN to predict the RUL of bearings. Shaopeng Dong

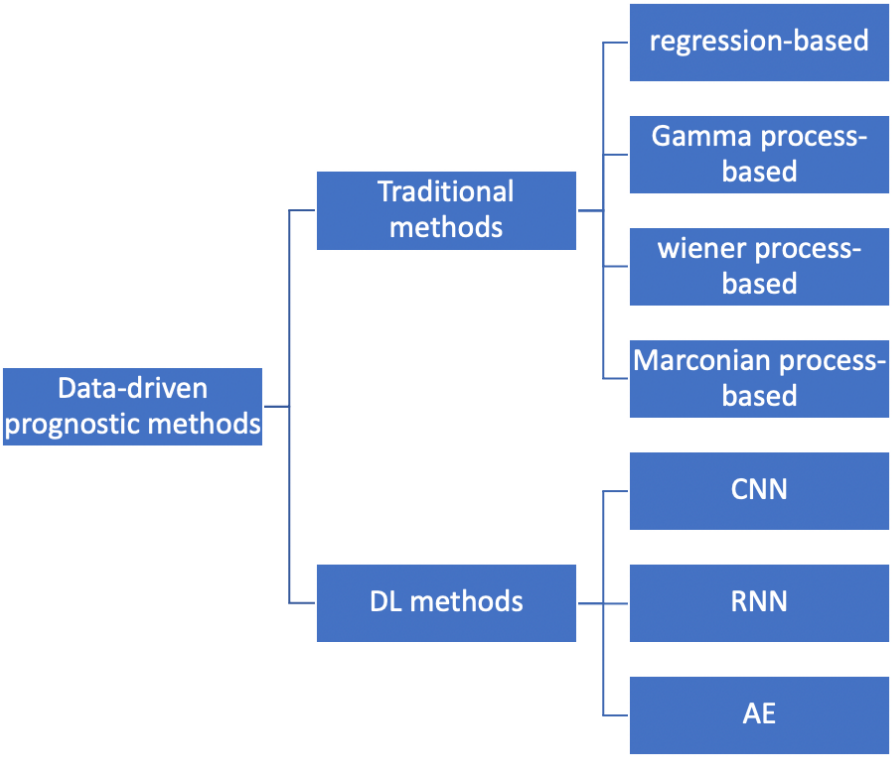


Figure 1-2 Hierarchy of data-driven prognostic approaches

et al. [23] utilized LSTM network (a variant of RNN) to predict the RUL and tested the performance on different case-studies. Long Wang *et al.* [24] introduced a deep autoencoder to derive an indicator from available sensor signals to present the trend of the blade breakage. Then, the exponentially weighted moving average (EWMA) was applied to monitor the variation and detect the shift of the indicator trend.

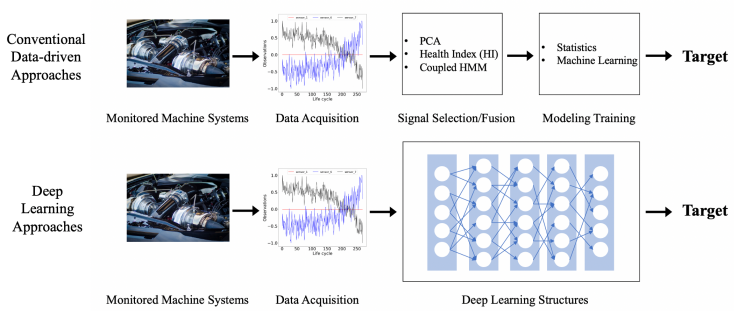


Figure 1-3 Traditional data-driven and deep learning approaches for prognostic analysis

Figure 1-3 presents the working process of traditional data-driven approaches and DL approaches respectively. From Figure 1-3, we can see that there are many differences between two approaches. Compared with traditional data-driven approaches, DL approaches are end-to-end models and don't require to select or fuse sensors manually according to expertise or prior knowledges. They can learn the latent representations of sensor signals automatically from input data and targets [25]. In this case, a great deal of human labors can be saved, and the problem of lacking expertise can be solved. And above all, DL approaches proved to be able to achieve superior performances compared with traditional data-driven methods [25]. The comparison between two approaches is summarized in Table 1-1.

Table 1-1 Comparison between traditional data-driven approaches and DL approaches

Traditional data-driven approaches	DL approaches
• Expertise and prior knowledge required • Multiple steps to achieve the target • Hard to capture complicated relationships between input data and targets • Limited on addressing large scale data	• End-to-end structure without intermediate step to select/fuse sensor signals • Capability of capturing complicated relationships between input data and targets • Good at addressing large scale data

1.4 Problem Statement and Objectives

DL is a powerful solution for prognostic analysis, but not a skeleton key. It is not appropriate to use DL as a "black box", where sensor signals come in and prediction results come out directly. Domain knowledge is still important and contributes to the success of condition monitoring and prognostic analysis, but it is often ignored by most existing papers. For example, machine systems usually start at a normal operation level and begin to degrade at an uncertain change point. As Figure 1-4 shows, the degradation trend should be divided into two stages, healthy stage before the change point, and unhealthy stage after the change point. The sensor signals at the healthy stage provide very limited information about the degradation process, which are redundant for analyzing. [26] and [27] demonstrated that one major step before prognostic analysis is to detect the change point and divide the signals into two stages as Figure 1-4 shows.

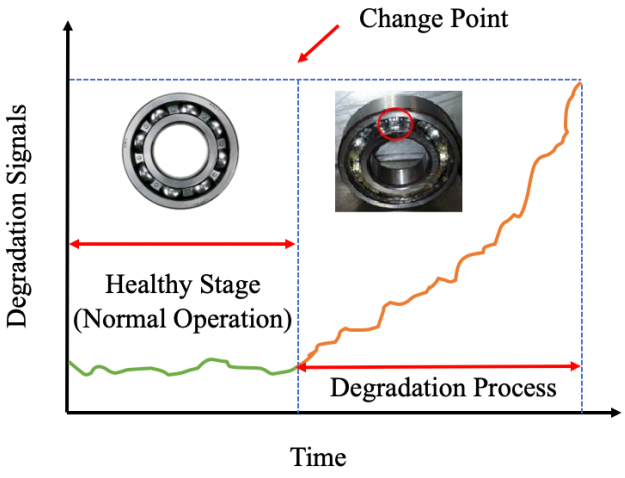


Figure 1-4 Health stage division of degradation process

Therefore, this thesis mainly focuses on using domain knowledge to develop more powerful and targeted DL approaches for prognostic analysis and applying them in different application areas. Besides, we also investigate how to fuse/combine multiple sensor signals and evaluate the quality of signal to provide more accurate and robust inferences for degradation modeling and prognostic analysis. Finally, we also build a web-based application to make it easier for non-technical users to solve practical problems with ML and DL technologies.

Chapter 2 Literature Review

Many research efforts have been made for prognostic analysis to improve the safety, reliability, and productivity of machine systems. Due to the rapid development of sensing technology, large-scale sensor signals are generated to monitor the health condition of the machine system, which offers an unprecedented opportunity for data-driven prognostic methods. This Chapter reviews the existing relevant techniques in this field, which involves sensor fusion, health index construction, statistical and ML based degradation models, as well as DL techniques for prognostic analysis.

2.1 Sensor Fusion and Health Index Construction

Sensor fusion is a technology to combine the information from different sensors to form a robust and accurate reading for a more accurate prognostic analysis [2], [28] [29]. Most existing sensor fusion methods can be generally classified into three categories, which are decision-level fusion, feature-level fusion, and data-level fusion [30]. Decision-level fusion targets on fusing prognostic results [5], while feature-level fusion integrates high-level features extracted from raw sensor data [31], [32]. More details about these two fusion methods can be found in [33]. Considering that the performance of these two methods are highly dependent on the data quality, and they also require prior knowledge about how to extract features and combine decisions, more and more researchers have focused on developing data-level fusion methods to combines raw sensor signals directly [5].

Most existing data-level fusion methods combine multiple sensor signals by constructing a composite health index to indicate the health status of the machine system. Kaibo Liu *et al.* [34] is the first one who suggested two properties that degradation signals should have, i) the trend of the degradation signal should be monotonic; ii) given the same environment and failure mode, the variance in the failure threshold of different units should be minimal. Based on this idea, Kaibo Liu *et al*. [34] constructed a composite health index via a weighted average of multiple sensor signals to maximize these two properties. Kaibo Liu and Shuai Huang [5] upgraded the fusion method of Kaibo Liu *et al.* [34] by integrating the sensor fusion with degradation modeling in a

unified manner. To further improve the prognostic results, Kaibo Liu *et al*. [35] put forward the third property that the health index should possess, which is iii) given the selected degradation model, the model fitting error for the developed health index should be minimal. Raffaele Gravina *et al.* [36] defined a signal quality metric for prognostic analysis which is similar in nature to the signal-to-noise ratio (SNR) metric. However, all these methods mainly focus on predicting when the constructed health index will reach to the failure threshold (RUL) but ignore the importance of the remaining time to different degradation states (health status). In this case, Abdallah Chehade and Zunya Shi [37] developed a novel sensor fusion framework to estimate the remaining time to each degradation state including the failure state. A new health index was constructed with this

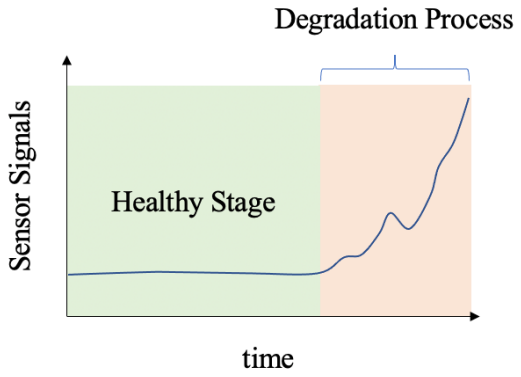


Figure 2-1 Degradation process with 2 HSs

framework by maximizing the separability between any two degradation states via a series of hypothesis testing.

In addition to the method of health index construction, there are other types of data-level fusion approaches. For example, Xiaolei Fang *et al.* [38] fused degradation signals using multivariate functional principal component analysis (FPCA). Ming Dong and David He [39] developed an integrated sensor fusion framework which is based on hidden semi-Markov model (HSMM). Qinyin Chen *et al.* [40] proposed a new sensor fusion method which combines backpropagation neural network (BPNN) and Wavelet Packet algorithm.

2.2 Health Stage Division and Change Point Detection

Health stage (HS) division is to divide the degradation process of the machine system into different HSs [41]. The change point is defined as the time point when the degradation process

transits from one stage to another. The HS is accordingly defined based on which type the degradation trend is. For example, the degradation of rolling element bearing in [42] suddenly appears at some point as Figure 2-1 shows. Therefore, the authors in [42] divided the degradation process into two HSs, which are healthy stage and degradation stage. Another example is [43]. In [43], the degradation process of a double row bearing was clustered into 3 health stages, which are normal stage, degradation stage, and failure stage, as Figure 2-2 shows. Considering that the degradation signals show totally different patterns at each HS, we need to detect the change point of degradation signals at first so as to assign different models or strategies accordingly [44].

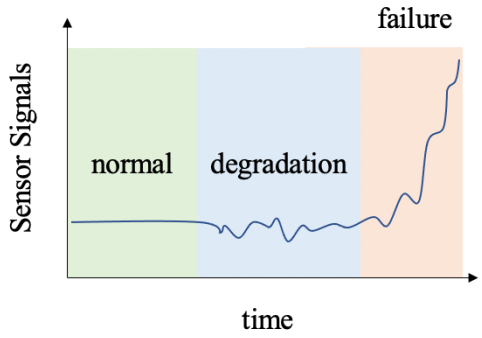


Figure 2-2 Degradation process with 3 HSs

Consequently, researchers have proposed some methods to detect the change point of the degradation process. For example, Wenbin Wang [45] used a control chart to detect the vibration signal variation from its normal values at the healthy stage. Xiaohang Jin *et al.* [46] first transformed the health index to Gaussian distributed data with the Box-Cox method, and then detected the change of health stage when the transformed data went beyond or below 3σ around the mean value. Olga Fink *et al.* [47] divided the degradation process into two classes, pre-defined and predicted time intervals. And they assigned different RUL prediction strategies in these two classes. Yuting Wu *et al.* [48] used the support vector machine (SVM) to detect the change point of the turbofan engine degradation process.

However, most existing change point detection methods are simplistic and highly dependent on population, so that they cannot work well when the machine system is operated under complicated conditions and the degradation trend shows complex patterns.

2.3 Statistical Approaches for Prognostic Analysis

Statistical prognostic approaches predict the RUL by fitting a probabilistic model with available observed sensor data without relying on any comprehensive physical understanding of machine systems [8], [44]. In statistical models, the uncertainties of degradation process caused by various variability sources, like unit-to-unit variability, measurement variability, and temporal variability, are generally described by introducing random variance in model parameters [49]. By doing so, statistical approaches are more robust and powerful for RUL prediction. There are four categories of statistical prognostic approaches. They are Wiener process-based models, Gamma process-based models, Markovian process-based models, and regression-based models [8], [9].

Wiener process is a kind of stochastic process with a drift term and a noise term by Brownian motion, which has been widely used for degradation modeling (monotonic and non-monotonic) [50], [51]. For example, Xiaoshegn Si *et al.* [52] proposed a Wiener-process-based degradation model with a recursive filter algorithm to predict the RUL from degradation data. Qingqing Zhai and Zhisheng Ye [15] proposed a new Wiener process model utilizing Brownian motion for the adaptive drift to predict the RUL of deteriorating products. Shengjin Tang *et al.* [16] developed a novel RUL prediction model for lithium-ion batteries based on the Wiener process with measurement error (WPME).

Gamma process is usually utilized to model the monotonically increasing degradation trend, where the increments of degradation data are assumed to be independent random variables following a Gamma distribution [44], [53]. Jian Liu *et al.* [11] proposed a method which combines indirect health indicator and multiple Gaussian process regression (GPR) model to predict the RUL of lithium-ion batteries under the situation when the capacity of the operating battery is not measurable. Shah Limon and Om Prakash Yadav [13] considered to use a multi-stress dependent gamma process to model the degradation process of the complex engineering system. T.B. Lien Nguyen *et al.* [14] presented a three-step data-driven prognostic method for batch manufacturing processes (BMP). This method constructed a health index to represent the operating state of the system at first, and then isolated the useful information of the health index from noises. Finally, the health index was modeled by Gamma process to predict the RUL of the system.

Markovian process-based models assume that the health stage transformation of the degradation process transform is a Markovian process following the principal of the Markov property [44].

Jeffrey P. Kharoufeh *et al.* [54] presented a semi-Markov degradation model for the degradation of a single unit system. Kong Fah Tee and Ejiroghene Ekpiwhre [55] proposed a Markov chain based degradation model by using a linear transition probability (LTP) matrix method and a median life expectancy (MLE) algorithm. Tongshun Liu *et al.* [56] improved the hidden semi-Markov model (HSMM) by including sequential transition of states and dependent durations of adjacent degradation states to describe the degradation process.

Nowadays, regression methods are popular in many applications and also in prognostic analysis. The most essential assumption of regression methods for prognostic analysis is that the health status or RUL of the machine system can be mapped by degradation signals or constructed health index [44]. There have been many research papers using different types of regression methods to predict the RUL of the machine system. For example, Jianfang Jia *et al.* [57] proposed to predict the state of health (SOH) and RUL of lithium-ion batteries through a Gaussian process regression (GPR) model with three health indicators extracted from raw sensor signals collected during the charging and discharging cycles. Francesco Di Maio *et al.* [58] developed an exponential regression model to predict the RUL of bearings. Peng Guo and Nan Bai [10] utilized an auto-associative kernel regression (AKKR) to model the normal behavior of the gearbox and leveraged a moving window statistical method to detect the failure. Racha Khelif *et al.* [12] developed a support vector regression model to predict the RUL of the equipment directly from sensor values without the need for degradation state estimation or a predetermined failure threshold.

In addition to those four types of prognostic models, there are some other statistical (including machine learning) methods also used for RUL prediction. For example, R. Jegadeeshwaran and V. Sugumaran [17] uses decision tree to select statistical features at first, then detects faults with support vector machine (SVM). Selina S. Y. Ng *et al.* [18] proposed a naïve Bayes (NB) model to predict the state-of-health (SOH) of lithium-ion batteries under different usage conditions and ambient temperatures.

Compared with physical-based models, statistical models for prognostic analysis are easy and quick to apply, but there are still some limitations. First, they are less capable of addressing large-scale sensor signals. Second, they usually underperform when the machine system works under various operating conditions or multiple failure modes.

2.4 Deep Learning Approaches for Prognostic Analysis

During the last few years, DL technologies have grown rapidly and attracted great attention due to their excellent capacities of addressing large-scale data. Under such a data-rich environment, modern manufacturing system has started to embrace the big data revolution and applied DL techniques in the field of prognostic analysis. Until now, various deep neural networks have been proposed to achieve high-precision RUL prediction. Three most widely used networks in the field of prognostic analysis are RNN, CNN, and AE.

RNN is a special neural network, which is designed for handling sequential/time series data. Its cell structure is shown in Figure 2-3, where $\boldsymbol{x}_t$ and $\boldsymbol{h}_t$ are the input and hidden state at time t respectively. As the loop shows in Figure 2-3, the hidden state $\boldsymbol{h}_t$ obtained from previous time based on $\boldsymbol{x}_t$ can be used as the input of the network at the next time step. This kind of recursion enables RNN store the memories in the hidden states for a long time in theory. Therefore, there

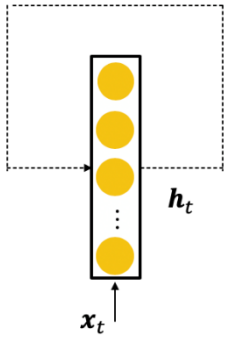


Figure 2-3 RNN cell structure

are some researchers using RNN and its variants, such as GRU and LSTM, to predict RUL of machine systems, since the sensor signals are in natural time series [6]. Heimes [20] is the first one who utilized RNN to predict the RUL of turbofan engines. Liang Guo *et al.* [26] designed a HI based RNN to predict the RUL of bearings. Both Shuai Zheng *et al.* [59] and Shaopeng Dong *et al.* [23] utilized the LSTM network to predict the RUL and tested the performance on different case-studies. Jianjing Zhang *et al.* [60] developed a novel Bi-directional LSTM to predict the RUL based on a dimensionless health index constructed. Elsheikh *et al.* [61] established a new bidirectional handshaking LSTM (BHLSTM) to predict the RUL when given short sequences of

sensor signals with random initial wear. Miao *et al.* [62] designed a dual-task LSTM network to assess the degradation stages and predict the RUL of turbofan engines in a parallel way.

Considering that the multiple sensor data can be presented in 2D format, some researchers proposed to use CNN for prognostic analysis because it shows a great potential to learn complex and robust representations [6]. For example, Levent Eren *et al.* [19] developed an adaptive CNN classifier to detect different types of fault conditions of bearings. Lei Ren *et al.* [22] used CNN to predict the RUL of bearings. Boyuan Yang *et al.* [63] presented a double-CNN model to predict RUL in two stages. First, the change point when the system transforms from normal operation to degradation (referred as IF point in [63]) is identified by the first CNN model. Second, the second CNN model is constructed for RUL prediction. Lei Ren, *et al.* [22] also proposed a deep CNN prediction of bearing RUL.

AE is a powerful network of automatically extracting latent representations from unlabeled data [24], [64]–[66]. A simplified AE structure is shown in Figure 2-4, which contains an encoder and a decoder [65]. The latent representations in low-dimensional space extracted from AE remove the "noise" and keep the most important information of the original inputs. Therefore, AE is actually

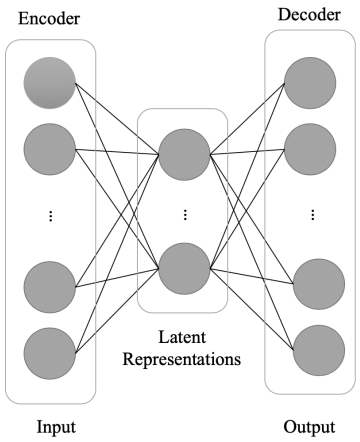


Figure 2-4 Structure of a three-layer AE

a nonlinear dimension reduction technique, which has been widely used for feature extraction to achieve more accurate prognostic analysis. For example, Long Wang *et al.* [24] introduced a deep autoencoder to derive an indicator from sensor signals to present the degradation trend of the blade breakage. Lei Ren *et al.* [67] used the AE to compress the time-domain features of bearing vibration signals. Jian Ma *et al.* [68] extracted degradation features from multiple sensors on the

aircraft engine with the stacked sparse autoencoder. Ya Song *et al.* [69] proposed a hybrid model which integrates AE and bi-directional LSTM to improve the performance of RUL prediction. AE in this paper is used as a feature extractor.

Existing research have proved that DL is a powerful solution for prognostic analysis, they still bring out some challenging problems to be solved. For example, existing deep neural networks are regarded as "black box", of which the inner computation mechanism is invisible [6]. And there is a lack of evidence and theories to explain why these neural networks work. Consequently, DL approaches are effective but unconvincing. Besides, the performance of DL approaches is highly dependent on the size of the data. The larger, the better. However, in most practical cases, the data related to failure are limited. The shortage of failure data is a bottle neck of prognostic analysis [70]. Finally, domain knowledge is important and contributes to the success of prognostic analysis, but it is often ignored by most existing pap ers using DL approaches for RUL prediction.

Chapter 3 Sensor Fusion via Statistical Hypothesis Testing for Prognosis and Degradation Analysis

Due to the rapid development of sensing technologies, multiple sensors became available for real-time monitoring of the degradation status of machine systems. With such a wealth of data collected from multiple sensors, researchers have proposed different sensor fusion approaches for prognosis and condition monitoring, thus allowing accurate inference of the remaining useful life (RUL) of machine systems. However, almost no method provides a statistical metric to evaluate the quality of degradation signals for prognosis and condition monitoring. To fill this literature gap, this paper develops a sensor fusion framework to check the reliability of given degradation signals for prognosis and degradation analysis through a series of statistical hypothesis tests. A health index is constructed in the sensor fusion framework to help differentiate between distinct degradation states. Based on the health index, the remaining times to various degradation states are estimated, including the RUL to failure. A published degradation data set of aircraft engines is used to evaluate and compare the prognostic and condition monitoring performance of the proposed method with benchmark methods.

3.1 Introduction

Prognosis essentially focuses on predicting the RUL of a machine system before a failure occurs, based on the current machine condition and past operation profile [71]–[73]. Prognosis plays a key role in logistics planning, maintenance scheduling, and other production processes because it can improve safety and reliability while reducing life-cycle costs of a machine system [74]. In recent years, advanced sensing technologies have significantly facilitated the development of prognosis. Sensors have been widely used for monitoring the degradation process of complex systems, which creates a data-rich environment for prognostic analysis [38], [75]. Therefore, many researchers have put effort into developing data-driven prognostic models to describe the degradation process and predict the RUL [71], [72], [76].There have been different data-driven prognostic approaches

proposed, such as logistic regression models [77], Gaussian processes [78], [79], hidden Markov models [80]–[82], Kalman filters [83], and so on. However, most of the existing literatures use single-sensor data to develop the prognostic model for machine systems [84]–[86]. This simplified approach may not be valid in many applications because single-sensor data contains limited information, which is usually far from enough to fully characterize the nature of the degradation process and, thus, cannot ensure accurate RUL predictions [34], [75]. Therefore, it is necessary to develop fusion methodologies for multi-sensor prognostic and degradation analysis.

With multiple sensors installed, the data obtained are much richer, thus leading to more accurate RUL predictions. However, it is possible that only a few sensors are related to the degradation process, while others are not [34], [71], [75]. Another possibility is that more than one sensor may capture the same feature of the degradation process, which will result in a redundancy problem [38]. Therefore, one major challenge of sensor fusion for prognosis and degradation analysis is to appropriately combine the information obtained from those sensors. There exist several fusion methods, such as Kalman filter [87], [88], neural network [89]–[91], fuzzy logic [92]–[94], principal components analysis (PCA) [7], [38], locally weighted project regression [95], and so forth. These fusion methods are applied to different fields such as image processing [96], [97], navigation [98], [99], model predictive control [95], and other areas [36], [100]. However, sensor fusion approaches that focus on RUL prognosis and degradation analysis are limited.

Most of the existing data fusion approaches for prognostic and degradation analysis focus on constructing a composite health index from the multi-sensor monitoring data. For example, Liu *et al.* [34] developed a health index fusion model to characterize the performance of the machine system. The authors then estimated the RUL by extending the health index signal until it hits the predefined failure threshold. Also, Liu *et al.* [35] developed a data-level fusion model that constructs the health index with the optimal signal quality for prognosis. Gravina *et al.* [36] defined a signal quality metric for prognosis that is similar in nature to the signal-to-noise ratio (SNR) metric (more details are provided in the literature review). There also exist other approaches that are not based on a composite health index. For example, Fang *et al.* [38] developed a PCA-based prognostic methodology, multivariate functional PCA (MFPCA), to predict the RUL of machine systems with multiple sensors installed. However, such an approach is a generic modeling technique and may not be reliable for degradation analysis in many real-life applications.

Other relevant prognosis approaches are based on key performance indicators (KPIs). For more details, Jiang and Yin [101] investigated recent advances in KPI-oriented prognosis and diagnosis with a MATLAB Toolbox: DB-KIT.

In summary, while there exist many effective sensor fusion techniques, some of them are not statistically intuitive or not specifically designed for prognosis and degradation analysis. In addition, most of the existing studies focus on predicting the RUL before hitting a predefined failure threshold. However, it would be more sensible to construct sensor fusion techniques that can real-time identify the degradation states of a system and predict the remaining time to futuristic degradation states.

In order to make up for these limitations in the existing literature, a novel sensor fusion framework is developed for prognosis and degradation analysis. Specifically, the framework constructs a health index that statistically maximizes the separability between any two degradation states via a series of hypothesis tests. Given the constructed health index, this paper proposes a novel approach to estimate the remaining life to each degradation state including the failure state. The contributions of the proposed sensor fusion framework are: 1) it aims to separate the cycles of the health index signal via a series of statistical hypothesis tests with more focus on separating cycles closer to failure; 2) it defines the degradation (health) states of the system based on the constructed health index, and it allows determination of the probability mass function of health states at any cycle given the multi-sensor data at that cycle; and 3) it predicts not only the RUL of the system but also the remaining time to all states before failure.

3.2 Proposed Sensor Fusion

This section proposes a sensor fusion approach for applications with enough historical units that already failed. Furthermore, the approach is shown to be accurate and reliable for scenarios where units share similar degradation characteristics with some unit-to-unit variability.

3.2.1 Motivation and Problem Formulation

Figure 3-1 and 3-2 show simulated measurements for multiple units collected from two different sensors. Both sensor signals are truncated after ten observations and the tenth observation is the failure observation. Note that the sensor measurements may be pressure, temperature, and so on. The objective of the figures is to visualize that some sensors are more suitable for prognosis and

degradation analysis. The figures show multiple stars at every cycle and each star corresponds to one unit. The cycle values are not necessarily integers, and they can be continuous (e.g., seconds).

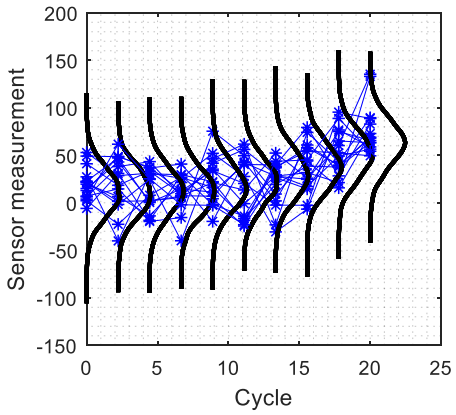


Figure 3-1 Sensor 1 measurements until failure for multiple units and their distributions

The figures also show the empirical distributions for the sensor measurements at each cycle.

From the figures, Sensor 2 shows less variability between different units (i.e., at any given cycle, the stars are closer to each other), and Sensor 2 also shows more separable distributions at different cycle values, especially at cycles that are closer to failure. The above-mentioned observations indicate that Sensor 2 is more reliable to approximate the degradation state than Sensor 1.

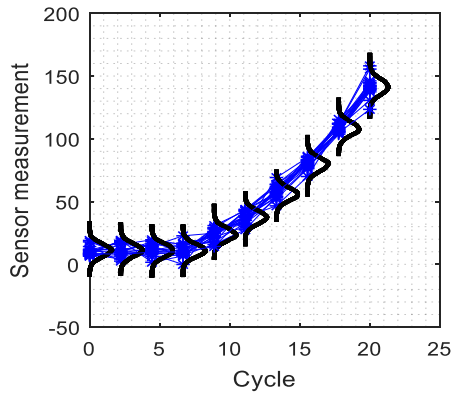


Figure 3-2 Sensor 2 measurements until failure for multiple units and their distributions

Proposition 1:

The more separable the cycles in a signal, the more reliable that signal is for degradation analysis and condition monitoring.

From Proposition 1, the goal is to construct a health index signal via sensor fusion that accurately separates any two cycles in that signal. Furthermore, it is possible to test if any two measurements are separable via hypothesis testing such as the two-sample t -test.

Proposition 2:

Separating consecutive cycles of a signal from one side is sufficient to separate any two cycles in that signal.

Proposition 2 states that it is sufficient to separate consecutive cycles either from the left or the right tail to separate any two cycles in a signal. Note that this will automatically result in a monotonic health index signal.

3.2.2 Hypothesis Testing for Consecutive Cycles

Let X be the health state such that $X = 0$ is the failure state. Assuming that the units are independent, the measurements at state $X = p$ are approximated by a Normal Distribution with mean $\mu_{j,p}$ and variance $\sigma_{j,p}^2$.

Accordingly, the two-sample t-test with unequal variance can be leveraged to test if the observation at state p and $p-1$ belong to the same population:

$$\begin{aligned} H_0&: \mu_p = \mu_{p-1} \\ H_a&: \mu_p \neq \mu_{p-1} \end{aligned} \tag{3-1}$$

The null hypothesis H_0 is rejected if

$$P\left(T_\nu \leq \sqrt{m}\frac{\overline{y}_{j.p} - \overline{y}_{j.p-1}}{\sqrt{s_{j,p}^2 + s_{j,p-1}^2}} \right) \geq 1 - \alpha \tag{3-2}$$

where T_ν is the students' t distribution with ν degrees of freedom, $\overline{y}_{j.p}$ is the sample mean, $s_{j,p}^2$ is the sample variance of the sensor measurements, and m is the number of historical units.

Therefore, if the t-score is large enough, the null hypothesis is rejected and two states p and $p-1$ are considered separable.

Consequently, sensors with higher t-scores are better for prognosis and degradation analysis because they are more robust to measurement noise and supports Proposition 1.

In summary, a series of t-tests is designed to quantify statistical power of separating consecutive states until failure with the expectation that the statistical power should increase as the state

approaches failure. This concludes the first contribution of this paper which is checking the quality and reliability of the available sensor signals for prognosis and condition monitoring.

3.2.3 Health Index

Let $h_{i,t}$ be the fused measurement of unit i at cycle t:

$$h_{i,t} = f\left(y_{i,t}^{(1)}, y_{i,t}^{(2)}, \dots, y_{i,t}^{(S)}; \boldsymbol{w}\right) \tag{3-3}$$

where $y_{i,t}^{(j)}$ is sensor j measurement for unit i at cycle t, and $\boldsymbol{w}$ is the vector of hyper parameters for the fusion model $f(.;\boldsymbol{w})$.

Theorem 1:

If $(T_v \le \sqrt{m}(\overline{h}_p - \overline{h}_{p-1})/(s_p^2 + s_{p-1}^2)^{1/2}) = q_p \ge 1-\alpha \; \forall 1 \le p \le N$, then $P(T_v \le \sqrt{m}(\overline{h}_p - \overline{h}_{p-1})/(s_p^2 + s_{p-1}^2)^{1/2}) \ge q_p \ge 1-\alpha \; \forall 1 \le p \le N$ and $\forall 1 \le j \le N-p$

Theorem 1 is a validation for Proposition 2 using the two-sample t-test. This further shows that it is sufficient to separate consecutive cycles when constructing the health index.

