Fault-Type-Aware Remaining Useful Life Prediction of Aircraft Engines Using an Integrated Deep Learning Framework

Junwon Seo[1]

[1]*The 10th Fighter Wing, Republic of Korea Air Force, Suwon, Gyeonggi-Do, 16659, Republic of Korea*
ericseo30@gmail.com

ABSTRACT

Accurate Remaining Useful Life (RUL) prediction is essential for reducing maintenance costs and improving operational efficiency in high-value, complex systems such as aircraft engines. Data-driven approaches have emerged as a primary methodology in RUL estimation research, demonstrating significant improvements in performance. However, discrepancies in degradation trajectories across multiple failure modes can adversely affect the prediction accuracy. To address this challenge, this study proposes an integrated framework based on a Time Series K-Means – Bidirectional Long Short-Term Memory (TS K-Means–Bi-LSTM) to perform RUL prediction considering different failure modes. Specifically, Time Series K-Means Clustering (TS K-Means) is used to cluster time series data into latent failure-mode groups, and a Bidirectional Long Short-Term Memory (Bi-LSTM) network is subsequently employed to predict the RUL for each group. The proposed framework is validated using the Commercial Modular Aero-Propulsion System Simulation dataset provided by NASA. Experimental results show that the proposed model outperforms existing methods. In addition, it achieves better results than the Bi-LSTM baseline trained under the same conditions but without fault type separation. This improvement likely results from minimizing interference among degradation patterns, allowing the model to better distinguish the unique behaviors associated with each fault type. Consequently, the proposed approach demonstrates strong potential for practical RUL prediction tasks.

1. INTRODUCTION

In response to the growing complexity of system operations and increasing maintenance demands in modern industries, Prognostics and Health Management (PHM) has gained significant attention. PHM enables optimized maintenance decisions through real-time health assessment and failure prognosis, serving as a cornerstone technology for intelligent asset management (Cuesta et al., 2025). Maintenance strategies are generally categorized into three types: reactive maintenance, preventive maintenance, and predictive maintenance. Among these, predictive maintenance is considered a key enabler of the future-oriented maintenance paradigm, as it helps minimize unplanned downtime and reduce maintenance costs. Due to these advantages, it has been increasingly adopted in safety-critical domains such as aerospace, railway, and energy sectors (Asif et al., 2022).

The feasibility of predictive maintenance largely depends on the ability to accurately predict a system's RUL. RUL refers to the remaining operational time of a system from its current state until the point of failure. There are two principal approaches to RUL estimation: model-based and data-driven approaches.

The model-based approach constructs mathematical models that are grounded in the physical structure and dynamic behavior of the system. Bolander et al. (2009) introduced a Bayesian updating technique based on particle filters. This method continuously updated the degradation state by incorporating diagnostic information, thereby improving the accuracy of the model-based approach.

However, this approach faces inherent limitations in fully capturing all complex factors of real-world systems, potentially leading to inaccurate RUL predictions. On the other hand, data-driven methods have increasingly gained attention, as they can predict RUL without requiring prior knowledge of system physics. The development of such methods has been further accelerated by advancements in machine learning and deep learning technologies (Chao et al., 2020).

Several studies have proposed machine learning techniques for RUL estimation. Khelif et al. (2017) proposed a Support Vector Regression (SVR)-based method for predicting RUL directly from sensor data, without the need to explicitly estimate failure states or thresholds. Meanwhile, Adhikari et

https://doi.org/10.36001/IJPHM.2025.v16i2.4305

al. (2018) developed a data-driven framework integrating multiple machine learning techniques. Their framework employed One-Class Support Vector Machine (One-Class SVM) for anomaly detection, K-Nearest Neighbors (KNN) for fault classification, and both Relevance Vector Regression (RVR) and Autoregressive Integrated Moving Average (ARIMA) models for RUL prediction.

More recently, deep learning has shown great promise in RUL prediction tasks. Jayasinghe et al. (2019) proposed a temporal convolutional memory network, which extracts features from time-series sensor data using a Convolutional Neural Network (CNN) and captures both short- and long-term dependencies through Long Short-Term Memory (LSTM). This model demonstrated robust prediction performance on aircraft engine datasets. Wen et al. (2024) introduced a Temporal and Heterogeneous Graph Neural Network (THGNN) that simultaneously considers temporal dynamics and inter-sensor relationships. In addition, Zhang et al. (2022) proposed a model that combines a Bidirectional Gated Recurrent Unit (BiGRU) with a Temporal Self-Attention Mechanism (TSAM), validating its effectiveness on both aircraft engine and milling datasets.

In addition, several studies have demonstrated performance improvements through various feature engineering techniques applied during the Exploratory Data Analysis (EDA) stage. Hong et al. (2020) selected features that exhibit high correlation with RUL by utilizing the Pearson correlation coefficient and trained their model using only the selected features. They also employed Shapley Additive Explanations (SHAP) to identify the features that contributed most significantly to prediction accuracy. These findings suggest that incorporating relevant environmental conditions and appropriate parameters can enhance RUL prediction accuracy.

Operational systems often operate under dynamic conditions and are exposed to multiple simultaneous fault modes. These diverse characteristics directly influence the accuracy of RUL prediction, necessitating their explicit consideration during the EDA phase. However, to the best of my knowledge, only a limited number of studies have systematically partitioned or labeled the data according to specific fault modes. Most prior works have relied on single-model training across the entire dataset, without accounting for variations across fault types. Table 1 presents five representative fault modes in turbofan engines and the sensors associated with each fault type (Wang et al., 2023). Detailed descriptions of the sensors are provided in Chapter 3 of Hong et al. (2020). As summarized in Table 1, each fault type impacts multiple sensors at the same time. Such behavior has also been observed in real-world operations, where different failure modes lead to varying sensor patterns across subsystems (Luo, 2006). Ignoring these variations may introduce noise and conflicting patterns in model training, leading to suboptimal prediction outcomes. Therefore, fault-type-aware experimentation has the potential to mitigate this issue. Furthermore, as the problem of degradation pattern interference in RUL prediction exists across various industries, this approach can be extended to other sectors beyond the aviation industry.

Table 1. Fault modes in turbofan engines and sensors associated with each fault type.

Fault mode	Key associated sensors
HPC fault	- PS30 (total HPC outlet pressure) - NC (core speed)
LPC fault	- Nf (fan speed) - P2 (LPC inlet pressure) - T2 (LPC inlet temperature)
HPT fault	- T50 (exhaust temperature) - T24 (temperature before HPT)
LPT fault	- Nf - T50 - P30 (HPC outlet pressure)
Fan fault	- Nf - P15 (fan outlet pressure)

With a similar focus, Peng et al. (2024) applied the FC-AMSLSTM model exclusively to engine units exhibiting high-pressure compressor (HPC) degradation faults in the C-MAPSS FD004 dataset. To isolate HPC fault cases, a decline index was utilized, leveraging trends observed across specific sensor measurements such as bypass ratio, HPC outlet pressure, and fuel flow-to-pressure ratio. However, from a system-level perspective, engine components are interdependently linked, and actual faults often trigger complex, concurrent responses across multiple sensors, rather than being confined to a single sensor (Li et al., 2021). Such an approach may overlook the holistic behavior of the system and omit critical features necessary for accurate RUL prediction. Furthermore, during in-flight operation, explicit fault type labels are typically unavailable at the time of data acquisition. Engineers are often unaware of the actual fault type until physical inspection or maintenance is conducted. Given these uncertainties, approaches that depend on predefined fault labels are challenging to implement in field environments.

