Remaining Useful Life Estimation using Event Data

Mahbubul Alam[1], Laleh Jalali[2], Dipanjan Ghosh[3], Ahmed Farahat[4] and Chetan Gupta[5]

[1,2,3,4,5]*Industrial AI Lab, Hitachi America Ltd. R&D, Santa Clara, CA, 95054, USA*

mahbubul.alam@hal.hitachi.com
laleh.jalali@hal.hitachi.com
dipanjan.ghosh@hal.hitachi.com
ahmed.farahat@hal.hitachi.com
chetan.gupta@hal.hitachi.com

ABSTRACT

Prognostics aims to predict the degradation of equipment by estimating their remaining useful life (RUL) and/or the failure probability within a specific time horizon. The high demand of equipment prognostics in the industry have propelled researchers to develop robust and efficient prognostics techniques. Among data driven techniques for prognostics, machine learning and deep learning (DL) based techniques, particularly Recurrent Neural Networks (RNNs) have gained significant attention due to their ability of effectively representing the degradation progress by employing dynamic temporal behaviors. RNNs are well known for handling sequential data, especially continuous time series sequential data where the data follows certain pattern. Such data is usually obtained from sensors attached to the equipment. However, in many scenarios, sensor data is not readily available and often very tedious to acquire. Conversely, event data is more common and can easily be obtained from the error logs saved by the equipment and transmitted to a backend for further processing. Nevertheless, performing prognostics using event data is substantially more difficult than that of the sensor data due to the unique nature of event data. Though event data is sequential, it differs from other seminal sequential data such as time series and natural language in the following manner, i) unlike time series, events are aperiodic and scarce, i.e., the appearance of events lacks periodicity; ii) unlike natural languages, event data do not follow any specific linguistic rule. Additionally, there may be a significant variability in the event types appearing within the same sequence. Therefore, this paper proposes an RUL estimation framework to effectively handle the intricate and novel event data. The proposed framework takes discrete events generated by an equipment (e.g., type, time, etc.) as input, and generates for each new event an estimate of the remaining operating cycles in the life of a given component. To evaluate the efficacy of our proposed method, we conduct extensive experiments using benchmark datasets such as the C-MAPSS data after converting the time-series data in these datasets to sequential event data. Furthermore, we propose several deep learning and machine learning based solution for the event-based RUL estimation problem. Our results suggest that the deep learning models, 1D-CNN, LSTM, and multi-head attention show similar RMSE, MAE and Score performance. Foreseeably, the XGBoost model achieve lower performance compared to the deep learning models since the XGBoost model fails to capture ordering information from the sequence of events.

1. INTRODUCTION

Prognostics is concerned with the prediction of future health and performance and any potential failure. Prognostics techniques are typically applied when a fault or degradation is detected in the unit to predict when a failure or severe degradation will happen. The problem of predicting a failure or estimating the remaining useful life of an equipment has been extensively studied in the Prognostics and Health Management (PHM) research community (Goebel et al., 2017).

Failure Prediction (FP) can be defined as predicting whether a monitored unit will fail within a given time horizon. The prediction methods receive as input the raw measurements from the unit and produce as output the probability of a certain failure type. For different failure types, multiple models can be constructed. If there are many failure examples, classification models can be learned from the data to distinguish between failure and non-failure cases.

On the other hand, Remaining Useful Life (RUL) estimation is concerned with estimating how much time or how many operating cycles are left in the life of the unit till a failure event of a given type happens. The prediction methods receive as input the raw measurements from the unit and produce as output a continuous output that reflect the remaining useful life (in time or operating cycles units). RUL estimation is the most-studied problem in the PHM literature and one of the few problems in which there are benchmark datasets such as the C-MAPSS dataset from NASA (Saxena et al., 2008) and other datasets with fewer number of run-to-failure examples (Gao et al., 2015). If there are many run-to-failure examples, the RUL problem can be formulated as a regression problem. Traditionally several regression-based approaches have been used to solve the RUL problem such as neural networks (Peel, 2008)(Lim et al., 2014), Hidden Markov Models (Ghasemi et al., 2010), and similarity-based methods. (T. Wang et al.,

2008). Recently, many deep learning models have been applied to the RUL problem. For instance, Deep Convolutional Neural Network (CNN) (Babu et al., 2016) applies the convolution and pooling filters along the temporal dimension over the multi-channel sensor data. Long Short-Term Memory (LSTM) (Zheng et al., 2017) uses multiple layers of LSTM cells in combination with standard feed forward layers to discover hidden patterns from sensor and operational data.

However, most of the existing techniques for RUL are designed to work on cases where the available data are multivariate time-series of sensor measurements that were recorded before failures. For most of the equipment, such sensor measurements are not available. Instead, most of equipment control units record and communicate events that reflect important changes in the underlying sensors (e.g., an event to reflect high pressure or low temperature) instead of maintaining the raw sensor measurements every few seconds (e.g., pressure and temperature measures). These events are typically defined by the equipment designers to summarize many raw signals and encode the important domain knowledge that need be communicated to the equipment users and repair technicians. In addition, for Internet of Things (IoT) solutions, managing these events instead of raw sensor measurements significantly reduces storage and communication costs. For these types of equipment, traditional techniques for RUL estimation will not be able to handle discrete events and are not designed to benefit from the domain knowledge encoded in such events.

