EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems
Research Article EAI.EU

A Novel Hybrid Transformer for RUL Prediction in Predictive Maintenance for Smart Manufacturing

Nguyen Huu Du[1], Nguyen Xuan Hoang[2], Dao Bich Thuong[3], Do Cong Ngon[3], Truong Thu Huong[3,*]

[1]Faculty of Mathematics and Informatics, Hanoi University of Science and Technology, Vietnam
[2]Queen's University of Belfast, United Kingdom
[3]School of Electrical and Electronic Engineering, Hanoi University of Science and Technology, Hanoi, Vietnam

Abstract

Ensuring continuous operation and minimizing unexpected failures are critical priorities in industrial manufacturing. Due to this requirement, smart manufacturing has transformed maintenance strategies, moving from traditional scheduled approaches to predictive maintenance (PdM), which leverages actual machine health conditions. A powerful technique that enables accurate PdM is Remaining Useful Life (RUL) estimation. Accurate RUL prediction allows the assessment of machine health, thereby facilitating timely and appropriate maintenance decisions to sustain continuous operation and reduce repair costs. While several existing models have been developed for RUL estimation, they often struggle to capture long-term dependencies in time series data, limiting their predictive accuracy. In this study, we propose a novel architecture that combines a one-dimensional Convolutional Neural Network (1D-CNN) with two consecutive transformer encoders enhanced by Fourier transforms to address these challenges. The performance of the proposed model was evaluated based on two well-known benchmark datasets, C-MAPSS and its improved version, N-CMAPSS. The experiment results show that our approach outperforms current state-of-the-art methods in both prediction accuracy and computational efficiency, demonstrating its potential for practical applications.

Received on 18 March 2026; accepted on 16 May 2026; published on 20 May 2026

Keywords: Remaining Useful Life, Bayesian optimization, Transformer, Fourier transform, Convolutional Neural network

doi:10.4108/eetinis.132.12274

1. Introduction

In modern industrial production, machinery has become the core of every manufacturing sector, from food processing and textiles to complex fields like high technology and energy. These systems encompass a wide range of equipment, including industrial robots, automated machinery, precision mechanical devices, and industrial automation systems. The development and application of this equipment do more than just optimize production processes; they also deliver significant benefits in productivity, product quality, error reduction, and cost savings. It can be said that these industrial systems are the backbone of modern manufacturing. Accordingly, ensuring their continuous and reliable operation is a top priority. Due to constant operational demands, these machines inevitably experience gradual degradation over time, which can lead to performance deterioration and unexpected failures. Such incidents may result in unpredictable consequences; they not only incur substantial costs and time for repairs, but also disrupt the entire production line, affecting the progress of product completion. Furthermore, a delay in detecting and addressing a malfunction can exacerbate the machine's condition, turning a minor issue into a major problem.

Previously, maintenance activities were often meticulously arranged in advance for a specific period of

*Corresponding author. Email: huong.truongthu@hust.edu.vn

time. However, these traditional methods had significant limitations. Machines could fail unexpectedly before scheduled maintenance, or, conversely, be serviced while still operating optimally, leading to inefficiency and waste. With the rapid advancements in Industry 4.0 nowadays, maintenance strategies have changed remarkably. It has become more efficient and precise through real-time monitoring of machine health status. Indeed, the use of advanced technologies, such as sensors, cameras, and the Internet of Things (IoT), has become increasingly prevalent in smart manufacturing. These technologies enable precise monitoring of critical machine health indicators, including vibration, temperature, pressure, surface imaging, and other essential parameters. Data derived are then transmitted to a centralized system, where analytical models and algorithms process them to assess the overall health status of machinery. This comprehensive evaluation supports the prediction of potential failures before they occur, enabling proactive maintenance actions. That is to say, maintenance has shifted from corrective and preventive methods to predictive maintenance (PdM) [1].