To address these issues, this study proposes a framework based on Time Series K-Means – Bidirectional Long Short-Term Memory (TS K-Means–Bi-LSTM), which segments input data into latent fault types and performs fault-type-specific RUL prediction. By addressing the interference among degradation patterns, this study offers a methodological innovation in aero-engine RUL prediction. The methods and contributions for the key steps of the study are as follows:

- **Optimal Degradation Zone Extraction for Enhanced Clustering Reliability:** A common degradation zone is extracted from both the training and testing datasets by selecting only the segments where degradation patterns are clearly observable. This method is based on the critical insight that distinctive fault patterns become prominent primarily during the latter stages of a component's life cycle. By focusing on these segments, it has the potential to enhance the reliability of clustering.

- **TS K-Means Based Knowledge-Guided Clustering Applicable to Real-World Operations:** The Time Series K-Means (TS K-Means) algorithm is used to cluster sequences into latent fault types. This approach takes degradation sequence data as input, performing clustering based on the progression of degradation. Rather than identifying explicit fault types, it categorizes data according to the underlying fault mode structures. Notably, it is effective in practical operational environments where the specific fault occurring is unknown during operation. Specifically, system-level Failure Mode, Effects & Criticality Analysis (FMECA) or Fault Tree Analysis (FTA) can provide in advance detailed information on component-level fault modes, including their failure characteristics, criticality, and specific impacts on embedded sensors. As shown in Table 1, fault modes tend to influence particular sensors while exerting negligible effects on unrelated ones. This selective influence results in common degradation patterns and directional trends that are unique to each fault type. Such prior knowledge can be leveraged to determine the number of clusters and to perform knowledge-guided clustering based on similarities in degradation progression.

- **Fault-Type-Specific Bi-LSTM Models for Improving RUL Prediction:** Separate Bidirectional Long Short-Term Memory (Bi-LSTM) models are trained for each fault type. This mitigates the interference effects common in heterogeneous degradation data, allowing the model to capture the unique bidirectional temporal dependencies within a specific degradation pattern. This approach aims to enhance RUL prediction accuracy by leveraging a steady stream of focused and consistent inputs.

The rest of the paper is organized as follows: Section 2 provides an overview of the dataset and experimental setup. Section 3 introduces the proposed method. The experimental procedure is detailed in Section 4. The experimental results and analysis are provided in Section 5. Finally, Section 6 concludes the paper and discusses directions for future work.

2. DATASET OVERVIEW AND EXPERIMENTAL SETUP

2.1. Dataset Description

The C-MAPSS dataset, developed by the NASA Prognostics Center of Excellence, is a widely used benchmark in the field of PHM. It was simulated based on a commercial turbofan engine with a thrust rating of 400 kN and was designed to reflect a variety of operating conditions and fault modes. The turbofan engine architecture is illustrated in Figure 1. The engine consists of key components such as the fan, low-pressure compressor (LPC), high-pressure compressor (HPC), shaft, combustion chamber, high-pressure turbine (HPT), and low-pressure turbine (LPT).

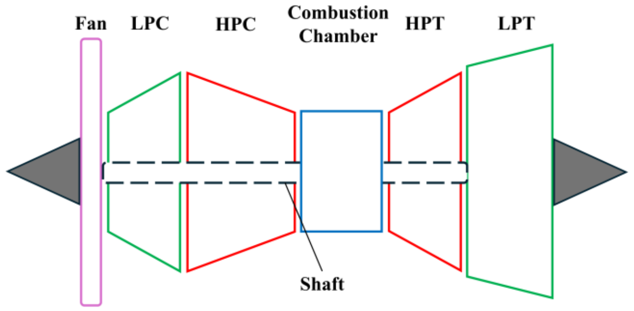

Figure 1. Architecture of the turbofan engine

The training set in the C-MAPSS dataset comprises run-to failure trajectories, where each unit starts from an arbitrary initial condition and operates until failure. In contrast, the testing set is only partially observed, with some sequences ending before the engine reaches complete failure. The full dataset is divided into four subsets FD001 through FD004, based on the combinations of fault modes and operational conditions. This study focuses on the FD004 subset, which represents the most complex configuration among the other subsets, with six operating conditions and two fault modes, as detailed in Table 2. It consists of 26 columns, including engine ID, cycle count, three operational conditions, and 21 sensor measurements. To better capture system-level degradation patterns, eighteen input features were selected, comprising the cycle count, three operational conditions, and fourteen sensor signals (sensor_2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20, and 21), all of which exhibited observable variation. Conversely, variables that are irrelevant or exhibited low variance were excluded from the input features.

Table 2. Summary of the FD004 dataset

Dataset	FD004
Operational conditions	6
Fault modes	2
Training engine units	249
Testing engine units	248

2.2. RUL Modeling and Data Filtering

In general, turbofan engines maintain stable health conditions during the early stage of operation, followed by a near-linear decline in RUL as degradation progresses (Berghout et al., 2020). This trend is effectively represented by a piecewise linear degradation model, as illustrated in Figure 2. To define the onset of degradation, the maximum RUL was set to 115 cycles. This threshold is not a fixed standard but an empirically determined parameter that can be adjusted based on dataset characteristics and modeling objectives. Similar approaches have been adopted in previous studies. For example, Zheng et al. (2017) set the RUL threshold at 125 cycles, while Asif et al. (2022) dynamically detected the onset of degradation and applied unit-specific maximum RUL values.

This study targets only the engine units exhibiting clear signs of degradation, as they are more informative for clustering based on fault-related behavior. As the C-MAPSS testing dataset does not follow a run-to-failure structure, some units remain in healthy condition throughout their recorded cycles. Including such units may introduce noise and hinder clustering performance. Therefore, testing units with RUL exceeding 115 cycles were excluded from evaluation, resulting in a final test set comprising 168 out of 248 units.

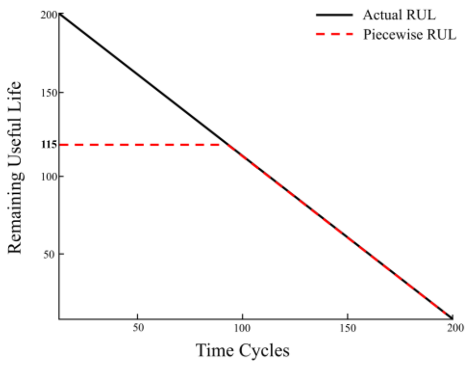

Figure 2. Piecewise linear degradation model

2.3. Common Degradation Zone

In the classification phase, this study focuses on specific segments where signs of degradation are more pronounced. Using the entire long-term time series as input can lead to overfitting issues due to noise, complexity, and non-stationarity. To address these challenges, a common practice is to extract key cycles or patches from the complete time series for the model's input. Wang et al. (2024) presented various case studies on utilizing only significant segments in deep learning model applications. The input data used for clustering reflected the observed trend in which sensor behavior becomes increasingly distinguishable as the system approaches failure, and this trend was also observed across multiple sensor measurements in this study. As illustrated in Figure 3, sensor_14 is presented as a representative example. Its values remain relatively constant in the early stages but exhibit a gradual increase as failure nears, followed by a sharp rise immediately prior to failure. Although this pattern is clearly observed in sensor_14, the magnitude and onset timing of variations vary depending on the sensor type and operating conditions. Furthermore, this approach is in line with prior research. Saeidi et al. (2019) segmented engine health into three classes based on RUL and defined the 50–0 RUL range as the 'class red', indicating urgent maintenance requirements. Similarly, Peng et al. (2024) reported that RUL prediction accuracy improves in the late degradation phase, attributing this effect to increased sensor responsiveness as the engine nears failure.