Consequently, several studies (Costello et al., 2017)(Xu et al., 2020) explore the potential of event data for solving the RUL estimation task. Costello et al. (Costello et al., 2017) propose a machine learning based RUL estimation technique for gas circulator (GC) using event data. The proposed method captures low power refueling (LPR) events from the time series load and vibration data which are the indicator of fuel replenishment state for the GC. First, a classifier identifies the states of the LPR for all historical data using the vibration response to segment the data into three classes, {Online, Upper, Lower}. Next, the data is further segmented into four temporal slices, {early, mid1, mid2, late}. Finally, a classifier identifies the likelihood of an LPR event to be in the late temporal state which acts as an implicit RUL estimate. Though promising, the proposed method is rigidly designed for the GC and may not be easily applicable to other equipment and domains. Another work in (Xu et al., 2020) proposes an echo state network (ESN) based RUL estimation technique using event data. The technique utilizes synthetically generated event data considering a system made by four non-repairable components. The task is to estimate the RUL of the last component using the six measurements corresponding to the events. However, the use of controlled synthetic data may limit the use of this technique for more complex practical event data.

Accordingly, to overcome the above challenges, we propose a generalized event-based prognostics technique using the discrete events (e.g., type, timestamp, etc.) captured from the equipment to estimate the RUL of the equipment. To demonstrate the practicality of our proposed method, we use the well-known publicly available C-MAPSS data. More specifically, we convert the C-MAPSS time series data to sequential event data by careful exploration and application of appropriate transformation techniques[1]. Subsequently, the event data is used to evaluate the performance of several deep learning and machine learning based techniques for solving the RUL estimation task. Our results suggest that the deep learning models, 1D-CNN, LSTM, and multi-head attention show similar RMSE, MAE and Score performance. Foreseeably, the XGBoost model achieve lower performance compared to the deep learning models since the XGBoost model fails to capture ordering information from the sequence of events.

The rest of the paper is organized as follows. Section 2 briefly discusses the deep learning and machine learning techniques used in this study. Section 3 formally defines the event-based RUL estimation problem, discusses the data pre-processing and feature engineering technique, and explains the proposed deep learning and machine learning based RUL solution for event data. Finally, Section 4 discusses the experiments and results followed by conclusion in Section 5.

2. BACKGROUND

In this paper, we utilize four machine learning and deep learning techniques for solving the event-based RUL estimation problem, Extended Gradient Boosting (XGBoost), Long Short-Term Memory (LSTM) networks, one-dimensional Convolutional Neural Networks (1D-CNN) and Multi-head attention model. In the next few sections, we briefly explain the above-mentioned models.

2.1. Extreme Gradient Boosting (XGBoost)

Extreme Gradient Boosting (XGBoost) (Chen & Guestrin, 2016) is an efficient extension of the Gradient boosting algorithm. Gradient boosting machine (GBM) is an ensemble of weak learner. GBM improves the overall model performance by training a new learner on top of a weak learner which predicts the errors made by the weak learner. This process is repeated an arbitrary number of times to improve the model performance. XGBoost utilizes the gradient boosting trees (gbtree) as the error predictor. Rather than optimizing the gbtree predictor in a conventional way, the XGBoost learns the gbtree to minimize the overall loss function while preventing the model from overfitting by adding an additional regularization term to the loss function. Furthermore, XGBoost uses an efficient optimization technique to improve the training speed and accuracy. Besides, the optimization algorithm allows parallelization at the feature level for faster computation.

[1]The code to generate events using C-MAPSS data is available at: https://github.com/Mahbubul-Alam-PhD/PHM-Society-2021

2.2. 1D Convolutional Neural Newtwork (1D-CNN)

Convolutional neural networks (CNNs) (Krizhevsky et al., 2012) are one of the most successful invention in the deep learning domain. The unique feature learning technique of CNN from raw image data in an end-to-end fashion crowned CNN as the *de facto* standard for computer vision applications. However, the presence of CNN time-dependent applications is relatively scarce. Yet, the 1-D variant of CNN (Ince et al., 2016) is successfully applied in applications involving text data, Electrocardiogram (ECG) data, vibration data etc. The working mechanism of 1D-CNN is similar to that of the 2D-CNN except the input data is a 1D vector and the convolution is applied in one direction. Nonetheless, understandably the computational complexity of 1D-CNN is significantly lower than the 2D CNN. As such, 1D-CNN model is suitable for real-time and small footprint device applications.

2.3. Long Short Term Memory (LSTM) Networks

LSTM (Gers et al., 1999) is one of the most popular recurrent neural network (RNN) architecture for handling time dependent and text data. The conventional RNN models suffer from vanishing/exploding gradient problem and fail to memorize long-term dependencies. LSTM alleviates these issues by introducing four carefully designed gate units in the architecture known as, input modulation gate, input gate, forget gate, and output gate. The combination of input and forget gate constitutes the cell state which functions as the long-term memory for the LSTM model. Due to the unique architectural advantages, the LSTM model has been applied in many different applications comprising time dependent and sequential data.

2.4. Multi-head Attention Model

Conventional RNN models such as LSTM handle time dependent data sequentially which is a major limiting factor to achieve parallel processing. Consequently, the multi-head attention model (Vaswani et al., 2017) is proposed to effectively process time dependent and sequential data. The multi-head attention model achieves parallel processing by removing the ordering of the time dependent data. However, the ordering information of time dependent data is a valuable feature. Therefore, the multi-head attention model captures the ordering information by feeding the same input multiple times to the "multiple heads" of the model and, hence the name multi-head. The special type of attention mechanism used in the multi-head attention model known as self-attention. The multi-head attention has shown state-of-the-art performance in various applications such as machine translation, document generation, biological sequence analysis etc.

3. METHODOLOGY

This section discusses the methods utilized to solve the event-based RUL estimation task. Firstly, we formally define the event-based RUL estimation problem. Secondly, we explain the data pre-processing techniques for the deep learning and machine learning models. Finally, we discuss our proposed deep learning and machine learning based models for solving event-based RUL estimation task.