A key technique for PdM is the estimation of Remaining Useful Life (RUL), the time remaining until machine failure [2]. By estimating RUL, which indicates the anticipated timing of potential machine failures, maintenance can be strategically scheduled to prevent disruptions, optimize operational efficiency, and significantly reduce overall maintenance costs [3]. In the literature, many studies have been conducted to provide accurate RUL prediction. In general, these methods can be categorized into two main groups: model-based methods and data-driven methods [4]. Model-based techniques rely on physical or mathematical representations of the degradation process. However, their application is often challenging due to the complexity of machine structures and the extreme operating environments, making it difficult to develop accurate models. Meanwhile, data-driven methods leverage collected sensor data only to learn patterns of degradation. With the increasing availability of high-quality sensor data, and especially the application of machine learning (ML) and deep learning (DL) techniques, these methods have become preferred. Several ML and DL algorithms have been employed for the RUL prediction of machines, such as Linear Regression [5], Support Vector Machine (SVM) [6], Convolutional Neural Network (CNN) [7], and Recurrent Neural Network (RNN) [8]. However, the use of these models presents certain disadvantages. For example, the ML models are ineffective when dealing with high-dimensional and time-sequence data. Meanwhile, DL-based models like CNNs and RNNs struggle to capture long-term dependencies in time series data, which can impair the model's predictive accuracy. Recently, the Transformer architecture [9] has emerged as a powerful solution for capturing long-term dependencies in sequences. Thanks to its self-attention mechanism, the model is less sensitive to increasing sequence lengths. In the context of RUL prediction, Transformer-based models have demonstrated better performance compared with CNN and RNN-based models [10].

A standard Transformer architecture comprises two main components, namely the encoder and the decoder, each performing distinct functions [11]. The encoder processes the input sequence to generate a set of hidden representations capturing contextual information, while the decoder uses these representations to produce an output sequence. In the context of Remaining Useful Life (RUL) estimation, most existing studies have employed both components. However, this is not strictly necessary. It is not compulsory to include both encoder and decoder in a Transformer model; the choice depends on the specific task for which the model is designed. Unlike sequence-to-sequence tasks, which require the decoder to generate sequential outputs (as in machine translation), RUL estimation involves analyzing time series data to predict a single scalar value, representing the estimated remaining lifetime of the machine. Consequently, the decoder can be omitted to simplify the architecture of the model and reduce computational costs, avoiding mechanisms like masked self-attention and encoder-decoder attention, which are redundant for this task. Following this perspective, [12] proposed a Transformer variant that relies solely on an encoder. In particular, the encoder was used to extract temporal features from the input, and then these representations were mapped to the RUL value by a regression layer. Nevertheless, the use of an encoder only may constrain its ability to effectively capture both long-term and short-term dependencies in time series data. Specifically, while self-attention is effective for capturing relationships across a sequence, it may struggle with very long or complex sequences without additional techniques. It may also overlook short-term temporal patterns critical for precise RUL predictions, thus compromising predictive performance. Moreover, the attention mechanism in the encoder usually requires considerable computation and memory. Thus, designing an efficient model for predicting RUL that minimizes training time, supports deployment on resource-constrained devices, and maintains competitive predictive accuracy is still a key challenge.

To address the aforementioned issues, this study proposes a novel transformer-based model for RUL prediction. The model integrates a one-dimensional Convolutional Neural Network (1D-CNN) followed by two stacked transformer encoders. The 1D-CNN extracts local temporal features from the time series input, while the use of two transformer encoders, arranged sequentially, is to increase model depth and enhance the capture of both long-term and

short-term dependencies critical for accurate RUL estimation. The first encoder processes the 1D-CNN output to extract lower-level features and initiate learning of long-term dependencies in the time series. The second encoder refines these representations, with the self-attention mechanism replaced by a Discrete Fourier Transform (DFT) [13] to focus on frequency patterns and reduce computational complexity. This parameter-free DFT reduces hardware requirements while maintaining predictive accuracy, offering improvements over conventional self-attention. In addition, a Bayesian Optimization (BO) algorithm is employed to optimize the model's hyperparameters. The performance of the proposed approach is evaluated on the two well-known benchmark datasets, namely, C-MAPSS and N-CMAPSS, using the metric of root mean square error and scoring function. To sum up, the main contributions of the study are as follows.