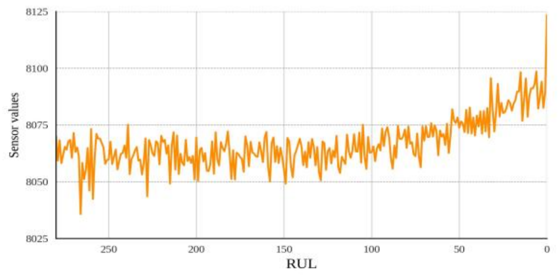

Figure 3. Variation of sensor_14 values over RUL

Motivated by these findings, this study focuses on a more informative portion of the degradation sequence, rather than using the entire timeline. A uniform 50-cycle degradation segment was extracted from the portion closest to the failure point. This choice was made to minimize potential bias in the classification results arising from variations in sequence length. In the training data, the RUL 50–0 interval was used, while in the testing data, which is truncated before failure, the last 50 observed cycles within the degradation phase were considered. Figure 4 illustrates the extraction process for the common degradation zone in both the training and testing datasets.

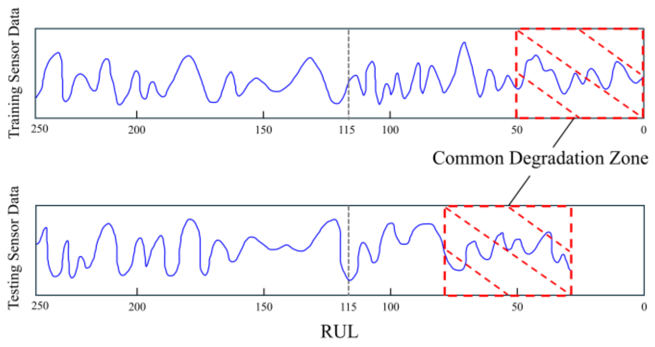

Figure 4. Extraction of the common degradation zone from the training and testing datasets.

3. PROPOSED METHOD

This study proposes a framework that combines TS K-Means clustering and Bi-LSTM regression. The overall process is divided into two phases: classification and regression, and this section provides a detailed explanation of the models and techniques employed in each phase.

3.1. Classification Phase

3.1.1. Data Normalization

In sensor data, input features vary in scale, which can bias model training by amplifying the influence of certain variables. Feature normalization is employed to rescale all features to a uniform range (Zheng et al., 2017). In the classification phase, Min-Max normalization is applied to transform each feature into the range [0, 1]. In Eq. (1), x denotes the original value, x_{min} and x_{max} represent the feature's minimum and maximum values, respectively. The resulting value X_{norm} is the normalized output.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

3.1.2. Dimensionality Reduction

High-dimensional input features can result in the curse of dimensionality, which hinders the model's generalization performance. To mitigate this issue and improve classification efficiency, dimensionality reduction is performed using an LSTM-Autoencoder. As shown in Figure 5, the encoder compresses the original 18-dimensional input into an 8-dimensional latent vector. Compared to standard methods such as PCA or t-SNE, the LSTM-Autoencoder is better suited for time-series data as it preserves temporal dependencies (Simpson, Dervilis, and Chatzi, 2021).

The encoder architecture consists of two LSTM layers followed by three dense layers. The LSTM layers contain 64 and 128 units, respectively, with a 20% dropout applied after each layer to reduce overfitting. The outputs are then passed through three dense layers with 32, 16, and 8 units. LeakyReLU activation functions are used to introduce nonlinearity, and batch normalization is applied after each dense layer to stabilize training. The model is trained using the Adam optimizer.

After dimensionality reduction, the resulting latent vectors are reshaped into 3D tensors of shape (number of samples, timesteps, number of latent dimension), where the timestep is set to 8. This configuration is determined empirically based on multiple experimental iterations.

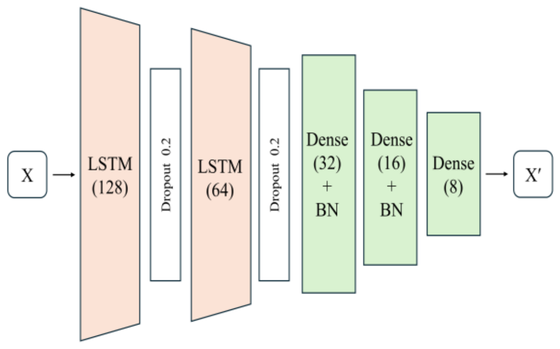

Figure 5. Architecture of the proposed LSTM-Autoencoder for dimensionality reduction

3.1.3. Time Series K-Means Clustering

In this study, a Soft-DTW-based TS K-Means clustering method is employed to effectively identify and group temporal fault patterns. Clustering is one of the techniques based on unsupervised learning that partitions data into groups based on similarities among data samples. Traditional K-Means clustering is an iterative algorithm that assigns each data point to the nearest centroid based on Euclidean distance and then updates the centroids until convergence. However, Euclidean distance is ill-suited for time series data, which often involves temporal shifts and distortions. As shown in Figure 6, even sequences with similar underlying patterns are incorrectly assessed under Euclidean distance when their time axes are not aligned, since it compares values at fixed time points. In contrast, Figure 7 illustrates how Dynamic Time Warping (DTW) addresses this limitation by flexibly aligning sequences through non-linear warping of the time axis (Holder, Bagnall, and Lines, 2024).

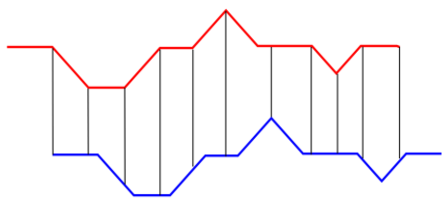

Figure 6. Distance measurement using Euclidean metric

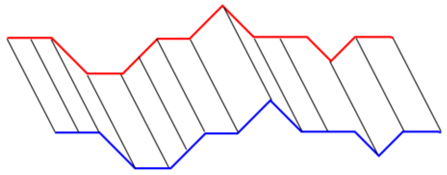

Figure 7. Distance measurement using DTW metric

DTW finds the optimal alignment path between two sequences via dynamic programming. Equation (2) defines the recursive formula used to compute the DTW distance. Specifically, at each step, the local distance $d(x_i, y_j)$ is added to the minimum cumulative cost among the three neighboring cells — $D(i-1,j)$, $D(i,j-1)$, $D(i-1,j-1)$. This recurrence enables DTW to accumulate the minimum alignment cost over the entire sequence and find the optimal warping path (Keogh and Ratanamahatana, 2005).

$$D(i,j) = d(x_i, y_j) + \min\{D(i-1,j), D(i,j-1), D(i-1,j-1)\} \qquad (2)$$

$$\mathrm{SoftDTW}_{(Q,C)} = -\lambda \cdot \log\left(\sum_{\pi \in \mathcal{P}} \exp\left(-\frac{1}{\lambda} \cdot \sum_{i,j \in \pi} d(Q_i, C_j)\right)\right) \qquad (3)$$

In this study, Soft-DTW is adopted as a differentiable extension of DTW. It employs a log-sum-exp formulation that smooths the minimum operator, making the distance function differentiable with respect to input sequences. As defined in Eq. (3), $d(Q, C)$ represents the local distance between sequence elements, and $\mathcal{P}$ denotes the set of all alignment paths. λ is a parameter that controls the smoothness of path selection. In Soft-DTW–based models, it regulates the smoothness of the soft-min operation, thereby influencing the differentiability of the loss function. As λ approaches zero, Soft-DTW behaves similarly to standard DTW, heavily weighting the shortest path. Conversely, as λ increases, the soft-min operation acts as an exponentially weighted average of all path costs, attenuating the differences between individual paths.