3.1. Problem Definition

Let, $\boldsymbol{X} = [X_1, X_2, X_3, \dots, X_n]$ be an event dataset, where $X_i = [x_{i1}, x_{i2}, x_{i3}, \dots, x_{im}]$, where $i = 1, 2, 3 \dots, n$ is a data instance contains a collection of un-correlated sequential events obtained from a pool of equipment/asset. Each x_{ik} represents a specific event type. Please note that both repetition and reappearance of any event x_{ik} is allowed, i.e., x_{11}, x_{11}, x_{12} and x_{11}, x_{12}, x_{11} are both valid sequence of events. Also, events from any instance of $\boldsymbol{X}$ may appear in other instances(s), i.e., x_{11}, x_{21}, x_{32} is a valid sequence. These sequential events may appear at any point of time without following any specific pattern or periodicity. Example events may include fault codes, error codes or any predefined codes which carry a meaning for that event. A collection of these codes collected from different equipment in the same domain and organized in a historical fashion form an event dataset. Such dataset may be obtained from a database that collects all the information captured by the device attached to the equipment. Additionally, each event x_{ik} may contain optional information o_{ik} which appear along with the event. Therefore, by definition o_{ik} is sequential and expressed as $O_i = [o_{i1}, o_{i2}, o_{i3}, \dots, o_{im}]$ where o_{ik} may be a single value or a collection of multiple values tied to the event. Example optional information may include the time of the event occurrence, the part number which is affected by the event etc. Moreover, each equipment/asset of interest may have some unique static attributes expressed as $\boldsymbol{C} = \{c_1, c_2, c_3, \dots, c_q\}$. Example static attributes may include equipment manufacturer, model number, year, subcategory etc. Finally, the event-based prognostics problem can be defined as follows. Given, the input $[\boldsymbol{X}, \boldsymbol{O}, \boldsymbol{C}]$ for an equipment, estimate failure time of the equipment $\boldsymbol{Y} = [Y_1, Y_2, Y_3, \dots, Y_n]$ where, $Y_i = \{y_{i1}, y_{i2}, y_{i3}, \dots, y_{im}\}$ and $y_{i1} \geq y_{i2} \geq y_{i3} \geq \dots \geq y_{im}$. The following equation formally defines the event-based prognostics problem,

$$\boldsymbol{Y} = f(\boldsymbol{X}, \boldsymbol{O}, \boldsymbol{C}) \tag{1}$$

where, f is a function that performs prognostics.

3.2. Data Pre-processing and Feature Engineering

From a machine learning context, the event-based prognostics problem can be posed as either regression or classification task. Accordingly, we process the input and output data to fit the regression or classification problem as shown in Figure. 1. The input data in our event-based prognostics problem formulation contains both sequential and static variables. These variables can be represented by either numerical or categorical values. Numerical values are

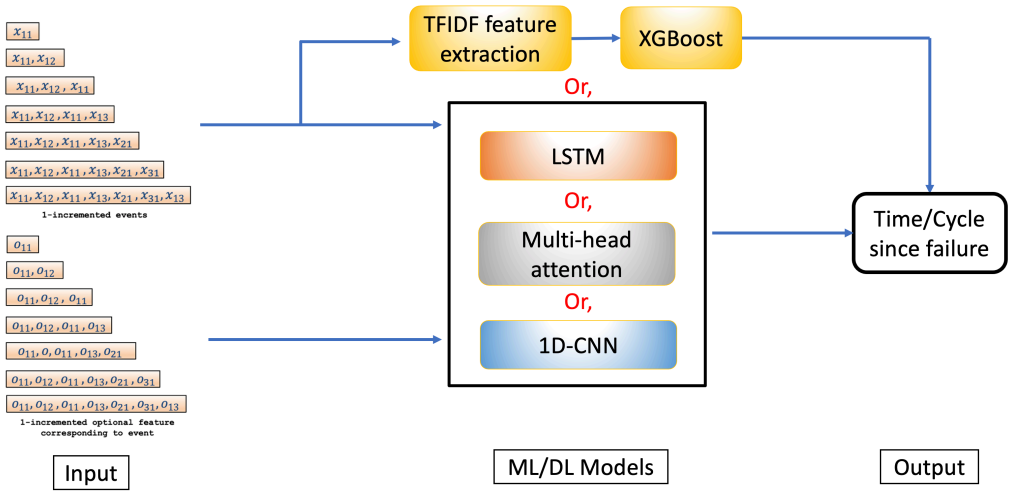


Figure. 1. Machine Learning and Deep Learning models for RUL estimation using event data.

processed with optional normalization technique. Categorical values are processed using appropriate category to numeric mapping technique. Besides, we convert the sequential event data in a "1-step increment" fashion and consider the corresponding target value of the last event in the 1-incremented sequence as the output. This transformation of the input converts the many-to-many mapping problem to a many-to-one problem. Additionally, the number of instances increase significantly which is beneficial when limited training samples are present. Optional pre-processing of the event sequences may be applied by keeping only the first appearance of an event ignoring the consecutive repetition of that event. This optional repeated event occurrence dropping step depends on the application and may require domain expert confirmation. For example, for a sequence x_{11}, x_{11}, x_{12} we convert it to x_{11}, x_{12}. However, for x_{11}, x_{12}, x_{11}, we do not make any changes as the repetition of x_{11} is not consecutive. Subsequently, we remove the target/output of the corresponding repeated event. In our example, when we convert the input x_{11}, x_{11}, x_{12} to x_{11}, x_{12}, we also remove the corresponding output of the second x_{11}, i.e., the output y_1, y_2, y_3 becomes y_1, y_3. Figure. 1 further illustrates the input and output data pre-processing for the event-based prognostics task for the following example which is an instance from one row of $\boldsymbol{X}$: input: $[(x_{11}, x_{11}, x_{12}, x_{11}, x_{13}, x_{13}, x_{21}, x_{31}, x_{13}),$ $(o_{11}, o_{11}, o_{12}, o_{11}, o_{13}, o_{13}, o_{21}, o_{31}, o_{13}),$ $(c_1, c_2, c_3)]$ and target/output: $[(y_{11}, y_{12}, y_{13}, y_{14}, y_{15}, y_{16}, y_{17}, y_{18}, y_{19})]$.