Therefore, a sufficient objective is to maximize $\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})/(s_p^2 + s_{p-1}^2)^{1/2}$ for every state $p = 1, \dots, N$ to achieve reliable and robust prognostic analysis. Mathematically, the optimization problem is written as:

$$\max_{\boldsymbol{w}} \frac{\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})}{\sqrt{s_p^2 + s_{p-1}^2}}, \forall p = 1,..N \tag{3-4}$$

which is equivalent to

$$\min_{\boldsymbol{w}} \frac{\sqrt{s_p^2 + s_{p-1}^2}}{\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})}, \forall p = 1,..N \tag{3-5}$$

where $\overline{h}_p$ is the sample mean and s_p^2 is the sample variance of the health index at state $X = p$ from the historical units, and N is the number of states until failure.

Unfortunately, there are additional challenges: 1) $\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})/(s_p^2 + s_{p-1}^2)^{1/2}$ at different values of p are not necessarily independent and may be competing and 2) the number of states N is not known and it is not necessarily equal to the number of recorded cycles. To address the first challenge, we propose to add up the quantities the t-scores for different states $X = 1, \dots, N$. To

address the second challenge, the number of states is overestimated by $N = \min_i n_i$ assuming that any unit will pass through all the health states or to consider consecutive states that are not separable as one state (e.g., Figure 2-3, shows six states where the first three cycles belong to the

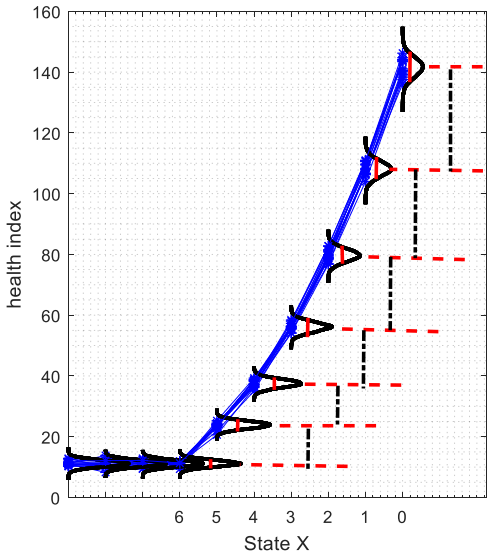


Figure 3-3 The state X for several historical units

same state). Here, n_i is the number of available cycles for training unit i. Therefore, a unit is assumed to experience at most $\min_i n_i$ distinct degradation states. Ideally, focusing on the last N cycles of historical units is preferred under the assumption that a unit starts degrading at most N cycles before failure.

Figure 3-3 shows an example of a well-constructed health index that addresses the above-mentioned challenges and requirements. The black dotted-dashed lines represent $(\overline{h}_p - \overline{h}_{p-1})$ and should be maximized. The solid redlines in the distributions quantifies s_p^2 and should be minimized. Also, it is shown in the figure that only the last 6 cycles tend to be different, and therefore, the choice of $N \geq 6$ is intuitive.

From the figure, it is shown that the six states are separable. Therefore, given the health index signal, which is a function of multi-sensor signals, the degradation state can be reliably predicted. Note that although the measurements at the first, second and third observations will imply that there are still at least six cycles to failure that is expected because it is hard to separate observations at an early stage of degradation.

Mathematically, under any fusion model $f(.;\boldsymbol{w})$, the weights are the solution of the following optimization problem:

$$\min_{\boldsymbol{w}} \frac{\sqrt{s_p^2 + s_{p-1}^2}}{\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})} \tag{3-6}$$

where

$$\overline{h}_p = \frac{1}{m}\sum_{i=1}^{m} h_{i,n_i-p} \tag{3-7}$$

and

$$s_p^2 = \frac{1}{m-1}\sum_{i=1}^{m}\left(h_{i,n_i-p} - \overline{h}_p\right)^2 \tag{3-8}$$

Then, weights c_p is introduced to quantify the importance of separating states p from $p-1$. This is critical when: 1) the number of sates N is overestimated and clearly the preference should be on separating states closer to failure and/or 2) it is hard to separate all consecutive states and the focus should be on separating states closer to failure.

Therefore, the preferred weights must satisfy $c_1 \geq c_2 \geq \cdots \geq c_p \geq \cdots \geq c_N > 0$ and $\boldsymbol{w}$ is now the solution of the following updated problem:

$$\min_{\boldsymbol{w}} \sum_{p=1}^{N} c_p \left\{ \frac{\sqrt{s_p^2 + s_{p-1}^2}}{\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})} \right\} \tag{3-9}$$

The optimization problem in Eq. 3-9 is nonlinear and complex because it aggregates competing ratios, but it shares similar merits with the convex optimization problem described in the following equation:

$$\min_{\boldsymbol{w}} \sum_{p=1}^{N} c_p \{\sqrt{s_p^2 + s_{p-1}^2} - \sqrt{m}(\overline{h}_p - \overline{h}_{p-1})\} \tag{3-10}$$

The optimization problem in Eq. 3-10 is still nonlinear because of the term $(s_p^2 + s_{p-1}^2)^{1/2}$, which may cause additional computational complexities. Therefore, the following linear relaxation is proposed for the optimization problem in Eq. 3-10.

$$\min_{\boldsymbol{w}}\{\sum_{p=1}^{N} c_p(\sqrt{s_p^2 + s_{p-1}^2})\} - \lambda\sqrt{m}\{\sum_{p=1}^{N} c_p(\overline{h}_p - \overline{h}_{p-1})\} \tag{3-11}$$

where λ is a tuning parameter.

3.2.4 Linear Fusion Model

This section discusses the choice of the fusion model, which is a key factor that influences: 1) the computational speed for solving Eq. 3-11; 2) the capability of separating the states X ; and 3) sensor selection.

Therefore, the linear fusion model is developed computationally efficient and helps in sensor selection because $|wj|$ is the weight of sensor j in constructing the fusion model. Specifically, the health index for any unit i at cycle t is calculated as

$$h_{i,t} = \begin{bmatrix} y_{i,t}^{(1)} \\ y_{i,t}^{(2)} \\ \vdots \\ y_{i,t}^{(S)} \end{bmatrix}^T * \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_S \end{bmatrix} = \boldsymbol{y}_{i,t}^T \boldsymbol{w} = \boldsymbol{w}^T \boldsymbol{y}_{i,t} \tag{3-12}$$

Consequently

$$\begin{aligned} \overline{h}_p = \frac{1}{m}\sum_{i=1}^{m} h_{i,n_i-p} &= \frac{1}{m}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}^T \begin{bmatrix} h_{1,n_1-p} \\ h_{2,n_2-p} \\ \vdots \\ h_{m,n_m-p} \end{bmatrix} = \frac{1}{m}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}^T \begin{bmatrix} \boldsymbol{y}_{1,n_1-p}^T \boldsymbol{w} \\ \boldsymbol{y}_{2,n_2-p}^T \boldsymbol{w} \\ \vdots \\ \boldsymbol{y}_{m,n_m-p}^T \boldsymbol{w} \end{bmatrix} \\ &= \frac{1}{m}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}^T \begin{bmatrix} \boldsymbol{y}_{1,n_1-p}^T \\ \boldsymbol{y}_{2,n_2-p}^T \\ \vdots \\ \boldsymbol{y}_{m,n_m-p}^T \end{bmatrix} \boldsymbol{w} = \frac{\boldsymbol{w}^T}{m} \begin{bmatrix} \boldsymbol{y}_{1,n_1-p}^T \\ \boldsymbol{y}_{2,n_2-p}^T \\ \vdots \\ \boldsymbol{y}_{m,n_m-p}^T \end{bmatrix}^T \begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} \\ &= \frac{\boldsymbol{w}^T}{m}\left[\boldsymbol{y}_{1,n_1-p}, \boldsymbol{y}_{2,n_2-p}, \dots, \boldsymbol{y}_{m,n_m-p}\right] \begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} \\ &= \frac{\boldsymbol{w}^T}{m} \begin{bmatrix} y_{1,n_1-p}^{(1)} & y_{2,n_2-p}^{(1)} & & y_{m,n_m-p}^{(1)} \\ y_{1,n_1-p}^{(2)} & y_{2,n_2-p}^{(2)} & \dots & y_{m,n_m-p}^{(2)} \\ \vdots & \vdots & & \vdots \\ y_{1,n_1-p}^{(S)} & y_{2,n_2-p}^{(S)} & & y_{m,n_m-p}^{(S)} \end{bmatrix} \begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} = \frac{\boldsymbol{w}^T}{m} \boldsymbol{\Upsilon}_p \mathbb{I} = \frac{\mathbb{I}^T}{m} \boldsymbol{\Upsilon}_p^{\mathrm{T}} \boldsymbol{w} \end{aligned} \tag{3-13}$$

Similarly, the sample standard deviation is calculated as

$$s_p^2 = \boldsymbol{w}^T \boldsymbol{\Upsilon}_p \left(\frac{\boldsymbol{I}_m - \mathbb{1}\mathbb{1}^T/m}{m-1} \right) \boldsymbol{\Upsilon}_p^{\mathrm{T}} \boldsymbol{w} \tag{3-14}$$

where $\boldsymbol{I}_m$ is the identity matrix with the size $m \times m$.

Thus, for a linear fusion model, (3-11) simplifies to

$$\min_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{B} \boldsymbol{w} - \lambda \sqrt{\boldsymbol{m}} \boldsymbol{c}^T \boldsymbol{A} \boldsymbol{w} \tag{3-15}$$

where

$$\boldsymbol{c}^T \boldsymbol{A} = [\boldsymbol{c_1}, \dots, \boldsymbol{c_N}] \begin{bmatrix} \frac{\mathbb{1}^T (\boldsymbol{\Upsilon}_1^T - \boldsymbol{\Upsilon}_0^T)}{\boldsymbol{m}} \\ \cdots \\ \frac{\mathbb{1}^T (\boldsymbol{\Upsilon}_N^T - \boldsymbol{\Upsilon}_N^T)}{\boldsymbol{m}} \end{bmatrix} \tag{3-16}$$

and

$$\begin{aligned} \boldsymbol{B} = {} & c_0^2 \boldsymbol{\Upsilon}_0 \left(\frac{\boldsymbol{I}_m - \mathbb{1}\mathbb{1}^T/m}{m-1} \right) \boldsymbol{\Upsilon}_0^T + c_N^2 \boldsymbol{\Upsilon}_N \left(\frac{\boldsymbol{I}_m - \mathbb{1}\mathbb{1}^T/m}{m-1} \right) \boldsymbol{\Upsilon}_N^T \\ & + \sum_{p=1}^{N-1} \left\{ 2 c_p^2 \boldsymbol{\Upsilon}_p \left(\frac{\boldsymbol{I}_m - \mathbb{1}\mathbb{1}^T/m}{m-1} \right) \boldsymbol{\Upsilon}_p^T \right\} \end{aligned} \tag{3-17}$$

The optimization problem in (3-15) can be solved using CPLEX, Gurobi, MATLAB, and R or directly via the closed-form solution $\boldsymbol{w}^* = \lambda \sqrt{m} \boldsymbol{B}^{-1} \boldsymbol{A}^T \boldsymbol{c}$.

Lemma 2:

The computational complexity for solving $\boldsymbol{w}^* = \lambda \sqrt{m} \boldsymbol{B}^{-1} \boldsymbol{A}^T \boldsymbol{c}$ is $O(max(S^{2.376}, N * S))$ using the Coppersmith-Winograd algorithm, where S is the number of available sensors.

Lemma 2 shows that solving for $\boldsymbol{w}$ is computationally efficient and depends only on the number of available sensors and the predefined number of health states by the user. Therefore, it may be computationally feasible to real-time update w when multi-sensor data from new failed units are added to the training data set. Such an iterative approach helps to continuously update the fusion model to achieve a more reliable prognostic and degradation analysis, which is critical for applications where collecting data are challenging and expensive.

Lemma 3:

Any multiple of optimal solution $\boldsymbol{w}^* = \lambda\sqrt{m}\boldsymbol{B}^{-1}\boldsymbol{A}^T\boldsymbol{c}$ results in the same t-score value $\sqrt{m}(\overline{h}_p - \overline{h}_{p-1})/(s_p^2 + s_{p-1}^2)^{1/2}$ and $P(T_{m-1} \leq \sqrt{m}(\overline{h}_p - \overline{h}_{p-1})/(s_p^2 + s_{p-1}^2)^{1/2})$ for $0 \leq p \leq N$.

Lemma 3 shows that any multiple of $\boldsymbol{w}^*$ results in an equivalent prognostic and diagnostic performance. Therefore, the value of the tuning parameter λ in optimization problem (3-15) is redundant and can be replaced by ± 1. If $\lambda = \pm 1$, then the health index signal is decreasing/increasing. Furthermore, this provides critical evidence that the optimization problem in (3-15) is an appropriate approximation for the optimization problems in (3-5) and (3-9).

3.2.5 Extension to Nonlinear Fusion Models

The idea is to construct a mapping $\boldsymbol{\eta}(.;\beta)$ from the available data space (sensor data) $\mathcal{X}$ to a feature space $\mathcal{H}$. This may help find better features than using a linear combination of the sensor data. Furthermore, by

$$h_{i,t} = \begin{bmatrix} \eta_1\left(y_{i,t}^{(1)}, y_{i,t}^{(2)}, \dots, y_{i,t}^{(S)}\right) \\ \eta_2\left(y_{i,t}^{(1)}, y_{i,t}^{(2)}, \dots, y_{i,t}^{(S)}\right) \\ \vdots \\ \eta_Q\left(y_{i,t}^{(1)}, y_{i,t}^{(2)}, \dots, y_{i,t}^{(S)}\right) \end{bmatrix}^T * \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_Q \end{bmatrix} = \boldsymbol{\beta}^T\boldsymbol{\eta}(\boldsymbol{y}_{i,t}) \tag{3-18}$$

Comparing to (3-18) and (3-12), $\boldsymbol{w}$ is substituted by $\boldsymbol{\beta}$ and $\boldsymbol{\Upsilon}_p$ is substituted by $\boldsymbol{\mathcal{K}}_p$ in (3-15).

$$\boldsymbol{\mathcal{K}}_p = \left[\boldsymbol{\eta}(\boldsymbol{y}_{1,n_1-p})\boldsymbol{\eta}(\boldsymbol{y}_{2,n_2-p})\dots\boldsymbol{\eta}(\boldsymbol{y}_{m,n_m-p})\right] \in \mathbb{R}^{Q\times m} \tag{3-19}$$

For better illustration, the quadratic fusion model is further discussed in the case study.

3.2.6 Discussion

This chapter proposed a generalized sensor fusion framework to maximize the statistical capability of separating any two consecutive states until failure. Unlike the existing *ad hoc* approaches, this approach relies on statistical hypothesis testing and a systematic framework. It is important to note the assumptions of the proposed framework: 1) there exist historical units that ran until failure to construct the architecture of the health index and 2) the measurements at any specific state of interest come from one population.

In summary, two fusion models are discussed: 1) the linear fusion model and 2) extended linear fusion models. The linear fusion model and its extension are computationally efficient and

interpretable. However, depending on the case study, the choice of the fusion model may be challenging because it may be hard to construct a health index (via commonly used fusion models) that effectively separates consecutive cycles. Therefore, more fusion models will be investigated in the future studies.

Although this section focuses on the two-sample t -test with unequal variance, it can be extended to other statistical hypothesis tests depending on the distribution of the historical measurements. For future studies, we are interested in developing a sensor fusion framework for scenarios where observations at any given state belong to multiple populations (e.g., multiple reasons for failure).

3.3 RUL Estimation

This section proposes a novel approach for RUL estimation. The approach tackles the RUL estimation problem from a different perspective than most of the existing literature.

3.3.1 Maximum Likelihood Estimation

Here, a unit fails when it first reaches the failure state (i.e., $X = 0$). Accordingly, the failure time is the solution for the following optimization problem:

$$\tilde{T}_f = \inf_t(X_t = 0) \tag{3-20}$$

Furthermore, the likelihood that a unit i is at state p given its predicted health index at cycle t can be estimated as:

$$P\left(X_{i,t} = p \middle| \tilde{h}_{i,t}\right) \propto f_{h_p}\left(\tilde{h}_{i,t}\right) \tag{3-21}$$

where $\tilde{h}_{i,t}$ is the distribution of the health index of unit i at cycle t and $f_{h_p}(.)$ is the PDF of the health index at state p.

Consequently, $X_{i,t}|\tilde{h}_{i,t}$ is estimated by the maximum likelihood estimator (MLE)

$$X_{i,t}^{\mathrm{MLE}} = \underset{p}{\operatorname{argmax}}\, f_{h_p}\left(\tilde{h}_{i,t}\right) \tag{3-22}$$

and unit i is estimated to fail at the first cycle t that $X_{i,t}^{\mathrm{MLE}} = 0$.

Under the assumption that $\tilde{h}_{i,t} \sim N\left(\tilde{u}_{i,t}, \tilde{\sigma}_{i,t}^2\right)$ and $h_p \sim N\left(\bar{h}_p, s_p^2\right)$, the MLE in (3-22) simplifies to

$$X_{i,t}^{\mathrm{MLE}} = \underset{p}{\operatorname{argmax}} \frac{1}{\sqrt{\tilde{\sigma}_{i,t}^2 + s_p^2}} \varphi\left(\frac{\tilde{u}_{i,t} - \bar{h}_p}{\sqrt{\tilde{\sigma}_{i,t}^2 + s_p^2}}\right) \tag{3-23}$$

where $\varphi(.)$ is the PDF of the standard normal distribution.

Although Eq. 3-23 is only a special outcome from Eq. 3-24 under the normality assumptions of $h_p \; and \; \tilde{h}_{i,t}$, these assumptions are valid for a wide set of applications. Specifically, the normality assumption of h_p is driven from the central limit theorem, where the units are assumed to be independent and share a similar degradation process with some unit-to-unit variability due to external factors, and the normality assumption for $\tilde{h}_{i,t} \sim N(\tilde{u}_{i,t}, \tilde{\sigma}_{i,t}^2)$ is common in the literature of degradation modeling. The case study provides an example based on a recent Bayesian-based approach. The major advantage of the Bayesian approach is that even for a wrong decision of the prior distribution of $\tilde{h}_{i,t}$, the posterior distribution is regulated to be a better estimate of the true distribution. Therefore, even if the normal prior distribution is far from the true distribution, the posterior normal distribution is expected to be closer to the true distribution.

Recall that n_i is the number of available cycles. Then, the RUL, $\tilde{T}_i$, is calculated as

$$\tilde{T}_i = \tilde{T}_{f,i} - n_i \tag{3-24}$$

3.3.2 Prediction of Health Index

This chapter proposes a general framework of sensor fusion for prognostic and degradation analysis via constructing a health index. The fusion model is expected to perform well for any degradation-based or prognostic task such as the RUL. Furthermore, the fusion model is not expected to perform well if the health index signal is not modeled accurately. Fortunately, there exist various approaches for extrapolating signals, such as multioutput Gaussian processes [102], [103], mixed-effects models (MEMs) [104], and physics-based model [105].

Most existing degradation models are based on the MEM, which was first introduced by Lu and Meeker [104].

$$y(t) = \zeta(\boldsymbol{\alpha}, \boldsymbol{\gamma}, t) + \epsilon(t) \tag{3-25}$$

where $\zeta(\cdot)$ is the parametric form of the degradation model, $y(t)$ is the signal measurement at cycle t; $\boldsymbol{\alpha}$ is a vector of fixed-effect parameters that represents common characteristics of the population, $\boldsymbol{\gamma}$ is a vector of random-effect parameters that characterize the unit-to-unit variability,

and $\epsilon(t)$ is an error term that represents the measurement noises. The MEM is a parametric model and dependent on the choice of $\zeta(\cdot)$; it can describe various functional forms for the degradation signals.

There are various extensions to the MEM and one common approach is proposed by Gebraeel *et al.* in [85]. Gebraeel *et al.* [85] updated the random effects in real-time based on the available *in-situ* data.

For demonstration and case study purposes, the health index is modeled via the following MEM [34], [35], [75], [106]:

$$h(t) = \sum_{\alpha=0}^{q}\{\gamma_\alpha g_\alpha(t)\} + \epsilon(t) = \boldsymbol{\Gamma}_t\boldsymbol{\gamma} + \epsilon(t) \tag{3-26}$$

where $g_\alpha(t)$ is the function of t; $\boldsymbol{\gamma} = [\gamma_0, \cdots, \gamma_q]^T$ is the random-effect parameters and often assumed to follow a multivariate normal distribution, $\boldsymbol{\gamma} \sim N_{p+1}(\boldsymbol{u}^0, \boldsymbol{\Sigma}^0)$; $\epsilon(t)$ is the random noise and follows $N(0, \tilde{\sigma}_\epsilon^2)$; and $\boldsymbol{\Gamma}_t = [g_\alpha(t), g_1(t), \ldots, g_q(t)]$ is the row of the design matrix that corresponds to time t.

For online extrapolation of the health index signal for an operating unit i that has not failed yet, the posterior distribution of the parameters given the partially observed health index signal, $\boldsymbol{h}_i^{\text{obs}}$, from the operating unit is calculated as:

$$\boldsymbol{\gamma}|\boldsymbol{h}_i^{\text{obs}} \sim N_{q+1}(\boldsymbol{u}_i^1, \boldsymbol{\Sigma}_i^1) \tag{3-27}$$

where

$$\boldsymbol{u}_i^1 = \left(\frac{\boldsymbol{\Psi}_i^T\boldsymbol{Q}_i\boldsymbol{\Psi}_i}{\sigma^2} + (\boldsymbol{\Sigma}^0)^{-1}\right)^{-1}\left(\frac{\boldsymbol{\Psi}_i^T\boldsymbol{Q}_i\boldsymbol{h}_i^{\text{obs}}}{\tilde{\sigma}_\epsilon^2} + (\boldsymbol{\Sigma}^0)^{-1}\boldsymbol{u}^0\right) \tag{3-28}$$

$$\boldsymbol{\Sigma}_i^1 = \left(\frac{\boldsymbol{\Psi}_i^T\boldsymbol{Q}_i\boldsymbol{\Psi}_i}{\tilde{\sigma}_\epsilon^2} + (\boldsymbol{\Sigma}^0)^{-1}\right)^{-1} \tag{3-29}$$

$$\boldsymbol{\Psi}_i = \begin{bmatrix} g_0(1) & \cdots & g_q(1) \\ \cdots & \cdots & \cdots \\ g_0(t) & \cdots & g_q(t) \\ \cdots & \cdots & \cdots \\ g_0(n_i) & \cdots & g_q(n_i) \end{bmatrix} \in \mathbb{R}^{n_i\times(q+1)} \tag{3-30}$$

and $\boldsymbol{Q_i}$ is a diagonal weighting matrix (typically higher weights are assigned for points closer to failure) and n_i the number of available cycles from the unit.

Therefore, the predictive distribution for the health index of unit i at cycle t given the observed health index signal is calculated as:

$$\tilde{h}_{i,t}|\boldsymbol{h}_i^{\text{obs}} \sim N\left(\tilde{u}_{i,t}, \tilde{\sigma}_{i,t}^2\right), \tag{3-31}$$

where $\tilde{u}_{i,t} = \boldsymbol{\Gamma}_t \boldsymbol{u}_i^1$; and $\tilde{\sigma}_{i,t}^2 = \boldsymbol{\Gamma}_t \boldsymbol{\Sigma}_i^1 \boldsymbol{\Gamma}_t^T + \tilde{\sigma}_\epsilon^2$.

3.3.3 Summary and Conclusion

In summary, a health index is first constructed as discussed in Section 3.2. The observed health index signal is then calculated from the available multisensory data. The observed health index value is then compared with the learned distributions of the degradation states to identify the most likely degradation state as described in Eq. 3-22 and 3-23. Note that $\tilde{\sigma}_{i,t}^2 = 0$ for all the observed health index values.

To estimate the RUL of an operating unit that has not failed yet, the health index signal is forecast via the approach developed by Gebraeel *et al.* [85] as shown in Eq. 3-26, 3-27 and 3-31. Next, the failure cycle is the first forecast cycle at which the most likely degradation state is the failure state (i.e., $X_{i,\tilde{T}_i}^{MLE} = 0$). Note that there exist other approaches to forecast the health index signal.

3.4 Case Study

The prognostic performance of the proposed sensor fusion approach is investigated in this section for a published degradation data set for aircraft engines. The data set is heavily used in the existing literature for sensor fusion, degradation modeling, prognostic analysis, and fault diagnosis. It is generated from the commercial modular aero-propulsion system simulation (C-MAPSS) developed at NASA [107] and published online for research investigations. The true physical model, initial conditions, and environmental and operational conditions of the engines are not provided. However, it is known that the testing and training engines degrade due to a fan defect. With such limited knowledge about the system, we rely solely on the multi-sensor data from the training and testing engines to understand the degradation process of the engines. The data set can be downloaded from

https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan.

In particular, the data set contains multi-sensor data from $m = 100$ historical engines that failed with a total of $\sum_{i=1}^{m} n_i$ = 20631observations. For each observation, data are recorded from 21 sensors that measure different characteristics of the engine performance [108]. The data set also

contains multi-sensor data from 100 operating units until some random point before failure. The remaining lifetime of those operating units is given in a separate file. Therefore, the goal is to predict the RUL from the available multi-sensor data of the 100 operating engines and then compare it to the true RUL of the operating units. The sensor descriptions are shown in Table 3-1.

Table 3-1 Detailed sensor description

symbol	description	units
T2	Total temperature at fan inlet	°R
T24	Total temperature at LPC inlet	°R
T30	Total temperature at HPC inlet	°R
T50	Total temperature at LPT inlet	°R
P2	Pressure at fan inlet	psia
P15	Total pressure in bypass-duct	psia
P30	Total pressure at HPC outlet	psia
Nf	Physical fan speed	rpm
Nc	Physical core speed	rpm
Epr	Engine pressure ratio	-
Ps30	Static pressure at HPC outlet	psia
Phi	Ratio of fuel flow to Ps30	pps
NRf	Corrected fan speed	rpm
NRc	Corrected core speed	rpm
BPR	Bypass ratio	-
farB	Burner fuel-air ratio	-
htBleed	Bleed enthalpy	-
NF_dmd	Demanded fan speed	rpm
PCNR_dmd	Demanded corrected fan speed	rpm
W31	HPT coolant bleed	lbm/s
W32	LPT coolant bleed	lbm/s

3.4.1 Data Preprocessing and Benchmark Methods

To be consistent with the literatures [34], [35], [38], [75], [106], a similar logarithmic transformation is applied to each individual sensor's signal and focuses on the standardized logged signals.

For the proposed approach, the transformed multi-sensor data are used to construct the health index: 1) for the linear fusion function, the health index is modeled as a quadratic model and 2) for the quadratic fusion function, the health index is modeled as a fourth-order polynomial model. The health index signal for each operating unit is then extrapolated for future cycles. Finally, the degradation status is estimated via the MLE state at each extrapolated cycle of the health index.

A demonstration of the linear fusion model is shown in Figure 3-4. The figure shows that the fused signal is less noisy than the individual sensor signals and the model for the fused signal accurately fits the data. This validates the choice of the data transformation and the degradation model.

The benchmark methods are as follows:

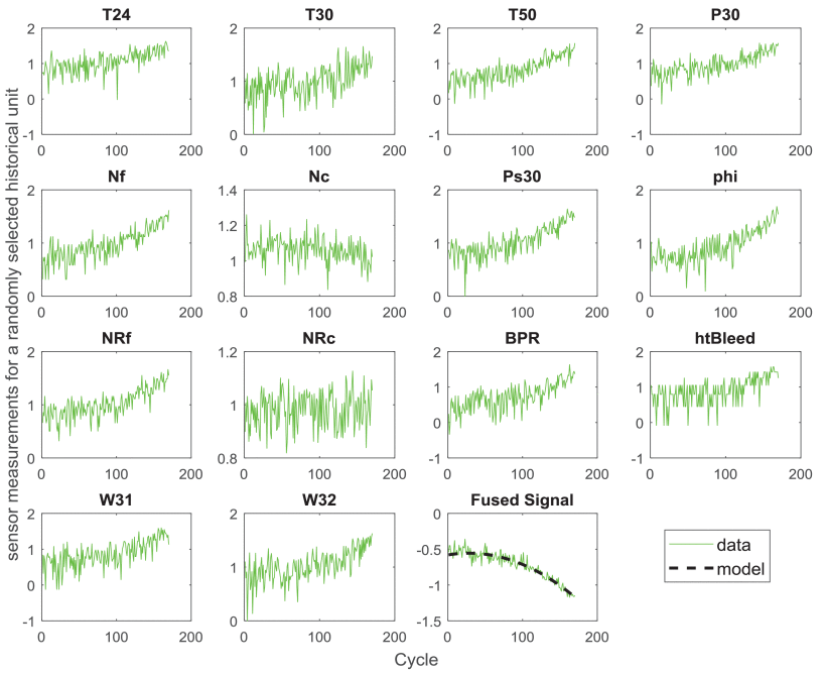


Figure 3-4 Demonstration of the fusion framework for a randomly selected historical unit

MFPCA: A sensor fusion approach based on function PCA by Xiaolei Fang *et al.* [38].

HI-NON: A non-parametric sensor fusion approach by Kaibo Liu *et al*. [34].

HI-SEMI: A parametric based fusion approach by Kaibo Liu and Shuai Huang [5] that depends on the parametric structure of the degradation model.

HI-SNR: A sensor fusion approach based on a SNR ratio metric for constructing a health index by Kaibo Liu et al. [35].

Single frequency (SF)-MLE (L): The proposed sensor fusion approach with a linear fusion model and an MLE estimator for the RUL.

SF-MLE (Q): The proposed sensor fusion approach with a quadratic feature function and an MLE estimator for the RUL.

3.4.2 Degradation Score and State Estimation

This task is critical for preventive maintenance and other decision-making tasks. Fortunately, the proposed sensor fusion framework allows estimation of the state of the system given the multi-sensor data at the cycle of interest. With the simple linear transformation in the following equation, the Degradation Score (DS) is calculated as

$$DS_t = \frac{N - X_t^{MLE}}{N} \tag{3-32}$$

where DS_t is the DS at cycle t with 0 for a fully healthy unit and 1 for a failed unit, X_t^{MLE} can be

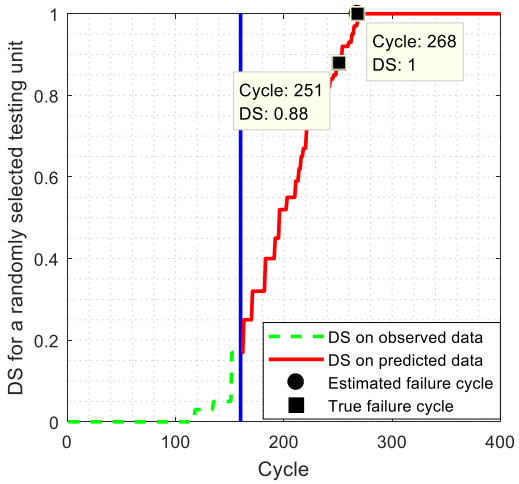


Figure 3-5 DS for a random testing unit with observed data truncated at an early stage

obtained from Eq. 3-23, and N is the number of degradation states until failure.

Figure 3-5 shows the DS for a randomly selected testing unit. The blue vertical line indicates the time of the last observed cycle for the unit, and it separates between the predicted/forecast part of DS (solid red) score and the observed part (dashed green). It can be shown from the figure that the DS increases with time, which is expected and the predicted DS at the true failure cycle is high enough (0.88) to raise an alarm. The second observation shows another major contribution of the proposed approach, which is the validity of raising alarms at a DS close to 1. For this selected testing unit, the error is (268 – 251) /251 ≈ 6.8%, which is considerably small given the fact that the observed data are truncated at an early stage of degradation with DS = 0.18. All the above are indicators of the efficacy of the proposed approach in practical case studies to raise reliable early alarms about the condition of an operating engine.

It is important to note that most of the existing fusion approaches have two states: normal or failed. Accordingly, the DS of such approaches would be either 0 or 1, which cannot be utilized to raise early alarms and requires extremely accurate prediction of the failure cycle.