- We propose a novel hybrid transformer-based model for RUL prediction, combining a one-dimensional Convolutional Neural Network (1D-CNN) with two stacked Transformer encoders, where the second encoder replaces the self-attention mechanism with a DFT. This combination leverages the 1D-CNN's ability to extract local temporal features and the DFT's efficiency to reduce computational costs. The architecture effectively captures both long-term and short-term dependencies, significantly improving prediction accuracy.

- We investigate the effectiveness of the BO algorithm in determining optimal hyperparameters for the proposed model. This algorithm facilitates the identification of configurations that enhance model performance, achieving lower RMSE values on both C-MAPSS and N-CMAPSS datasets. It also provides insights into the impact of hyperparameters on predictive accuracy and computational efficiency.

- We conduct extensive experiments not only on the C-MAPSS but also on its enhanced version, i.e., N-CMAPSS, to validate the model's performance across diverse scenarios. The proposed model consistently achieves robust performance under varying operating conditions and fault modes, demonstrating its practical utility.

- The proposed model achieves competitive performance compared to the state-of-the-art RUL prediction models in terms of accuracy and training efficiency. Specifically, it is the first to achieve scoring values below 100 and RMSE below 7 on FD001 and FD003, and scoring below 200 with RMSE around 8.0 on FD002 within the C-MAPSS dataset. The model also achieves superior performance across other sub-datasets of C-MAPSS and N-CMAPSS, marking a substantial improvement over existing methods on these well-known benchmark datasets for RUL prediction.

The rest of the paper is organized as follows. Section 2 briefly reviews related work in the literature. The components and architecture of the proposed model are described in detail in Section 3. Section 4 presents the experiments and the obtained results to evaluate the performance of the proposed model. The discussion of potential challenges and future work is presented in Section 5. Finally, conclusions are given in Section 6.

2. Related work

Due to the challenges that ML algorithms face in processing high-dimensional and time-series data, DL-based approaches for RUL prediction are commonly applied in the literature. For example, several studies have explored CNN-based models, such as [14], [15], and [16]. [17] investigated the capability of the Long Short-Term Memory (LSTM) network to handle sequential data. [18] designed a deep transfer reinforcement learning network based on an LSTM network for tool wear monitoring and RUL prediction. Bidirectional Gated Recurrent Unit (GRU) networks were employed to assess machine health status and predict RUL in [19]. Other hybrid models combining both CNN and LSTM can be found in [20], [21], [22], [23] and [24].

Recently, a growing number of studies have examined the potential of transformer networks to address the RUL prediction problem. Thanks to the attention mechanism, transformers can effectively capture dependencies among elements within a sequence. As a result, they have shown notable improvements in RUL prediction tasks. [25] introduced a deep network architecture that incorporates a transformer layer after a gated convolutional unit for local feature extraction. Similarly, [26] used the transformer as a backbone to capture long-term dependencies. [27] proposed a dual aspect self-attention model with two encoders, which work in parallel to simultaneously extract features of different sensors and time steps. [10] addressed domain adaptation by aligning data distributions in both feature and semantic spaces before feeding them into the transformer. [28] introduced temporal and split-feature attention blocks to analyze multivariate time-series data and detect degradation patterns for RUL estimation. Several works have also combined deep learning architectures with transformers to leverage the advantages of both. [29] proposed a hybrid architecture that integrates a transformer encoder with a convolutional neural network (CNN) to extract localized features. A similar idea was adopted in [30], where a hybrid model

combines LSTM with a transformer. [31] presented an end-to-end approach that incorporates global attention (GA), self-attention (SA), and a temporal convolutional network (TCN). With two pooling operations, the GA mechanism enables parallel computation, allowing for adaptive learning of important inputs while avoiding resource waste. In addition, transformer-based architectures followed by Bayesian inference have been used to quantify uncertainty under complex fault modes and operating conditions, as can be seen in [32, 33] and [34].

The structure of a standard transformer neural network consists of two main layers: an encoder and a decoder. Both layers rely on the attention mechanism, which represents the importance of other tokens in the input when encoding a given token. However, this mechanism incurs high computational and memory costs, causing models that adopt both layers to face significant computational overhead and limitations when deployed on resource-constrained devices. To address this, an alternative approach that uses only the encoder was suggested in [12]. Another factor that significantly affects a model's performance is the choice of hyperparameters. Many existing studies rely on the authors' experience or heuristic methods. In contrast, others use more automated techniques, such as Grid Search and Random Search, to determine the hyperparameters of the designed models. However, these algorithms either fail to fully exploit the model's potential or consume a considerable amount of training time, especially with large datasets.