3.2. Regression Phase

3.2.1.Sliding Window

A sliding window technique is used to segment a continuous time series into fixed-length subsequences by moving a window across the data. This transformation converts the raw time series into a structure suitable for model training. It offers several advantages in time series modeling. By generating multiple overlapping samples, it enables the model to capture subtle temporal variations and recurring patterns (Al-Khazraji et al., 2022). In this study, the optimal window size was determined through repeated experiments using candidate values of 20, 30, 50, and 60. The selection of these candidates is guided by previous studies employing the sliding window technique, including Wang et al. (2023), Al-Khazraji et al. (2022), and Peng et al. (2024). As a result, the window size is set to 50 with a stride of 1, which minimizes information loss at boundaries and the final input data are organized into 3D tensors of shape (number of samples, window size, number of features).

3.2.2. Data Standardization

In the regression phase, each time series window is standardized using the TimeSeriesScalerMeanVariance. As a result, all windows are scaled to have zero mean and unit variance, enabling the model to learn from uniformly scaled inputs. This standardization is defined in Eq. (4), where $X_{i,t}$ denotes the value of feature i at timestep t, and μ_i, σ_i represent the mean and standard deviation of feature i within the window, respectively. The standardized value, denoted by $X'_{i,t}$, is used as the input.

$$X'_{i,t} = \frac{X_{i,t} - \mu_i}{\sigma_i} \qquad (4)$$

3.2.3. Bi – LSTM

Bi-LSTM is an extension of the LSTM architecture by processing sequential data in both forward and backward directions. Traditional LSTM networks incorporate a cell state and gating mechanisms to selectively retain relevant information. This structure effectively mitigates the vanishing gradient problem and manages long-range dependencies in time series data. While the LSTM processes information in a single forward direction—from past to present, Bi-LSTM incorporates both a forward path and a backward path to process input data. By integrating both directions, Bi-LSTM provides a more comprehensive understanding of data dependencies. This architecture has been widely adopted in various fields, including speech recognition and automated language translation (Wang et al., 1997). Equations (5) and (6) define the forward and backward paths, respectively, and Equation (7) represents the final output at timestep t obtained by combining both directional outputs. The structure of the Bi-LSTM network is illustrated in Figure 8.

$$h_t^{\rightarrow} = \mathrm{LSTM}(x_t, h_{t-1}^{\rightarrow}) \qquad (5)$$

$$h_t^{\leftarrow} = \mathrm{LSTM}(x_t, h_{t+1}^{\leftarrow}) \qquad (6)$$

$$y_t = \sigma(\overrightarrow{h}_t + \overleftarrow{h}_t) \qquad (7)$$

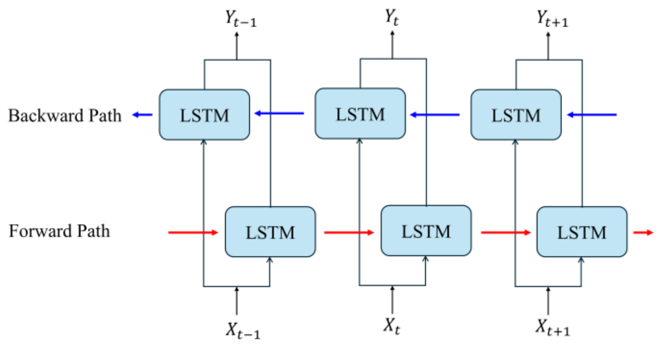

Figure 8. Architecture of the Bi-LSTM network

The proposed RUL prediction model consists of three Bi-LSTM layers followed by three dense layers. The Bi-LSTM layers contain 256, 128, and 64 units, respectively, and use the *tanh* activation function. The dense layers are composed of 128, 64, and 1 units, with the *ReLU* activation function applied to all but the final output layer, which generates the RUL values. To prevent overfitting and stabilize training, dropout and batch normalization are applied throughout the network. In this study, candidate dropout rates for each layer were set to the values [0.1, 0.2, 0.3]. A grid search is conducted to identify the optimal combination, resulting in a configuration of 0.3–0.2–0.2–0.2–0.2 across the five dropout layers. The structure of the Deep Bi-LSTM model is presented in Figure 9.

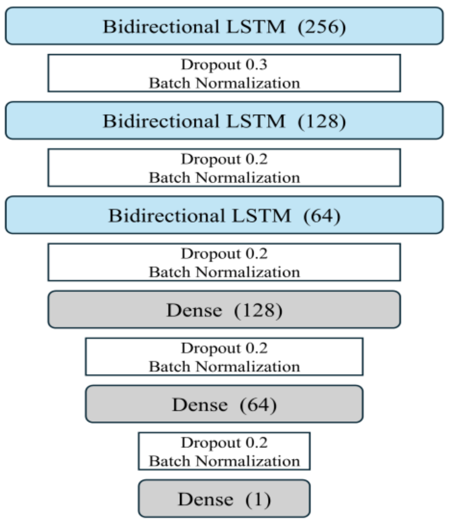

Figure 9. Structure of the proposed regression model

4. EXPERIMENTAL DETAILS

4.1. Overall Flowchart

The overall flowchart is presented in Figure 10, and the steps are summarized as follows:

1. To use only degradation progressing data, units with an RUL exceeding 115 are excluded from both the training and testing datasets. A common degradation zone of 50 timesteps is then extracted from each sequence.
2. Eighteen variables are selected as input features and scaled using Min-Max normalization. Dimensionality reduction is performed via LSTM-Autoencoder. Next, the resulting latent vectors are clustered into two fault types using the TS K-Means model. At this stage, as the specific fault modes corresponding to each cluster are inherently unknown in clustering, this study designated them as anonymous clusters, namely Fault 0 and Fault 1.
3. Fault type labels are assigned to the raw dataset. A sliding window approach is applied to convert the sequences into 3D tensors, and each window is standardized using TS-ScalerMeanVariance.
4. Finally, the Deep Bi-LSTM model is trained separately for each fault type. Model performance is evaluated using RMSE and Score metrics.