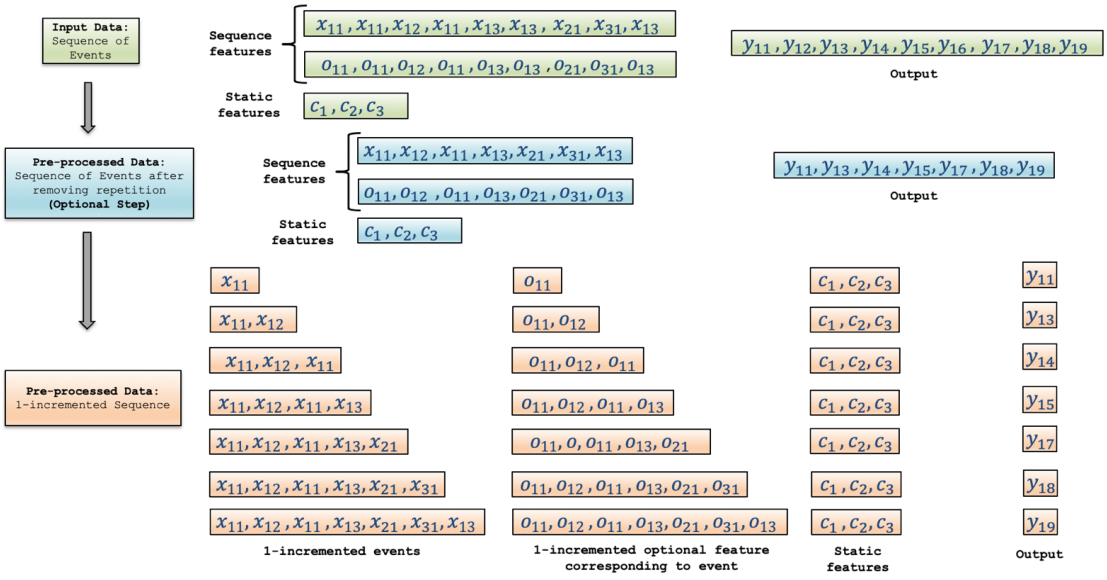


Figure. 2. Data pre-processing for event-based prognostics.

In this paper, we consider the sequence of events and corresponding continuous features obtained from the C-MAPSS data as input to the machine learning and deep learning models. The static features are absent for the C-MAPSS event data. Further details regarding the event data conversion process of the C-MAPSS data is provided in Section 0.1. Furthermore, a concrete example of the input $\boldsymbol{X, O}$ and the output $\boldsymbol{Y}$ is provided in Section 4.1.

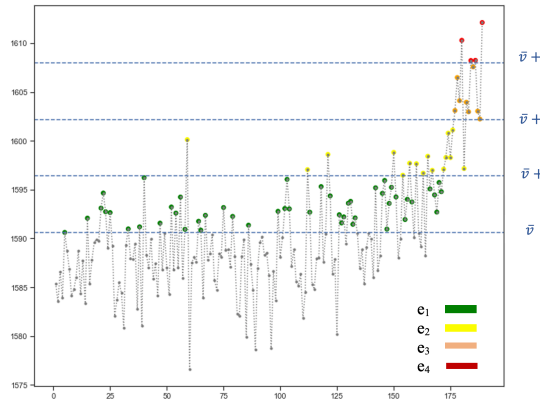
3.3. Machine Learning and Deep Learning Models for Event-based RUL Estimation

A high-level generalized flow diagram of the machine learning and deep learning based RUL estimation model is shown in Figure. 1. In this study, we consider three state-of-the-art deep learning models designed for sequential data, LSTM, Multi-head attention and 1D-CNN. The inputs to the deep learning models are the 1-incremented sequence of events and the corresponding continuous features. First, the categorical event values are converted to fixed length compact vectors. We treat the categorical event values as words in natural language, and hence, the fixed length vector conversion is performed using Word2Vec. Next, the corresponding continuous features for each event are concatenated to the fixed length vector. The final concatenated features are provided as input to the deep learning models. The loss function for the deep learning models is the root mean squared error (RMSE) i.e., we obtain the RMS difference between the network output and the target remaining useful life value. Subsequently, the models are trained by minimizing the RMSE loss using Adaptive Moment estimation (ADAM) (Kingma & Ba, 2014) optimizer. The output of the deep learning models is the continuous remaining useful life value. Additionally, we use a state-of-the-art machine learning model, XGBoost to perform the event-based RUL estimation task. The input data processing for the XGBoost model is different than that of the deep learning models. Only 1-incremented sequence of events are considered as the input and converted to fixed size TFIDF feature vectors. These features are used to train the XGBoost model. The output of the XGBoost is the same as the deep learning models, continuous remaining useful life value.