Furthermore, Figure 3-6 shows the boxplots of the DS of the true failure cycle at different levels

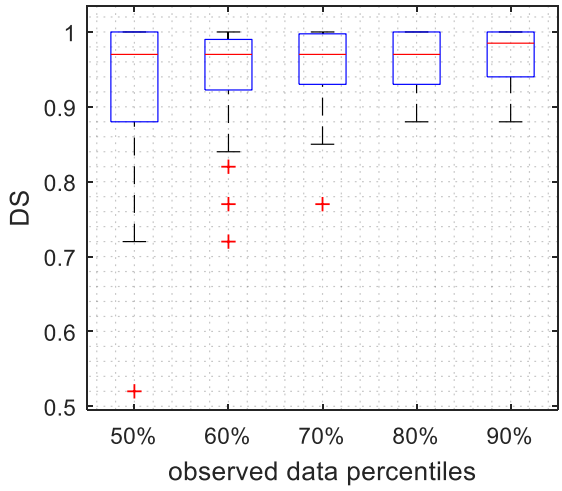


Figure 3-6 DS at failure cycles for the testing unit at different percentiles of observed data

of observed data. The figure shows that even if only 50% of the data is observed, the median and the 25the percentile of the DS at the true failure cycle are 0.97 and 0.88 respectively. In general, the medians of the DS at failure for all the observed data percentiles are close to 1, which shows the accuracy of the proposed approach in predicting the degradation level near failure. Furthermore, the figure shows that most of the testing units show a DS that is large enough at failure; therefore,

it is possible to consider a safety DS threshold (e.g., DS=0.7) to flag that an operating unit is close to failure.

3.4.3 RUL Estimation

To evaluate the performance of the benchmark and proposed approaches, we calculate the absolute difference between the estimated failure time and the true failure time for the testing dataset. Mathematically, the error is calculated as

$$err_i = \frac{\left|\tilde{T}_{f,i} - T_{f,i}\right|}{T_{f,i}} = \frac{\left|\tilde{T}_{f,i} - n_i + T_{f,i} - n_i\right|}{T_{f,i}} = \frac{\left|\tilde{T}_i - T_i\right|}{T_i + n_i}. \quad (3\text{-}33)$$

Figures 3-7 and 3-8 show the mean and standard deviation of the absolute error in Eq. 3-33 at different percentiles of observed cycles. Recall that only partial data of the testing units are observed, and the goal is to estimate the RUL based on the observed data.

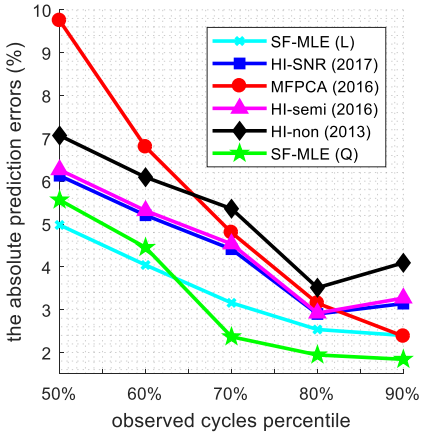


Figure 3-7 Mean of absolute error at different percentiles of observed cycles

The results in Figures 3-7 and 3-8 shows that: 1) the proposed approach outperforms the existing approaches, with lower mean error and standard deviation in the RUL prediction error; 2) the quadratic feature function outperforms the linear feature function for units with more available data; and 3) the linear feature function outperforms the quadratic feature function for units with less available data. The first observation is expected because, unlike the benchmark approaches that consider only failure and normal states, the proposed approach considers multiple states until failure. Furthermore, the proposed sensor fusion is designed to separate between those states until failure. The second and third observations are also expected because more information is gathered from a higher feature space; however, a higher feature space is more complex and requires more data to be effectively learned. Therefore, the quadratic fusion function is expected to be less efficient at lower levels of available data and it is also expected to improve and show more accurate RUL estimations with more available data. Overall, Figures 3-7 and 3-8 show that the constructed health indices under the proposed sensor fusion approach (SF-MLE and SF-MLE-Q) are more effective for RUL estimation than recent existing approaches.

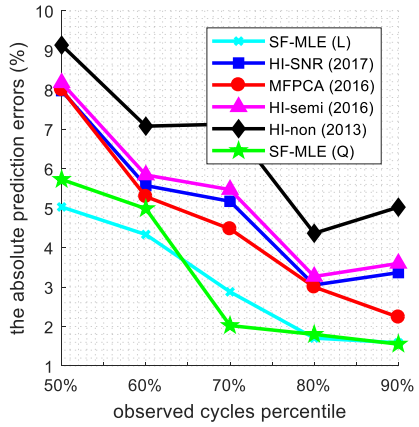


Figure 3-8 Standard deviation of absolute error at different percentiles of observed cycles

Chapter 4 A Multi-Sensor Fusion Approach Based on a Novel Coupled Duration Dependent Hidden Semi-Markov Model

Sensor fusion can reduce uncertainty and produce more reliable information by combining sensors from different sources. Hidden Markov Model (HMM), a common method of modeling stochastic process, can be also used for sensor fusion. This function can be achieved by attaching multiple observations from different sensors to each state of the stochastic process. Although HMM may work well in some cases, but it still suffers from several limitations. Firstly, HMM cannot capture the interactions among multiple stochastic processes. Secondly, the geometric distribution of state duration in HMM is not in line with most practical situations. To overcome the limitations mentioned above, researchers have developed some extensions of HMM, like hidden semi-Markov model (HSMM), coupled hidden semi-Markov model (CHMM), etc. But none of these extensions can handle both issues at the same time. Besides, sensor signals should be affected by deterioration of each state, but this time-varying characteristic of observation sequence is usually ignored in HMM and its extensions. Therefore, we developed a new sensor fusion method, coupled duration-dependent hidden semi-Markov model (CDD-HSMM), which i) captures the interactions among multiple stochastic processes; ii) explicitly defines the distribution of state duration according to actual situation; iii) assumes the observation variable to be state duration dependent. To prove the model effect, we applied CDD-HSMM on two fields in case study, remaining useful life (RUL) prediction of turbofan engines and Alzheimer's Disease (AD) stage recognition. The results illustrate that CDD-HSMM is more effective than the conventional HMM and its extensions.

4.1 Theoretical Methods

HSMM is an extension of HMM. HMM has been widely applied in many fields like speech recognition, protein folding, handwriting recognition, and so forth [35]–[38]. Figure 4-1 shows the structure of a stochastic process modeled by HMM from time $t = 1$ to T: an observation sequence $[O_1, O_2, \dots O_T]$ produced by the hidden state sequence $[s_1, s_2, \dots s_T]$ [39]. Here, $[s_1, s_2, \dots s_T]$ is

assumed to be a Markov chain, where the state transition in HMM is only dependent on the current state. An HMM is described with five parameters, N, V, $\boldsymbol{\pi}$, $\boldsymbol{A}$, and $\boldsymbol{B}$. Here, N is the number of hidden states and V is the observation set. Specifically, for the stochastic process molded by HMM in Figure 4-1, $s_t \in [1, N]$, and $O_t \in V$, for $\forall t \in [1, T]$. N and V are usually determined empirically based on the application and data to be modeled. $\boldsymbol{\pi}$, $\boldsymbol{A}$, and $\boldsymbol{B}$ are parameters that need to be learned through training. $\boldsymbol{\pi}$ is the initial state probability distribution, where $\boldsymbol{\pi} = \{\pi_i = P(s_1 = i), 1 \leq i \leq N\}$. $\boldsymbol{A}$ is the state transition probability distribution, where $\boldsymbol{A} = \{a_{ij} =$

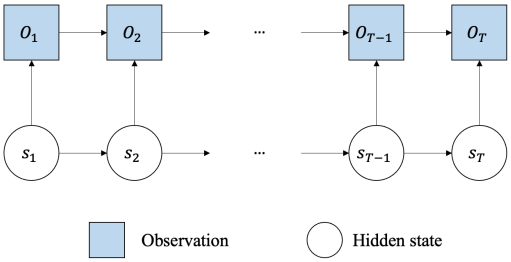


Figure 4-1 Structure of HMM

$P(s_t = j | s_{t-1} = i), 1 \leq i, j \leq N\}$. And $\boldsymbol{B}$ is the observation probability distribution, where $\boldsymbol{B} = \{b_i(O_t) = P(O_t | s_t = i), 1 \leq i \leq N\}$. A complete HMM can be written as $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B})$.

Although HMM has been proven successful in many applications, there is still a limitation. Due to the non-zero probability of self-transition, the duration of each state is geometrically distributed, not coincident with actual facts [40]–[42]. To overcome this limitation, HSMM is proposed.

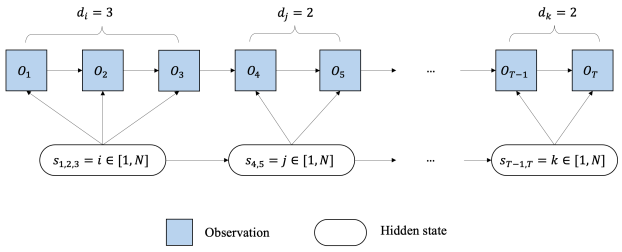


Figure 4-2 Structure of HSMM

Figure 4-2 takes an example to demonstrate the structure of HSMM. Like HMM, HSMM also has an observation sequence $[O_1, O_2, \ldots O_T]$ and a state sequence $[s_1, s_2, \ldots s_T]$, where $O_t \in V$, $s_t \in [1, N]$, for $\forall t \in [1, T]$. However, different from HMM, each state in HSMM can emit a sequence of observation. For example, in Figure 4-2, the subsequence $[O_1, O_2, O_3]$ is generated by

state i, that is $s_{t=1,2,3} = i$. d_n for $\forall n \in [1, N]$ in Figure 4-2 represents the duration for each state, which is the length of time when the state has been visited without transition. The state duration d_n usually follows an explicitly defined probability distribution instead of a geometric distribution as HMM does. This unique variable makes the stochastic process modeled by HSMM not satisfying Markov property anymore. In other word, the state transition is not only dependent on current state but also relies on how long the state has been visited.

A complete HSMM is denoted as $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{D})$.

1) initial state probability distribution:

$$\boldsymbol{\pi} = \{\pi_i = P(s_1 = i), 1 \leq i \leq N\}$$

2) state transition probability distribution:

$$\boldsymbol{A} = \{a_{ij} = P(s_t = j | s_{t-1} = i), 1 \leq i \leq N\}$$

3) observation probability distribution:

$$\boldsymbol{B} = \{b_i(O_t) = P(O_t | s_t = i), 1 \leq i \leq N\}$$

4) duration probability distribution:

$$\boldsymbol{D} = \{p_i(d) = P(t_2 - t_1 = d | s_{t1-1}, s_{t2+1} \neq i, s_{t1:t2} = i\ ), 1 \leq i \leq N\}$$

CHMM is another advanced extension of HMM with a more complex model structure. In many applications, such as activity recognition, speech recognition and so forth, numerous sensors are collected. To characterize the interactions among different sensors, CHMM is first proposed by [43].

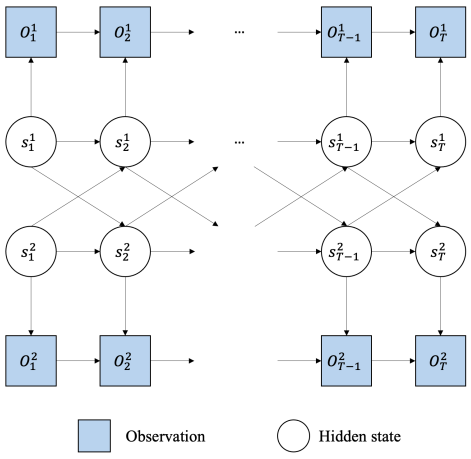


Figure 4-3 Structure of CHMM

As Figure 4-3 shows, CHMM couples two stochastic processes which satisfy Markov property.

In other words, CHMM is comprised of two HMM chains. Therefore, the parameters of HMM, like initial state probability, state transition probability, and observation probability, also exist in CHMM. However, unlike HMM, the state transition of one chain in CHMM is not only dependent on the current state of its own chain, but also that of the other chain.

A complete CHMM is usually described as follows. Here, we use superscript c to distinguish different chains.

1) initial state probability distribution:

$$\boldsymbol{\pi} = \{\pi_{i^c}^c = P(s_1^c = i^c), c \in \{1,2\}, 1 \le i^c \le N^c\}$$

2) state transition probability distribution:

$$\boldsymbol{A} = \{a_{i^{c'},j^c}^c = P\left(s_{t+1}^c = j^c \middle| s_t^{c'} = i^{c'}\right), c', c \in \{1,2\}, 1 \le j^c \le N^c, 1 \le i^{c\prime} \le N^{c\prime}\}$$

3) observation probability distribution:

$$\boldsymbol{B} = \{b_{i^c}^c = P(O_t^c | s_t^c = i^c), c \in \{1,2\}, 1 \le i^c \le N^c\}$$

N^c and $N^{c\prime}$ are state numbers for chain c and c' respectively. As mentioned above, CHMM assumes the state transition of one chain depends on the current states of two chains. Therefore, as Eq.4-1 shows, the probability $P({s_t}^c | {s_{t-1}}^1, {s_{t-1}}^2)$ should be calculated as the product of marginal condition probabilities $P({s_t}^c | {s_{t-1}}^{c\prime})$, for $\forall c, c' \in \{1,2\}$ based on the probability theory.

$$P({s_t}^c | {s_{t-1}}^1, {s_{t-1}}^2) = \prod_{c\prime \in \{1,2\}} P({s_t}^c | {s_{t-1}}^{c\prime}) = \prod_{c\prime \in \{1,2\}} a_{i^{c\prime},j^c}^c \tag{4-1}$$

4.2 Proposed Coupled Hidden Semi-Markov Model for Sensor Fusion

In order to preserve advantages of both HSMM and CHMM, we combine them together and make modification to propose a novel coupled duration-dependent hidden semi-Markov model (CDD-HSMM). The proposed CDD-HSMM, which takes the advantages of HSMM and CHMM, can capture the duration-dependent characteristics of state transition and describe the interactions among them. Besides, it can also characterize the time-varying characteristics of the observation sequences. In this section, we will give a detailed introduction about CDD-HSMM, including the model structure, the parameter space, and the inference and algorithm for parameter re-estimation.

4.2.1 Structure of CDD-HSMM

The structure of CDD-HSMM is given by Figure 4-4. As Figure 4-4 shows, there are two chains

in the model, interacted with each other. However, unlike CHMM, each chain in CDD-HSMM is a semi-Markov chain, which means that the duration probability distribution $\boldsymbol{D}$ is explicitly defined as HSMM does. Besides, to make model more realistic, we take the duration-dependent characteristics of observation sequences into consideration. We assume that both state transition probability distribution $\boldsymbol{A}$ and observation probability distribution $\boldsymbol{B}$ are state duration dependent.

A complete CDD-HSMM model is defined as below:

initial state probability distribution:

$$\boldsymbol{\pi} = \{\pi_{i^c}^c = P(s_1^c = i^c), c \in \{1,2\}, 1 \le i^c \le N^c\}$$

state transition probability distribution:

$$\boldsymbol{A} = \{\ a_{i^{c'},j^c}^c(d^{c'}) = P\left(s_{t+1}^c = j^c \middle| s_{t-d^{c'}}^{c'} \ne i^{c'}, s_{t-d^{c'}+1}^{c'} = \ldots = s_t^{c'} = i^{c'}\right), c', c \in \{1,2\}, 1 \le j^c \le N^c, 1 \le i^{c'} \le N^{c'}\}$$

observation probability distribution:

$$\boldsymbol{B} = \{b_{i^c}^c(d^c) = P\left(O_t^c \middle| s_{t-d^c}^c \ne i^c, s_{t-d^c+1}^c = \ldots = s_t^c = i^c\right), \quad c \in \{1,2\}, 1 \le i^c \le N^c\}$$

duration probability distribution:

$$\boldsymbol{D} = \{p_{i^c}^c(d^c) = P(t_2 - t_1 = d^c | s_{t1-1}^c, s_{t2+1}^c \ne i^c, s_{t1:t2}^c = i^c), c \in \{1,2\}, 1 \le i^c \le N^c\}$$

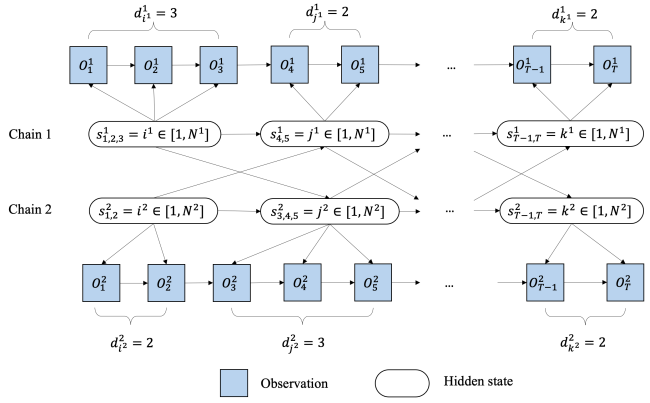


Figure 4-4 Structure of CDD-HSMM

Considering that there are two chains involved, the state in CDD-HSMM is two-dimensional, denoted as $\boldsymbol{i} = (i^1, i^2)$, where superscripts 1 and 2 represent two chains respectively. Likewise, the observation at time t is written as $\boldsymbol{O_t} = (O_t^1, O_t^2)$ and the duration is $\boldsymbol{d} = (d^1, d^2)$.

According to probability theory, the joint probability that two chains start with composite state $\boldsymbol{i}$, denoted as $\pi_{\boldsymbol{i}}$, should be calculated as follows:

$$P(s_1 = \boldsymbol{i}) = \prod_{c\in\{1,2\}} P(s_1^c = i^c) = \prod_{c\in\{1,2\}} \pi_{i^c}^c \tag{4-2}$$

Similarly, the conditional probability that two chains transit from composite state $\boldsymbol{i} = (i^1, i^2)$ to $\boldsymbol{j} = (j^1, j^2)$ with duration time $\boldsymbol{d} = (d^1, d^2)$ can be calculated as follows:

$$\begin{aligned} &P(s_t = \boldsymbol{j} | s_{t-1} = \boldsymbol{i}, d_t = \boldsymbol{d}\,) \\ &\quad = \prod_{c\in\{1,2\}} \prod_{c\prime\in\{1,2\}} P(\, s_{t+1}^c = j^c | s_{t-d^{c'}}^{c'} \neq i^{c'}, s_{t-d^{c'}+1}^{c'} = \ldots = s_t^{c'} = i^{c'}) \\ &\quad = \prod_{c\in\{1,2\}} \prod_{c\prime\in\{1,2\}} a_{i^{c'},j^c}^c(d^{c'}) \end{aligned} \tag{4-3}$$

Finally, the conditional probability $P(\,\boldsymbol{O_t}|s_t = \boldsymbol{i}, d_t = \boldsymbol{d})$ can be calculated as follows:

$$\begin{aligned} P(\,\boldsymbol{O_t}|s_t = \boldsymbol{i}, d_t = \boldsymbol{d}) &= \prod_{c\in\{1,2\}} P\big(\, O_t^c \big| s_{t-d^c}^c \neq i^c, s_{t-d^c+1}^c = \ldots = s_t^c = i^c\big) \\ &= \prod_{c\in\{1,2\}} b_{i^c}^c(d^c) \end{aligned} \tag{4-4}$$

For purpose of simplification in the derivation, we denoted $P(s_1^1 = i^1, s_1^2 = i^2)$ as $\pi_{\boldsymbol{i}}$, $P(s_t = \boldsymbol{j}|s_{t-1} = \boldsymbol{i}, d_t = \boldsymbol{d}\,)$ as $a_{\boldsymbol{i},\boldsymbol{j}}(\boldsymbol{d})$, and $P(\,\boldsymbol{O_t}|s_t = \boldsymbol{i}, d_t = \boldsymbol{d})$ as $b_{\boldsymbol{i},\boldsymbol{d}}(\boldsymbol{O_t})$.

4.2.2 Inference of CDD-HSMM

Due to the existence of duration-dependent parameters $\boldsymbol{A}$ and $\boldsymbol{B}$, we need to modify the classic Baum-Welch algorithm, which is designed for updating HMM parameters, to make it suitable for the new model. In this sub-section, we will mainly focus on introducing some variables which play key roles in the modified Baum-Welch algorithm.

Forward variable is the joint probability of emitting observation sequence $\boldsymbol{O_{1:t}} = (\boldsymbol{O_1}, \boldsymbol{O_2}, \ldots, \boldsymbol{O_t})$ and staying in the composite state $\boldsymbol{i}$ with duration time $\boldsymbol{d}$ at time t given model $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{D})$.

$$\alpha_t(\boldsymbol{i}, \boldsymbol{d}) = P(\boldsymbol{O_{1:t}}, s_t = \boldsymbol{i}, d_t(\boldsymbol{i}) = \boldsymbol{d}|\boldsymbol{\lambda}) \tag{4-5}$$

When $t = 1$, the value of forward variable is:

$$\alpha_1(\boldsymbol{i}, \boldsymbol{d}) = \begin{cases} \pi_{\boldsymbol{i}} b_{\boldsymbol{i},\boldsymbol{d}}(\boldsymbol{O_1}), \text{when } d^c = 1 \text{ for } \forall c \in \{1,2\} \\ 0, \text{otherwise} \end{cases} \tag{4-6}$$

According to Eq. 4-6, if there is any duration time d^c greater than 1, the forward variable at time

$t = 1$ will equal to 0. It conforms to reality, because it is impossible that the chain will stay in one state for a period longer than the observation length.

When $t > 1$, there are several complicated cases we need to give a think. After analyzing all possibilities of state transitions for two chains, we determined four situations to be considered when developing the forward recursion formulae.

Situation 1: both chains stay at state $\boldsymbol{i}$ without transition at time t.

$$\alpha_t(\boldsymbol{i}, \boldsymbol{d}+\boldsymbol{1}) = \alpha_{t-1}(\boldsymbol{i}, \boldsymbol{d}) b_{\boldsymbol{i},\boldsymbol{d}}(\boldsymbol{O_t}) \tag{4-7}$$

Where $\boldsymbol{d}+\boldsymbol{1} = (d^1+1, d^2+1)$ for $\forall d^1, d^2 \geq 1$.

Situation 2: state transits from $\boldsymbol{i}$ to $\boldsymbol{j}$ at time t, where $i^1 \neq j^1$, and $i^2 \neq j^2$. In this case, the duration of both chains at time t will equal to 1, denoted as $\boldsymbol{d} = \boldsymbol{1}$ ($d^c = 1\ for\ \forall c \in \{1,2\}$).

$$\alpha_t(\boldsymbol{j}, \boldsymbol{1}) = \sum_{\boldsymbol{d \geq 1}} \alpha_{t-1}(\boldsymbol{i}, \boldsymbol{d}) p_{\boldsymbol{i}}(\boldsymbol{d}) a_{\boldsymbol{i},\boldsymbol{j}}(\boldsymbol{d}) b_{\boldsymbol{j},\boldsymbol{1}}(\boldsymbol{O_t}) \tag{4-8}$$

Situation 3: state transits from $\boldsymbol{i}$ to $\boldsymbol{j}$ at time t, where $i^1 \neq j^1$ but $i^2 = j^2$. Under this situation, the duration of chain 1 at time t will equal to 1, whereas that of chain 2 will equal to $d^2 + 1$.

$$\alpha_t(\boldsymbol{j}, \boldsymbol{d^{1*}}) = \sum_{d^1 \geq 1} \alpha_{t-1}(\boldsymbol{i}, \boldsymbol{d}) p^1_{i^1}(d^1) a^1_{i^1,j^1}(d^1) b_{\boldsymbol{j},\boldsymbol{d^{1*}}}(\boldsymbol{O_t}) \tag{4-9}$$

where $\boldsymbol{d} = (d^1, d^2)$ for $\forall d^1, d^2 \geq 1$, $\boldsymbol{d^{1*}} = (1, d^2+1)$.

Situation 4: state transits from $\boldsymbol{i} = (i^1, i^2)$ to $\boldsymbol{j} = (j^1, j^2)$ at time t, where $i^2 \neq j^2$ but $i^1 = j^1$. Opposite from situation 3, the duration of chain 2 at time t will equal to 1, whereas that of chain 1 will equal to $d^1 + 1$.

$$\alpha_t(\boldsymbol{j}, \boldsymbol{d^{2*}}) = \sum_{d^2 \geq 1} \alpha_{t-1}(\boldsymbol{i}, \boldsymbol{d}) p^2_{i^2}(d^2) a^2_{i^2,j^2}(d^2) b_{\boldsymbol{j},\boldsymbol{d^{2*}}}(\boldsymbol{O_t}) \tag{4-10}$$

where $\boldsymbol{d} = (d^1, d^2)$ for $\forall d^1, d^2 \geq 1$, $\boldsymbol{d^{2*}} = (d^1+1, 1)$.

Backward variable is defined as the joint probability of generating observation sequence $\boldsymbol{O_{t+1:T}} = (\boldsymbol{O_{t+1}}, \boldsymbol{O_{t+2}}, \ldots, \boldsymbol{O_T})$ given model $\boldsymbol{\lambda}$, state $\boldsymbol{i}$ and duration time $\boldsymbol{d}$.

$$\beta_t(\boldsymbol{i}, \boldsymbol{d}) = P(\boldsymbol{O_{t+1:T}} | s_t = \boldsymbol{i}, d_t(\boldsymbol{i}) = \boldsymbol{d}, \boldsymbol{\lambda}) \tag{4-11}$$

At time $t = T$, the initial value of backward variable is set as 1 no matter which state and

duration time it is, as Eq. 4-12 shows.

$$\beta_T(\boldsymbol{i}, \boldsymbol{d}) = 1 \tag{4-12}$$

For time $1 \leq t < T$, there are four possibilities of $\boldsymbol{\beta_{t+1}}$ that $\boldsymbol{\beta_t}(\boldsymbol{i}, \boldsymbol{d})$ will transit to. Therefore, for the backward recursion, we also need to take four situations of state transition into consideration as we did for forward variable. Here, superscripts 1, 2, 3, and 4 of β stand for these four situations.

Situation 1: both chains stay at state $\boldsymbol{i}$ without transition at time $t+1$.

$$\beta_{t+1}^1 = \beta_{t+1}(\boldsymbol{i}, \boldsymbol{d}+\boldsymbol{1}) b_{\boldsymbol{i},\boldsymbol{d}+\boldsymbol{1}}(\boldsymbol{O_{t+1}}) \tag{4-13}$$

Situation 2: state transits from $\boldsymbol{i}$ to $\boldsymbol{j}$ at time $t+1$, where $i^1 \neq j^1$, and $i^2 \neq j^2$.

$$\beta_{t+1}^2 = \sum_{\boldsymbol{d} \geq \boldsymbol{1}} \beta_{t+1}(\boldsymbol{j}, \boldsymbol{1}) p_{\boldsymbol{i}}(\boldsymbol{d}) a_{\boldsymbol{i},\boldsymbol{j}}(\boldsymbol{d}) b_{\boldsymbol{j},\boldsymbol{1}}(\boldsymbol{O_{t+1}}) \tag{4-14}$$

Situation 3: state transits from $\boldsymbol{i}$ to $\boldsymbol{j}$ at time $t+1$, where $i^1 \neq j^1$ but $i^2 = j^2$.

$$\begin{aligned} \beta_{t+1}^3 = \sum_{d^1 \geq 1} & \beta_{t+1}\big((j^1, j^2), (1, d^2+1)\big) p_{i^1}^1(d^1) \\ & a_{i^1,j^1}^1(d^1) b_{j^1}^1(1) b_{i^2}^2(d^2+1) \end{aligned} \tag{4-15}$$

Situation 4: state transits from $\boldsymbol{i}$ to $\boldsymbol{j}$ at time $t+1$, where $i^2 \neq j^2$ but $i^1 = j^1$.

$$\begin{aligned} \beta_{t+1}^4 = \sum_{d^1 \geq 1} & \beta_{t+1}\big((j^1, j^2), (d^1+1, 1)\big) p_{i^2}^2(d^2) \\ & a_{i^2,j^2}^2(d^2) b_{i^1}^1(d^1+1) b_{j^2}^2(1) \end{aligned} \tag{4-16}$$

Considering all these four possibilities of state transition, the backward recursion formula is determined as follows:

$$\beta_t(\boldsymbol{i}, \boldsymbol{d}) = \beta_{t+1}^1 + \beta_{t+1}^2 + \beta_{t+1}^3 + \beta_{t+1}^4 \tag{4-17}$$

To re-estimate parameters of CDD-HSMM based on observation sequences, there are some other variables that need to be defined.

The first one is the posterior probability of state transition from state $\boldsymbol{i}$ to $\boldsymbol{j}$ with duration time $\boldsymbol{d}$ at time t given the observation sequence and model parameters, usually named as forward-backward variable.

$$\xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d}) = P(s_t = \boldsymbol{i}, s_{t+1} = \boldsymbol{j}, d_t(\boldsymbol{i}) = \boldsymbol{d} | \boldsymbol{O_{1:T}}, \boldsymbol{\lambda}) \tag{4-18}$$

When both chains change states, the forward-backward variable should be calculated based on Eq. 4-19. In Eq. 4-19, $\boldsymbol{i} = (i^1, i^2)$, $\boldsymbol{j} = (j^1, j^2)$, where $i^1 \neq j^1$, and $i^2 \neq j^2$.

$$\xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d}) = \frac{\alpha_t(\boldsymbol{i}, \boldsymbol{d}) p_{\boldsymbol{i}}(\boldsymbol{d}) a_{\boldsymbol{i},\boldsymbol{j}}(\boldsymbol{d}) \beta_{t+1}(\boldsymbol{j}, \boldsymbol{1}) b_{\boldsymbol{j},\boldsymbol{1}}(\boldsymbol{O_{t+1}})}{P(\boldsymbol{O_{1:T}}|\boldsymbol{\lambda})} \tag{4-19}$$

When only chain 1 changes state, the forward-backward variable should be calculated according to Eq. 4-20, where $i^1 \neq j^1$, $i^2 = j^2$.

$$\xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d}) = \frac{\alpha_t(\boldsymbol{i}, \boldsymbol{d}) p^1_{i^1}(d^1) a^1_{i^1,j^1}(d^1) \beta_{t+1}(\boldsymbol{j}, \boldsymbol{d^{1*}}) b_{\boldsymbol{j},\boldsymbol{d^{1*}}}(\boldsymbol{O_{t+1}})}{P(\boldsymbol{O_{1:T}}|\boldsymbol{\lambda})} \tag{4-20}$$

Likewise, we should refer Eq. 4-21 when only chain 2 changes state, where $i^2 \neq j^2$, $i^1 = j^1$.

$$\xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d}) = \frac{\alpha_t(\boldsymbol{i}, \boldsymbol{d}) p^2_{i^2}(d^2) a^2_{i^2,j^2}(d^2) \beta_{t+1}(\boldsymbol{j}, \boldsymbol{d^{2*}}) b_{\boldsymbol{j},\boldsymbol{d^{2*}}}(\boldsymbol{O_{t+1}})}{P(\boldsymbol{O_{1:T}}|\boldsymbol{\lambda})} \tag{4-21}$$

There is a final situation when both chains stay in current state $\boldsymbol{i} = (i^1, i^2)$. However, before we discuss this situation, let us introduce the second variable to be defined for the purpose of model parameter re-estimation. It is the posterior probability that state stay at $\boldsymbol{i}$ with duration time $\boldsymbol{d}$ at time t, which can be calculated as follows:

$$\begin{aligned} \gamma_t(\boldsymbol{i}, \boldsymbol{d}) &= P(s_t = \boldsymbol{i}, d_t(\boldsymbol{i}) = \boldsymbol{d}|\boldsymbol{O_{1:T}}, \boldsymbol{\lambda}) \\ &= \frac{\alpha_t(\boldsymbol{i}, \boldsymbol{d}) \beta_t(\boldsymbol{i}, \boldsymbol{d})}{P(\boldsymbol{O_{1:T}}|\boldsymbol{\lambda})} \end{aligned} \tag{4-22}$$

Given $\gamma_t(\boldsymbol{i}, \boldsymbol{d})$, we can go further to obtain the posterior probability under the final situation $\xi_t(\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{d})$ when both chains stay in current state $\boldsymbol{i}$ without transition.

$$\xi_t(\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{d}) = P(s_t = \boldsymbol{i}, s_{t+1} = \boldsymbol{i}, d_t(\boldsymbol{i}) = \boldsymbol{d}|\boldsymbol{O_{1:T}}, \boldsymbol{\lambda}) = \gamma_t(\boldsymbol{i}, \boldsymbol{d}) - \xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d}) \tag{4-23}$$

4.2.3 Model Parameter Re-estimation

In this subsection, we will illustrate in detail about how to re-estimate parameters of CDD-HSMM via the variables that we defined as shown above.

Baum-Welch algorithm is an efficient and commonly used method for tuning unknown

parameters of HMM [44]–[46]. Two major steps are involved, E-step and M-step. In E-step, $\alpha_t(\boldsymbol{i},\boldsymbol{d})$ and $\beta_t(\boldsymbol{i},\boldsymbol{d})$ are calculated to tells us the probability of hidden states given observations and the parameters before tuning. In M-step, parameters are tuned to best fit the observations and expected hidden states. These two steps are repeated until the probability of observation sequences given parameters is convergent to a maximum value.