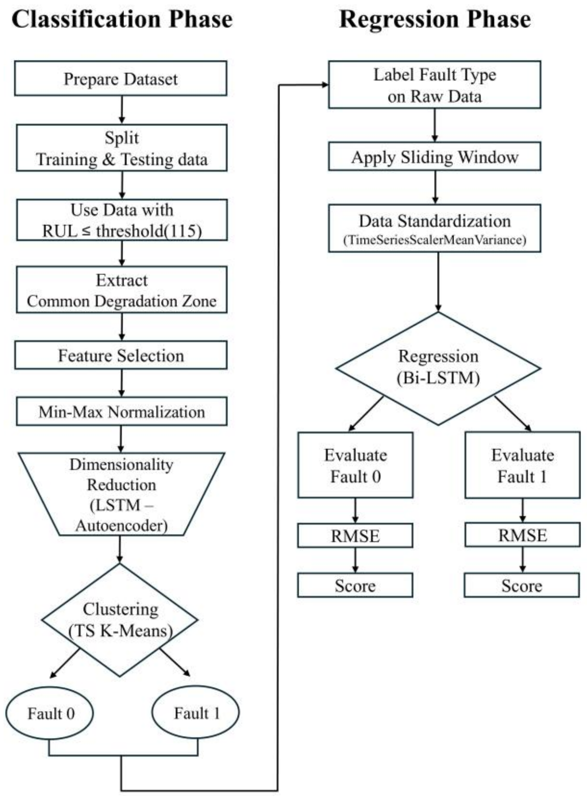

Figure 10. Overall flowchart of the proposed framework

4.2. Hyperparameter Setup

The hyperparameters used in this study are summarized in Tables 3 and 4. In the regression phase, the training dataset is split into 80% for training and 20% for validation. The optimal hyperparameter configuration for the models is determined through a systematic and iterative optimization process, prioritizing data characteristics and the model's training stability. The candidate ranges considered during the iterative hyperparameter tuning are as follows: window size ∈ {20, 30, 50, 60}, batch size ∈ {32, 64, 128, 256}, and learning rate decay factor ∈ {0.1, 0.2}. The initial learning rate is set to 1e-3 and is allowed to decay to 1e-7, with an early stopping criterion configured with a patience of 10 epochs. In addition, the smoothing parameter λ is evaluated over five candidate values {1e-3, 1e-2, 1e-1, 1, 10}, selected with reference to prior studies on Soft-DTW-based clustering, as discussed in Section 5.1.

Table 3. Hyperparameters for the TS K-Means model

Hyperparameter	Value
Number of clusters	2
Max iterations	100
Distance Function	Soft-DTW
λ	1

Table 4. Hyperparameters for the Deep Bi-LSTM model

Hyperparameter	Value
Number of hidden layers	6 (3, 3)
Number of neurons	256-128-64-128-64-1
Window size	50
Epochs	Early stopping (100)
Batch size	32
Learning rate	1e-3 → 1e-7
Learning rate decay factor	0.2
Loss function	Huber
Activation	Tanh(Bi-LSTM), ReLU(Dense)
Optimizer	Adam

4.3. Evaluation Metric

This study evaluates model performance using one metric in the classification phase and five in the regression phase. The Silhouette score is used to assess clustering quality. For each data point i, $a(i)$ denotes the average distance to all other points within the same cluster, while $b(i)$ represents the minimum average distance to points in the nearest neighboring cluster, as defined in Eq. (8). The score ranges from −1 to +1, with higher values indicating better clustering characterized by compact intra-cluster structure and well-separated clusters.

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{8}$$

For the regression phase, Root Mean Squared Error (RMSE) serves as the primary evaluation metric, offering an intuitive measure of prediction accuracy. In Eq. (9), D_i represents the prediction error between the predicted and actual RUL for unit i, and N is the total number of testing units. To summarize performance across the two fault types, the Mean RMSE is reported as a representative score to show overall performance. In both metrics, lower values indicate superior predictive accuracy, as formalized in Eq. (9) and (10) respectively.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(D_i)^2} \quad (D_i = p_i - a_i) \tag{9}$$

$$\text{Mean RMSE} = \frac{\text{RMSE}_{\text{fault0}} + \text{RMSE}_{\text{fault1}}}{2} \tag{10}$$

While RMSE is a useful metric for evaluating overall prediction accuracy, it does not differentiate between early and late predictions, which is critical in safety-sensitive operational environments. The Score metric addresses this limitation by employing an asymmetric loss function that penalizes late predictions ($D_i > 0$) more heavily than early ones ($D_i < 0$). This reflects the operational risk posed by delayed maintenance actions, leading to system failures or safety incidents. Overall model performance is evaluated using the Total Score, computed as the sum of individual scores from both fault types. In addition, an Average Score (AS) is introduced to account for different sample sizes across testing units. Lower values indicate better performance, and the definitions of Score, Total Score, and AS are provided in Eq. (11), (12), and (13), respectively.

$$\text{Score} = \sum_{i=1}^{N} \begin{cases} \exp\left(\dfrac{D_i}{10}\right) - 1 & \text{if } D_i \geq 0 \\ \exp\left(-\dfrac{D_i}{13}\right) - 1 & \text{if } D_i < 0 \end{cases} \tag{11}$$

$$\text{Total Score} = \text{Score}_{\text{fault0}} + \text{Score}_{\text{fault1}} \tag{12}$$

$$\text{AS} = \frac{\text{Score}_{(\text{Total Score})}}{N} \tag{13}$$

5. RESULTS AND DISCUSSION

This section is structured into three parts. First, the fault type clustering results obtained from the dataset are presented. Second, clustering performance is compared using three distance metrics—Euclidean, DTW, and Soft-DTW—to evaluate the influence of the distance metric on clustering quality. Third, based on the clustering results, the RUL prediction performance is analyzed and compared with previous studies. All experiments were conducted on Google Colab using an NVIDIA Tesla T4 GPU with 16GB of RAM, running Python 3.11.11 and TensorFlow 2.18.0.

Table 5. Clustering results on the C-MAPSS FD004 dataset

Data	Fault type (units)	Engine units	Silhouette score
Training dataset	Fault 0 (107)	1, 2, 3, 6, 9, 10, 11, 12, 14, 18, 20, 23, 26, 29, 30, 34, 35, 37, 42, 46, 47, 48, 49, 51, 52, 53, 55, 56, 58, 61, 62, 65, 69, 71, 72, 79, 80, 85, 86, 87, 88, 91, 94, 95, 98, 101, 108, 109, 111, 112, 116, 117, 118, 126, 127, 128, 131, 133, 134, 139, 144, 145, 146, 149, 151, 154, 155, 157, 158, 159, 161, 162, 163, 164, 166, 167, 169, 171, 172, 173, 174, 179, 180, 184, 188, 189, 190, 191, 200, 201, 203, 206, 207, 208, 215, 223, 225, 227, 228, 229, 230, 232, 233, 239, 241, 243, 244	0.34
	Fault 1 (142)	4, 5, 7, 8, 13, 15, 16, 17, 19, 21, 22, 24, 25, 27, 28, 31, 32, 33, 36, 38, 39, 40, 41, 43, 44, 45, 50, 54, 57, 59, 60, 63, 64, 66, 67, 68, 70, 73, 74, 75, 76, 77, 78, 81, 82, 83, 84, 89, 90, 92, 93, 96, 97, 99, 100, 102, 103, 104, 105, 106, 107, 110, 113, 114, 115, 119, 120, 121, 122, 123, 124, 125, 129, 130, 132, 135, 136, 137, 138, 140, 141, 142, 143, 147, 148, 150, 152, 153, 156, 160, 165, 168, 170, 175, 176, 177, 178, 181, 182, 183, 185, 186, 187, 192, 193, 194, 195, 196, 197, 198, 199, 202, 204, 205, 209, 210, 211, 212, 213, 214, 216, 217, 218, 219, 220, 221, 222, 224, 226, 231, 234, 235, 236, 237, 238, 240, 242, 245, 246, 247, 248, 249	
Testing dataset	Fault 0 (67)	8, 9, 12, 13, 18, 25, 29, 32, 35, 37, 40, 41, 44, 47, 49, 56, 61, 62, 64, 68, 71, 74, 86, 89, 99, 102, 105, 110, 111, 118, 119, 123, 124, 126, 131, 134, 135, 138, 143, 149, 150, 151, 161, 176, 178, 179, 180, 181, 185, 190, 192, 193, 194, 197, 202, 206, 211, 212, 213, 216, 223, 231, 235, 237, 240, 244, 248	0.53
	Fault 1 (101)	1, 2, 3, 4, 6, 7, 14, 15, 20, 21, 22, 23, 26, 27, 31, 36, 38, 42, 43, 45, 48, 50, 57, 58, 65, 67, 69, 70, 73, 77, 78, 79, 80, 81, 82, 85, 87, 88, 90, 91, 92, 96, 97, 98, 100, 101, 103, 106, 107, 112, 113, 114, 116, 117, 121, 142, 145, 146, 147, 152, 154, 155, 158, 159, 160, 162, 163, 165, 166, 167, 168, 172, 173, 174, 177, 183, 184, 187, 189, 196, 198, 199, 200, 201, 208, 209, 210, 214, 217, 219, 221, 222, 224, 226, 230, 233, 234, 236, 238, 242, 247	