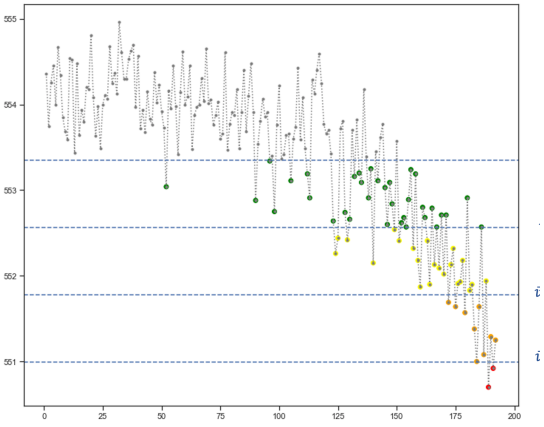
4. Experiments and Results

4.1. Event Data Generation

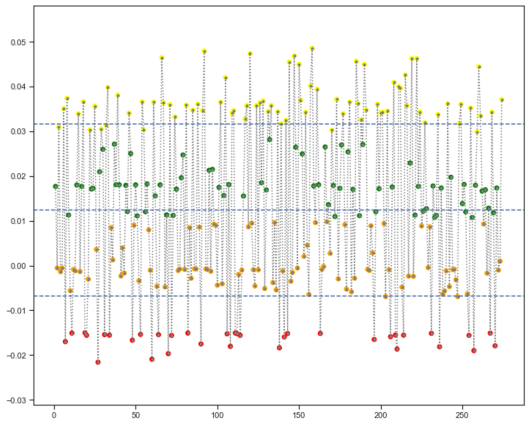
This section describes the event data generation process using the popular NASA C-MAPSS (Commercial Modular Aero-Propulsion System Simulation) dataset (Frederick et al., 2007). This dataset is widely used benchmark data for the RUL estimation task. The data has 4 subsets, FD001, FD002, FD003, and FD004 with different number of operating conditions and fault conditions. Specifically, the FD002 and FD004 datasets have six operating conditions which makes the RUL estimation task unattainable using the raw sensor time series only without incorporating information about



(a)



(b)



(c)

Figure. 3. (a) shows generated events for a signal with positive trend, (b) shows generated events for a signal with negative trend. The equipment starts in a normal state until a failure happened in cycle 193 and (c) shows generated events for a signal without an

operating conditions. In order to achieve that, apply a normalization technique to the FD002 and FD004 datasets

which removes the effects of operating conditions on the sensor data and discovers hidden trends in the sensor time series (Q. Wang et al., 2019). Therefore, generating events from such time series data is a complicated process. It involves proper understanding of the data and the view of a domain expert to identify the critical events in the time series. Furthermore, the translation from continuous to discrete data is crucial in generating meaningful events which has a significant impact on the performance of failure prognostics algorithms. Threshold discretization is one of the simplest methods to discretized real-valued raw signals into a finite number of discrete events. Threshold discretization converts a real-valued vector V = (v_1, v_2, …, v_N) to a symbolic-valued vector E = (e_1, …, e_M) where M is usually a small number, called discretization degree. We define a set of thresholds (cut points) that assign a symbolic value to some of the real-valued data points in the vector, V. This discretization process mainly depends on the choice of the cut points.

One way to generate meaningful events from continuous real-valued vector is to consider potential trends in the raw signals. From engineering perspective, a fault-related event might have happened in the equipment when the signal value deviates from normal range (either decreases or increases). We define multiple cut points based on mean and standard deviation of each signal defined as follows.

$$\bar{v} = \frac{1}{N}\sum_{i=1}^{N} v_i \tag{2}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(v_i - \bar{v})^2} \tag{3}$$

where, V = (v_1, v_2, …, v_N) represents a real-valued vector. If the signal values have an obvious trend as the component gets closer to a failure, the trend property can be used to define meaningful cut points. A signal might have positive, negative, or no trend. The followings are the rules for defining cut points:

- For positive-trend signals, we define the following events based on multiple cut point derived from mean and standard variation.

$$\begin{array}{lll} \forall v_i & if\ \bar{v} \le v_i < \bar{v} + \sigma & Then\ e_1 \\ \forall v_i & if\ \bar{v} + \sigma \le v_i < \bar{v} + 2\sigma & Then\ e_2 \\ \forall v_i & if\ \bar{v} + 2\sigma \le v_i < \bar{v} + 3\sigma & Then\ e_3 \\ \forall v_i & if\ v_i \ge \bar{v} + 3\sigma & Then\ e_4 \end{array} \tag{4}$$

- For negative-trend signals, we define the following events.

$$\begin{array}{lll} \forall v_i & if\ \bar{v} - \sigma \le v_i < \bar{v} & Then\ e_1 \\ \forall v_i & if\ \bar{v} - 2\sigma \le v_i < \bar{v}\,\sigma & Then\ e_2 \\ \forall v_i & if\ \bar{v} - 3\sigma \le v_i < \bar{v}\,2\sigma & Then\ e_3 \\ \forall v_i & if\ v_i < \bar{v} - 3\sigma & Then\ e_4 \end{array} \tag{5}$$

- For a signal without an obvious trend, we define the following events.

$$\begin{array}{lll} \forall v_i & if\ \bar{v} \le v_i < \bar{v} + \sigma & Then\ e_1 \\ \forall v_i & if\ \bar{v} + \sigma \le v_i & Then\ e_2 \\ \forall v_i & if\ \bar{v} - \sigma \le v_i < \bar{v} & Then\ e_3 \\ \forall v_i & if\ v_i \ge \bar{v} - \sigma & Then\ e_4 \end{array} \tag{6}$$

Figure. 3 (a), (b) and (c) demonstrates the event generation procedure for a signal with positive, negative and without an obvious trend, respectively.