Table 4-1 shows the general Baum-Welch algorithm for HMM. We modified it accordingly to make the algorithm more suitable for CDD-HSMM.

Table 4-1 Baum-Welch algorithm for HMM parameter re-estimation

Step 1: Initialization:

Observation sequences $\boldsymbol{O} = \{O_t, 1 \le t \le T\}$

Iteration number $n = 0$

Initialize parameters, $\boldsymbol{\lambda}^{(\mathbf{0})} = (\boldsymbol{\pi}^{(\mathbf{0})}, \boldsymbol{A}^{(\mathbf{0})}, \boldsymbol{B}^{(\mathbf{0})})$

Step 2: Iterative Calculation:

if $P = (\boldsymbol{O}|\boldsymbol{\lambda})$ **is not convergent to a maximum value:**

1. E-step

1) $\alpha_t(i) = \begin{cases} \pi_i b_i(O_1), \text{if } t = 1 \\ \sum_j \alpha_{t-1}(j) a_{ji}\, b_i(O_t), \text{ o.s.} \end{cases}$

2) $\beta_t(i) = \begin{cases} 1, \text{if } t = T \\ \sum_j \beta_{t+1}(j) a_{ij}\, b_j(O_{t+1}), \text{ o.s.} \end{cases}$

3) $\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_j \alpha_t(j)\beta_t(j)}$

4) $\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_i \sum_k \alpha_t(i) a_{ik} b_k(O_{t+1}) \beta_{t+1}(k)}$

2. M-step

1) $\overline{\pi_\iota} = \frac{\gamma_1(i)}{\sum_j \gamma_1(j)}$

2) $\overline{a_{\iota\jmath}} = \frac{\sum_t \xi_t(i,j)}{\sum_t \sum_k \xi_t(i,k)}$

3) $b_i(v) = \frac{\sum_t 1_{O_t=v} \gamma_t(i)}{\sum_t \gamma_t(i)}, 1_{O_t=v} = \begin{cases} 1, \text{if } O_t = v \\ 0, \text{ o.s} \end{cases}$

3. $\boldsymbol{\lambda} = \bar{\boldsymbol{\lambda}}, P = (\boldsymbol{O}|\boldsymbol{\lambda}), n = n + 1$

Else: End

Step 3: Return Estimated Model Parameters $\bar{\boldsymbol{\lambda}}$

With the observation sequences generated by two chains, the parameters of CDD-HSMM $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{D})$ can be re-estimated through variables we defined in the previous section.

First, the initial state probability distribution can be re-estimated as follows:

$$\overline{\pi_{i^1}^1} = \sum_{i^2} \gamma_1(i^1, i^2, 1,1) \tag{4-24}$$

$$\overline{\pi_{i^2}^2} = \sum_{i^1} \gamma_1(i^1, i^2, 1,1) \tag{4-25}$$

Next, the state transition probability distribution $a_{i^c,j^c}^c(d^c)$ for $\forall c \in \{1,2\}$, can be updated through Eq. 4-26 and 4-27.

$$\overline{a_{i^1,j^1}(d^1)} = \frac{\sum_{t=1}^T \sum_{i^2} \sum_{j^2} \sum_{d^2} \xi_t(i^1, i^2, j^1, j^2, d^1, d^2)}{\sum_{t=1}^T \sum_{i^2} \sum_{d^2} \gamma_t(i^1, i^2, d^1, d^2)} \tag{4-26}$$

$$\overline{a_{i^2,j^2}(d^2)} = \frac{\sum_{t=1}^T \sum_{i^1} \sum_{j^1} \sum_{d^1} \xi_t(i^1, i^2, j^1, j^2, d^1, d^2)}{\sum_{t=1}^T \sum_{i^1} \sum_{d^1} \gamma_t(i^1, i^2, d^1, d^2)} \tag{4-27}$$

For observation probability distribution, we assume that $b_{i^c}^c(d^c)$ for $\forall c \in \{1,2\}$ follows Normal distribution. In this case, we have two elements, mean and variance, to be updated, which can be derived through Eq. 4-28 - Eq. 4-31.

$$\overline{\mu_{i^1,j^1}(d^1)} = \frac{\sum_{t=1}^T \sum_{i^2} \sum_{j^2} \sum_{d^2} \gamma_t(i^1, i^2, j^1, j^2, d^1, d^2) O_t^1}{\sum_{t=1}^T \sum_{i^2} \sum_{d^2} \gamma_t(i^1, i^2, d^1, d^2)} \tag{4-28}$$

$$\overline{\mu_{i^2,j^2}(d^2)} = \frac{\sum_{t=1}^T \sum_{i^1} \sum_{j^1} \sum_{d^1} \gamma_t(i^1, i^2, j^1, j^2, d^1, d^2) O_t^2}{\sum_{t=1}^T \sum_{i^1} \sum_{d^1} \gamma_t(i^1, i^2, d^1, d^2)} \tag{4-29}$$

$$\overline{\sigma_{i^1,j^1}(d^1)} = \frac{\sum_{t=1}^T \sum_{i^2} \sum_{j^2} \sum_{d^2} \gamma_t(i^1, i^2, j^1, j^2, d^1, d^2)\, \phi^1}{\sum_{t=1}^T \sum_{i^2} \sum_{d^2} \gamma_t(i^1, i^2, d^1, d^2)} \tag{4-30}$$

$$\overline{\sigma_{i^2,j^1}(d^2)} = \frac{\sum_{t=1}^T \sum_{i^1} \sum_{j^1} \sum_{d^1} \gamma_t(i^1, i^2, j^1, j^2, d^1, d^2)\, \phi^2}{\sum_{t=1}^T \sum_{i^1} \sum_{d^1} \gamma_t(i^1, i^2, d^1, d^2)} \tag{4-31}$$

Where $\phi^1 = (O_t^1 - \mu_{i^1,j^1}(d^1))$ and $\phi^2 = (O_t^2 - \mu_{i^2,j^2}(d^2))$.

Finally, the state duration probability can be written as follows:

$$\overline{p_{i^1}(d^1)} = \frac{\sum_{t=1}^T \sum_{i^2} \sum_{j^2} \sum_{d^2} \sum_{j^1} \xi_t(i^1, i^2, j^1, j^2, d^1, d^2)}{\sum_{t=1}^T \xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d})} \tag{4-32}$$

$$\overline{p_{i^2}(d^2)} = \frac{\sum_{t=1}^T \sum_{i^1} \sum_{j^1} \sum_{d^1} \sum_{j^2} \xi_t(i^1, i^2, j^1, j^2, d^1, d^2)}{\sum_{t=1}^T \xi_t(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{d})} \tag{4-33}$$

With the parameter re-estimation formulae shown above, we can update all parameters $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{D})$ by training CDD-HSMM model iteratively until the probability $P = (\boldsymbol{O}|\boldsymbol{\lambda})$ is convergent to a maximum value.

$$P = (\mathbf{O}|\boldsymbol{\lambda}) = \alpha_t(\boldsymbol{i},\boldsymbol{d})\beta_t(\boldsymbol{i},\boldsymbol{d}), \text{ for } \forall t \in \{1,2,\ldots,T\} \quad (4\text{-}34)$$

Table 4-2 Modified Baum-Welch algorithm for CDD-HSMM parameter re-estimation

Step 1: Initialization:

Observation sequences $\boldsymbol{O_{1:T}}$

Iteration number $n = 0$

Initial model, $\boldsymbol{\lambda^{(0)}} = (\boldsymbol{\pi^{(0)}}, \boldsymbol{A^{(0)}}, \boldsymbol{B^{(0)}}, \boldsymbol{D^{(0)}})$

Step 2: Iterative Calculation:

if $P = (\boldsymbol{O}|\boldsymbol{\lambda})$ **is not convergent to a maximum value:**

1. E-step

5) $\alpha_t(\boldsymbol{i},\boldsymbol{d})$: Eq. 6-6 – 6-10

6) $\beta_t(\boldsymbol{i},\boldsymbol{d})$: Eq. 6-12 – 6-17

7) $\gamma_t(\boldsymbol{i},\boldsymbol{d})$: Eq. 6-22

8) $\xi_t(\boldsymbol{i},\boldsymbol{j},\boldsymbol{d})$: Eq. 6-19 – 6-21, Eq. 6-23

2. M-step

4) $\overline{\pi_{i^c}^c}$: Eq. 6-24 and 6-25

5) $\overline{a_{i^c,j^c}^c(d^c)}$: Eq. 6-26 and 6-27

6) $\overline{\mu_{i^c,j^c}(d^c)}$: Eq. 6-29 and 6-30

7) $\overline{\sigma_{i^c,j^c}(d^c)}$: Eq. 6-31 and 6-32

8) $\overline{p_{i^c}(d^c)}$: Eq. 6-33 and 6-34

3. $\boldsymbol{\lambda} = \bar{\boldsymbol{\lambda}}$, $P = (\boldsymbol{O}|\boldsymbol{\lambda})$, $n = n + 1$

Else: End

Step 3: Return Estimated Model Parameters $\bar{\boldsymbol{\lambda}}$

4.3 Case Study

This section investigates the performance of proposed CDD-HSMM through two case studies on different datasets. The first dataset is Turbofan Engine Degradation Dataset provided by NASA. In the first case study, we applied CDD-HSMM on recognizing the health state behind the observation sequences to predict the RUL of turbofan engines. The other dataset we used in the

second case study comes from Alzheimer's Disease Neuroimaging Initiative (ADNI) study. We tried to accurately diagnose Alzheimer's Disease (AD) stage with the longitudinal measurements of participants provided by ADNI study via CDD-HSMM. Both datasets contain multiple different sensor signals/ clinical measurements in time order (multi-dimensional time series data). Therefore, the two case studies designed in this paper can evaluate the effectiveness of CDD-HSMM as a new approach for sensor fusion. The model performance of CDD-HSMM in both case studies will be compared with other three popular HMM-based models, which are HMM, HSMM, and CHMM.

Table 4-3 Description of sensors

#	symbol	description	units
1	T2	Total temperature at fan inlet	°R
2	T24	Total temperature at LPC inlet	°R
3	T30	Total temperature at HPC inlet	°R
4	T50	Total temperature at LPT inlet	°R
5	P2	Pressure at fan inlet	psia
6	P15	Total pressure in bypass-duct	psia
7	P30	Total pressure at HPC outlet	psia
8	Nf	Physical fan speed	rpm
9	Nc	Physical core speed	rpm
10	Epr	Engine pressure ratio	-
11	Ps30	Static pressure at HPC outlet	psia
12	Phi	Ratio of fuel flow to Ps30	pps
13	NRf	Corrected fan speed	rpm
14	NRc	Corrected core speed	rpm
15	BPR	Bypass ratio	-
16	farB	Burner fuel-air ratio	-
17	htBleed	Bleed enthalpy	-
18	NF_dmd	Demanded fan speed	rpm
19	PCNR_dmd	Demanded corrected fan speed	rpm
20	W31	HPT coolant bleed	lbm/s
21	W32	LPT coolant bleed	lbm/s

4.3.1 Turbofan Engine Dataset

The turbofan engine degradation dataset is simulated by Commercial Modular Aero-Propulsion System Simulator (C-MAPSS) of NASA [47]. This dataset records 21 sensor signals monitoring 100 engines during their whole service life cycles, from normal operation to failure. The detailed information of 21 sensors is listed in Table 4-3.

Through visualization, we found these sensor signals have different trends as turbofan engines degrade, as Figure 4-5 shows. For example, sensor 2 reveals an obvious increasing trend while sensor 7 displays an opposite decreasing trend. And sensor 1 doesn't show any significant variation on measurement values. According to [18], [48], [49], the original sensor signals in this dataset are not qualified for prognostic analysis, because some of them either don't show any monotonic

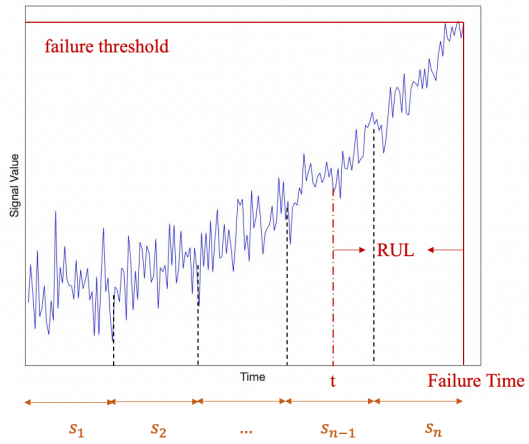


Figure 4-5 Health state division

trend or have large variances under same environmental conditions. Therefore, we select two widely cited health indexes developed from original sensors, which better characterize the degradation behavior of the system. Each health index is considered as one chain in the CDD-HSMM model we developed. These two health indexes are proposed by [49] and [50] respectively.

4.3.1.1 Problem Setup and Training

The goal of this case study is to accurately predict the RUL of engine unit given two health indexes we select with CDD-HSMM. However, CDD-HSMM cannot directly predict RUL but is able to identify the hidden states and duration time based on the time-series observations. Therefore, to make data fully prepared for training, we determined to divide the full life-cycle

health indexes of all engines into n health states first, as Figure 4-6 shows. But before state division, there is still more work we need to do. After analyzing, we found that the turbofan engines in the dataset have different lengths of life cycle. According to the statistics in Table 4-4, the maximum engine life span is 362 cycles, whereas the minimum life span is only 128 cycles. This is because all engines started to be monitored at a random time point when they are under the normal operation condition. Considering that the observations collected under the normal condition provide limited clue for degradation modeling, we decided to remove the health indexes at the very beginning and make all life spans equal to 126 cycles. Then, we divided the 126 cycles into n health states. After testing different n numbers, we finally set n equal to 7, taking account of accuracy and time efficiency.

Table 4-4 Engine life span statistics

Max life span (cycles)	Min life span (cycle)	Average life span (cycles)
362	128	206

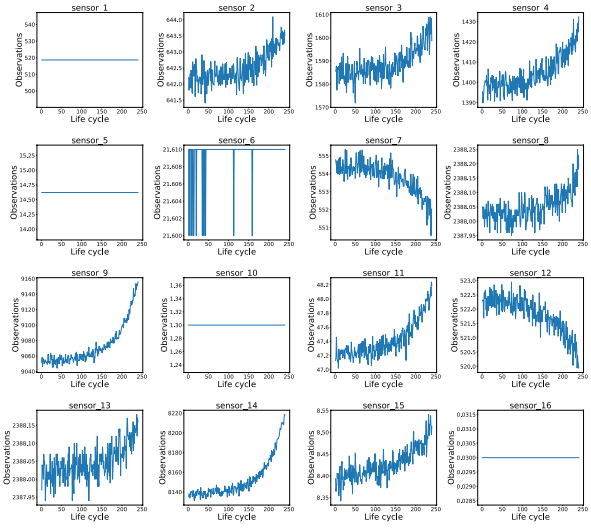


Figure 4-6 Sensor (numbered 1-16) visualization on engine 11

Given the health indexes available and number of states, we can estimate parameters of CDD-HSMM through training with the modified Baum-Welch algorithm as mentioned previously.

CDD-HSMM in this case is designed to have a left-to-right structure, where the health state can only transit to the adjacent right state or stay in itself [28]. Because the health condition of engines cannot go into reverse without maintenances. With this left-to-right structure, the model

parameters are assigned as: the initial state distribution $\pi_i = [1, 0, 0, 0, 0, 0, 0]$, for $\forall i = 1,2,\ldots,7$; the state transition probability $a_{ij}^{c,c'} = 0.5$, for $\forall i = 1,2,\ldots,7, \forall\, c, c' = 1,2$, where $j = i$, or $j = i + 1$, otherwise, $a_{ij}^{c,c'} = 0$; the initial duration probability distribution is set as $p_{i,d}^{c} = \frac{1}{D}$, for $\forall i = 1,2,\ldots,7, \forall c = 1,2, \forall d = 1,2,\ldots,D$, where $D = \frac{126}{7} = 18$, which is the maximum length of duration for each health state; the observation probability follows Normal distribution, $b_{i,d}^{c} = N(\mu_{i,d}^{c}, \sigma_{i,d}^{c})$, for $\forall i = 1,2,\ldots,7, \forall c = 1,2, \forall d = 1,2,\ldots,D$. To get the initial estimation of $\mu_{i,d}^{c}$ and $\sigma_{i,d}^{c}$, we first separated health indexes based their state number i and duration length d respectively. Then we calculate the sample mean and standard deviation for each group, which are further assigned as initial values of $\mu_{i,d}^{c}$ and $\sigma_{i,d}^{c}$.

With the initial estimation of model parameters, we can start training. Among the 100 engines involved, we randomly chose 90 engines for training and remaining 10 engines for testing. Besides, to assess the consistency of the model performance, we developed 10 CDD-HSMM models with random selection of training and testing engines and same initial assignments of model parameters.

Next, we can go further to obtain the most likely hidden state sequences and corresponding durations with Viterbi algorithm. Viterbi algorithm is a dynamic programming method and has

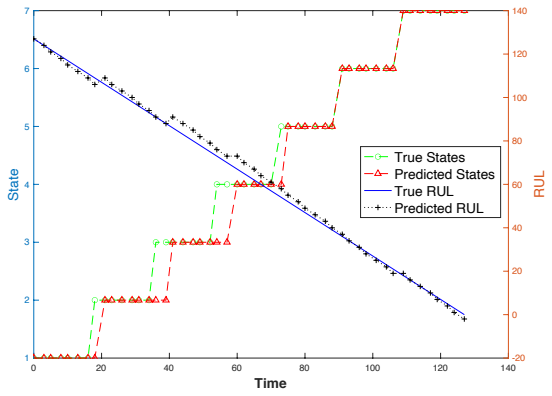


Figure 4-7 Health state identification and RUL prediction results of engine 95

been widely used for solving the decoding problem of HMM. Table 4-5 shows the modified Viterbi algorithm we particularly designed for CDD-HSMM.

Considering that both health indexes are generated from the same turbofan degradation process, so we specially define that $\overline{s_t^1} = \overline{s_t^2}, for\ \forall t \in [1, T]$ in our case study.

Table 4-5 Modified Viterbi algorithm for CDD-HSMM

Input:

observation sequences $\boldsymbol{O} = (\boldsymbol{O_1}, \boldsymbol{O_2}, \dots, \boldsymbol{O_T}), \boldsymbol{O_t} = (O_t^1, O_t^2)$, for $\forall t \in [1, T]$

estimated model parameters $\bar{\boldsymbol{\lambda}} = (\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{A}}, \bar{\boldsymbol{B}}, \bar{\boldsymbol{D}})$

Output:

most likely hidden state sequences $\bar{\boldsymbol{S}} = (\overline{\boldsymbol{s_1}}, \overline{\boldsymbol{s_2}}, \dots, \overline{\boldsymbol{s_T}}), \overline{\boldsymbol{s_t}} = \left(\overline{s_t^1}, \overline{s_t^2}\right)$, for $\forall t \in [1, T]$

and corresponding duration sequences $\bar{\boldsymbol{Z}} = \left(\overline{\boldsymbol{d_1}}, \overline{\boldsymbol{d_2}}, \dots, \overline{\boldsymbol{d_T}}\right), \overline{\boldsymbol{d_t}} = \left(\overline{d_t^1}, \overline{d_t^2}\right)$, for $\forall t \in [1, T]$

Step 1: Create multi-dimensional matrices $\boldsymbol{Q}$ and $\boldsymbol{X}$:

When $t = 1$, for each composite state $\boldsymbol{i}$ and duration $\boldsymbol{d}$:

$$\boldsymbol{Q_{t=1}}(\boldsymbol{i}, \boldsymbol{d}) = \alpha_1(\boldsymbol{i}, \boldsymbol{d})$$

$$\boldsymbol{X_{t=1}}(\boldsymbol{i}, \boldsymbol{d}) = 0$$

End for

When $t > 1$, for each composite state $\boldsymbol{i}$ and duration $\boldsymbol{d}$:

$$\boldsymbol{Q_t}(\boldsymbol{i}, \boldsymbol{d}) = \max_{\boldsymbol{i},\boldsymbol{d}} \alpha_t(\boldsymbol{i}, \boldsymbol{d})$$

$$\boldsymbol{X_{t=1}}(\boldsymbol{i}, \boldsymbol{d}) = arg \max_{\boldsymbol{i},\boldsymbol{d}} \alpha_t(\boldsymbol{i}, \boldsymbol{d})$$

End for

Step 2: Backtrack:

When $t = T$:

$$(\overline{\boldsymbol{s_T}}, \overline{\boldsymbol{d_T}}) = arg \max_{\boldsymbol{i},\boldsymbol{d}} \boldsymbol{Q_T}(\boldsymbol{i}, \boldsymbol{d})$$

For $t = T, T-1, \dots, 2, 1$:

If $\overline{d_t^c} > 1$, for $\forall c \in \{1,2\}$:

$$\overline{d_{t-1}^c} = \overline{d_t^c} - 1$$

$$\overline{s_{t-1}^c} = \overline{s_t^c}$$

Else:

$$\overline{\boldsymbol{s_{t-1}}}, \overline{\boldsymbol{d_{t-1}}} = \boldsymbol{X_t}\left(\overline{\boldsymbol{s_t}}, \overline{\boldsymbol{d_t}}\right)$$

Step 3: Return $\bar{\boldsymbol{S}}$ and $\bar{\boldsymbol{Z}}$

Once the estimated state sequences $\bar{\boldsymbol{S}}$ and corresponding durations $\bar{\boldsymbol{Z}}$ are returned, we can use Eq. 4-36 to calculate RUL based on the estimated health state number $\overline{s_t}$ and duration time in that

state $\overline{d_t}$. l in Eq. 4-35 is the maximum time length of health states, which is equal for all states.

$$\overline{RUL} = l(\overline{s_t} - 1) + \overline{d_t} \tag{4-35}$$

4.3.1.2 Performance Evaluation

As mentioned above, we need to identify the hidden health states as well as state duration with CDD-HSMM at first in our case study. With the estimated health states and duration time, we can further infer the RUL accordingly through Eq. 4-36. Therefore, there are two major tasks in this case study, health state identification and RUL prediction. Figure 4-7 visually shows the prediction results of these two tasks on a randomly selected engine from the testing set. In Figure 4-7, the green dotted line represents the true health states as the engine degrades with time, and the red dotted line stands for predicted health states. The blue solid line illustrates the real RUL values which linearly decreases as time goes on. While the black dash line represents the predicted RUL values. From Figure 4-7, we can see that CDD-HSMM successfully recognized each health state and predicted RUL in the case of Engine 95, since most of green dots are covered by red dots. And the blue solid line and black dash line almost coincide with each other.

To evaluate the model performances on these two tasks, we use precision score and RMSE respectively to demonstrate the effectiveness of the proposed CDD-HSMM model and compare it with other three benchmark models, which are HMM, HSMM, CHMM models.

Precision score is a commonly used evaluation metric for classification, calculated as Eq. 4-36.

$$precison\ score = \frac{TP}{TP + FP} \tag{4-36}$$

For each health state in our case study, TP is defined as the correctly predicted instances, and FP is the instance, which is wrongly predicted as that health state, but it should not be. Figure 8 presents the precision score comparison results of CDD-HSMM model with other three benchmark models. The x-axis in Figure 4-8 is the health state number from 1 to 7. The y-axis is the precision score from 0 to 1. As Figure 4-8 shows, CDD-HSMM almost outperforms all other benchmark models at all health states, especially in the states at the middle of degradation process. Additionally, the precision score of CDD-HSMM model on the health states at the beginning and the end of the degradation can go above 0.9, demonstrating the advantageous performance on health state recognition.

Accurate health state recognition naturally leads to accurate RUL prediction, which is evaluated by RMSE in our case study. RMSE is used to measure the differences between the true RUL and predicted RUL, denoted as RUL and $\widehat{RUL}$ respectively in Eq. 4-37. From Eq. 4-37, we can conclude that the lower the RMSE, the more accurate the prediction.

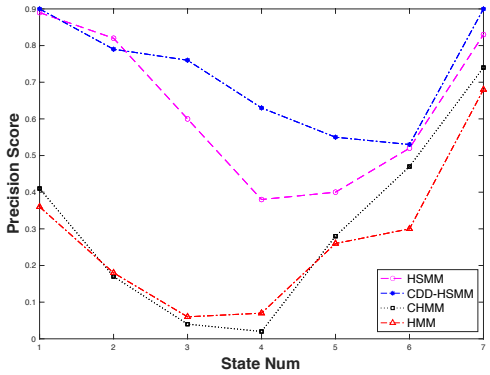


Figure 4-8 Precision score of each health state on testing set

$$RMSE = \sqrt{\frac{\sum_t^T \left(\widehat{RUL}(t) - RUL(t)\right)^2}{T}} \tag{4-37}$$

Table 4-6 lists the RMSE comparison results of all models at different observation points. The percentage in the parentheses is RMSE reduction in percent if we compare the benchmark model with CDD-HSMM.

Table 4-6 RMSE comparison results on testing set

Cycles Before Failure	HMM	HSMM	CHMM	CDD-HSMM
5	1.77(27.11%)	1.31(1.53%)	2.28(47.37%)	1.29
10	1.70(18.82%)	1.27(-8.66%)	2.22(37.8%)	1.38
30	2.70(55.56%)	1.38(13.04%)	2.51(52.19%)	1.20
50	4.79(74.95%)	2.40(50.00%)	4.42(72.85%)	1.20

As we can see from Table 4-6, CDD-HSMM owns the smallest RMSE scores at three observation points (5, 30, and 50 cycles before failure). For observation point, which is 50 cycles before failure, RMSE is decreased by over 70% if we compare CDD-HSMM with HMM/CHMM,

which is a huge improvement of performance. And for other observation points, the performance improvement measured by RMSE is also significant. Table 4-6 convincingly proves the efficacy of the proposed CDD-HSMM model on the application of RUL prediction.

4.3.2 ADNI Dataset

The second dataset we used is from the Alzheimer's Disease Neuroimaging Initiative (ADNI) research study. ADNI aims to support investigations that help to slow or stop the Alzheimer's Disease (AD). In their study, clinical, imaging, generic and biospecimen biomarkers are collected to track the progression of AD in human brain.

In our case study, we chose ADNI Merge dataset from the whole ADNI database to diagnose AD stages of participants. ADNI Merge dataset utilizes measurements like Mini-Mental State Examination (MMSE), Clinical Dementia Rating Sum of Boxes (CDRSB), Alzheimer's Disease Assessment Scale (ADAS) and Rey Auditory Verbal Learning Test (RAVLT) to track the progression of AD over 2000 participants. Here, we selected two widely used measurements, MMSE and CDRSB, as the major predictors because of their accessibility and effectiveness [51], [52]. Additionally, for purpose of training, we only considered the participants who keep taking regular examination every half year for over at least 2 years. In this case, 789 participants were left in our experiment.

In ADNI Merge dataset, the Alzheimer's Disease is described in 5 stages with a progressive pattern of cognitive impairment. These 5 stages in order are Cognitively Normal (CN), Significant Memory Concerns (SMC), Early Mild Cognitive Impairment (EMCI), Late Mild Cognitive Impairment (LMCI), and AD. Every half year, the AD stage of each participant is re-evaluated and recorded in the dataset. After reviewing existing papers and official statements of ADNI study [53]–[57], we finally decided to combine CN and SMC into CN, EMCI and LMCI into MCI separately.

4.3.2.1 Problem Setup and Training

The goal of second case study is to develop a well-performed CDD-HSMM model to provide accurate diagnosis of AD stage for each participant based on the longitudinal measurements provided by the dataset. Therefore, a CDD-HSMM model with three AD stages (CN, MCI, and AD) is trained offline using the two selected longitudinal measurements (MMSE and CDRSB)

with the modified Baum-Welch algorithm. Each measurement is considered as one chain in our CDD-HSMM model.

Same as the first case study, we need to assign initial values to model parameters before training. The initial state probability distribution of each chain is set as $\pi_i = [0.33, 0.33, 0.33]$, for $\forall i = 1,2$. The initial state transition probability distribution of each chain is also set as equiprobability, that is $a_{ij}^{c,c'} = 0.33$, for $\forall i, j = 1,2,3, \forall c, c' = 1,2$.

The initial assignments of duration probability distribution and observation probability distribution are more complicated. After exploratory data analysis, we found that the maximum length of participant longitudinal measurements is equal to 5 (to the best situation, some participants keep taking semiannual examination for two and a half years). Therefore, we assume that maximum length of duration for each stage is also equal to 5. Consequently, the initial duration probability distribution is set up as $p_i^c(d) = 0.2$, for $\forall i = 1,2,3, \forall c = 1,2, \forall d = 1,2,3,4,5$, which is also equiprobability. The observation probability is assumed to follow the Normal distribution with two elements. To set the initial values, we first separated the longitudinal measurements according to their stage number and duration. Then, we calculate the sample mean and standard deviation for each group, which are further assigned as initial values of the observation probability distribution.

During training stage, we randomly chose 710 participants for training, and remaining 79 participants for testing. Besides, in order to assess the consistency of the model performance, we developed 10 CDD-HSMM models with random selection of training and testing participants and same initial assignments of parameters.

4.3.2.2 Performance Evaluation

After re-estimating model parameters, we can identify the AD stage behind measurements with the modified Viterbi algorithm shown in Table 4-5. For each experimental run, we will summarize the AD stage classification results of all participants from the testing set in a confusion matrix. For example, Figure 4-9 is the confusion matrix of four models on the 6th experimental run. Based on confusion matrix, we use two metrics, recall and precision score, to compare the classification performance of CDD-HSMM with other benchmark models on each AD stage.

The formula of precision score in this case study has been shown in Eq. 4-36. Eq. 4-38 shows the formula of recall. Like the first case study, both metrics are used to evaluate the model performance at each AD stage. For each AD stage, FN is the instance that is wrongly predicted as another different AD stage, but it should be in effect.

$$recall = \frac{TP}{TP + FN} \tag{4-38}$$

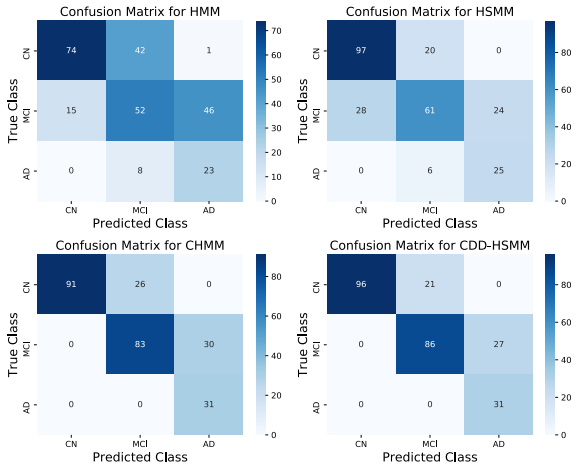


Figure 4-9 Confusion matrix of four models on experiment 6

Table 4-7 and 4-8 present the recall and precision score comparison results on all AD stages for each experimental run, where the percentage in parentheses is the improvement in percent achieved by CDD-HSMM.

Based on Table 4-7, we can see that CDD-HSMM outperforms other three models on each stage with the highest recall value. Take the stage of AD as an example, the recall value of CDD-HSMM can reach to 1 at highest, and the mean value is equal to 0.96. The high recall value returned by CDD-HSMM means that our model has the great power to recognize most of patients on the stage of AD. By comparison, the mean recall values of HMM and HSMM are only 0.80 and 0.78 respectively, about 20% lower than that of CDD-HSMM. Similar situations can be seen on other two stages.

Table 4-8 shows the precision score comparison results on four models. It is clear to see the precision score of CDD-HSMM is higher than any other model on CN and MCI stages, which are 0.91 and 0.84 respectively. For AD stage, the precision score of CDD-HSMM equals to 0.48, indicating that our model returns lots of false positives (predicted as AD but it is not actually).

This is because that the ADNI Merge dataset is imbalanced, and AD stage is the minority class (around 11.1% of the whole data). In this case, the characteristics of AD stage are not fully captured and there are more instances from other two stages to be wrongly predicted as AD stage.