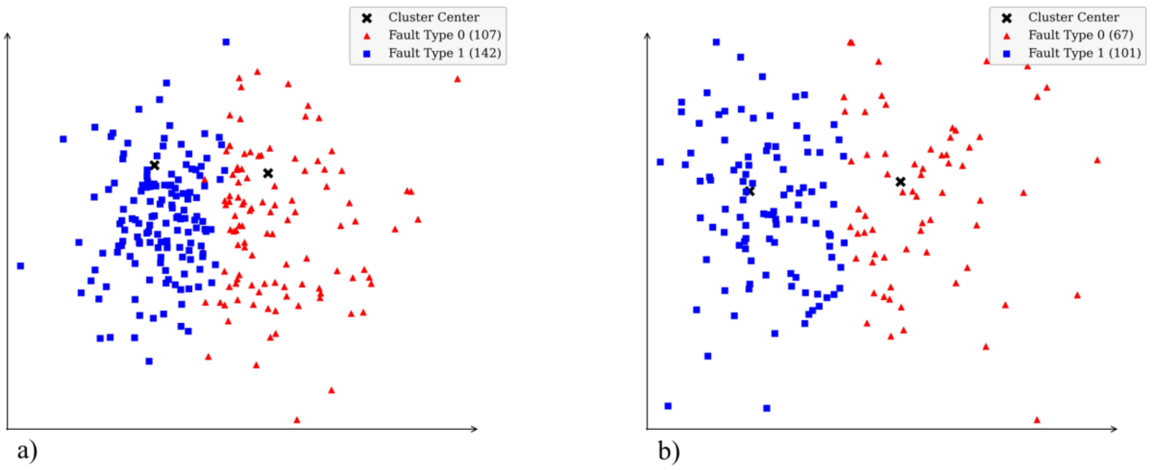

Figure 11. Clustered scatter plots of the C-MAPSS FD004 dataset: a) training dataset, b) testing dataset

5.1. Fault Type Clustering Results

The training and testing datasets are separately partitioned into two latent fault types using the knowledge-guided clustering approach. This simulates a blind operational scenario, in which the exact fault location and type are typically unknown. Table 5 summarizes the final clustering results. Of the 249 engine units in the training dataset, 107 were assigned to Fault 0 and 142 to Fault 1, with a Silhouette score of 0.34. In the testing dataset, 67 out of 168 units were classified as Fault 0 and 101 as Fault 1, yielding a Silhouette score of 0.53. Figure 11 presents scatter plots of the clustering results: (a) training dataset and (b) testing dataset. In both figures, the data are separated into two clusters with clearly visible boundaries. The training dataset exhibited a slightly lower score than the testing dataset. As shown in Figure 11(a), some samples are ambiguously located between clusters.

This may be attributed to the temporal position of the input segments within the degradation timeline. In particular, the training segments are primarily extracted from the final portion of the degradation trajectory, immediately preceding failure, where patterns tend to be more fluctuating and complex. This higher variability may reduce intra-cluster cohesion and result in a lower Silhouette score. In contrast, the testing segments do not extend to the point of failure, as data collection is terminated beforehand. These pre-failure degradation segments typically exhibit less abrupt and more stable patterns compared to those near the failure point, contributing to lower temporal complexity and better cluster separation.

In addition, the effect of the smoothing parameter λ on the results of Soft-DTW-based clustering is analyzed. Experiments are conducted on both training and testing datasets using five λ values: 0.001, 0.01, 0.1, 1, and 10, and the corresponding silhouette scores are recorded. The results are presented in Table 6. The search space for λ is defined with reference to Cuturi and Blondel (2017), as widely adopted in prior work on time-series clustering, where this parameter is typically set in powers of ten. Also, the maximum number of iterations is set to 100 to ensure convergence of the clustering process. Experimental results show that the optimal performance is achieved at $\lambda = 1$ for both the training and testing datasets. When λ is less than 1, the algorithm exhibits sensitivity to noise and local fluctuations, resulting in degraded performance. Conversely, values of λ greater than 1 likely cause oversmoothing, which diminishes the discriminative differences among individual time-series paths. These findings highlight the importance of tuning λ to achieve an optimal balance between filtering out noise and preserving the inherent features of the data.

Table 6. Silhouette scores on training and testing datasets for various λ values

Data	***λ =0.001***	***λ =0.01***	***λ =0.1***	***λ =1***	***λ =10***
Train	0.17	0.29	0.31	0.34	0.26
Test	0.24	0.31	0.26	0.53	0.4

5.2. Comparison of Clustering Results Using Different Distance Functions

A comparative analysis of clustering performance is conducted using three distance metrics: Euclidean, DTW, and Soft-DTW. Table 7 summarizes the resulting Silhouette scores and cluster distributions for each metric. Among them, Soft-DTW yielded the highest Silhouette scores — 0.34 for the training dataset and 0.53 for the testing dataset — outperforming the other two approaches. In contrast, Euclidean-based clustering resulted in the lowest scores. Because Euclidean distance relies on strict pointwise alignment at identical time indices, it cannot effectively capture nonlinear temporal variations arising from differences in sequence length or structure. DTW mitigates this issue by allowing flexible alignment through time warping. However, its non-differentiable nature makes it incompatible with centroid-based algorithms such as K-Means, which require iterative updates of cluster centers. Ultimately, Soft-DTW overcomes these limitations and is therefore well-suited for time-series clustering applications. Furthermore, this ablation study demonstrates that the Soft-DTW distance function is the most effective choice within the TS K-Means algorithm for clustering nonlinear time-series degradation patterns. This provides a practical guideline for selecting distance functions to achieve optimal performance in future research.

Table 7. Clustering performance of TS K-Means with different distance functions

Distance function	**Silhouette (Train)** **[fault0, fault1]**	**Silhouette (Test)** **[fault0, fault1]**
Euclidean	0.08 [118, 131]	0.11 [71, 97]
DTW	0.11 [114, 135]	0.17 [78, 90]
Soft-DTW	0.34 [107, 142]	0.53 [67, 101]

5.3. RUL Prediction Results

In this section, this study presents the RUL prediction results categorized by fault type. In addition, the results are compared with the Bi-LSTM model trained without fault type separation. Finally, the proposed model is compared with previous studies to assess its relative effectiveness.