Following the above procedure mentioned above, an example of the input $\boldsymbol{X}, \boldsymbol{O}$ and the output $\boldsymbol{Y}$ is as follows, $\boldsymbol{X}$ = *[['e11$_1$', 'e8$_2$', 'e11$_1$', 'e4$_1$', 'e11$_1$', 'e7$_1$'], ['e13$_1$', 'e12$_1$', 'e2$_1$', 'e12$_1$', 'e2$_1$', 'e21$_1$', 'e12$_1$']]* where, "N" represents the identifier for the event eN_j and *"j"* represents *1, 2, 3, 4* in Equation (4)-(6); $\boldsymbol{O}$ = *[[0, 0, 2, 0, 4, 0], [0, 0, 0, 2, 2, 0, 5]]* where each integer represents the number of cycles elapsed since an event first appeared; $\boldsymbol{Y}$ = *[[158, 158, 156, 152, 150, 151], [157, 155, 155, 155, 154, 153, 150]]* where each integer represents the number of cycles remaining (RUL) to failure when an event appears which is the target/ground truth of the RUL estimation problem. However, in many practical industrial applications the target value, $\boldsymbol{Y}$ is unavailable. In such cases, the target value may be estimated by observing the data (data centric approach), using the knowledge of a domain expert, and/or designing a physics model of the component.

4.2. Performance Evaluation

Traditionally, in a RUL estimation task the RUL values decreases linearly which reflects the degradation of the system/component over time. However, in practical applications the degradation is insignificant at the beginning and increases towards the end of life of the system/component. As such, we model the RUL changes over time using a piece-wise linear target function (Babu et al., 2016) as shown in Figure. 4. Figure. 4 demonstrates that the maximum RUL value is fixed to a constant value and the linear degradation starts after the system/component is used up to a certain degree. For C-MAPSS dataset we set the maximum limit to 130 (Q. Wang et al., 2019). The performance of the proposed event-based RUL regression models is measured using two widely used performance metric, root mean squared error (RMSE) and mean absolute error (MAE) as follows.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{7}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{8}$$

where, y denotes the true RUL value and $\hat{y}$ denotes the estimated RUL value by the model and N indicates the number of test samples.

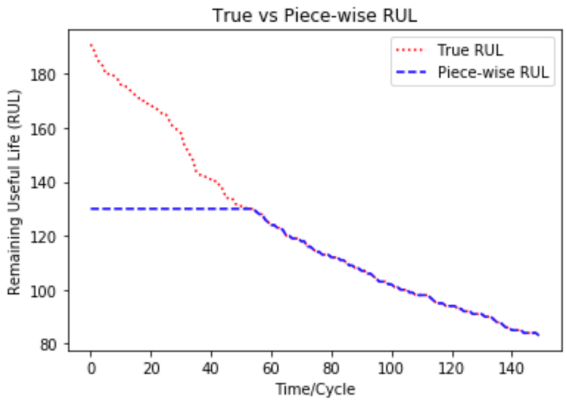


Figure. 4. True vs Piece-wise RUL of C-MAPSS data. Piece-wise maximum RUL is 130 cycles.

Additionally, we use a score-based evaluation metric to compare the proposed RUL estimation models as follows (Q. Wang et al., 2019).

$$Score = \begin{cases} \sum_{i=1}^{N} \left(e^{-\frac{(\hat{y}-y)}{13}} - 1 \right), (\hat{y} - y) < 0 \\ \sum_{i=1}^{N} \left(e^{-\frac{(\hat{y}-y)}{10}} - 1 \right), (\hat{y} - y) \geq 0 \end{cases} \quad (9)$$

where, y denotes the true RUL value and $\hat{y}$ denotes the estimated RUL value by the model and N indicates the number of test samples. The significance of the score function over the RMSE or MAE is that the score function tends to penalize the estimated RUL values that are larger than the true RUL value whereas, the RMSE/MAE metric treats the larger and smaller estimated RUL values equally. From a practical standpoint, the estimated RUL value smaller than the true RUL value is more acceptable than the larger estimated RUL value which indicates that the estimated RUL value is after the component failure. Please note that the smaller RMSE, MAE and score function values indicate better accuracies according to the definition of the metrics shown in Equation (7) – (9).

4.3. Results and Analysis

We use four state-of-the-art deep learning and machine learning algorithms, LSTM, Multi-head attention, 1D-CNN, and XGBoost for solving the event-based RUL estimation task. The input data is pre-processed following the techniques mentioned in Section 3.2. and the RUL estimation task is performed using the deep learning and machine learning models explained in Section 3.3. Essentially, the input to the deep learning and machine learning models are sequence of events and the corresponding attributes associated with the events. An important factor for the models to capture meaningful information from the sequence of events is to select an upper bound on the number of events in the sequences. The number of events in the sequences may vary from 1 to 5000. The events that appear at the beginning of a very long sequence may not have meaningful impact on the component failure. Accordingly, we run experiments to determine the optimal number of maximum events to achieve the best performance. Particularly, we run the multi-head attention model for different sequence lengths. Figure. 5 shows the effect of sequence on the performance of the multi-head attention model in terms of RMSE. Figure. 5 demonstrates that the multi-head attention model achieves the lowest RMSE for maximum sequence length 10. The model performance deteriorates for sequence length larger or smaller than 10. This in turn suggests that the last few events contain meaningful information regarding the component failure. Therefore, we use the maximum sequence length of 10 for the subsequent experiments.