Table 4-7 Recall comparison results

		CDD-HSMM	CHMM	HSMM	HMM
CN	1	0.85	0.61	0.69	0.68
	2	0.78	0.75	0.88	0.75
	3	0.93	0.85	0.88	0.85
	4	0.84	0.7	0.65	0.6
	5	0.92	0.92	0.83	0.89
	6	0.82	0.78	0.82	0.63
	7	0.67	0.58	0.79	0.79
	8	0.97	0.97	1	1
	9	0.76	0.66	0.77	0.83
	10	0.75	0.67	0.63	0.81
	mean	0.83	0.75(10.67%)	0.79(5.06%)	0.78(6.41%)
MCI	1	0.68	0.66	0.47	0.16
	2	0.67	0.5	0.51	0.36
	3	0.66	0.67	0.62	0.4
	4	0.65	0.61	0.4	0.32
	5	0.52	0.51	0.41	0.29
	6	0.76	0.73	0.59	0.5
	7	0.7	0.67	0.49	0.29
	8	0.46	0.63	0.41	0.32
	9	0.68	0.66	0.61	0.28
	10	0.62	0.56	0.58	0.43
	mean	0.64	0.62(3.23%)	0.51(25.49%)	0.34(88.24%)
AD	1	1	1	0.85	1
	2	1	1	0.67	0.81
	3	1	1	0.69	0.62
	4	1	1	0.88	0.71
	5	1	1	0.83	0.92
	6	1	1	0.81	0.9
	7	0.87	0.87	0.83	0.77
	8	0.76	0.76	0.57	0.52
	9	1	1	0.91	0.97
	10	0.92	0.92	0.77	0.82
	mean	0.96	0.96(0.00%)	0.78(23.08%)	0.80(20.00%)

Table 4-8 Precision score comparison results

		CDD-HSMM	CHMM	HSMM	HMM
CN	1	0.93	1	0.59	0.58
	2	0.91	0.90	0.59	0.55
	3	0.79	0.90	0.49	0.69
	4	1	1	0.54	0.53
	5	0.94	1	0.55	0.55
	6	1	1	0.78	0.83
	7	1	1	0.68	0.59
	8	0.74	0.88	0.59	0.56
	9	1	1	0.48	0.58
	10	0.79	0.77	0.45	0.45
	mean	0.91	0.95(4.21%)	0.57(59.65%)	0.59(52.54%)
MCI	1	0.88	0.73	0.63	0.44
	2	0.84	0.77	0.69	0.76
	3	0.96	0.91	0.79	0.86
	4	0.87	0.79	0.62	0.62
	5	0.94	0.94	0.80	0.82
	6	0.80	0.76	0.70	0.71
	7	0.70	0.63	0.64	0.52
	8	0.74	0.79	0.58	0.37
	9	0.81	0.74	0.68	0.61
	10	0.83	0.81	0.87	0.72
	mean	0.84	0.79(6.33%)	0.70(20.00%)	0.64(31.25%)
AD	1	0.51	0.44	0.59	0.38
	2	0.46	0.35	0.47	0.48
	3	0.50	0.45	0.70	0.53
	4	0.41	0.37	0.41	0.41
	5	0.32	0.30	0.37	0.32
	6	0.53	0.51	0.51	0.33
	7	0.67	0.63	0.67	0.63
	8	0.62	0.62	0.80	0.56
	9	0.42	0.41	0.44	0.33
	10	0.42	0.35	0.54	0.45
	mean	0.48	0.44(9.09%)	0.55(-12.7%)	0.44(9.09%)

4.4 Conclusion and Future Work

In this paper, an advanced HMM extension, CDD-HSMM, is proposed to provide more accurate and robust performance by fusing multiple sensor signals. Compared with existing HMM-based models, CDD-HSMM proposed can not only describe the interactions among multiple sensors, but also taking the time-varying characteristics of both state transitions and observations. Therefore, due to the complex model structure and modified model parameters, CDD-HSMM is more consistent with reality, thus performing better than HMM and its extensions according to the case study results shown in this paper. Besides, we also contributed modified Baum-Welch and Viterbi algorithms for the proposed CDD-HSMM to re-estimate parameters and find the most-likely sate sequences behind the observations.

Future research studies can be directed to i) investigate how to improve the efficiency of modified Baum-Welch algorithm by decreasing the calculation time but without increasing the space needed; ii) explore the model performance under the situation when there are more than two chains involved.

Chapter 5 A Dual-LSTM Framework Combining Change Point Detection and Remaining Useful Life Prediction

Remaining useful life (RUL) prediction is a key task of prognostics and health management, which helps to schedule condition-based maintenance. With the massive, collected data from multiple sensors and the rapid development of computing technology, Deep Learning (DL) techniques become prevalent in the field of RUL prediction. Specifically, the Long Short-Term Memory (LSTM) neural networks show excellent RUL prediction performance compared to traditional techniques due to its capability of capturing both long- and short-term temporal patterns. However, most existing LSTM-based models assume that (i) operating units start to degrade at the same time in operation or (ii) restrict a constant time length of the degradation process. This paper addresses these non-practical assumptions by proposing a Dual-LSTM framework that real-time detects the starting point of degradation and then models the RUL. Different from existing LSTM-based models, the Dual-LSTM relaxes the strong assumption of the fixed change point and detects the uncertain change point unit by unit at first. Then, the Dual-LSTM predicts the health index beyond the change point which can be leveraged to calculate the RUL based on a piecewise decreasing health index function proposed in this paper. The proposed Dual-LSTM (i) filters out the sensor signals irrelevant to the RUL prediction through the change point detection, (ii) introduces a novel one-dimension piecewise decreasing health index function to differentiate the normal operation and degradation process, as well as indicate the health condition of the unit, (iii) makes full use of historical information to achieve detection and prediction tasks by characterizing both long and short-term dependencies of sensor signals through LSTM network. The effectiveness of the proposed Dual-LSTM framework is validated and compared to state-of-art benchmark methods on a publicly available turbofan engine degradation dataset.

5.1 Introduction

Maintenance plays an important role by directly affecting the service life and production efficiency of the industrial system [109], [110]. There have been so many different maintenance strategies proposed by existing literatures [111], including corrective maintenance [112], [113], periodic inspections [114], [115], risk-based maintenance [116], [117], opportunistic maintenance [118]–[120] and condition-based maintenance (CBM) [71], [111], [121], [122]. In recent years, CBM has been growing rapidly and applied extensively [109], [123] to avoid the unnecessary and excessive costs generated by the uncertain system failures. Besides, the advanced monitoring and computing technologies also boost the development of CBM [123]. Remaining useful life (RUL) prediction is a key task in the field of CBM, which provide information about when the unit or system will comes to the end of useful life [8], [124]. The accurate RUL prediction supports smart maintenance scheduling since it helps to raise early alarms and avoid unexpected failures, improve production efficiency and reduce life-cycle costs [109], [125], [126].

Due to the rapid development of sensing technology, a huge variety of sensors are used to monitor the health condition of the units [5], [35], [37], [106]. With the large number of sensor signals generated on hand, researchers proposed many data-driven approaches to model the degradation process to predict the RUL of the units. The classical data-driven approaches are statistics-based. For example, C. Ordóñez et al. [127] proposed a hybrid ARIMA-SVM model for RUL prediction. At first, an ARIMA model is designed to predict the values of sensor signals in advance. Then, the prediction results of sensor signals are further used as the inputs of an SVM model to predict the RUL. A. Chehade et al. [37] developed a sensor fusion framework via a series of statistical hypothesis testing. With this new framework, the health index (HI) maximizing the separability between any two degradation states is constructed. Based on the constructed HI, the remaining time to each degradation state including the failure can be predicted. A. Chehade et al. [106] also proposed a convex quadratic formulation to combine the weighted models of historical units, thus reconstructing the Bayesian updated degradation model of the operating units. Li et al. [30] developed an RUL prediction method to describe the degradation process with a general expression of age- and stage-dependent models. Wang et al. [128] presented a Continuous Hidden Markov Model (CHMM) based approach for RUL prediction. They diagnosed the wear state of milling tool with CHMM first, then predicted RUL with the Gaussian regression model. Si et al.

[52] established a Wiener-process-based degradation model with a recursive filter algorithm to predict the RUL. Compared with physics-based models, statistical data-driven models are usually more accurate, quicker and cheaper to deploy, and don't require a comprehensive physical understanding of the unit [129].

Recently, the recurrent neural network (RNN), a Deep Learning (DL) technique, became prevalent in the application of RUL prediction. RNN shows strong capabilities of capturing the complex non-linear relationships between inputs and outputs, as well as the short-term dependencies in time series [62], [130]. Heimes [20] is the first one who utilized RNN to predict the RUL of turbofan engines. Liang Guo et al. [26] designed a HI based RNN to predict the RUL of bearings. However, RNN usually underperforms when learning long-term dependencies of time series due to the vanishing and exploding gradient problems [131]–[134].

LSTM, a variant version of RNN, extends the ideas of RNN and can prevent the vanishing and exploding gradient problems. This advantage attributes to the cell states and gate structures of the LSTM. Within the LSTM network, inputs and historical memories selectively pass over the network by the cell states. The selection process is carefully regulated by the gate structures. By doing so, the cell states and gate structures create a path in the network for the truly relevant and important information to flow, which will not be forgotten during the backpropagation. In this case, the vanishing and exploding problem can be solved and both long- and short- term memories are characterized [134]–[136]. There are some efforts that leveraged LSTM networks for RUL prediction [23], [59], [60]. Both Shuai Zheng *et al.* [59] and Shaopeng Dong *et al.* [23] utilized the LSTM network to predict the RUL and tested the performance on different case-studies. For most case-studies, the LSTM network showed better RUL prediction performance in comparison to other benchmark models, such as multilayer perceptron (MLP), support vector machine (SVM), convolutional neural network (CNN), RNN, and gated recurrent units (GRU). Different from S. Zheng et al. and Shaopeng Dong et al., Jianjing Zhang et al. [60] didn't develop an LSTM which outputs the predicted RUL from the sensor data directly. Instead, they derived a dimensionless HI through a single-layer perceptron at first. Then they continuously fed a sequence of consecutive HIs into a Bi-directional LSTM to predict the HI at one-cycle ahead until the predicted HI hit the predetermined failure threshold. Finally, the RUL was calculated as the length from the current time to the time when the HI went beyond the threshold. Wang et al. [137] proposed a hybrid

model for RUL prediction. In Wang et al., the LSTM network was used to predict the RUL based on the long sequences of sensor signals, and the gradient boosting regression (GBR) was adopted to predict the RUL with the short sequences. Finally, the time series of predicted RUL returned by LSTM and GBR respectively were used as the input of the backpropagation neural network (BPNN) to get the final predicted RUL. Besides, some variants of the LSTM network are further proposed to predict the RUL under more complex conditions and achieve higher precisions. Elsheikh et al. [61] established a new bidirectional handshaking LSTM (BHLSTM) to predict the RUL when given short sequences of sensor signals with random initial wear. Miao et al. [62] designed a dual-task LSTM network to assess the degradation stages and predict the RUL of turbofan engines in a parallel way. Song et al. [69] proposed a hybrid prediction model integrating the autoencoder and bidirectional LSTM (BLSTM) to improve the RUL prediction accuracy. Li et al. [129] developed a directed acyclic graph (DAG) network combining CNN and LSTM in a parallel manner for RUL prediction. With this DAG network, CNN for feature extraction could be corrected based on the prediction error returned by the LSTM.

Although the LSTM has presented an outstanding ability in the application field of RUL prediction, there still exists a drawback that needs to be overcome. In practice, it is common to see that the engine unit starts at the normal operation level and begins to degrade at an uncertain change point. It is unreasonable to predict the RUL before the change point because the degradation is negligible during this period [52], [59], [138]. Additionally, the sensor signals collected before the change point provide very limited information about the degradation process. Therefore, it is necessary to detect the change point and mainly focus on the degradation process beyond this change point. However, most existing LSTM networks developed for RUL prediction assume that there is no change point existed and RUL decreases linearly with time from the beginning to the end [137]. Other LSTMs simply assign a fixed change point for any unit regardless of the operational conditions [59]–[61], [69], [129], [130]. Therefore, due to the lack of accuracy in change point detection, most of those approaches are not practically reliable for degradation modeling. There are only a few research efforts that apply change point detection prior to RUL prediction using LSTMs [48]. For example, Y. Wu et al. [31] used the support vector machine (SVM) to detect the change point but showed poor performance for complex systems. To

overcome this major limitation in existing deep learning models, this paper proposes the Dual-LSTM Network for both change point detection and RUL prediction.

The main contributions of the paper are:

(1) Proposing a new Dual-LSTM framework which combines the change point detection and RUL prediction. Within the framework, the change point detection is a prerequisite which serves to filter out the unnecessary sensor signals irrelevant to the RUL prediction. By doing so, the performance of RUL prediction can be efficiently improved, which has been proved by the experiment results in this paper.

(2) Introducing a novel one-dimension piecewise decreasing HI construction function. On the one hand, the HI function helps to differentiate the normal operation and degradation process. On the other hand, the HI function can accurately indicate the health condition and it is easy to use HI to infer the RUL accordingly.

(3) Characterizing both long- and short-term dependencies within each sensor via the Dual-LSTM framework. Therefore, the historical information is preserved as much as possible and fully

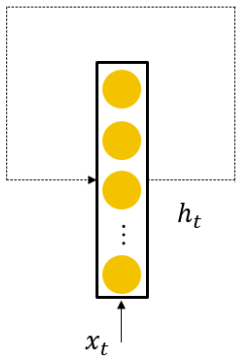


Figure 5-1 RNN cell structure

utilized for change point detection and RUL prediction.

5.2 Theoretical Methods

5.2.1 RNN

RNN is a neural network with a feedback loop as Figure 1 shows. In Figure 5-1, $\boldsymbol{x_t}$ and $\boldsymbol{h_t}$ are the input and hidden state at time step t respectively. At the hidden layer, $\boldsymbol{h_t}$ is a function of the linear combination of $\boldsymbol{h_{t-1}}$ and $\boldsymbol{x_t}$, as shown in Eq. 5-1. At the output layer, the output $\boldsymbol{y_t}$ is

calculated as a linear combination of $\boldsymbol{h_t}$ as shown in Eq. 5-2. The $\boldsymbol{W_h}$, $\boldsymbol{V_h}$, and $\boldsymbol{b_h}$ in Eq. 5-1 as well as the $\boldsymbol{W_y}$, $\boldsymbol{b_y}$ in Eq. 5-2 are the parameters of the RNN that are predicted during the training process.

$$\boldsymbol{h_t} = \sigma(\boldsymbol{W_h h_{t-1}} + \boldsymbol{V_h x_t} + \boldsymbol{b_h}) \tag{5-1}$$

$$\boldsymbol{y_t} = \sigma(\boldsymbol{W_y h_t} + \boldsymbol{b_y}) \tag{5-2}$$

The recursive function of Eq. 5-1 is the feedback loop of the RNN, as shown in Figure 5-1. It allows the RNN to store the memories in the hidden state and pass the hidden state from one-time step to the next. With the feedback loop, the output is not only dependent on the input at the current time but also relies on the memories from previous time steps.

However, RNN proves to be problematic for characterizing long-term dependencies in practice [18]–[21], since it usually experiences vanishing/exploding gradient problems, like the multilayer perceptron (MLP) does. Specifically, during the backpropagation step in the training phase, the derivative of $\boldsymbol{h_t}$ is a product of all derivatives of hidden states at time steps after t based on the chain rule. If those derivatives are small or large, then the derivative of $\boldsymbol{h_t}$ is highly possible to vanish or explode, and so are the derivatives $\boldsymbol{W_h}$ of and $\boldsymbol{V_h}$. This problem will stop the RNN from further training and results in a poorly trained network.

5.2.2 LSTM

To overcome the gradient vanishing/exploding limitation, researchers proposed the LSTM network. LSTM prevents gradients from vanishing or exploding by keeping both long- and short-term memories in the cell state. Besides, three special gates are designed to control the information flow in the LSTM network. They are the forgot gate, input gate, and output gate. The cell structure of the LSTM is shown in Figure 5-2 and the three gates are shaded in yellow.

$$\boldsymbol{f_t} = \sigma(\boldsymbol{W_f h_{t-1}} + \boldsymbol{V_f x_t} + \boldsymbol{b_f}) \tag{5-3}$$

$$\boldsymbol{i_t} = \sigma(\boldsymbol{W_i h_{t-1}} + \boldsymbol{V_i x_t} + \boldsymbol{b_i}) \tag{5-4}$$

$$\boldsymbol{o_t} = \sigma(\boldsymbol{W_o h_{t-1}} + \boldsymbol{V_o x_t} + \boldsymbol{b_o}) \tag{5-5}$$

$$\widetilde{\boldsymbol{c_t}} = tanh(\boldsymbol{W_a h_{t-1}} + \boldsymbol{V_a x_t} + \boldsymbol{b_a}) \tag{5-6}$$

$$\boldsymbol{c_t} = \boldsymbol{f_t} \circ \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \circ \widetilde{\boldsymbol{c_t}} \tag{5-7}$$

$$\boldsymbol{h_t} = \boldsymbol{o_t} \circ tanh(\boldsymbol{c_t}) \tag{5-8}$$

$$\boldsymbol{y_t} = \sigma(\boldsymbol{W_y h_t} + \boldsymbol{b_y}) \tag{5-9}$$

Eq. 5-3 to 5-9 illustrate mathematical formulas behind the LSTM cell. Here, $\boldsymbol{f_t}$, $\boldsymbol{i_t}$, and $\boldsymbol{o_t}$ are the forget, input, and output gate's activation vectors respectively. Each value of the vector ranges from 0 to 1 to control the information flow within the network. The cell state at current time t is noted as $\boldsymbol{c_t}$, which is used to store the memories from prior inputs. From Eq. 5-7, we can see that $\boldsymbol{c_t}$ is updated based on the previous cell state $\boldsymbol{c_{t-1}}$ and a candidate cell state $\widetilde{\boldsymbol{c_t}}$. In RNN, all values in the hidden state will be updated at each time step, as shown in Eq. 5-1. However, the situation is different in LSTM. Not all values of $\boldsymbol{c_t}$ will be updated, which is determined by the forget gate and input gate. More specifically, in Eq. 5-7, $\boldsymbol{f_t}$ decides which value of $\boldsymbol{c_{t-1}}$ should be added into $\boldsymbol{c_t}$, and $\boldsymbol{i_t}$ decides which value of $\widetilde{\boldsymbol{c_t}}$ should be added into $\boldsymbol{c_t}$. Next, $\boldsymbol{o_t}$ decides which value of $\boldsymbol{c_t}$

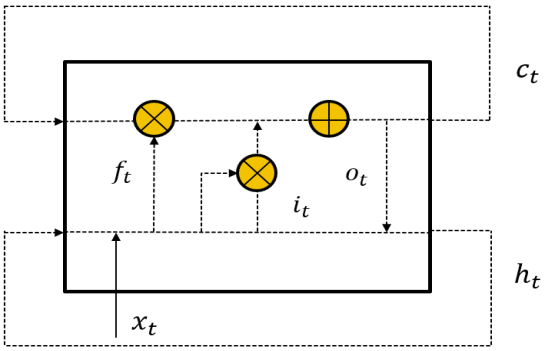


Figure 5-2 LSTM cell structure

should be outputted through the hidden state $\boldsymbol{h_t}$. Finally, the output $\boldsymbol{y_t}$ is calculated based on the hidden state $\boldsymbol{h_t}$, as shown in Eq. 5-9.

With the cell state and three special gates, the memories can be carried for a long period without changes, which avoids the gradient from vanishing or exploding.

5.3 Proposed Dual-LSTM Framework

To achieve a high-precision RUL prediction, a Dual-LSTM framework combining change point detection and RUL prediction is proposed. With the Dual-LSTM framework, we can characterize the unit-to-unit heterogeneity of the change points to mainly focus on the degradation process

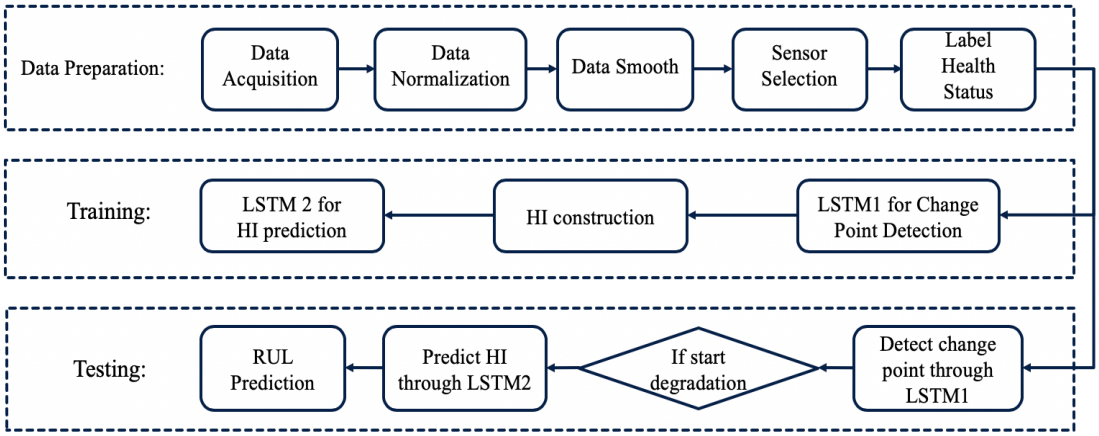


Figure 5-3 Flowchart of the proposed Dual-LSTM framework

beyond the uncertain change point. Figure 5-3 explains how the proposed Dual-LSTM framework works.

As Figure 5-3 shows, the Dual-LSTM framework consists of three stages, data preparation stage, training stage, and testing stage. The data preparation stage aims to make the sensor signals more appropriate for training and testing. In the training stage, the LSTM1 designed for change point detection is learned on the training dataset at first. Then, the HI values of the training units at different time points are constructed based on the change points detected by LSTM1. Then, the constructed HI becomes the target of the LSTM2 which is designed for HI prediction. Finally, the two LSTM networks learned in the training stage are used in the testing stage to predict the RUL of the testing units in real-time.

5.3.1 Change Point Detection

As Figure 5-4 shows, once the data is fully prepared, the first step is to detect the uncertain change point for each unit based on the multi-sensor signals with LSTM1. The network structure of LSTM1 is shown in Figure 5-4.

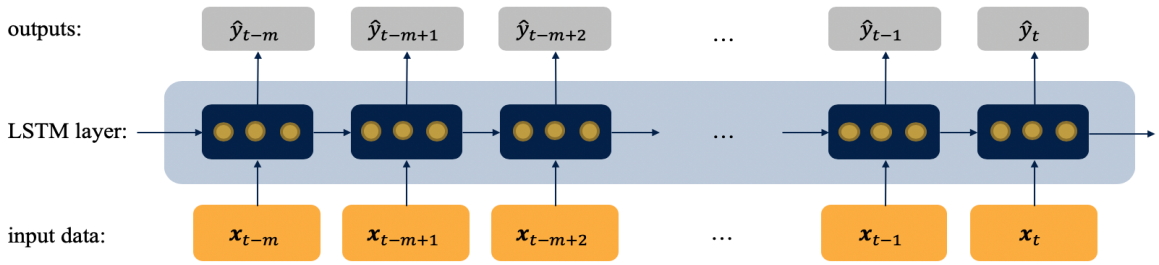


Figure 5-4 Structure of LSTM1 for change point detection

In Figure 5-4, the input $\boldsymbol{x}_t = [x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(n)}]$ represents the multi-sensor data at time t, where n is the number of sensors, and m is the number of time steps looked back. With multiple-dimension inputs $\boldsymbol{X}_t = [\boldsymbol{x}_{t-m}, \boldsymbol{x}_{t-m+1}, \ldots, \boldsymbol{x}_{t-1}, \boldsymbol{x}_t]$ at time t. y_t indicates if the unit is in the normal operation phase or the degradation process phase, which is labeled as 0 before the change point (normal operation) and 1 (degradation process) otherwise. LSTM1 models the $\hat{y}_t$ as a non-linear autoregressive function of the multi-sensor data $\boldsymbol{X}_t$ as shown in Eq. 5-10 to capture the complex relationship between the multi-sensor data and the change point. Here, $\boldsymbol{\theta}^1$ is the set of parameters of LSTM1. The LSTM1 designed for change point detection is a two-class classifier with two categories and the optimal parameters $\boldsymbol{\theta}^1$ are determined to minimize the binary cross-entropy in Eq. 5-11. T here is the number of time points in the training dataset. Finally, the change point, denoted as T_{cp}, is defined as the first time when the output $\hat{y}_t$ goes beyond the threshold, as Eq. 5-12 shows.

$$\hat{y}_t = F(\boldsymbol{X}_t, \boldsymbol{\theta}^1) \tag{5-10}$$

$$\mathcal{L}1 = -\frac{1}{T}\sum_t^T y_t \log(\hat{y}_t) + (1 - y_t)\log(1 - \hat{y}_t) \tag{5-11}$$

$$T_{cp}(t) = \inf(t^* \leq t | \hat{y}_{t^*} = F(\boldsymbol{X}_{t^*}, \boldsymbol{\theta}^1) > threshold) \tag{5-12}$$

The threshold should be determined based on the ROC curve and maximize the value of true positive ratio minus false positive ratio.

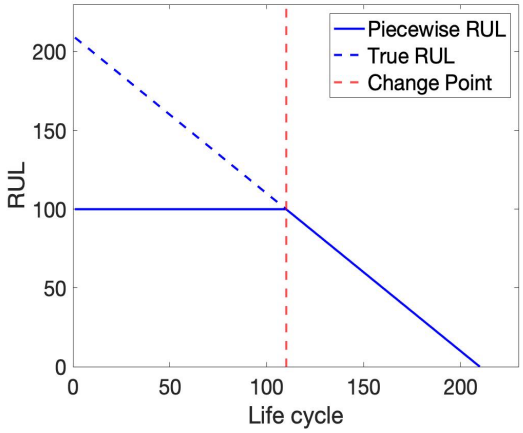


Figure 5-5 Piecewise linear RUL target function

5.3.2 Health Index Prediction

Most existing LSTM-based networks designed for RUL prediction are one-step models, which feed multi-sensor signals into neural networks to predict the RUL directly, without any intermediates. Usually, the true RUL is labeled according to the piecewise linear RUL target function shown in Figure 5-5. The red dash line suggests the position of the change point. As Figure 5-5 shows, the RUL value before the change point is defined as a constant number, implying that the unit has not started to degrade, and the degradation is negligible. After the change point, the unit starts to degrade, and the RUL linearly decreases until it reaches to zero.

However, the position of the change point (red dash line) is assumed to be fixed for different units in most existing models. First, it is unrealistic in most cases, which has been argued in section I. As Figure 5-6 shows, different units may have different change points and different length of life span. And second, the same health condition level may have different RUL values under this RUL target function. In this case, a unified HI construction function should be introduced to accurately indicate the health condition of the unit.

Figure 5-7 shows the HI curves of two units in Figure 5-6 based on the unified HI construction function created in this paper, as Eq. 5-13 shows. In Eq. 5-13, T_f is the total life span of the unit and $T_{cp}(t)$ is the detected change point from previous step. According to Fig 5-7 and Eq. 5-13, the

unified HI is a piecewise decreasing function. Before the change point, the HI always equals to 1, representing the unit is under the normal operation. Once the unit starts to degrade (after the change point), the HI will linearly decrease from 1 to 0 until the unit comes to the failure. With this unified HI function, we can make sure that the same health condition shares the same HI value.

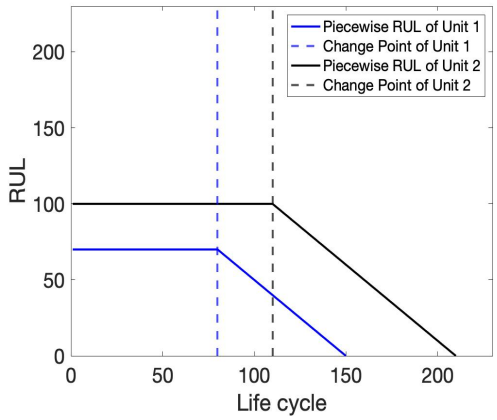


Figure 5-6 Piecewise decreasing RUL function on two units with different change

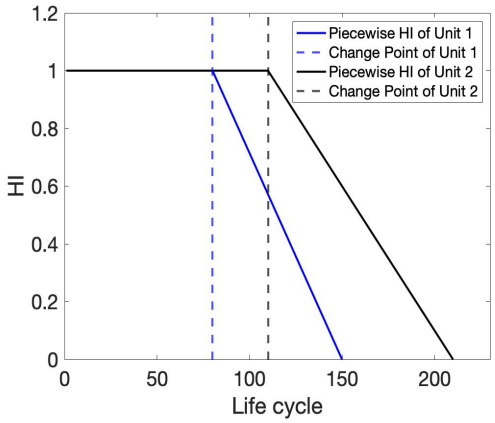


Figure 5-7 Piecewise HI function on two units with different change points and life spans

$$HI(t) = 1 - \frac{max(t - T_{cp}, 0)}{T_f - T_{cp}} \tag{5-13}$$

To predict the $HI(t)$ based on sensor signals collected, we design an LSTM2 to characterize the non-linear relationship between inputs $\boldsymbol{X}_t$ and $HI(t)$, expressed by Eq. 5-14. Here, $\boldsymbol{\theta}^2$ is the set of parameters for LSTM2 and it is optimized by minimizing the customized loss function in Eq.

5-15. The loss function is the squared difference between the predicted health index $\widehat{HI}(t)$ and true health index $HI(t)$ we labeled based on Eq. 5-13.

$$\widehat{HI}(t) = H(\boldsymbol{X}_t, \boldsymbol{\theta}^2) \tag{5-14}$$

$$\mathcal{L}2 = \sqrt{\frac{\sum_t^T \left(\widehat{HI}(t) - HI(t)\right)^2}{T}} \tag{5-15}$$

The overall structure of LSTM2 for HI prediction is shown in Figure 5-8.

5.3.3 RUL Prediction

The last step is to predict the RUL based on $\widehat{HI}(t)$. The RUL is defined as the length from the current time to the failure time, thus it should be calculated as the life span T_f minus the current time t. According to the unified HI function in Eq. 5-13, the HI is a one-to-one function of T_f. Therefore, T_f can be predicted give the predicted $\widehat{HI}(t)$, as Eq. 5-16 shows. $\widehat{T}_f(t)$ here is the predicted T_f.

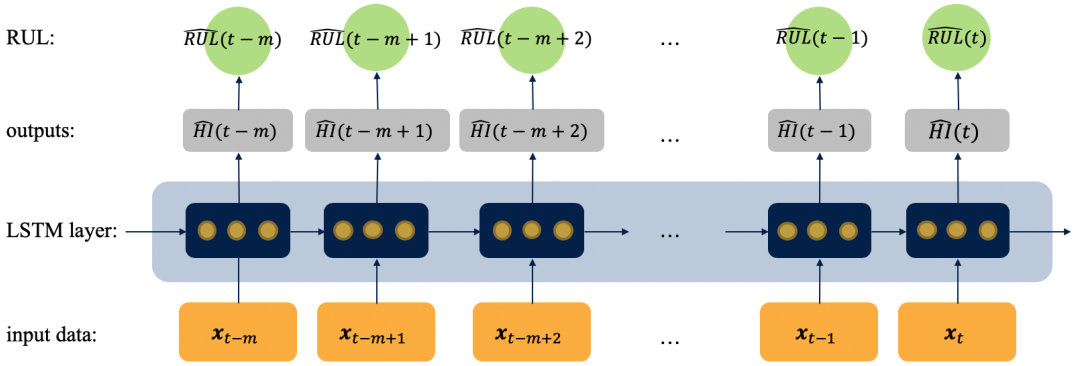


Figure 5-8 Structure of LSTM2 for HI prediction

$$\widehat{T}_f(t) = \frac{t - T_{cp}(t)}{1 - \widehat{HI}(t)} + T_{cp}(t) \tag{5-16}$$

Finally, the predicted $\widehat{RUL}(t)$ can be obtained by subtracting $\widehat{T}_f(t)$ from t.

$$\widehat{RUL}(t) = \widehat{T}_f(t) - t = \frac{\widehat{HI}(t)\left(t - T_{cp}(t)\right)}{1 - \widehat{HI}(t)} \tag{5-17}$$

The detailed algorithm of Dual-LSTM framework for RUL prediction from end to end is summarized in Table 5-1. Adam algorithm is used to train the LSTM network.