Motivated by these challenges, this study proposes a deep architecture that integrates two transformer encoders to capture long-term dependencies in time series data, along with a Bayesian optimization algorithm to automatically identify the optimal hyperparameters, thereby improving the model's performance in the RUL prediction task. Moreover, a Fourier Transform replaces the self-attention sublayer in the second encoder to reduce training time. Table 1 compares key information regarding the learning structures, hyperparameter selection methods, and efforts to reduce model complexity in recent studies using the well-known CMAPSS benchmark dataset.

3. Proposed architecture

This section describes the proposed architecture in detail. It is designed to harness the capabilities of a hybrid transformer-based network for RUL prediction.

3.1. Overview of the Architecture

As illustrated in Fig. 1, the proposed model consists of four main functional components: a 1D CNN layer, a standard transformer encoder layer, a Fourier transform-based encoder layer, and a linear layer. In this structure, two sequential encoders were selected to strike a balance between model depth and computational efficiency, aligning with the lightweight objective of the study.

The sequential operation of the proposed method is as follows.

- Firstly, the sliding window technique is applied to fetch data from the original dataset and input it into the model as batches in the form of an $b \times n$ matrix, where b is the batch size and n is the number of features. This approach leverages the long-term memory capability of the transformer encoder and enhances the feature extraction power of the CNN block. The data passes through the CNN block for high-level feature extraction, followed by padding and a 1D convolution layer. This maintains data dimensionality and improves information extraction during training.

- Then, the output from the CNN is combined with a positional vector generated by the Positional Encoding (PE) function to provide temporal order information for the Transformer encoder layer. The output is fed into the self-attention block of the first encoder, which computes attention weights to capture correlations within the time series input. Add&Norm and Position-wise Feedforward blocks are applied to transform the output, making it compatible with subsequent functional blocks.

- After that, the output from the first encoder is fed into the second Transformer encoder, where the self-attention mechanism is replaced by a DFT to process the time series in the frequency domain. The DFT extracts frequency patterns and takes the real part of the output, reducing computational complexity and hardware requirements. This approach enhances the model's efficiency and accuracy for RUL prediction.

- Finally, the entire output of the second Transformer encoder is passed through a linear layer to produce the predicted RUL value. After training and inference are complete, the BO algorithm is used to find an optimal set of hyperparameters based on current parameter performance. This process iterates until the model achieves the target performance.

The design of each layer in the model is presented in the sequel.

3.2. One-dimension (1D) CNN layer

When referring to Convolutional Neural Networks (CNNs), they are usually considered DL algorithms designed to handle 2D signals such as images