Table 8 summarizes the RMSE and Score for each fault type, along with the overall performance in terms of Mean RMSE and Total Score. To ensure consistency of the results, five independent trials are conducted using random seeds of 32, 42, 52, 62, and 72. Results are reported as the mean ± standard deviation, where the mean indicates overall prediction accuracy and the standard deviation reflects model stability. The proposed model achieved RMSE values of 17.00 ± 0.83 for Fault 0 and 17.65 ± 0.22 for Fault 1, resulting in a Mean RMSE of 17.33 ± 0.61. The relatively small RMSE gap between fault types suggests that the model provides unbiased predictions across different degradation patterns. In terms of Score, Fault 0 recorded

398.4 ± 39.9 and Fault 1 recorded 817.2 ± 59.9, yielding a Total Score of 1215.6 ± 50.9.

Table 8. RUL prediction results by fault type

Metric	**Fault 0**	**Fault 1**
RMSE	17.00 ± 0.83	17.65 ± 0.22
Mean RMSE	17.33 ± 0.61	
Score	398.4 ± 39.9	817.2 ± 59.9
Total Score	1215.6 ± 50.9	

Subsequently, I evaluate the performance of the proposed method against a conventional approach in which the model is trained on the entire dataset without fault type separation. The same Bi-LSTM architecture, as described in Section 3.2.3, was employed in both cases. To ensure a consistent comparison, the evaluation is conducted on the 168 testing units introduced in Section 2.2, using identical hyperparameter settings and preprocessing procedures. As shown in Table 9, the conventional approach yielded RMSE of 19.04 ± 1.82, while the proposed approach achieved a lower RMSE of 17.33 ± 0.61, indicating an average improvement of 1.71 points. In terms of the AS, the conventional method recorded 11.20, whereas the proposed method achieved 7.24, representing an improvement of approximately 35%. These improvements can be attributed to the effect of fault type separation, which reduces interference among heterogeneous degradation patterns and thereby enhances prediction accuracy. Notably, the proposed model outperformed the conventional method in both metrics despite using fewer training samples, demonstrating the effectiveness of fault-type-aware modeling in time-series–based RUL prediction. This offers an effective strategy that can enhance RUL prediction accuracy in the presence of heterogeneous degradation patterns. Given that systems in operational environments often experience multiple concurrent failures, the proposed approach could be applicable across a wide range of industries, including manufacturing, energy, and transportation, not just aviation engines.

Table 9. Evaluation of RUL prediction performance with and without fault type separation

Methods	**Training units**	**RMSE (Mean RMSE)**	**AS**
Bi-LSTM (Conventional approach)	249	19.04 ± 1.82	11.20
TS K-Means – Bi-LSTM	107 (Fault 0) 142 (Fault 1)	17.33 ± 0.61	7.24

Finally, the proposed method is compared with previous studies based on the C-MAPSS FD004 dataset. Among the compared models, Deep LSTM, Bi-LSTM, LSTM Attention, AEQRNN, Double Attention-based Architecture, and ISG-McMsDCNN-LSTM were trained on the full dataset of 249 engine units without fault type separation. In contrast, the proposed method and FC-AMSLSTM were trained separately for each fault type. The proposed method achieved the best RMSE performance among all methods. Although its AS was slightly higher than that of the Double Attention-based architecture, the result remains competitive considering the reduced training data due to fault-type clustering. Notably, it outperformed FC-AMSLSTM, which was optimized specifically for HPC fault cases, thus demonstrating superior accuracy across two latent fault types. This demonstrates the model's robustness in real-world operational settings where the exact fault type is often not clearly identified. Detailed comparison results are presented in Table 10.

Table 10. Performance comparison between the proposed method and previous models

Methods	**Training units**	**RMSE**	**AS**
Deep LSTM (2017)	249	28.17	22.38
Bi-LSTM (2018)	249	24.86	21.89
LSTM Attention (2021)	249	27.08	22.78
AEQRNN (2022)	249	20.67	18.54
Double Attention based Architecture (2022)	249	19.86	7.02
ISG-McMsDCNN-LSTM (2024)	249	17.81	9.75
FC-AMSLSTM (2024)	111	19.48	8.42
TS K-Means – Bi-LSTM	107 / 142	17.33	7.24

6. CONCLUSION

This study proposes a framework that combines the TS K-Means clustering with the Deep Bi-LSTM regression model for RUL prediction. Given that sensor degradation patterns in aircraft engines can vary by fault types and potentially reduce prediction accuracy, this study reflects this characteristic in the deep learning process by first distinguishing fault modes and then performing fault-specific RUL prediction. While previous studies often overlooked the diversity of degradation behaviors, the proposed framework introduces a novel perspective by enabling degradation pattern separation, thus contributing a technical advancement to the PHM domain. Experimental results demonstrated that the proposed method achieved higher accuracy than the conventional approach, which does not incorporate fault type separation. In addition, it outperformed previously published models. These results suggest that incorporating fault type separation helps mitigate the effects of interference among different degradation patterns, resulting in more balanced and accurate learning. Furthermore, when fault types have been reasonably characterized in advance, such as through comprehensive FTA or FMECA, the proposed method—by leveraging knowledge-guided clustering to identify latent fault patterns—can be effectively applied to real-world scenarios where the exact fault is not identifiable during operation.

In this study, healthy data with RUL values beyond the threshold are excluded to focus on the degradation phase. Future research will aim to preserve such data and examine its contribution to model performance. In addition, the current framework adopts knowledge-guided clustering strategy. However, this approach can lead to clustering instability when data are scarce or the fault modes are diverse. To address this, semi-supervised clustering emerges as a promising solution. It allows a small amount of labeled data to serve as a guide, enabling simultaneous learning with large volumes of unlabeled data, thus contributing to more reliable clustering results. Furthermore, this method can leverage a few labeled data points to identify the specific fault type corresponding to each cluster, significantly enhancing the interpretability of the research outcomes. Building on these advantages, the future work will investigate applying semi-supervised clustering to various domain-specific datasets and analyzing its contributions.

REFERENCES

Adhikari, P., Rao, H. G., & Buderath, M. (2018, October). Machine learning based data driven diagnostics & prognostics framework for aircraft predictive maintenance. In *10th International Symposium on NDT in Aerospace* (pp. 1–15), October 24–26, Dresden, Germany. Retrieved from https://www.ndt.net/article/aero2018/papers/We.5.B.3.pdf

Al-Khazraji, H., Nasser, A. R., Hasan, A. M., Al Mhdawi, A. K., Al-Raweshidy, H., & Humaidi, A. J. (2022). Aircraft engines remaining useful life prediction based on a hybrid model of autoencoder and deep belief network. *IEEE Access*, *10*, 82156–82163. doi:10.1109/ACCESS.2022.3188681

Alomari, Y., Andó, M., & Baptista, M. L. (2023). Advancing aircraft engine RUL predictions: An interpretable integrated approach of feature engineering and aggregated feature importance. *Scientific Reports*, *13*, 13466. doi:10.1038/s41598-023-40315-1

Asif, O., Haider, S. A., Naqvi, S. R., Zaki, J. F. W., Kwak, K.-S., & Islam, S. M. R. (2022). A deep learning model for remaining useful life prediction of aircraft turbofan engine on C-MAPSS dataset. *IEEE Access*, *10*, 95425–95440. doi:10.1109/ACCESS.2022.3203406