Table 5-1 Dual-LSTM framework for RUL prediction

Training:

Initialization:

$iter = 0$, $\boldsymbol{m}_{iter} = 0$, $\boldsymbol{v}_{iter} = 0$

Assign values of α, β_1 , β_2

Randomly initialize $\boldsymbol{\theta}^1_{iter}, \boldsymbol{\theta}^2_{iter}$

While $\boldsymbol{\theta}^{1/2}_{iter}$ not converged:

if change point detection:

gradient $\boldsymbol{g}^1_{iter} = \frac{\partial \mathcal{L}1}{\partial \boldsymbol{\theta}^1_{iter}}$

else if: HI prediction:

gradient $\boldsymbol{g}^2_{iter} = \frac{\partial \mathcal{L}2}{\partial \boldsymbol{\theta}^2_{iter}}$

$$\boldsymbol{m}_{iter+1} = \beta_1 \boldsymbol{m}_{iter} + (\mathbf{1} - \beta_1)\, \boldsymbol{g}^{1/2}_{iter}$$

$$\boldsymbol{v}_{iter+1} = \beta_1 \boldsymbol{v}_{iter} + (\mathbf{1} - \beta_1)\, \boldsymbol{g}^{1/2}_{iter}$$

$$\widehat{\boldsymbol{m}}_{iter+1} = \frac{\boldsymbol{m}_{iter+1}}{1 - \beta_1^{iter+1}}$$

$$\widehat{\boldsymbol{v}}_{iter+1} = \frac{\boldsymbol{v}_{iter+1}}{1 - \beta_2^{iter+1}}$$

$$\boldsymbol{\theta}^{1/2}_{iter+1} = \boldsymbol{\theta}^{1/2}_{iter} - \frac{\alpha \widehat{\boldsymbol{m}}_{iter+1}}{\sqrt{\widehat{\boldsymbol{v}}_{iter+1}} + \varepsilon}$$

$$iter = iter + 1$$

Return $\boldsymbol{\theta}^{1/2}_{iter}$

Testing:

Change Point Detection:

$$\hat{y}_t = F(\boldsymbol{X}_t, \boldsymbol{\theta}^1_{iter})$$

if $\hat{y}_t < 0.5$: break

else:

Return:

$$T_{cp}(t) = \inf(t^* \le t | \hat{y}_{t^*} = F(\boldsymbol{X}_{t^*}, \boldsymbol{\theta}^1) > 0.5)$$

HI prediction:

$$\widehat{HI}(t) = H(\boldsymbol{X}_t, \boldsymbol{\theta}^2_{iter})$$

$$\widehat{RUL}(t) = \frac{\widehat{HI}(t)\big(t - T_{cp}(t)\big)}{1 - \widehat{HI}(t)}$$

Return: $\widehat{\boldsymbol{RUL}}(\boldsymbol{t})$

5.4 Experiments and Results

This section demonstrates the RUL prediction performance of the proposed Dual-LSTM framework, compared with benchmark deep learning models. Performance evaluation is carried out on the aircraft turbofan engine degradation dataset. To assess the consistency of the proposed Dual-LSTM framework performs consistently, we developed 10 Dual-LSTM models with random selection of the training/testing units and random initial weights of LSTM networks. More details on the dataset are provided in Section IV.A. In the training stage, the LSTM1 and LSTM2 networks of the Dual-LSTM framework are learned on the training data set. Then, the trained Dual-LSTM framework is applied to the testing data set to evaluate the performances. Three metrics, which are the scoring function (SF), RMSE, and relative prediction error, are used for evaluation.

Table 5-2 Detailed description of sensors

#	symbol	description	units
1	T2	Total temperature at fan inlet	°R
2	T24	Total temperature at LPC inlet	°R
3	T30	Total temperature at HPC inlet	°R
4	T50	Total temperature at LPT inlet	°R
5	P2	Pressure at fan inlet	psia
6	P15	Total pressure in bypass-duct	psia
7	P30	Total pressure at HPC outlet	psia
8	Nf	Physical fan speed	rpm
9	Nc	Physical core speed	rpm
10	Epr	Engine pressure ratio	-
11	Ps30	Static pressure at HPC outlet	psia
12	Phi	Ratio of fuel flow to Ps30	pps
13	NRf	Corrected fan speed	rpm
14	NRc	Corrected core speed	rpm
15	BPR	Bypass ratio	-
16	farB	Burner fuel-air ratio	-
17	htBleed	Bleed enthalpy	-
18	NF_dmd	Demanded fan speed	rpm
19	PCNR_dmd	Demanded corrected fan speed	rpm
20	W31	HPT coolant bleed	lbm/s
21	W32	LPT coolant bleed	lbm/s

5.4.1 Dataset Overview

The turbofan engine degradation dataset is simulated by the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) tool developed by NASA, which is widely used in the field of PHM [139], [140]. In the simulation, the engine model runs under a variety of operating conditions with different altitude levels, Mach numbers, and throttle resolver angles (TRA), as well as two fault modes, high-pressure compressor (HPC) degradation, and fan degradation. In our experiment, both settings of FD001 and FD003 with the single operation condition are investigated. FD001 has one failure mode and FD003 has two failure modes. In the dataset, 21 sensor signals are collected to monitor the health conditions of the turbofan engines, which randomly start from a normal operation level to the failure. Table 5-2 gives a detailed description of the 21 sensors.

Both FD001 and FD003 contain a training set and a testing set, and each set consists of 100 engine units. In the training set, the engines run until failure, whereas the engines in the testing set stop at some random time point before failure. For performance evaluation purposes, only the training set data is used. Among the training set data, 90 engines are randomly chosen for the training purpose, whereas the remaining 10 engines are used for testing.

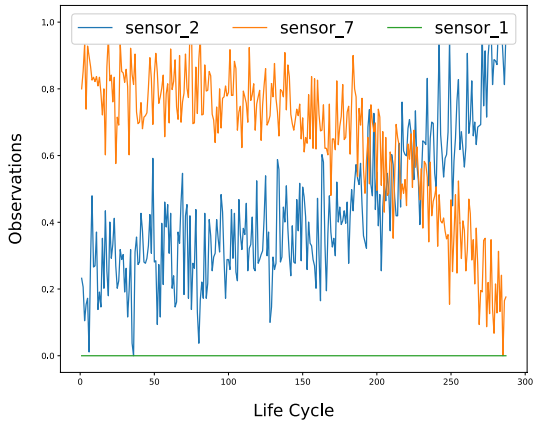


Figure 5-9 Normalized sensor measurements for a random engine

5.4.2 Data Preprocessing

Figure 5-9 shows the normalized measurements from three sensors for a random engine. It is worth noting that different sensor signals show different trends. For example, sensor 2 and sensor 7 reveal increasing and decreasing trends respectively, while sensor 1 doesn't show significant

variations on values. Based on a fundamental assumption of most existing prognostic models [141], [142] , there should be inherent monotonic features of the degradation process. Therefore, in order to filtrate useful information related to degradation process, only sensor signals with monotonically increasing or decreasing trends should be selected. Table 3 records the trend information of each sensor. According to Table 5-3, we decide to remove sensor 1, 5, 6, 10, 14, 16, 18, and 19 without any monotonic features inside. 14 sensors with monotonic increasing/decreasing trends are kept for further analysis. Second, the Min-Max rescaling method is used to normalize the data to be in the range of 0 to 1. Third, in order to eliminate the effect of noise, the Savitzky-Golay filter with frame length 51 and polynomial order 3 is used to smooth the sensor signals.

Table 5-3 Sensor trends summary

Trend type	Sensor number
Increasing	2, 3, 4, 8, 9, 11, 13, 15, 17
Decreasing	7, 12, 20, 21
Irregular/ unchanged	1, 5, 6, 10, 14, 16, 18, 19

As discussed in the proposed framework, LSTM1 is trained to predict the change point of degradation, which requires pre-labeled true change points for training. However, such labeled true change points are not available for this case-study. Therefore, to label the true change points of the training engines, we first assume that there are k health states in total from the normal operation to failure during the degradation. Then, we use a binary segmentation method [143], [144] to find out the time when the unit transfers from current state to the next state. The best value

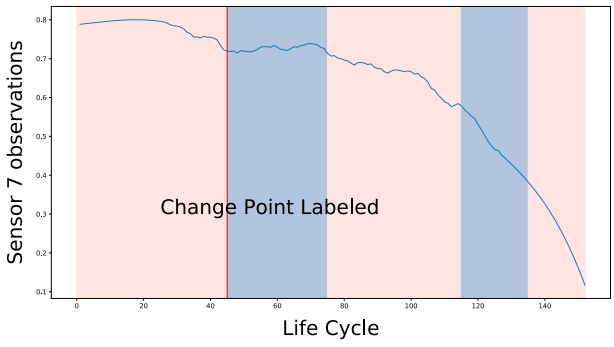


Figure 5-10 Health state segmentation of engine 100 from FD001 (k=5)

of k ($k = 5$) is determined by the information-theoretic approach. Figure 5-10 shows the health state segmentation result on engine 100 of the FD001 dataset. The blue curve is the observations of sensor 7 over time. The pink and light blue shadows represent different health state in order.

Those time points which are grouped into the first health state is under the normal operation. And the time point transforming from first health state to the second is defined as the true change point, as the red solid line in Figure 5-10 shows.

5.4.3 Training State

There are two steps in the training stage. The first step is to learn the parameters of the LSTM1 designed for change point detection. The 14 sensor signals selected as inputs are fed into the LSTM1 network. The parameters of LSTM1 are learned to minimize the cross-entropy loss function in Eq. 5-11. Finally, the determined LSTM1 is a three-layer neural network: one input layer (multi-sensor data), one LSTM layer with 32 hidden neurons and 20 look-back time-steps, and one output layer (0 or 1). For training, Adam optimizer is used with a learning rate of 0.001 and a batch size of 100 for 100 epochs.

The second step is to train the LSTM2 for modeling the HI as a function of the 14 sensor signals and the HI is given as the output. The determined LSTM2 consists of four layers: one input layer (multi-sensor data), two hidden LSTM layers with 32 neurons and 20 look-back time-steps, and one output layer (HI). For training, Adam optimizer is used with a learning rate of 0.0001 and a batch size of 100 for 80 epochs.

5.4.4 Performance Evaluation

Three metrics are used to evaluate the RUL prediction performance of the proposed Dual-LSTM framework. They are scoring function (SF), RMSE, and relative prediction error. The formulas of these three metrics are given in Eq. 5-18 - 5-20. N is the number of testing units.

SF is defined by the data creator [108] and commonly used to evaluate the performance of RUL prediction in many existing papers [48], [61], [69], [129], [130], [137]. It is an asymmetric function, which prefers early prediction (predicted RUL is less than true RUL) rather than late prediction error (predicted RUL is larger than true RUL).

$$SF_t = \begin{cases} \dfrac{\sum_i^N e^{\frac{d_{it}}{13}} - 1}{N}, \text{where } d_{it} = \widehat{RUL}_{it} - RUL_{it} < 0 \\ \dfrac{\sum_i^N e^{\frac{d_{it}}{10}} - 1}{N}, \qquad \text{otherwise} \end{cases} \tag{5-18}$$

RMSE is also a common evaluation metric for RUL prediction performance. However, different from SF, RMSE gives equal penalties to early and late prediction errors.

$$RMSE_t = \sqrt{\frac{1}{N}\sum_i^N {d_{it}}^2} \tag{5-19}$$

The relative prediction error is defined as the absolute difference between true and predicted RUL values divided by the total lifespan.

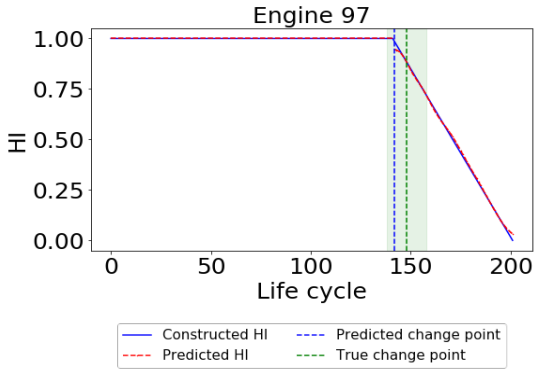


Figure 5-11 Prediction results on engine 97 from FD001

$$err_t = \frac{1}{N}\sum_i^N \frac{|d_{it}|}{T_{f_i}} \tag{5-20}$$

According to the definitions of three metrics shown above, the best model for RUL prediction is the one that results in the smallest SF, RMSE, and relative prediction error on the testing units.

Figures 5-11 and 5-12 visualize the experiment results on testing units of 97 from FD001 and 33 from FD003 respectively. The green dash line suggests the true change point and the blue dash

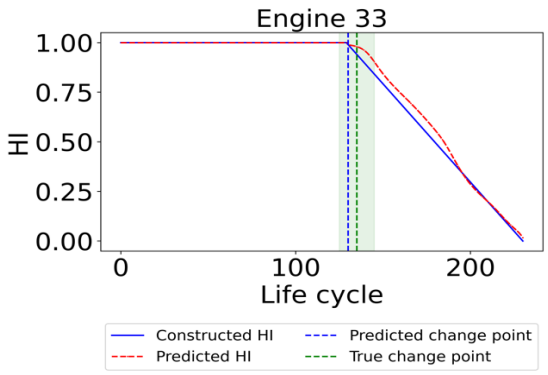


Figure 5-12 Prediction results on engine 33 from FD003

line represents the predicted change point. The green shadow reflects a time buffer of +/- 10 cycles, which allows other acceptable change points. Because it is possible to mislabel the true change point by the sensor fusion framework. Figures 5-11 and 5-12 clearly show that both predicted change points are inside the green shadow, which means the trained LSTM1 successfully detects the change point.

Besides, the blue solid line is the HI constructed based on the change point detected. The red dash line is the predicted HI (the predicted HI before the change point is automatically assigned as 1). From the figures, we can see that the predicted HI values are close to the constructed HI beyond the change point, which proves that the trained LSTM2 performs well in HI prediction. Finally, the predicted HI is utilized to calculate the RUL using the function in Eq. 5-17.

To further explain the effectiveness of the proposed Dual-LSTM framework for RUL prediction, we compare it with two benchmark deep learning models. The first is an RNN with one hidden layer that consists of 32 neurons and one look-back time -step. The second is an LSTM network with one hidden layer that consists of 32 hidden neurons and 20 look-back time-steps. For the two benchmark models, the change point is assumed to be fixed (we adopt the same approach as [29] to set the fixed change point at the 130th cycle). Both neural networks are trained by Adam optimizer with a batch size of 100 for 100 epochs.

Figures 5-13, 14 and 15 present the comparison results on FD001 dataset based on three evaluation metrics mentioned above at different monitoring points with different remaining cycles to failure. The comparison results on FD003 dataset are summarized in Table 5-4. The results both in Figures 5-13, 14 and 15 as well as Table 4 are the average of the 10 Dual-LSTM networks. The comparison results show that:

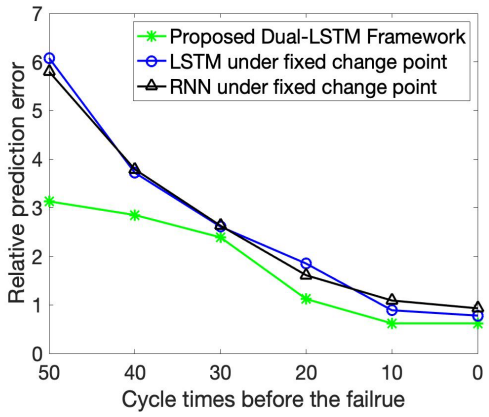


Figure 5-13 Relative prediction error on three neural networks

i) The proposed Dual-LSTM framework achieved the best RUL prediction performances (lowest SF, RMSE, and relative prediction errors) at all monitoring points. It means that the proposed Dual-LSTM framework can effectively improve the RUL prediction performance. The accurate RUL predictions of the Dual-LSTM at the earliest monitoring time (50 cycle times before the failure) provide confidence of raising early failure alarm and scheduling maintenance.

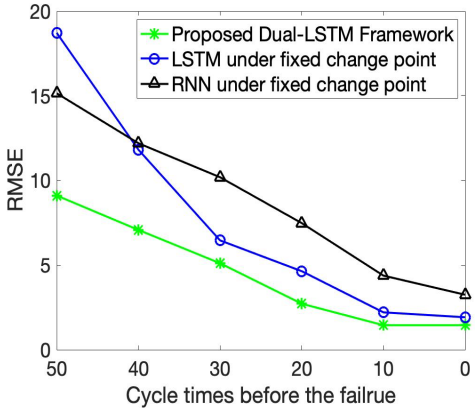


Figure 5-14 RMSE on three neural networks

ii) Particularly, the SF from the Dual-LSTM framework is an order magnitude smaller than that of the benchmark models. Unlike the RMSE relative prediction errors, the Dual-LSTM improvement in SF is much more evident. This may be caused by the change point detection within the framework, which helps to correct the RUL labeling errors (especially the late errors) induced by the fixed change point assumption.

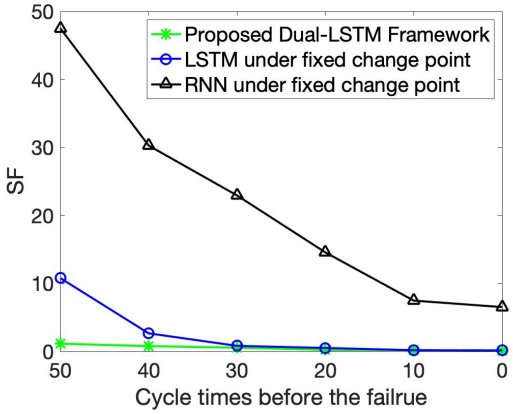


Figure 5-15 SF on three neural networks

iii) Both the proposed Dual-LSTM and the benchmark networks show a descending trend for three evaluation metrics. Therefore, more accurate results are expected as the unit approaches failure. This is a natural consequence because more data about the unit is collected as it approaches failure and that supports more accurate predictions.

Table 5-4 Comparison results on FD003

Cycles before failure	SF			RMSE			Relative prediction error		
	1*	2*	3*	1*	2*	3*	1*	2*	3*
50	1.86	1.25	1.1	7.85	7.05	8.19	3.14	3.55	3.4
40	0.98	1.88	0.95	7.61	11.55	7.09	3.1	3.82	2.54
30	0.71	0.86	0.62	6.56	8.96	5.27	2.66	2.9	2.03
20	0.52	0.61	0.49	4.92	5.28	5.43	1.86	2.12	1.88
10	0.23	0.49	0.2	2.61	4.35	2.53	0.93	1.81	0.87
0	0.51	0.4	0.1	4.16	3.63	1.38	1.9	1.49	0.44

1*: RNN under fixed change point
2*: LSTM under fixed change point
3*: Proposed Dual-LSTM framework

In summary, the first comparison study shows that the Dual-LSTM outperformed the standard RNN and LSTM neural networks at all monitoring times using three common evaluation metrics.

Figure 5-16 shows the boxplot of relative prediction errors on different monitoring points over 10 experiments on FD001. For each Dual-LSTM model, the training/testing units and the initial parameters were randomly selected. The boxplot shows small variations in the relative prediction errors between the 10 Dual-LSTM models, which validates the consistency of the proposed Dual-LSTM framework for RUL predictions. Furthermore, the boxplot shows that relative predictions errors.

The Dual-LSTM framework is also compared to two recently published LSTM-based models for RUL prediction that are applied to the same degradation dataset. The first benchmark model is the vanilla LSTM network introduced in [48]. The authors in [48] designed the vanilla LSTM network to predict the RUL of turbofan engines and also supported it by the SVM as the change point detector.

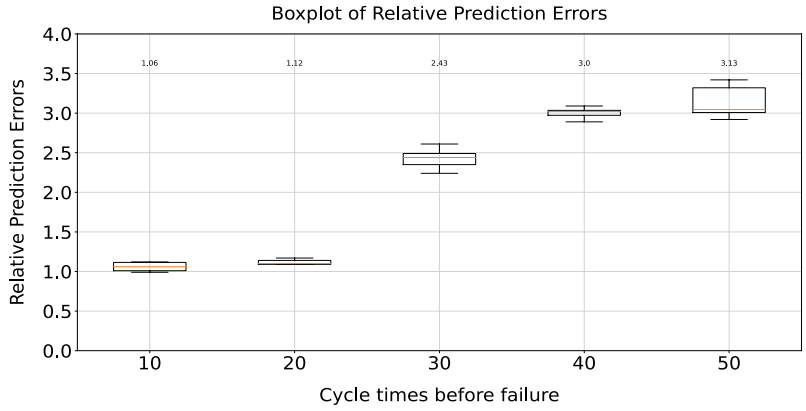


Figure 5-16 Prediction errors boxplot on monitoring points for 10 experiments on FD001

The comparison results both on FD001 and FD003 are summarized in Table 5-5. The standard RNN and GRU are additional benchmark models discussed in [48]. P_5, P_{10}, and P_{20} in Table 5-5 are percentages of samples in the testing set with relative prediction errors less than or equal to 5%, 10%, and 20% respectively. In this case, the larger the P_5, P_{10}, and P_{20}, the better the prediction performance. From Table 5-5, we can see that the Dual-LSTM framework has the largest P_5, P_{10}, and P_{20}, as well as the smallest SF at all monitoring points both FD001 and FD003. This suggests that the Dual-LSTM framework outperforms the vanilla LSTM network and other two benchmark models in [48], even based on their predefined evaluation metrics.

The Dual-LSTM framework is also compared to the dual-task LSTM networks proposed by [62]. The dual-task LSTM uses LSTM to predict the RUL and classify the degradation states simultaneously. The comparison results are summarized in Table 5-6. The authors of [62] compared the results of three different cell structures LSTM, GRU and peephole. GRU and peephole are variants of LSTM with different gate structures and activation functions. The evaluation metric RUL_n in Table 5-6 is the RMSE of the last n cycles before the failure.

$$RUL\_n = \sqrt{\frac{1}{n}\sum_{t=T_f-n+1}^{T_f}(\widehat{RUL}_t - RUL_t)^2} \tag{5-21}$$

Table 5-5 Comparison results with LSTM-based models for RUL prediction in [48]

Dataset setting	Models	50 cycles left				10 cycles left				5 cycles left			
		P_5(%)	P_{10}(%)	P_{20}(%)	SF	P_5(%)	P_{10}(%)	P_{20}(%)	SF	P_5(%)	P_{10}(%)	P_{20}(%)	SF
FD001	Standard RNN	10	35	60	5581.41	75	90	95	256.34	85	95	95	118.52
	GRU	45	85	85	17.96	100	100	100	0.35	100	100	100	0.53
	Vanilla LSTM	30	55	70	37.51	100	100	100	0.43	100	100	100	0.30
	Dual-LSTM	40	70	100	1.15	100	100	100	0.22	100	100	100	0.12
FD003	Standard RNN	35	60	85	48.96	100	100	100	256.34	95	100	100	118.52
	GRU	40	65	95	14.29	90	100	100	0.35	85	100	100	0.53
	Vanilla LSTM	50	70	90	30.58	100	100	100	0.43	100	100	100	0.30
	Dual-LSTM	60	90	100	4.05	100	100	100	0.29	100	100	100	0.24

Table 5-6 clearly shows that the peephole outperforms other two models with lower RMSE at different levels of RUL on FD001; however, it still underperforms the proposed Dual-LSTM at all the tested levels of RUL. Specifically, the proposed Dual-LSTM framework showed a lowest RMSE of 1.11 before 50 cycles of the failure, which manifests its effectiveness for raising early failure alarms and optimizing condition-based maintenance strategies. Besides, for FD003, the proposed Dual-LSTM framework also performs better than the peephole model according to the results provide by [62]. One major reason behind the outperforming performance of the Dual-LSTM is the robustness of its change point detection.

Table 5-6 RMSE comparison with three benchmark models in [62]

Cell	FD001				FD003	
	LSTM	GRU	peephole	Dual-LSTM	peephole	Dual-LSTM
RUL_50	7.03	6.37	3.58	1.11	5.23	3.30
RUL_20	5.01	6.09	3.16	0.86	5.87	1.01
RUL_10	4.17	4.86	3.37	1.06	5.02	0.76
RUL_5	2.32	2.45	2.38	1.16	2.45	0.93

In summary, the proposed Dual-LSTM is compared to five benchmark methods using multiple evaluation metrics. All the comparative results show that the Dual-LSTM outperforms existing benchmark models for RUL predictions.

5.5 Conclusion and Discussion

In this paper, a Dual-LSTM framework combining change point detection and RUL prediction is proposed to achieve the high-precision RUL prediction. With this proposed Dual-LSTM framework, we can detect the uncertain change point of each unit specifically and mainly focus on the degradation process beyond the change point to achieve more accurate RUL predictions.

The performance of the proposed Dual-LSTM framework is compared with the RNN and LSTM networks with fixed change points. The RUL performance comparisons demonstrate that the proposed Dual-LSTM framework outperforms the benchmark models. Additionally, the proposed Dual-LSTM framework also shows outperforming performances compared with LSTM-based

models in [48] and [62] that do not assume fixed change points. Besides, the outstanding RUL performance achieved at the early monitoring points demonstrates its ability to raise early failure alarms and provides strong support for optimizing condition-based maintenance strategies.

Future research studies must be directed to investigate advanced physics-based loss functions and develop other variants of LSTM networks for simultaneous change point detection and RUL prediction.

Chapter 6 A Long Short-Term Memory Network for Online State-of-Charge Estimation of Li-ion Battery Cells

This chapter proposes a new long short-term memory neural network model to estimate the state-of-charge of lithium-ion (Li-ion) battery cells. The proposed model improves the estimation accuracy by accounting for the changes in the battery parameters due to ageing by utilizing relevant knowledge from previous cycles when estimating the current state-of-charge. Derivation and details of the proposed model followed by experimental verification using commercial Li-ion battery cells are provided.

6.1 Introduction

Due to the unique characteristics such as high energy/power densities and prolonged service-life, Li-ion batteries have been widely used in numerous applications such as electric vehicles (EVs), consumer electronics, aircraft, etc. [145], [146]. These applications require a highly accurate estimation capability of the amount of remaining energy in the battery [147].

To identify the battery residual energy, the SOC is introduced. The SOC reflects the remaining battery capacity during the charge/discharge cycles as a percentage of its nominal capacity. Traditional techniques for SOC estimation are filter-based, like Kalman filter (KF) [148], extended Kalman filter (EKF) [149], etc. Most of these filter-based models are off-line models [150]. That is, the model parameters given by archived data are fixed. However, the parameters often affected by environmental factors such as temperature, battery ageing and so on [151]. Therefore, it is necessary to build on-line models to estimate SOC in real-time. In recent years and as a result of the rapid development in computing science, there have been some researchers using DL techniques, like feedforward neural networks [150], RNN [152], LSTM [145], [147], [153], to estimate SOC of batteries. This is because these DL techniques are highly efficient in processing large-scale data, which enables online updating model parameters based on real-time data of battery measurements.

Although existing DL techniques can estimate SOC of batteries in real-time, most of the existing neural networks ignore the impact of previous cycles on the SOC estimation. As mentioned, the model parameters are usually affected by battery ageing. In this case, the measurements related to the battery ageing should also be considered to improve the accuracy of SOC estimation. In this paper, an LSTM network that leverages information from previous cycles is proposed to achieve on-line SOC estimation with high precision by considering variation in the battery internal parameters due to ageing.

6.2. Proposed LSTM for On-line SOC Estimation

6.2.1 LSTM Cell Structure

LSTM is a variant of RNN. Like RNN, LSTM can pass memories from previous time steps to the following ones. However, compared with RNN, LSTM is not only able to keep short-term memories, but also maintain long-term memories of time series. Therefore, LSTM becomes a popular technique for time series analysis in recent years. This advantage attributes to the special gates inside the LSTM cell, shown in Figure 6-1.

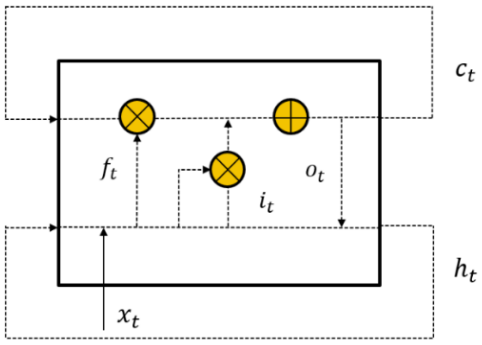


Figure 6-1 LSTM cell structure

We can see three gates inside the LSTM cell, which are forget gate f_t, input gate i_t, and output gate o_t. Each gate is an activation function (usually sigmoid function), as (6-1) to (6-3) show. Here, x_t is the input at time t. h_{t-1} is the hidden state from the previous time step. $W_{f,i,o}$, $V_{f,i,o}$, and $b_{f,i,o}$ are model parameters need to be estimated. These three gates together control the information flow within the LSTM network:

$$f_t = \sigma(W_f h_{t-1} + V_f x_t + b_f) \quad (6\text{-}1)$$

$$i_t = \sigma(W_i h_{t-1} + V_i x_t + b_i) \quad (6\text{-}2)$$

$$o_t = \sigma(W_o h_{t-1} + V_o x_t + b_o) \quad (6\text{-}3)$$

In Figure 6-1, c_t is the cell state at time t which stores prior memories. c_t is supposed to be updated at each time step based on previous cell state c_{t-1} and candidate cell state $\widetilde{c_t}$, shown in (6-4). $\widetilde{c_t}$ is calculated based on (6-5). From (6-4), we can see that how c_t is updated depends on the forget gate f_t and input gate i_t. f_t decides which value of c_{t-1} should be added into c_t, and i_t decides which value of $\widetilde{c_t}$ should be added into c_t. Furthermore, the output gate o_t decides which value of c_t should be outputted through the hidden state h_t, as (6-6) shows. Finally, with h_t, LSTM returns the output y_t at each time step based on (6-7). Therefore, through the cell state and three gates, the memories can be stored for a long period.

$$c_t = f_t \circ c_{t-1} + i_t \circ \widetilde{c_t} \quad (6\text{-}4)$$

$$\widetilde{c_t} = tanh(W_a h_{t-1} + V_a x_t + b_a) \quad (6\text{-}5)$$

$$h_t = o_t \circ tanh(c_t) \quad (6\text{-}6)$$

$$y_t = \sigma(W_y h_t + b_y) \quad (6\text{-}7)$$

6.2.2 Proposed LSTM for Online SOC Estimation

Figure 6-2 shows the LSTM network proposed in this paper for SOC estimation. There are 6 measurements selected as input to feed the LSTM network at each time step. The 6 measurements are cycle number c, time in hours at time t, the current in Amperes at time t of cycle c, I_t^c, the voltage in Volts at time t of cycle c, V_t^c, the battery temperature in ^oC at time t of cycle c, T_t^c , and the true SOC at the same time t from the previous cycle $c-1$, SOC_t^{c-1}. Among these measurements, the c, t, and SOC_t^{c-1} are battery ageing-related measurements. Based on the input data, LSTM estimates the SOC at each time step $\widehat{SOC}_t^c$. Here, m is the number of time steps we look back when estimating the SOC at the current time. In other words, the current SOC not only depends on the current measurements but also depends on the measurements from previous m time steps. For training purposes, the loss function is set to be the common root mean squared error

(RMSE) function, shown in (6-8). T is the time length of each cycle, and C is the number of cycles in the training set:

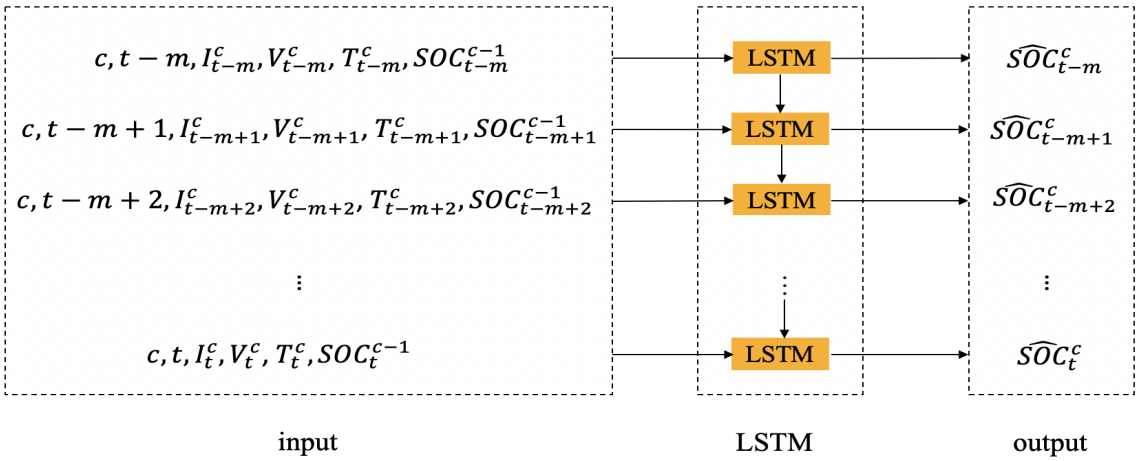


Figure 6-2 LSTM network for online SOC estimation

$$\mathcal{L} = \sqrt{\frac{\sum_c^C \sum_t^T (\widehat{SOC}_t^c - SOC_t^c)^2}{CT}} \tag{6-8}$$

6.3 Experiment

6.3.1 Data Description

The proposed LSTM network is evaluated on the dataset of 18650 Lithium-ion batteries provided by NASA Ames. In this experiment, only battery B0005 is studied. Here, we only focus on estimating the SOC during the discharging process. For training purposes, the min-max method is used for normalization. Besides, the records that happened after the true SOC reaches zero are removed from the dataset.