Table 1 shows the performance comparison of four state-of-the-art deep learning and machine learning models, LSTM, Multi-head attention, 1D-CNN, and XGBoost in terms of RMSE, MAE and Score function for solving the event-based RUL estimation task using all four subsets of the C-MAPSS data, FD001, FD002, FD003, and FD004. The hyperparameters of the machine learning and deep learning models are selected using random search method (Bergstra & Bengio, 2012). Table 1 illustrates that the 1D-CNN deep learning model achieves better RMSE, MAE and Score compared to that of the LSTM, Multi-head attention and XGBoost models except for the FD001 dataset. The LSTM model achieves better RMSE and MAE value compared to the Multi-head attention and 1D-CNN models for the FD001 dataset. The Multi-head attention model achieves better score value for the FD001 dataset. Furthermore, Table 1 demonstrates that the XGBoost model achieves significantly lower performance compared to all the deep learning models in terms of RMSE, MAE and Score. This may be due to the TFIDF feature extraction step performed on the sequence data before feeding to the XGBoost model. TFIDF feature extraction process loses the ordering information of the events which may be necessary to perform the correct RUL estimation. Moreover, the additional features related to the

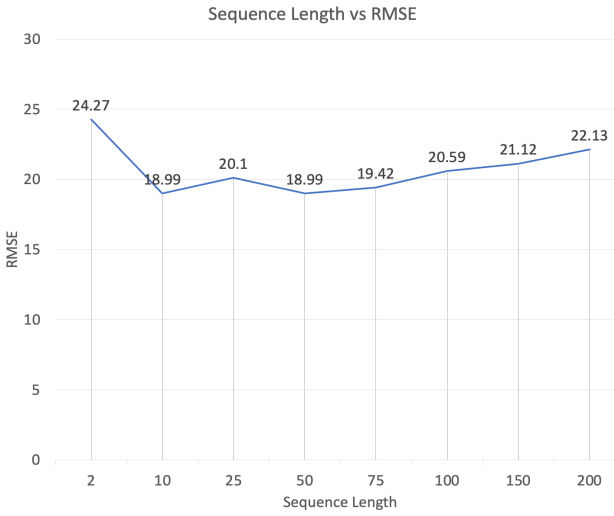


Figure. 5. Effect of sequence length on the performance (RMSE). The plot is generated by running the multi-head attention deep learning model for different sequence lengths.

Table 1. Performance comparison of 4 state-of-the-art deep learning and machine learning models in terms of RMSE, MAE and SCORE function for solving the Event-based RUL estimation task using c-mapss data (FD001-FD004). For each dataset the best values are shown using bold fonts and the second best values are shown using italic fonts.

Metrics	Models	Dataset			
		FD001	**FD002**	**FD003**	**FD004**
RMSE	**1D-CNN**	19.77	**28.87**	**20.99**	**30.13**
	LSTM	**18.70**	*29.60*	*21.58*	*30.63*
	Multi-head attention	*18.99*	30.49	22.21	32.00
	XGBoost	30.78	40.52	50.64	43.56
MAE	**1D-CNN**	*14.31*	**20.47**	**16.17**	**23.51**
	LSTM	**13.52**	*20.76*	*16.46*	*23.83*
	Multi-head attention	14.51	22.17	17.87	25.57
	XGBoost	25.23	31.61	39.17	35.94
Score	**1D-CNN**	*1.48E+03*	**1.09E+04**	**1.61E+03**	**7.54E+03**
	Multi-head attention	**7.02E+02**	*1.41E+04*	*1.88E+03*	8.64E+03
	LSTM	9.09E+02	1.69E+04	2.07E+03	*8.63E+03*
	XGBoost	4.89E+03	9.83E+04	3.41E+05	5.66E+04

events are omitted for the XGBoost model. Consequently, the deep learning models exploit the underlying ordering information concealed in the sequence of events. Moreover, the deep learning models utilize the additional features computed for each event in the sequence.

In summary, our results suggest that the deep learning models, 1D-CNN, LSTM, and multi-head attention show similar RMSE, MAE and Score performance. Foreseeably, the XGBoost model achieve lower performance compared to the deep learning models since the XGBoost model fails to capture ordering information from the sequence of events. Additionally, the additional features associated with the events are absent for the XGBoost model.

5. Conclusion

This paper introduces a novel RUL estimation framework using sequential event data. Conventional RUL estimation methods rely on time continuous time dependent data where the data follows certain pattern. However, such time dependent data is difficult to obtain in many practical scenarios. Conversely, the event data is more common and easier to obtain. Nevertheless, handling event data is challenging due to the random nature of the event appearance, lack of periodicity and absence of specific patterns. Therefore, a sophisticated framework is essential to solve the RUL estimation task using event data. Consequently, this paper proposes an end-to-end event-based RUL estimation framework. The framework involves pre-processing of the input event data, meaningful feature extraction, and applying suitable machine learning and deep learning techniques to perform the RUL estimation task. The proposed pre-processing and feature extraction technique is carefully designed by extensive data analysis to efficiently handle the event data which we obtained by handling real life event data. Unfortunately, such event data is not publicly available, and hence, we use the widely used C-MAPSS data for RUL estimation and convert the continuous time series data to sequential events by thoroughly analyzing the C-MAPSS data. Afterwards, we utilize several deep learning and machine learning techniques for solving the RUL estimation task using the generated event data. Concisely, this paper proposes a novel and innovative framework for effectively handling the event-based RUL estimation task. In future, we plan to apply the proposed framework using more complex event data. Furthermore, we plan to generalize the framework to be applicable in various domains involving event data.

Referrences

Babu, G. S., Zhao, P., & Li, X.-L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. *International Conference on Database Systems for Advanced Applications*, 214–228.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*(2).

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–794.

Costello, J. J. A., West, G. M., & McArthur, S. D. J. (2017). Machine learning model for event-based prognostics in gas circulator condition monitoring. *IEEE Transactions on Reliability*, *66*(4), 1048–1057.