Berghout, T., Mouss, L.-H., Kadri, O., Saïdi, L., & Benbouzid, M. (2020). Aircraft engines remaining useful life prediction with an improved online sequential extreme learning machine. *Applied Sciences*, *10*(3), 1062. doi:10.3390/app10031062

Bolander, N., Qiu, H., Eklund, N., Hindle, E., & Rosenfeld, T. (2009). Physics-based remaining useful life prediction for aircraft engine bearing prognosis. *Annual Conference of the PHM Society*, *1*(1). doi:10.36001/phmconf.2009.v1i1.1631

Chao, M. A., Kulkarni, C., Goebel, K., & Fink, O. (2022). Fusing physics-based and deep learning models for prognostics. *Reliability Engineering & System Safety*, *217*, 107961. doi:10.1016/j.ress.2021.107961

Chen, Z., Wu, M., Zhao, R., Guretno, F., Yan, R., & Li, X. (2020). Machine remaining useful life prediction via an attention-based deep learning approach. *IEEE Transactions on Industrial Electronics*, *68*(3), 2521–2531. doi:10.1109/TIE.2020.2972443

Cheng, Y., Hu, K., Wu, J., Zhu, H., & Shao, X. (2021). Autoencoder quasi-recurrent neural networks for remaining useful life prediction of engineering systems. *IEEE/ASME Transactions on Mechatronics*, *27*(2), 1081–1092. doi:10.1109/TMECH.2021.3079729

Cuesta, J., Leturiondo, U., Vidal, Y., & Pozo, F. (2025). A review of prognostics and health management techniques in wind energy. *Reliability Engineering & System Safety*, *260*, 111004. doi:10.1016/j.ress.2025.111004

Cuturi, M., & Blondel, M. (2017, August). Soft-DTW: a differentiable loss function for time-series. In *Proceedings of the 34th International Conference on Machine Learning (ICML)* (pp. 894–903), August 6–11, Sydney, Australia. PMLR. Retrieved from https://proceedings.mlr.press/v70/cuturi17a.html

Ensarioğlu, K., İnkaya, T., & Emel, E. (2023). Remaining useful life estimation of turbofan engines with deep learning using change-point detection based labeling and feature engineering. *Applied Sciences*, *13*(21), 11893. doi:10.3390/app132111893

Holder, C., Bagnall, A., & Lines, J. (2024). On time series clustering with k-means. *arXiv preprint*, arXiv:2410.14269. doi:10.48550/arXiv.2410.14269

Hong, C. W., Lee, C., Lee, K., Ko, M.-S., Kim, D. E., & Hur, K. (2020). Remaining useful life prognosis for turbofan engine using explainable deep neural networks with dimensionality reduction. *Sensors*, *20*(22), 6626. doi:10.3390/s20226626

Jayasinghe, L., Samarasinghe, T., Yuen, C., Low, J. C. N., & Ge, S. S. (2019, February). Temporal convolutional memory networks for remaining useful life estimation of industrial machinery. In *2019 IEEE International Conference on Industrial Technology (ICIT)* (pp. 915–920). IEEE. doi:10.1109/ICIT.2019.8754956

Keogh, E., & Ratanamahatana, C. A. (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, *7*, 358–386. doi:10.1007/s10115-004-0154-9

Khelif, R., Chebel-Morello, B., Malinowski, S., Laajili, E., Fnaiech, F., & Zerhouni, N. (2017). Direct remaining useful life estimation based on support vector regression. *IEEE Transactions on Industrial Electronics*, *64*(3), 2276–2285. doi:10.1109/TIE.2016.2623260

Li, X., Wang, L., Wang, C., Ma, X., Miao, B., Xu, D., & Cheng, R. (2024). A method for predicting remaining useful life using enhanced Savitzky–Golay filter and improved deep learning framework. *Scientific Reports*, *14*(1), 23983. doi:10.1038/s41598-024-74989-y

Li, X., Zhong, X., Shao, H., Han, T., & Shen, C. (2021). Multi-sensor gearbox fault diagnosis by using feature-fusion covariance matrix and multi-Riemannian kernel ridge regression. *Reliability Engineering & System Safety*, *216*, 108018. doi:10.1016/j.ress.2021.108018

Liu, L., Song, X., & Zhou, Z. (2022). Aircraft engine remaining useful life estimation via a double attention-based data-driven architecture. *Reliability Engineering & System Safety*, *221*, 108330. doi:10.1016/j.ress.2022.108330

Luo, M. (2006). Data-driven fault detection using trending analysis. Doctoral dissertation, Louisiana State University and Agricultural & Mechanical College, Baton Rouge, LA. Retrieved from https://www.proquest.com/openview/ace8248d483788bddeb4379c7426feee/1?cbl=18750&diss=y&pq-origsite=gscholar

Peng, Z., Wang, Q., Liu, Z., & He, R. (2024). Remaining useful life prediction for aircraft engines under high-pressure compressor degradation faults based on FC-AMSLSTM. *Aerospace*, *11*(4), 293. doi:10.3390/aerospace11040293

Shahapure, K. R., & Nicholas, C. (2020, October). Cluster quality analysis using silhouette score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)* (pp. 747–748), October 6–9, Sydney, Australia. IEEE. doi:10.1109/DSAA49011.2020.00096

Simpson, T., Dervilis, N., & Chatzi, E. (2021). Machine learning approach to model order reduction of nonlinear systems via autoencoder and LSTM networks. *Journal of Engineering Mechanics*, *147*(10), 04021061. doi:10.1061/(ASCE)EM.1943-7889.0001971

Wang, H., Zhang, Z., Li, X., Deng, X., & Jiang, W. (2023). Comprehensive dynamic structure graph neural network for aero-engine remaining useful life prediction. *IEEE Transactions on Instrumentation and Measurement*, *72*, 1–16. doi:10.1109/TIM.2023.3322481

Wang, J., Wen, G., Yang, S., & Liu, Y. (2018, October). Remaining useful life estimation in prognostics using deep bidirectional LSTM neural network. In *2018 Prognostics and System Health Management Conference (PHM-Chongqing)* (pp. 1037–1042), October 26–28, Chongqing, China. IEEE. doi:10.1109/PHM-Chongqing.2018.00184

Wang, Y., et al., 2024, "Deep time series models: A comprehensive survey and benchmark" *arXiv preprint arXiv:2407.13278*

Wen, Z., Fang, Y., Wei, P., Liu, F., Chen, Z., & Wu, M. (2025). Temporal and heterogeneous graph neural network for remaining useful life prediction. *IEEE Transactions on Neural Networks and Learning Systems*.

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. doi:10.1016/j.neucom.2017.05.063

Yun, Y., Kim, S., Cho, S., & Choi, J. (2019). Neural network based aircraft engine health management using C-MAPSS data. *Journal of Aerospace System Engineering*, 13(6), 17–25. doi:10.20910/JASE.2019.13.6.17

Zhang, J., Jiang, Y., Wu, S., Li, X., Luo, H., & Yin, S. (2022). Prediction of remaining useful life based on bidirectional gated recurrent unit with temporal self-attention mechanism. *Reliability Engineering & System Safety*, *221*, 108297. doi:10.1016/j.ress.2021.108297

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

BIOGRAPHIES

Junwon Seo received the bachelor's degree in system engineering from the Republic of Korea Air Force Academy, Cheongju, Korea, in 2022.

Currently, he is serving as an aircraft maintenance officer in the Republic of Korea Air Force. His research interests include reliability engineering, predictive maintenance, and artificial intelligence.