6.3.2 Training Stage

In our experiment, the first 100 cycles are used for training. Cycle 101 to 110 are used for testing. The model parameters of the weight and bias matrices in (6-1) - (6-3) are learned to minimize the RMSE loss function in (6-8) Different combinations of hyperparameters are tested to achieve the best performance in the experiment. Finally, the hyperparameters are set as the number of time steps looked back is 5, the number of hidden neurons is 5, and the learning rate is 0.001. Batch

size equals to one. Adam optimizer is used with 500 epochs. The training process needs an average of fewer than 10 minutes on the Tesla K80 GPU.

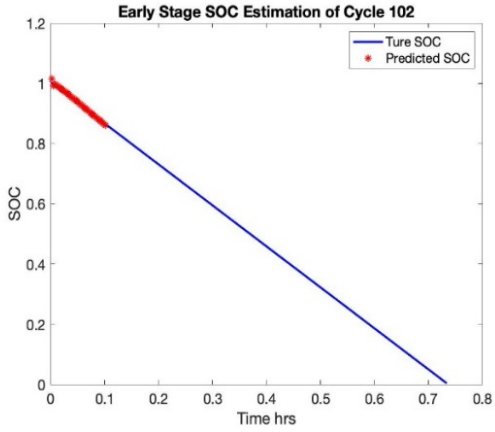


Figure 6-3 Early stage SOC estimation of cycle 102

6.3.3 Experimental Results

Figure 6-3 and 6-4 shows the SOC estimation results at different discharging stages on different testing cycles. The early stage in Figure 6-3 means that time t is about 15% percentile of the total lifetime. The late stage in Figure 6-4 refers to time t which is about 85% percentile of the total lifetime. The blue solid line in Figure 6-3 and 6-4 is the true SOC. And the red dots are estimated SOC values at each time t. From above figures, we can see that the estimated SOC highly coincidences with the true SOC no matter at the early or late stage.

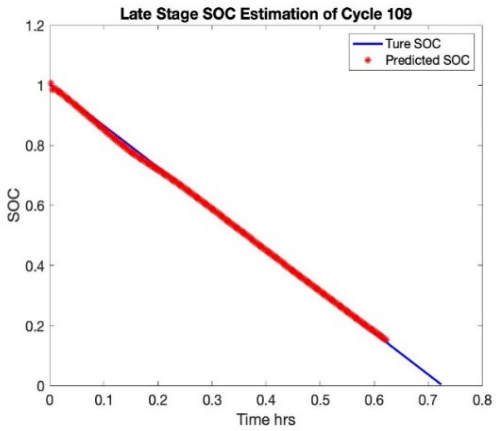


Figure 6-4 Late stage SOC estimation of cycle 109

To evaluate the experiment results quantitatively, mean absolute error (MAE) and max MAE are used. Figure 6-5 presents the MAE on each testing cycle from 101 to 110. The maximum MAE occurs at cycle 107, which is 0.98%. The minimum MAE occurs at cycle 103, which is 0.51%.

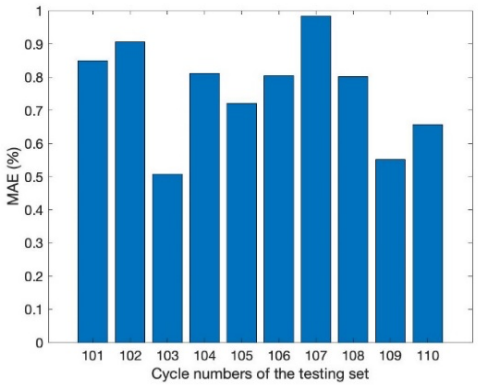


Figure 6-5 MAE (%) of testing units

The average MAE on all testing cycles is 0.76%. Figure 6-6 shows the max error of each testing cycle. Here, we can see that cycle 104 owns the highest max error (2.28%) and cycle 101 owns the lowest max MAE (1.72%).

Besides, the proposed LSTM in this paper is also compared with other benchmark models designed for SOC estimation, which are classic LSTM, RNN, and multilayer perceptron (MLP) networks without considering battery ageing-related measurements. The MAE (%) of these four models at different discharge stages are shown in Table 6-1. The early stage, middle stage, late stage, and end in Table 6-1 refer to time t, which are 15%, 50%, 85%, and 100% percentile of the

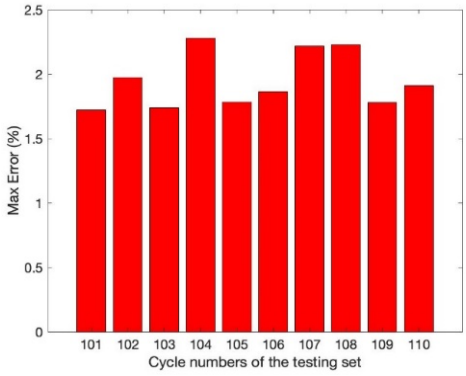


Figure 6-6 Max error (%) on testing units

total lifetime respectively. To show the comparison results intuitively, Figure 6-7 also visualizes Table 6-1 with the histogram.

From Table 6-1 and Figure 6-7, we can see that the proposed LSTM network owns the smallest MAE values at all stages. Furthermore, the MAE results of all the four models show a decreasing trend as more data is available. In summary, the proposed LSTM greatly improves the accuracy of SOC estimation at different stages of battery cell life by considering ageing-related measurements.

Table 6-1 MAE (%) comparison results with benchmark models

Stages / Models	Early Stage (15%)	Middle Stage (50%)	Late Stage (85%)	End (100%)
LSTM	5.02	5.28	2.67	0.91
RNN	6.93	7.03	2.34	1.64
MLP	8.49	6.69	3.73	0.71
LSTM considering ageing-related measurements	0.59	0.74	0.65	0.45

6.4 Conclusion

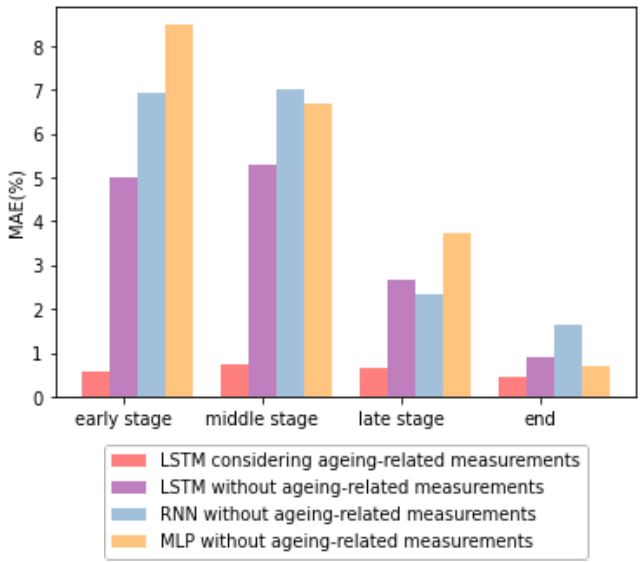


Figure 6-7 Comparison results with benchmarks based on MAE (%)

By taking the battery ageing-related and historical measurements into consideration, the proposed LSTM network is expected to achieve high precision SOC estimation performance. The experiment results validate the high accuracy of the proposed LSTM. For future studies, it is of

high-interest to couple this model with existing remaining useful life techniques [28], [35], [37], [106], [154].

Chapter 7 AutoML Website-based Application Development

7.1 Introduction

The ML workflow is typically composed of five steps, which are problem defining, data preparation, model designing, training and deploying. To complete an ML project is time-consuming and requires amount of expertise. Therefore, it is hard to conduct data analysis with ML technologies for non-professional users. To make ML workflow easier and more convenient, we developed an integrated and accessible website-based application, called AutoML, which allows users, especially non-professional users, to solve industrial problems based on data analysis with much less efforts.

AutoML follows the ML workflow step by step and provides various built-in ML algorithms and frameworks in a single toolset, such as decision tree, SVM, random forest, KNN, deep neural network, and so forth. Those built-in algorithms can be utilized in different types of applications, like RUL prediction, feature-based classification, image-based classification, etc.

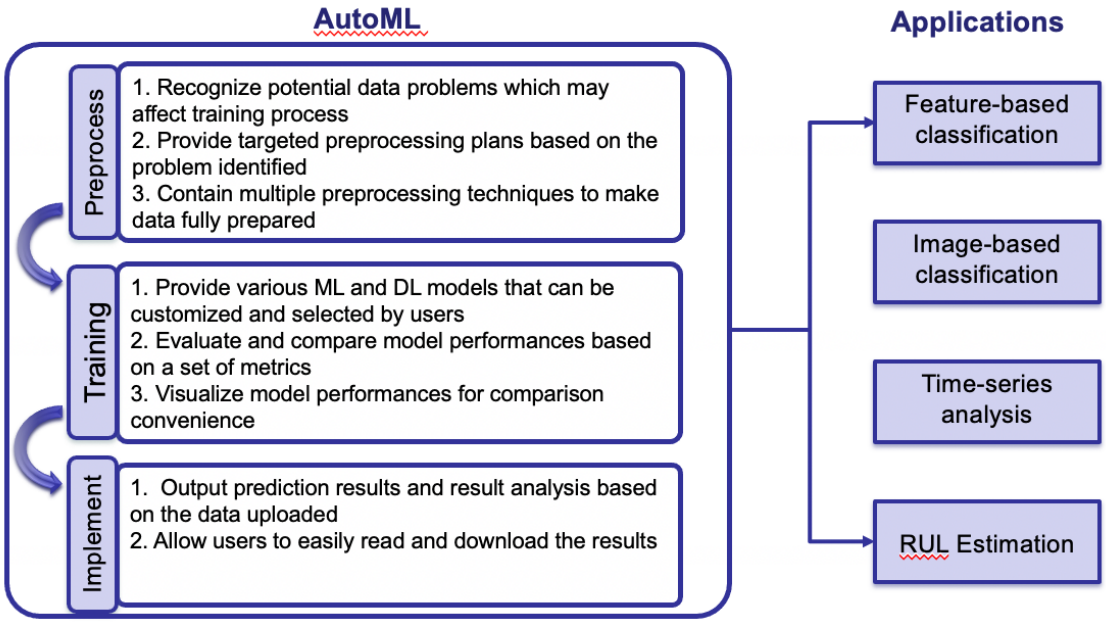


Figure 7-1 Structure of AutoML

7.2 AutoML Structure and Functionalities

There are there three parts involved within the AutoML application, which corresponds to three major steps in ML workflow. They are preprocessing part, training part and implementation part. Figure 7-1 shows the overall structure of the AutoML.

Preprocessing part provides several functionalities, such as feature filtering, handling missing values, standardization, correlation analysis, dimension reduction, etc., which enables users to prepare the dataset for further training.

Training part allows users to train different customized ML/DL models with the preprocessed dataset. In this part, users can choose target model from the pool and customize the model hyperparameters for training. The performances of selected models on training dataset are also provided and visualized in this part. Based on the performance, users can download the model which meets requirements.

By uploading the dataset and well-trained models, implementation part will output the prediction results and detailed explanation (e.g., probabilities) about the prediction results, which can be easily downloaded by a one-click button.

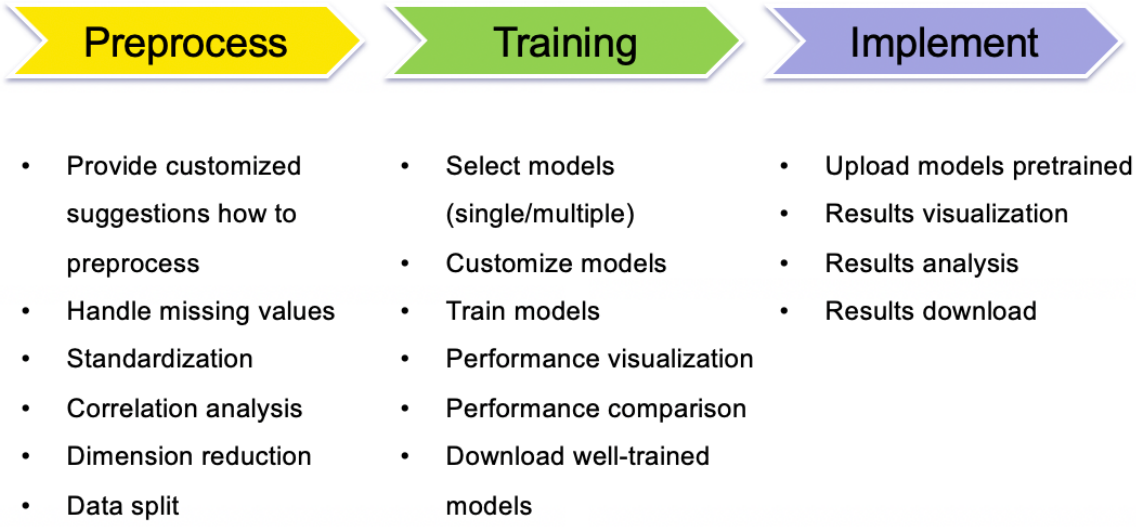


Figure 7-2 Workflow and functionalities of AutoML

7.3 Case Study

In this subsection, we will give a case study to introduce how AutoML works. The dataset we used for case study is downloaded from Kaggle: https://www.kaggle.com/divyansh22/crystal-system-properties-for-liion-batteries. This dataset contains 339 observations about the physical and chemical properties of the lithium-ion silicate cathodes. The detailed information about these properties is given in Table 7-1.

Table 7-1 Detailed description of dataset

Property Name	Description	Unit
Formation Energy	Formation energy	eV
E Above Hull	Energy if decomposition of material into most stable ones	eV
Band Gap	Band gap	eV
Nsites	Number of atoms in the unit cell of the crystal	\
Density	The density of bulk crystalline materials	gm/cc
Volume	The unit cell volume of the material	\
Has Band structure	Boolean variable for band structure	\

Based on the physical and chemical properties provided by the dataset, we can classify the lithium-ion batteries into three categories on the basis of their crystal system, which are monoclinic, orthorhombic and triclinic. Therefore, the objective of this case study is to develop an accurate classifier to recognize the type of the battery crystal system.

7.3.1 Data Preprocessing

At first, we upload the data, then AutoML creates a report about how to preprocess the dataset, as Figure 7-3 shows. The report is customized and highly dependent on the dataset the user upload. In other words, AutoML offers different suggestions in the report based on different dataset. In our case study, three suggestions are given, as Figure 7-3 shows. 1) Standardization, because AutoML found large differences between feature ranges. For example, the formation energy ranges from -2.98 to -2.01, while volume ranges from 122.58 to 1518.85; 2) Handling missing values; 3) Select features, because AutoML found there are highly correlated features found in the dataset. Users can follow this report to preprocess their dataset.

In addition, five techniques are provided for data preprocessing, which are handling missing values, standardization, correlation analysis, dimension reduction, and split data. Users can choose the technique by clicking on the corresponding component of the navbars, as Figure 7-3 shows.

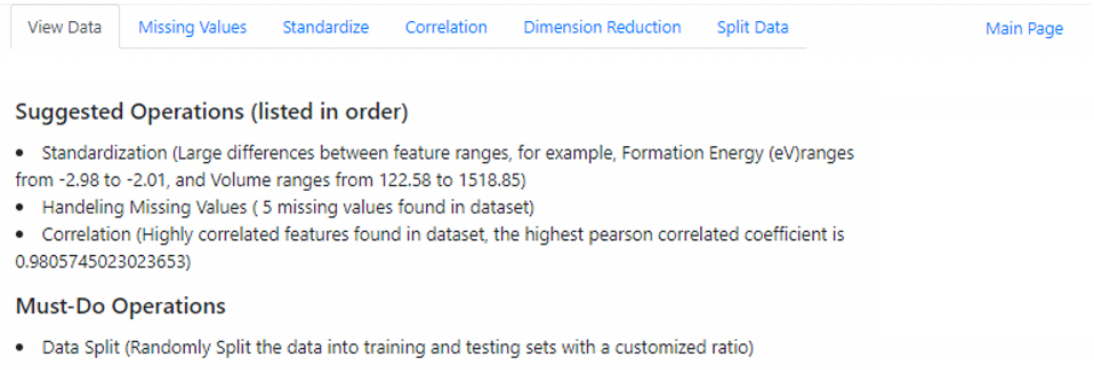


Figure 7-3 Data preprocessing report offered by AutoML

7.3.2 Model Training

Next, we go further to train the classification models with the dataset we have preprocessed. Figure 7-4 presents the training page of AutoML, offering 5 major functionalities in general, which are data uploading, data viewing, model customizing, model training, and results visualization.

As Figure 7-4 shows, there are three buttons designed in this page, named as "*View Data*", "*Customize Models*", and "*Train Models*" respectively. To view and check the dataset, users can click on the "*View Data*" button, and the data uploaded will be presented at the bottom of the page, under the "*Training Data*". To customize the model, users can click on the " *Customize Models* " and jump to another new page, as Figure 7-5 shows. This new page provides different kinds of models and posts the corresponding default hyperparameter values, which can be changed by users. Besides, users can also add or delete any model for training. After customizing the models, users come back to the main page to train those models by clicking on the " *Train Models* " button. Once the training process is finished, the results will be summarized and presented in the middle of the page under the "Learned Models". Here, the training page offers 6 metrics for comparison, which are accuracy for training set ("ACC FOR TRAIN"), accuracy for validation set ("ACC FOR VAL"),

cross entropy for training set, cross entropy for validation set, training time, and prediction time/instance.

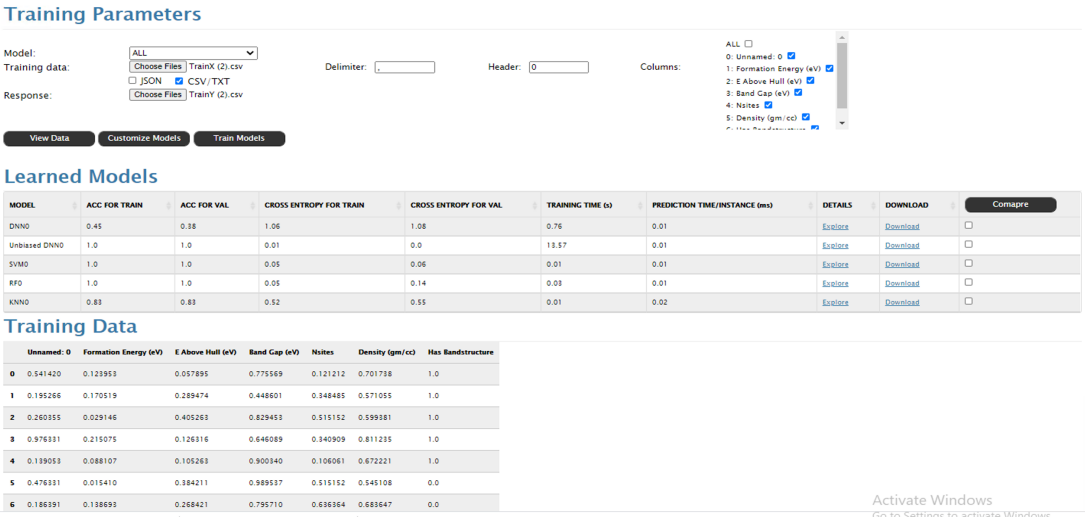


Figure 7-4 Training page of AutoML

Except for the basic metrics, AutoML also provides more detailed information about the training results. For example, users can click the link of "Explore" to see the ROC curve and confusion matrix of the corresponding model.

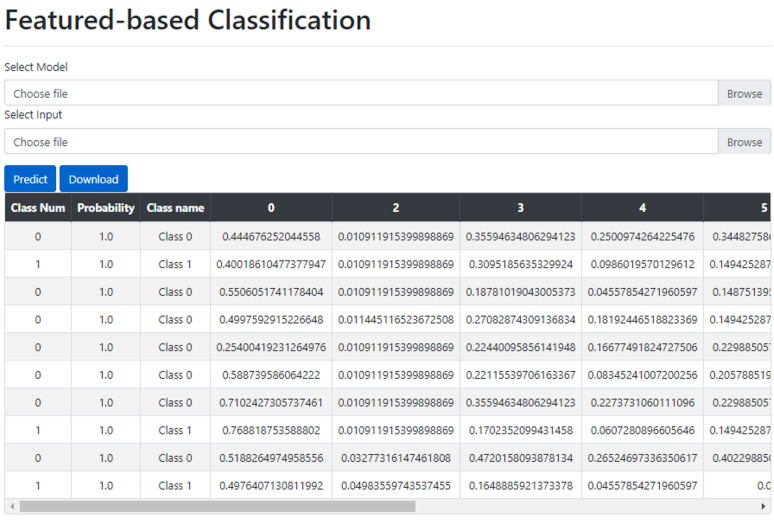
Figure 7-5 Customize models in AutoML

Given results based on a series of metrics, users are able to compare the performance from different perspectives. In our case study, SVM0 is finally selected after careful comparison.

7.3.3 Implementation

The last step is to deploy the well-trained models (SVM0 in our case study) to predict class of the new dataset without labels.

Figure 7-6 shows the implementation page of AutoML. Compared with preprocessing and training page, the implementation page looks much simpler, which only contains two input fields, model uploading field and data uploading field. The prediction results will be presented directly under these two input fields. In the prediction result table, each row is one observation. The first column is the predicted class number. The second column is the probability of the class. The third column is the predicted class name and the remaining columns are the features for each observation. The prediction result can be download easily by clicking the "*Download*" button.



Class Num	Probability	Class name	0	2	3	4	5
0	1.0	Class 0	0.444676252044558	0.010911915399898869	0.35594634806294123	0.2500974264225476	0.34482758
1	1.0	Class 1	0.40018610477377947	0.010911915399898869	0.3095185635329924	0.0986019570129612	0.149425287
0	1.0	Class 0	0.5506051741178404	0.010911915399898869	0.18781019043005373	0.04557854271960597	0.14875139
0	1.0	Class 0	0.4997592915226648	0.011445116523672508	0.27082874309136834	0.18192446518823369	0.149425287
0	1.0	Class 0	0.25400419231264976	0.010911915399898869	0.22440095856141948	0.16677491824727506	0.22988505
0	1.0	Class 0	0.588739586064222	0.010911915399898869	0.22115539706163367	0.08345241007200256	0.205788519
0	1.0	Class 0	0.7102427305737461	0.010911915399898869	0.35594634806294123	0.2273731060111096	0.22988505
1	1.0	Class 1	0.768818753588802	0.010911915399898869	0.1702352099431458	0.0607280896605646	0.149425287
0	1.0	Class 0	0.5188264974958556	0.03277316147461808	0.4720158093878134	0.26524697336350617	0.40229885
1	1.0	Class 1	0.4976407130811992	0.04983559743537455	0.1648885921373378	0.04557854271960597	0.0

Figure 7-6 Implementation page of AutoML

Chapter 8 Summary and Future Work

8.1 Summary of Original Contributions

This thesis contributes to the field of CBM with a focus on Deep Learning approaches for condition monitoring and prognostic analysis. Specifically, this thesis investigates how to apply deep learning methods for sensor fusion, health condition monitoring, degradation modeling, and RUL prediction under the data-rich environment.

The major contributions of this thesis are:

- *Sensor fusion via statistical hypothesis testing for prognosis and degradation analysis*

  The work: 1) separates the cycles of the health index signal via a series of statistical hypothesis tests with more focus on separating cycles closer to failure; 2) defines the degradation (health) states of the system based on the constructed health index and allows determination of the probability mass function of health states at any cycle given the multi-sensor data at that cycle; and 3) predicts not only the RUL of the system but also the remaining time to all states before failure. The proposed sensor fusion method was evaluated by using the published degradation data set of aircraft engines generated by C-MAPSS. The experiment results have shown that the proposed approach outperforms the existing approaches, with lower mean error and standard deviation in the RUL prediction error.

- *A multi-sensor fusion approach based on a novel coupled duration dependent hidden semi-Markov model*

  This work overcome the limitations of existing HMM-based sensor fusion models for degradation modeling and RUL prediction. Compared with existing HMM models and its extensions, this new CDD-HSMM proposed by this work i) captures the interactions among multiple stochastic processes; ii) explicitly defines the distribution of state duration according to actual situation; iii) assumes the observation variable to be state duration dependent. To prove the model effect, we applied CDD-HSMM on two fields

in case study, RUL prediction of turbofan engines and Alzheimer's Disease (AD) stage recognition. The experimental results illustrate that the proposed CDD-HSMM is more effective than the conventional HMM and its extensions.

- *A long short-term memory network for combining change point detection and remaining useful life prediction*

  The proposed Dual-LSTM in this work aims to improve the accuracy of RUL prediction by (i) filtering out the sensor signals irrelevant to the RUL prediction through the change point detection, (ii) introducing a novel one-dimension piecewise decreasing health index function to differentiate the normal operation and degradation process, as well as indicate the health condition of the unit, (iii) making full use of historical information to achieve detection and prediction tasks by characterizing both long and short-term dependencies of sensor signals through LSTM network. The effectiveness of the proposed Dual-LSTM framework is validated and compared to state-of-art benchmark methods on a publicly available turbofan engine degradation dataset. The proposed Dual-LSTM is compared to benchmark methods using multiple evaluation metrics. All comparative results show that the Dual-LSTM outperforms existing benchmark models for RUL predictions.

- *A long short-term memory network for online state-of-charge estimation of li-ion battery cells*

  This work proposes a new long short-term memory neural network model to estimate the state-of-charge of lithium-ion (Li-ion) battery cells. The proposed model improves the estimation accuracy by accounting for the changes in the battery parameters due to ageing by utilizing relevant knowledge from previous cycles when estimating the current state-of-charge. The proposed model was tested and validated on the dataset of 18650 Lithium-ion batteries provided by NASA Ames. The experiment results validate the high accuracy of the proposed LSTM.

8.2 Future Research

The work presented in this thesis contributes to the field of prognostic analysis, one of the most important tasks of CBM. The accurate RUL prediction achieved by the methods proposed in this

thesis helps to raise early alarms to avoid unexpected failures, improve production efficiency, and reduce life-cycle costs. There are some improvements which can be made in the future study.

- *Degradation models under the situation when there are missing or asynchronous sensor measurements*

  In many real-world applications, continuous or frequent sensor signals monitoring the health condition of the machine systems are not always possible or economical. For example, monitoring the cracks on gas turbine blades requires shutting down the turbine. Or it is possible that sensors may encounter unpredicted failures or are occasionally disconnected for some unknown reason. Therefore, in real cases, the sensor signals for degradation monitoring are usually collected sparsely, which will compromise the ability to find a suitable degradation model. Therefore, degradation models for sparse sensor signals are required to meet real needs.

- *Extend existed Deep Learning models for degradation modeling to cases with multiple failure processes and different working conditions*

  For complex systems, the failure usually comes from either operation of the system itself or from external sources. In some real cases, multiple failure processes may occur in one system and any of them can lead to the failure. And these multiple failure processes may be dependent on each other. This creates unique and challenging problems to accurate prognostic analysis, specifically RUL prediction. Therefore, we need to develop models to analyze the degradation process of the complex system which experiences multiple failure processes, to accurately the behavior and RUL of the system.

Bibliography

[1] Liu, K., Gebraeel, N. Z., & Shi, J. (2013). A data-level fusion model for developing composite health indices for degradation modeling and prognostic analysis. *IEEE Transactions on Automation Science and Engineering*, *10*(3), 652–664. https://doi.org/10.1109/tase.2013.2250282

[2] Jardine, A. K., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510. https://doi.org/10.1016/j.ymssp.2005.09.012

[3] Ahmad, R., & Kamaruddin, S. (2012). An overview of time-based and condition-based maintenance in industrial application. *Computers & Industrial Engineering*, *63*(1), 135–149. https://doi.org/10.1016/j.cie.2012.02.002

[4] Vogl, G. W., Weiss, B. A., & Helu, M. (2016). A review of diagnostic and prognostic capabilities and best practices for manufacturing. *Journal of Intelligent Manufacturing*, *30*(1), 79–95. https://doi.org/10.1007/s10845-016-1228-8

[5] Liu, K., & Huang, S. (2016). Integration of data fusion methodology and degradation modeling process to improve prognostics. *IEEE Transactions on Automation Science and Engineering*, *13*(1), 344–354. https://doi.org/10.1109/tase.2014.2349733

[6] Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P., & Gao, R. X. (2019). Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, *115*, 213–237. https://doi.org/10.1016/j.ymssp.2018.05.050

[7] Fang, X., Gebraeel, N. Z., & Paynabar, K. (2017). Scalable prognostic models for large-scale condition monitoring applications. *IISE Transactions*, *49*(7), 698–710. https://doi.org/10.1080/24725854.2016.1264646

[8] Si, X. S., Wang, W., Hu, C. H., & Zhou, D. H. (2011). Remaining useful life estimation – A review on the statistical data driven approaches. *European Journal of Operational Research*, *213*(1), 1–14. https://doi.org/10.1016/j.ejor.2010.11.018

[9] Tsui, K. L., Chen, N., Zhou, Q., Hai, Y., & Wang, W. (2015). Prognostics and health management: A review on data driven approaches. *Mathematical Problems in Engineering*, *2015*, 1–17. https://doi.org/10.1155/2015/793161

[10] Guo, P., & Bai, N. (2011). Wind turbine gearbox condition monitoring with AAKR and moving window statistic methods. *Energies*, *4*(11), 2077–2093. https://doi.org/10.3390/en4112077

[11] Liu, J., & Chen, Z. (2019). Remaining useful life prediction of lithium-ion batteries based on health indicator and Gaussian process regression model. *IEEE Access*, *7*, 39474–39484. https://doi.org/10.1109/access.2019.2905740

[12] Satishkumar, R., & Sugumaran, V. (2016). Estimation of remaining useful life of bearings based on support vector regression. *Indian Journal of Science and Technology*, *9*(10). https://doi.org/10.17485/ijst/2016/v9i10/88997

[13] Limon, S., & Yadav, O. P. (2019). Predicting remaining lifetime using the monotonic Gamma process and Bayesian inference for multi-stress conditions. *Procedia Manufacturing*, *38*, 1260–1267. https://doi.org/10.1016/j.promfg.2020.01.218

[14] Nguyen, T. L., Djeziri, M., Ananou, B., Ouladsine, M., & Pinaton, J. (2016). Fault prognosis for batch production based on percentile measure and Gamma process: Application to semiconductor manufacturing. *Journal of Process Control*, *48*, 72–80. https://doi.org/10.1016/j.jprocont.2016.10.003

[15] Zhai, Q., & Ye, Z. S. (2017). RUL prediction of deteriorating products using an adaptive Wiener process model. *IEEE Transactions on Industrial Informatics*, *13*(6), 2911–2921. https://doi.org/10.1109/tii.2017.2684821

[16] Tang, S., Yu, C., Wang, X., Guo, X., & Si, X. (2014). Remaining useful life prediction of lithium-ion batteries based on the Wiener process with measurement Error. *Energies*, *7*(2), 520–547. https://doi.org/10.3390/en7020520

[17] Jegadeeshwaran, R., & Sugumaran, V. (2015). Fault diagnosis of automobile hydraulic brake system using statistical features and support vector machines. *Mechanical Systems and Signal Processing*, *52–53*, 436–446. https://doi.org/10.1016/j.ymssp.2014.08.007

[18] Ng, S. S., Xing, Y., & Tsui, K. L. (2014). A naive Bayes model for robust remaining useful life prediction of lithium-ion battery. *Applied Energy*, *118*, 114–123. https://doi.org/10.1016/j.apenergy.2013.12.020

[19] Eren, L., Ince, T., & Kiranyaz, S. (2018). A generic intelligent fearing fault diagnosis system using compact adaptive 1D CNN classifier. *Journal of Signal Processing Systems*, *91*(2), 179–189. https://doi.org/10.1007/s11265-018-1378-3

[20] F. O. Heimes. (2008). Recurrent neural networks for remaining useful life estimation. *International Conference on Prognostics and Health Management*, pp. 1-6, https://doi.org/10.1109/PHM.2008.4711422.