Frederick, D. K., DeCastro, J. A., & Litt, J. S. (2007). *User's guide for the commercial modular aero-propulsion*

system simulation (C-MAPSS).

Gao, Z., Cecati, C., & Ding, S. X. (2015). A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches. *IEEE Transactions on Industrial Electronics*, *62*(6), 3757–3767.

Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). *Learning to forget: Continual prediction with LSTM*.

Ghasemi, A., Yacout, S., & Ouali, M.-S. (2010). Parameter estimation methods for condition-based maintenance with indirect observations. *IEEE Transactions on Reliability*, *59*(2), 426–439.

Goebel, K., Daigle, M. J., Saxena, A., Roychoudhury, I., Sankararaman, S., & Celaya, J. R. (2017). *Prognostics: The science of making predictions*.

Ince, T., Kiranyaz, S., Eren, L., Askar, M., & Gabbouj, M. (2016). Real-time motor fault detection by 1-D convolutional neural networks. *IEEE Transactions on Industrial Electronics*, *63*(11), 7067–7075.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25*, 1097–1105.

Lim, P., Goh, C. K., Tan, K. C., & Dutta, P. (2014). *Estimation of remaining useful life based on switching Kalman filter neural network ensemble*. Rolls Royce Singapore Singapore Singapore.

Peel, L. (2008). Data driven prognostics using a Kalman filter ensemble of neural network models. *2008 International Conference on Prognostics and Health Management*, 1–6.

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*, 1–9.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *ArXiv Preprint ArXiv:1706.03762*.

Wang, Q., Zheng, S., Farahat, A., Serita, S., & Gupta, C. (2019). Remaining useful life estimation using functional data analysis. *2019 Ieee International Conference on Prognostics and Health Management (Icphm)*, 1–8.

Wang, T., Yu, J., Siegel, D., & Lee, J. (2008). A similarity-based prognostics approach for remaining useful life estimation of engineered systems. *2008 International Conference on Prognostics and Health Management*, 1–6.

Xu, M., Baraldi, P., Al-Dahidi, S., & Zio, E. (2020). Fault prognostics by an ensemble of Echo State Networks in presence of event based measurements. *Engineering Applications of Artificial Intelligence*, *87*, 103346.

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.

Mahbubul Alam is a senior research scientist at the Industrial AI Laboratory at Hitachi America, Ltd. R&D where he is working on developing innovative algorithms for solving real life practical problems in the industrial domain. He completed his Ph.D. from the Vision Lab of the Department of Electrical and Computer Engineering at the Old Dominion University (ODU), Norfolk, Virginia, USA in December 2018. His doctoral dissertation contributed developing novel machine learning, more specifically deep learning algorithms for solving complex computer vision problems. Mahbubul received his BS and MS degree in Computer Science and Engineering from Jahangirnagar University, Bangladesh in 2008 and 2011, respectively. He has secured highest grade in both BS and MS final examination. He was awarded "Gold Medal" for obtaining the highest Cumulative GPA in the whole University during his undergrad study. He has more than 30 high quality publications and more than 1000 google scholar citations. His research interests are machine learning, deep learning, computer vision, predictive maintenance, visual inspection and other interesting and challenging problems in the industrial domain.

Laleh Jalali is a senior Machine Learning scientist with an extensive background in applying advanced machine learning, deep learning, and data mining techniques in different domains. At Hitachi, Laleh has been involved in many customer co-creation projects where the team owns the end-to-end process, from research to production models. Her professional interests focus on event-based frameworks, predictive maintenance, natural language understanding, and healthcare analytics. She published key elements of her work in "Event Mining for Explanatory Modeling" book, published by ACM in collaboration with Morgan and Claypool. Before Hitachi, Laleh graduated with a PhD in Computer Science from the University of California, Irvine.

Dipanjan Ghosh is a Senior Research Scientist at the Industrial AI Lab, Hitachi America Ltd. (Research and Development), Santa Clara, CA. He earned his Ph.D. in Mechanical Engineering in 2017 from University at Buffalo. His research interest includes application of data-driven method in the industrial domain, specifically in prognostics.

Ahmed Farahat is a principal research scientist at the Industrial AI Laboratory at Hitachi America, Ltd. R&D where he is working on developing innovative algorithms for solving real-world problems in industrial and societal domains. Ahmed has more than fifteen years research experience in academic and industrial environments. Previously, Ahmed was a research scientist at the Big Data

Laboratory at Hitachi America, Ltd. R&D. He also worked as postdoctoral fellow at the University of Waterloo and a research engineer at IBM Egypt. He also holds a Ph.D. degree from the University of Waterloo and M.Sc. and B.Sc. degrees from Cairo University, all in Computer Engineering. Ahmed's research interests lie in the areas of machine learning and data mining, and their applications to industrial use cases. Ahmed co-invented more than 15 patents and co-authored more than 30 high-quality publications.

Chetan Gupta works in the field of Industrial AI. During his career, he has played multiple roles: from a pure researcher, to leading multiple research and development teams towards successful execution of AI projects, to a thought leader laying out a vison of the future. At Hitachi, he heads the Industrial AI Lab and co-leads the Global AI CoE. He is proud to be a part of one of the strongest teams in the area of Industrial AI. The team consisting of data scientists, architects and developers is building fundamental horizontal technologies for Industrial AI, building solutions for verticals such as mobility, manufacturing, mining, energy management systems, and opening new frontiers in industrial analytics. Chetan has over 100 papers and patents in the area of Industrial AI, data mining/machine learning, data stream systems, complex event processing, workload management, etc. Chetan has a Ph.D. in Mathematics and M.S. in Mathematical Computer Science and Chemical Engineering from University of Illinois, Chicago.