[21] Ghate, V. N., & Dudul, S. V. (2010). Optimal MLP neural network classifier for fault detection of three phase induction motor. *Expert Systems with Applications*, *37*(4), 3468–3481. https://doi.org/10.1016/j.eswa.2009.10.041

[22] Ren, L., Sun, Y., Wang, H., & Zhang, L. (2018). Prediction of bearing remaining useful life with deep convolution neural network. *IEEE Access*, *6*, 13041–13049. https://doi.org/10.1109/access.2018.2804930

[23] Wu, Y., Yuan, M., Dong, S., Lin, L., & Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*, *275*, 167–179. https://doi.org/10.1016/j.neucom.2017.05.063

[24] Wang, L., Zhang, Z., Xu, J., & Liu, R. (2018). Wind turbine blade breakage monitoring with deep autoencoders. *IEEE Transactions on Smart Grid*, *9*(4), 2824–2833. https://doi.org/10.1109/tsg.2016.2621135

[25] Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P., & Gao, R. X. (2019). Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, *115*, 213–237. https://doi.org/10.1016/j.ymssp.2018.05.050

[26] Guo, L., Li, N., Jia, F., Lei, Y., & Lin, J. (2017). A recurrent neural network-based health indicator for remaining useful life prediction of bearings. *Neurocomputing*, *240*, 98–109. https://doi.org/10.1016/j.neucom.2017.02.045

[27] Atamuradov, V., Medjaher, K., Camci, F., Dersin, P., & Zerhouni, N. (2018). Degradation-level assessment and online prognostics for sliding chair failure on point machines. *IFAC-PapersOnLine*, *51*(24), 208–213. https://doi.org/10.1016/j.ifacol.2018.09.579

[28] Chehade, A., Song, C., Liu, K., Saxena, A., & Zhang, X. (2018). A data-level fusion approach for degradation modeling and prognostic analysis under multiple failure modes. *Journal of Quality Technology*, *50*(2), 150–165. https://doi.org/10.1080/00224065.2018.1436829

[29] Khaleghi, B., Khamis, A., Karray, F. O., & Razavi, S. N. (2013). Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, *14*(1), 28–44. https://doi.org/10.1016/j.inffus.2011.08.001

[30] Li, N., Lei, Y., Guo, L., Yan, T., & Lin, J. (2017). Remaining useful life prediction based on a general expression of stochastic process models. *IEEE Transactions on Industrial Electronics*, *64*(7), 5709–5718. https://doi.org/10.1109/tie.2017.2677334

[31] Goebel K. and Bonissone P. (2005). Prognostic information fusion for constant load systems. *2005 7th International Conference on Information Fusion*, 1247-1255, https://doi.org/ 10.1109/ICIF.2005.1592000

[32] Volponi A., Brotherton T., Luppold R., & Donald S. (2004). Development of an information fusion system for engine diagnostics and health management. *NASA/TM-2004-212924*, https://doi.org/10.2514/6.2004-6461.

[33] Gunatilaka, A., & Baertlein, B. (2001). Feature-level and decision-level fusion of noncoincidently sampled sensors for land mine detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(6), 577–589. https://doi.org/10.1109/34.927459

[34] Liu, K., Gebraeel, N. Z., & Shi, J. (2013). A data-level fusion model for developing composite health indices for degradation modeling and prognostic analysis. *IEEE Transactions on Automation Science and Engineering*, *10*(3), 652–664. https://doi.org/10.1109/tase.2013.2250282

[35] Liu, K., Chehade, A., & Song, C. (2017). Optimize the signal quality of the composite health index via data fusion for degradation modeling and prognostic analysis. *IEEE Transactions on Automation Science and Engineering*, *14*(3), 1504–1514. https://doi.org/10.1109/tase.2015.2446752

[36] Gravina, R., Alinia, P., Ghasemzadeh, H., & Fortino, G. (2017). Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges. *Information Fusion*, *35*, 68–80. https://doi.org/10.1016/j.inffus.2016.09.005

[37] Chehade, A., & Shi, Z. (2019). Sensor fusion via statistical hypothesis testing for prognosis and degradation analysis. *IEEE Transactions on Automation Science and Engineering*, *16*(4), 1774–1787. https://doi.org/10.1109/tase.2019.2897784

[38] Fang, X., Paynabar, K., & Gebraeel, N. (2017). Multistream sensor fusion-based prognostics model for systems with single failure modes. *Reliability Engineering & System Safety*, *159*, 322–331. https://doi.org/10.1016/j.ress.2016.11.008

[39] Dong, M., & He, D. (2007). Hidden semi-Markov model-based methodology for multi-sensor equipment health diagnosis and prognosis. *European Journal of Operational Research*, *178*(3), 858–878. https://doi.org/10.1016/j.ejor.2006.01.041

[40] Chen, Q., Hu, Y., Xia, J., Chen, Z., & Tseng, H. W. (2018). Data fusion of wireless sensor network for prognosis and diagnosis of mechanical systems. *Microsystem Technologies*, *27*(4), 1187–1199. https://doi.org/10.1007/s00542-018-4144-3

[41] Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical Systems and Signal Processing*, *104*, 799–834. https://doi.org/10.1016/j.ymssp.2017.11.016

[42] Nectoux P., Gouriveau R., Medjaher K., Ramasso E., Chebel-Morello B., et al. (2012). PRONOSTIA: An experimental platform for bearings accelerated degradation tests. *IEEE International Conference on Prognostics and Health Management, PHM'12.*, 1-8.

[43] Qiu, H., Lee, J., Lin, J., & Yu, G. (2003). Robust performance degradation assessment methods for enhanced rolling element bearing prognostics. *Advanced Engineering Informatics*, *17*(3–4), 127–140. https://doi.org/10.1016/j.aei.2004.08.001

[44] Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical Systems and Signal Processing*, *104*, 799–834. https://doi.org/10.1016/j.ymssp.2017.11.016

[45] Wang, W. (2002). A model to predict the residual life of rolling element bearings given monitored condition information to date. *IMA Journal of Management Mathematics*, *13*(1), 3–16. https://doi.org/10.1093/imaman/13.1.3

[46] Jin, X., Sun, Y., Que, Z., Wang, Y., & Chow, T. W. S. (2016). Anomaly detection and fault prognosis for bearings. *IEEE Transactions on Instrumentation and Measurement*, *65*(9), 2046–2054. https://doi.org/10.1109/tim.2016.2570398

[47] Fink, O., Zio, E., & Weidmann, U. (2015). A classification framework for predicting components' remaining useful life based on discrete-event diagnostic data. *IEEE Transactions on Reliability*, *64*(3), 1049–1056. https://doi.org/10.1109/tr.2015.2440531

[48] Wu, Y., Yuan, M., Dong, S., Lin, L., & Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*, *275*, 167–179. https://doi.org/10.1016/j.neucom.2017.05.063

[49] Lei, Y., Li, N., & Lin, J. (2016). A new method based on stochastic process models for machine remaining useful life prediction. *IEEE Transactions on Instrumentation and Measurement*, *65*(12), 2671–2684. https://doi.org/10.1109/tim.2016.2601004

[50] Ye, Z. S., Chen, N., & Shen, Y. (2015). A new class of Wiener process models for degradation analysis. *Reliability Engineering & System Safety*, *139*, 58–67. https://doi.org/10.1016/j.ress.2015.02.005

[51] Si, X. S., Wang, W., Hu, C. H., Chen, M. Y., & Zhou, D. H. (2013). A Wiener-process-based degradation model with a recursive filter algorithm for remaining useful life estimation. *Mechanical Systems and Signal Processing*, *35*(1–2), 219–237. https://doi.org/10.1016/j.ymssp.2012.08.016

[52] Si, X. S., Wang, W., Hu, C. H., Chen, M. Y., & Zhou, D. H. (2013). A Wiener-process-based degradation model with a recursive filter algorithm for remaining useful life estimation. *Mechanical Systems and Signal Processing*, *35*(1–2), 219–237. https://doi.org/10.1016/j.ymssp.2012.08.016

[53] Wang, H., Xu, T., & Mi, Q. (2015). Lifetime prediction based on Gamma processes from accelerated degradation data. *Chinese Journal of Aeronautics*, *28*(1), 172–179. https://doi.org/10.1016/j.cja.2014.12.015

[54] Kharoufeh, J. P., Solo, C. J., & Ulukus, M. Y. (2010). Semi-Markov models for degradation-based reliability. *IIE Transactions*, *42*(8), 599–612. https://doi.org/10.1080/07408170903394371

[55] Tee, K. F., Ekpiwhre, E., & Yi, Z. (2018). Degradation modelling and life expectancy using Markov chain model for carriageway. *International Journal of Quality & Reliability Management*, *35*(6), 1268–1288. https://doi.org/10.1108/ijqrm-06-2017-0116

[56] Liu, T., Zhu, K., & Zeng, L. (2018). Diagnosis and prognosis of degradation process via hidden semi-Markov model. *IEEE/ASME Transactions on Mechatronics*, *23*(3), 1456–1466. https://doi.org/10.1109/tmech.2018.2823320

[57] Jia, J., Liang, J., Shi, Y., Wen, J., Pang, X., & Zeng, J. (2020). SOH and RUL prediction of lithium-ion batteries based on Gaussian process regression with indirect health indicators. *Energies*, *13*(2), 375. https://doi.org/10.3390/en13020375

[58] Di Maio, F., Tsui, K. L., & Zio, E. (2012). Combining relevance vector machines and exponential regression for bearing residual life estimation. *Mechanical Systems and Signal Processing*, *31*, 405–427. https://doi.org/10.1016/j.ymssp.2012.03.011

[59] Zheng S., Ristovski K., Farahat A. & Gupta C. (2017). Long short-term memory network for remaining useful life estimation. *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 88-95, https://doi.org/10.1109/ICPHM.2017.7998311

[60] Zhang, J., Wang, P., Yan, R., & Gao, R. X. (2018). Long short-term memory for machine remaining life prediction. *Journal of Manufacturing Systems*, *48*, 78–86. https://doi.org/10.1016/j.jmsy.2018.05.011

[61] Elsheikh, A., Yacout, S., & Ouali, M. S. (2019). Bidirectional handshaking LSTM for remaining useful life prediction. *Neurocomputing*, *323*, 148–156. https://doi.org/10.1016/j.neucom.2018.09.076

[62] Miao, H., Li, B., Sun, C., & Liu, J. (2019). Joint learning of degradation assessment and RUL prediction for aeroengines via dual-task deep LSTM networks. *IEEE Transactions on Industrial Informatics*, *15*(9), 5023–5032. https://doi.org/10.1109/tii.2019.2900295

[63] Yang, B., Liu, R., & Zio, E. (2019). Remaining useful life prediction based on a double-convolutional neural network architecture. *IEEE Transactions on Industrial Electronics*, *66*(12), 9521–9530. https://doi.org/10.1109/tie.2019.2924605

[64] Xia, M., Li, T., Liu, L., Xu, L., & Silva, C. W. (2017). Intelligent fault diagnosis approach with unsupervised feature learning by stacked denoising autoencoder. *IET Science, Measurement & Technology*, *11*(6), 687–695. https://doi.org/10.1049/iet-smt.2016.0423

[65] Betechuoh, B., Marwala, T., & Tettey, T. (2006). Autoencoder networks for HIV classification. *Current Science, 91*(11), 1467-1473. Retrieved September 7, 2021, http://www.jstor.org/stable/24093843

[66] Chen, Z., & Li, W. (2017). Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network. *IEEE Transactions on Instrumentation and Measurement*, *66*(7), 1693–1702. https://doi.org/10.1109/tim.2017.2669947

[67] Ren, L., Sun, Y., Cui, J., & Zhang, L. (2018). Bearing remaining useful life prediction based on deep autoencoder and deep neural networks. *Journal of Manufacturing Systems*, *48*, 71–77. https://doi.org/10.1016/j.jmsy.2018.04.008

[68] Ma, J., Su, H., Zhao, W. L., & Liu, B. (2018). Predicting the remaining useful life of an aircraft engine using a stacked sparse autoencoder with multilayer self-learning. *Complexity*, *2018*, 1–13. https://doi.org/10.1155/2018/3813029

[69] Song, Y., Shi, G., Chen, L., Huang, X., & Xia, T. (2018). Remaining useful life prediction of turbofan engine using hybrid model based on autoencoder and bidirectional long short-term memory. *Journal of Shanghai Jiaotong University (Science)*, *23*(S1), 85–94. https://doi.org/10.1007/s12204-018-2027-5

[70] Zhao G., Zhang G., Ge Q. & Liu X. (2016). Research advances in fault diagnosis and prognostic based on deep learning. *Prognostics and System Health Management Conference*, Chengdu, China, (pp: 1-6), https://doi.org/10.1109/PHM.2016.7819786

[71] Jardine, A. K., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510. https://doi.org/10.1016/j.ymssp.2005.09.012

[72] Kan, M. S., Tan, A. C., & Mathew, J. (2015). A review on prognostic techniques for non-stationary and non-linear rotating systems. *Mechanical Systems and Signal Processing*, *62–63*, 1–20. https://doi.org/10.1016/j.ymssp.2015.02.016

[73] Veldman, J., Klingenberg, W., & Wortmann, H. (2011). Managing condition-based maintenance technology. *Journal of Quality in Maintenance Engineering*, *17*(1), 40–62. https://doi.org/10.1108/13552511111116240

[74] Liao, L. (2014). Discovering prognostic features using genetic programming in remaining useful life prediction. *IEEE Transactions on Industrial Electronics*, *61*(5), 2464–2472. https://doi.org/10.1109/tie.2013.2270212

[75] Liu, K., & Huang, S. (2016b). Integration of data fusion methodology and degradation modeling process to improve prognostics. *IEEE Transactions on Automation Science and Engineering*, *13*(1), 344–354. https://doi.org/10.1109/tase.2014.2349733

[76] Dragomir O. E., Gouriveau R., Dragomir F., Minca E., & Zerhouni N. (2009). Review of prognostic problem in condition-based maintenance. *2009 European Control Conference*, Budapest, Hungary, (pp: 1587–1592), https://doi.org/10.23919/ECC.2009.7074633

[77] Yan, J., Koç, M., & Lee, J. (2004). A prognostic algorithm for machine performance assessment and its application. *Production Planning & Control*, *15*(8), 796–801. https://doi.org/10.1080/09537280412331309208

[78] Saha, B., Goebel, K., Poll, S., & Christophersen, J. (2009). Prognostics methods for battery health monitoring using a Bayesian framework. *IEEE Transactions on Instrumentation and Measurement*, *58*(2), 291–296. https://doi.org/10.1109/tim.2008.2005965

[79] Saha S., Saha B., Saxena A. & Goebel K. (2010). Distributed prognostic health management with gaussian process regression. *2010 IEEE Aerospace Conference*, Big Sky, MT, USA, (pp. 1-8), https://doi.org/10.1109/AERO.2010.5446841

[80] Peng, Y., & Dong, M. (2011). A prognosis method using age-dependent hidden semi-Markov model for equipment health prediction. *Mechanical Systems and Signal Processing*, *25*(1), 237–252. https://doi.org/10.1016/j.ymssp.2010.04.002

[81] Bunks, C., Mccarthy, D., & Al-ani, T. (2000). Condition-based maintenance of machines using hidden Markov models. *Mechanical Systems and Signal Processing*, *14*(4), 597–612. https://doi.org/10.1006/mssp.2000.1309

[82] Zhou, Z. J., Hu, C. H., Xu, D. L., Chen, M. Y., & Zhou, D. H. (2010). A model for real-time failure prognosis based on hidden Markov model and belief rule base. *European Journal of Operational Research*, *207*(1), 269–283. https://doi.org/10.1016/j.ejor.2010.03.032

[83] Phelps E., Willett P. K., & Kirubarajan T. (2001) Statistical approach to prognostics. *Component and Systems Diagnostics, Prognostics, and Health Management*, 4389, 23-24. https://doi.org/10.1117/12.434249

[84] Park, C., & Padgett, W. (2005). New cumulative damage models for failure using stochastic processes as initial damage. *IEEE Transactions on Reliability*, *54*(3), 530–540. https://doi.org/10.1109/tr.2005.853278

[85] Gebraeel, N. Z., Lawley, M. A., Li, R., & Ryan, J. K. (2005). Residual-life distributions from component degradation signals: A Bayesian approach. *IIE Transactions*, *37*(6), 543–557. https://doi.org/10.1080/07408170590929018

[86] Wang, X., & Xu, D. (2010). An inverse Gaussian process model for degradation data. *Technometrics*, *52*(2), 188–197. https://doi.org/10.1198/tech.2009.08197

[87] Carlson, N. (1990). Federated square root filter for decentralized parallel processors. *IEEE Transactions on Aerospace and Electronic Systems*, *26*(3), 517–525. https://doi.org/10.1109/7.106130

[88] Carlson, N. (1990). Federated square root filter for decentralized parallel processors. *IEEE Transactions on Aerospace and Electronic Systems*, *26*(3), 517–525. https://doi.org/10.1109/7.106130

[89] Safari, S., Shabani, F., & Simon, D. (2014). Multirate multisensor data fusion for linear systems using Kalman filters and a neural network. *Aerospace Science and Technology*, *39*, 465–471. https://doi.org/10.1016/j.ast.2014.06.005

[90] Li, S., Kwok, J. T., & Wang, Y. (2002). Multifocus image fusion using artificial neural networks. *Pattern Recognition Letters*, *23*(8), 985–997. https://doi.org/10.1016/s0167-8655(02)00029-6

[91] Huang, W., & Jing, Z. (2007). Multi-focus image fusion using pulse coupled neural network. *Pattern Recognition Letters*, *28*(9), 1123–1132. https://doi.org/10.1016/j.patrec.2007.01.013

[92] Yen, J., & Pfluger, N. (1995). A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation. *IEEE Transactions on Systems, Man, and Cybernetics*, *25*(6), 971–978. https://doi.org/10.1109/21.384260

[93] Wichit, N., & Choksuriwong, A. (2015). Multi-sensor data fusion model based Kalman filter using fuzzy logic for human activity detection. *International Journal of Information and Electronics Engineering*, *5*(6), 450–453. https://doi.org/10.7763/ijiee.2015.v5.577

[94] Naeem, W., Sutton, R., & Xu, T. (2012). An integrated multi-sensor data fusion algorithm and autopilot implementation in an uninhabited surface craft. *Ocean Engineering*, *39*, 43–52. https://doi.org/10.1016/j.oceaneng.2011.11.001

[95] Gao, T., Yin, S., Gao, H., Yang, X., Qiu, J., & Kaynak, O. (2018). A locally weighted project regression approach-aided nonlinear constrained tracking control. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(12), 5870–5879. https://doi.org/10.1109/tnnls.2018.2808700

[96] Bhateja, V., Patel, H., Krishn, A., Sahu, A., & Lay-Ekuakille, A. (2015). Multimodal medical image sensor fusion framework using cascade of wavelet and contourlet transform domains. *IEEE Sensors Journal*, *15*(12), 6783–6790. https://doi.org/10.1109/jsen.2015.2465935

[97] Ghassemian, H. (2016). A review of remote sensing image fusion methods. *Information Fusion*, *32*, 75–89. https://doi.org/10.1016/j.inffus.2016.03.003

[98] Bergamini, E., Ligorio, G., Summa, A., Vannozzi, G., Cappozzo, A., & Sabatini, A. (2014). Estimating orientation using magnetic and inertial sensors and different sensor fusion approaches: accuracy assessment in manual and locomotion tasks. *Sensors*, *14*(10), 18625–18649. https://doi.org/10.3390/s141018625

[99] Santoso, F., Garratt, M. A., & Anavatti, S. G. (2017). Visual–inertial navigation systems for aerial robotics: sensor fusion and technology. *IEEE Transactions on Automation Science and Engineering*, *14*(1), 260–275. https://doi.org/10.1109/tase.2016.2582752

[100] Moon, S., Park, Y., Ko, D. W., & Suh, I. H. (2016). Multiple kinect sensor fusion for Human skeleton tracking using Kalman filtering. *International Journal of Advanced Robotic Systems*, *13*(2), 65. https://doi.org/10.5772/62415

[101] Jiang, Y., & Yin, S. (2019). Recent advances in key-performance-indicator oriented prognosis and diagnosis with a MATLAB toolbox: DB-KIT. *IEEE Transactions on Industrial Informatics*, *15*(5), 2849–2858. https://doi.org/10.1109/tii.2018.2875067

[102] Alvarez M., & Lawrence N. D. (2009). Sparse convolved Gaussian processes for multi-output regression. *Advances in Neural Information Processing Systems*, 21, 57–64.

[103] Alvarez M. A. & Lawrence N. D. (2011). Computationally efficient convolved multiple output Gaussian processes. *Journal of Machine Learning Research*, 12, 1459–1500.

[104] Lu, C. J., & Meeker, W. O. (1993). Using degradation measures to estimate a time-to-failure distribution. *Technometrics*, *35*(2), 161–174. https://doi.org/10.1080/00401706.1993.10485038

[105] Zafar, S. (2005). Statistical mechanics-based model for negative bias temperature instability induced degradation. *Journal of Applied Physics*, *97*(10), 103709. https://doi.org/10.1063/1.1889226

[106] Chehade, A., Bonk, S., & Liu, K. (2017). Sensory-based failure threshold estimation for remaining useful life prediction. *IEEE Transactions on Reliability*, *66*(3), 939–949. https://doi.org/10.1109/tr.2017.2695119

[107] DeCastro J., Litt J., & Frederick D. (2008). A modular aero-propulsion system simulation of a large commercial aircraft engine. *44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit,* Hartford, CT. https://doi.org/10.2514/6.2008-4579.

[108] Saxena A., Goebel K., Simon D., & Eklund N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. *2008 International Conference on Prognostics and Health Management*, Denvor, CO (pp. 1–9). https://doi.org/10.1109/PHM.2008.4711414

[109] Ghamlouch, H., Fouladirad, M., & Grall, A. (2019). The use of real option in condition-based maintenance scheduling for wind turbines with production and deterioration uncertainties. *Reliability Engineering & System Safety*, *188*, 614–623. https://doi.org/10.1016/j.ress.2017.10.001

[110] Igba, J., Alemzadeh, K., Henningsen, K., & Durugbo, C. (2014). Effect of preventive maintenance intervals on reliability and maintenance costs of wind turbine gearboxes. *Wind Energy*, *18*(11), 2013–2024. https://doi.org/10.1002/we.1801

[111] Kang, J., Sobral, J., & Soares, C. G. (2019). Review of condition-based maintenance strategies for offshore wind energy. *Journal of Marine Science and Application*, *18*(1), 1–16. https://doi.org/10.1007/s11804-019-00080-y

[112] Stenström, C., Norrbin, P., Parida, A., & Kumar, U. (2015). Preventive and corrective maintenance – cost comparison and cost–benefit analysis. *Structure and Infrastructure Engineering*, *12*(5), 603–617. https://doi.org/10.1080/15732479.2015.1032983

[113] Andrzejczak, K., Młyńczak, M., & Selech, J. (2018). Poisson-distributed failures in the predicting of the cost of corrective maintenance. *Eksploatacja i Niezawodnosc - Maintenance and Reliability*, *20*(4), 602–609. https://doi.org/10.17531/ein.2018.4.11

[114] Levitin, G., Xing, L., & Xiang, Y. (2020). Optimizing software rejuvenation policy for tasks with periodic inspections and time limitation. *Reliability Engineering & System Safety*, *197*, 106776. https://doi.org/10.1016/j.ress.2019.106776

[115] Alebrant Mendes, A., Ribeiro, J. L. D., & Coit, D. W. (2017). Optimal time interval between periodic inspections for a two-component cold standby multistate system. *IEEE Transactions on Reliability*, *66*(2), 559–574. https://doi.org/10.1109/tr.2017.2689501

[116] Arzaghi, E., Abaei, M. M., Abbassi, R., Garaniya, V., Chin, C., & Khan, F. (2017). Risk-based maintenance planning of subsea pipelines through fatigue crack growth monitoring. *Engineering Failure Analysis*, *79*, 928–939. https://doi.org/10.1016/j.engfailanal.2017.06.003

[117] Pui, G., Bhandari, J., Arzaghi, E., Abbassi, R., & Garaniya, V. (2017). Risk-based maintenance of offshore managed pressure drilling (MPD) operation. *Journal of Petroleum Science and Engineering*, *159*, 513–521. https://doi.org/10.1016/j.petrol.2017.09.066

[118] Patriksson M., Wojciechowski A., & Tjernberg L. B. (2009). An optimization framework for opportunistic maintenance of offshore wind power system. *2009 IEEE Bucharest PowerTech*, Bucharest, Romania (pp: 1-7). https://doi.org/10.1109/PTC.2009.5281868

[119] Saranga, H. (2004). Opportunistic maintenance using genetic algorithms. *Journal of Quality in Maintenance Engineering*, *10*(1), 66–74. https://doi.org/10.1108/13552510410526884

[120] Ding, F., & Tian, Z. (2011). Opportunistic maintenance optimization for wind turbine systems considering imperfect maintenance actions. *International Journal of Reliability, Quality and Safety Engineering*, *18*(05), 463–481. https://doi.org/10.1142/s0218539311004196

[121] Chen, N., Ye, Z. S., Xiang, Y., & Zhang, L. (2015). Condition-based maintenance using the inverse Gaussian degradation model. *European Journal of Operational Research*, *243*(1), 190–199. https://doi.org/10.1016/j.ejor.2014.11.029

[122] Yam, R. C. M., Tse, P., Li, L., & Tu, P. (2001). Intelligent predictive decision support system for condition-based maintenance. *The International Journal of Advanced Manufacturing Technology*, *17*(5), 383–391. https://doi.org/10.1007/s001700170173

[123] Alaswad, S., & Xiang, Y. (2017). A review on condition-based maintenance optimization models for stochastically deteriorating system. *Reliability Engineering & System Safety*, *157*, 54–63. https://doi.org/10.1016/j.ress.2016.08.009

[124] Arnold A., Nallapati R., & Cohen W. W. (2007). A comparative study of methods for transductive transfer learning. *Seventh IEEE International Conference Data Mining Workshops (ICDMW 2007)*, Omaha, NE (pp. 77–82). https://doi.org/10.1109/ICDMW.2007.109

[125] Schwabacher M. & Goebel K. (2007). A survey of artificial intelligence for prognostics. *AAAI Fall Symposium*, Arlington, VA (pp. 107–114).

[126] Okoh, C., Roy, R., Mehnen, J., & Redding, L. (2014). Overview of remaining useful life prediction techniques in through-life engineering services. *Procedia CIRP*, *16*, 158–163. https://doi.org/10.1016/j.procir.2014.02.006

[127] Ordóñez, C., Sánchez Lasheras, F., Roca-Pardiñas, J., & Juez, F. J. D. C. (2019). A hybrid ARIMA–SVM model for the study of the remaining useful life of aircraft engines. *Journal of Computational and Applied Mathematics*, *346*, 184–191. https://doi.org/10.1016/j.cam.2018.07.008

[128] Wang, M., & Wang, J. (2011). CHMM for tool condition monitoring and remaining useful life prediction. *The International Journal of Advanced Manufacturing Technology*, *59*(5–8), 463–471. https://doi.org/10.1007/s00170-011-3536-7

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

[130] Zheng C. *et al.* (2018). A data-driven approach for remaining useful life prediction of aircraft engines. *2018 21st International Conference on Intelligent Transportation Systems*

(ITSC), Maui, Hawaii, (pp. 184–189). https://doi.org/10.1109/ITSC.2018.8569915

[131] Pascanu, R., Mikolov, T., & Bengio, Y. (2012). Understanding the exploding gradient problem. *ArXiv*, abs/1211.5063.

[132] Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *06*(02), 107–116. https://doi.org/10.1142/s0218488598000094

[133] Pascanu, R., Mikolov, T. & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *Proceedings of the 30th International Conference on Machine Learning,* 28(3):1310-1318. https://proceedings.mlr.press/v28/pascanu13.html

[134] Gers F. A. & Schraudolph N. N. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3(2002), 115–143.

[135] Hochreiter S. & Schmidhuber J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[136] Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2009). A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(5), 855–868. https://doi.org/10.1109/tpami.2008.137

[137] Wang S. et al. (2018). A remaining useful life prediction model based on hybrid long-short sequences for engines, *2018 21st International Conference on Intelligent Transportation Systems (ITSC),* Maui, Hawaii (pp. 1757-1762). https://doi.org/10.1109/ITSC.2018.8569668

[138] Yuan M., Wu Y., & Lin L. (2016). Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network. *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, Beijing, China (pp. 135-140). https://doi.org/10.1109/AUS.2016.7748035.

[139] Ramasso, E., & Saxena, A. (2020). Performance benchmarking and analysis of prognostic methods for CMAPSS datasets. *International Journal of Prognostics and Health Management*, *5*(2). https://doi.org/10.36001/ijphm.2014.v5i2.2236

[140] Ramasso E., & Saxena A. (2014). Review and analysis of algorithmic approaches developed for prognostics on CMAPSS dataset. *Annual Conference of the Prognostics and Health Management Society 2014*, Fort Worth, TX, USA.

[141] Niknam, S. A., Kobza, J., & Hines, J. W. (2016). Techniques of trend analysis in degradation-based prognostics. *The International Journal of Advanced Manufacturing Technology*, *88*(9–12), 2429–2441. https://doi.org/10.1007/s00170-016-8909-5

[142] Liu L., Wang S., Liu D., & Peng Y. (2016). Quantitative description of sensor data monotonic trend for system degradation condition monitoring. *2016 Prognostics and System Health Management Conference*, Chengdu, China (pp. 1-5). https://doi.org/10.1109/PHM.2016.7819924.

[143] Badagián A. L., Regina A., & Peña D. (2013). Time series segmentation procedures to detect, locate and estimate change-points. In Beran J., Feng Y., & Hebbel H. (Eds.), *Emperical Economic and Financial Research: Theory, Methods and Practice* (pp: 45-59). Springer.

[144] Korkas, K., & Fryzlewicz, P. (2017). Multiple change-point detection for non-stationary time series using Wild Binary Segmentation. *Statistica Sinica*, 27(2017), 287-311. https://doi.org/10.5705/ss.202015.0262

[145] Zhang, C., Zhu, Y., Dong, G., & Wei, J. (2019). Data-driven lithium-ion battery states estimation using neural networks and particle filtering. *International Journal of Energy Research*, 43(14), 8230-8241. https://doi.org/10.1002/er.4820

[146] Chehade A. A., & Hussein A. A. (2019). A multi-output convolved Gaussian process model for capacity estimation of electric vehicle li-ion battery cells. *2019 IEEE Transportation Electrification Conference and Expo (ITEC)*, Novi, Michigan (pp. 1-4). https://doi.org/10.1109/ITEC.2019.8790463

[147] Khalid A., Sundararajan A., Acharya I. & Sarwat A. I. (2019). Prediction of li-ion battery state of charge using multilayer perceptron and long short-term memory models. *2019 IEEE Transportation Electrification Conference and Expo (ITEC)*, Detroit, Michigan (pp. 1-6). https://doi.org/10.1109/ITEC.2019.8790533

[148] Yu, Z., Huai, R., & Xiao, L. (2015). State-of-charge estimation for lithium-ion batteries using a Kalman filter based on local linearization. *Energies*, *8*(8), 7854–7873. https://doi.org/10.3390/en8087854

[149] Huria T., Ceraolo, M., Gazzarri, J., & Jackey, R. (2013) Simplified extended Kalman filter observer for SOC estimation of commercial power oriented LFP lithium battery cells. *SAE Technical Paper* 2013-01-1544. https://doi.org/10.4271/2013-01-1544

[150] Chemali, E., Kollmeyer, P. J., Preindl, M., & Emadi, A. (2018). State-of-charge estimation of Li-ion batteries using deep neural networks: A machine learning approach. *Journal of Power Sources*, *400*, 242–255. https://doi.org/10.1016/j.jpowsour.2018.06.104

[151] Zhang, R., Xia, B., Li, B., Cao, L., Lai, Y., Zheng, W., Wang, H., & Wang, W. (2018). State of the art of lithium-ion battery SOC estimation for electrical vehicles. *Energies*, *11*(7), 1820. https://doi.org/10.3390/en11071820

[152] Li, C., Xiao, F., & Fan, Y. (2019). An approach to state of charge estimation of lithium-ion batteries based on recurrent neural networks with gated recurrent unit. *Energies*, *12*(9), 1592. https://doi.org/10.3390/en12091592

[153] Song, X., Yang, F., Wang, D., & Tsui, K. L. (2019). Combined CNN-LSTM network for state-of-charge estimation of lithium-ion batteries. *IEEE Access*, *7*, 88894–88902. https://doi.org/10.1109/access.2019.2926517

[154] Chehade, A., & Liu, K. (2019). Structural degradation modeling framework for sparse data sets with an application on Alzheimer's disease. *IEEE Transactions on Automation Science and Engineering*, *16*(1), 192–205. https://doi.org/10.1109/tase.2018.2829770