Table 1. Summary of recent works on RUL prediction of turbofan engines

Work	Dataset	Proposed Structure	Hyperparameter Optimization	Lightweight	Year
[10]	C-MAPSS & N-CMAPSS	Transformer	Grid Search	No	2022
[27]	C-MAPSS	DAST	Grid Search	No	2022
[12]	C-MAPSS	Transformer Encoder	Bayesian Optimization	Yes	2023
[14]	C-MAPSS	Deep Attention Residual	Heuristic	No	2021
[15]	C-MAPSS	DA-TCN	Grid Search	No	2021
[16]	N-CMAPSS	Deep Gaussian Process	Grid Search	No	2021
[19]	C-MAPSS	BiGRU	Grid Search	No	2022
[22]	N-CMAPSS	CNN-LSTM	Heuristic	No	2022
[20]	N-CMAPSS	Deep Gaussian Process	Grid Search	No	2022
[21]	C-MAPSS	CNN-LSTM	Evolutionary Algorithm	No	2021
[23]	C-MAPSS	BiLSTM-CNN	Grid Search	No	2020
[24]	C-MAPSS	Conv–BiLSTM	Grid Search	No	2026
[25]	C-MAPSS	Transformer Encoder-GCU	Heuristic	Yes	2021
[26]	C-MAPSS	VAE-Transformer	Heuristic	No	2022
[28]	C-MAPSS	Transformer	Grid Search	No	2022
[29]	C-MAPSS	TCN-Transformer	Random Search	No	2022
[30]	C-MAPSS	LSTM & Transformer	Heuristic	No	2023
[31]	C-MAPSS	GA-TCN	N.A	No	2023
[32]	C-MAPSS & N-CMAPSS	Bayesian Gated-Transformer	N.A	No	2023
[33]	C-MAPSS	RMTF	Grid Search	No	2025
[34]	C-MAPSS	MCDAN	N.A	No	2024
[35]	C-MAPSS	GCN-TCN	N.A	No	2024
[36]	C-MAPSS	LSTM-Domain ANN	Grid Search	No	2020
[37]	C-MAPSS	PCA-LSTM	Heuristic	No	2021
[38]	C-MAPSS	GCN-GRU	Grid Search	No	2023
[39]	N-CMAPSS	DNN-BiLSTM	N.A	No	2022
[40]	C-MAPSS	Transformer ARR	Grid Search	No	2024
[41]	C-MAPSS	CNN-LSTM-Attention	Heuristic	No	2024
[42]	C-MAPSS	MSCCAN	N.A	No	2025
Ours	C-MAPSS & N-CMAPSS	CNN-Transformer-Fourier	Bayesian Optimization	Yes	—

and video frames. Their ability to learn complex features and patterns makes them the primary tool for many engineering applications in image classification and recognition. CNNs have also been widely used for feature extraction from raw data in various deep learning-based models. However, due to numerous hidden layers and parameters, 2D CNN models require a large amount of labeled

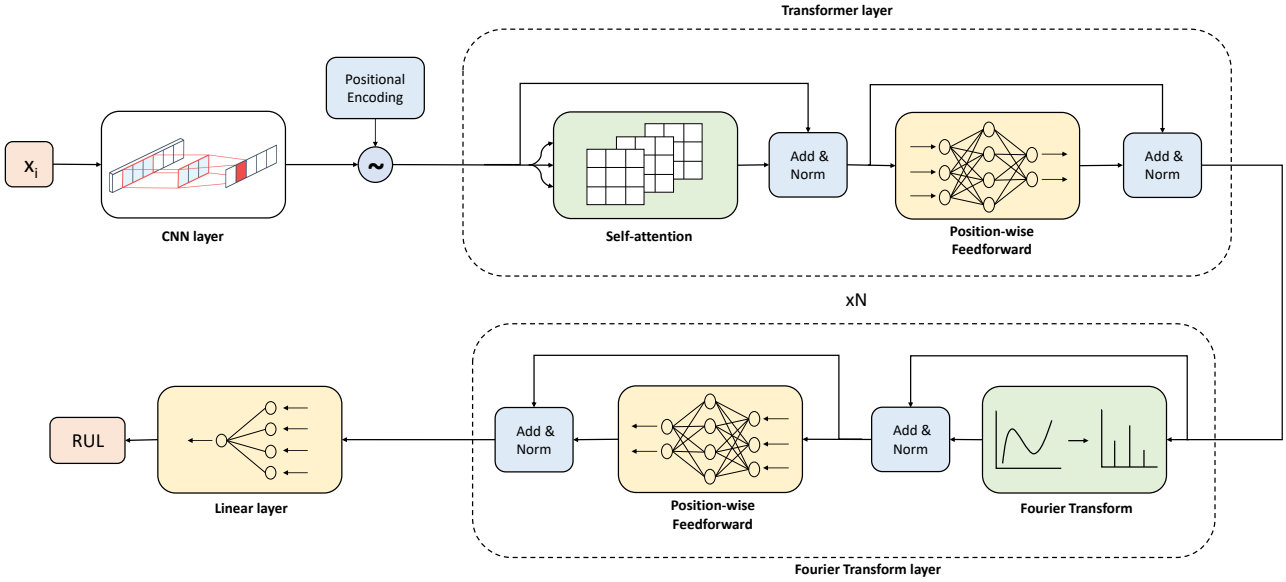

Figure 1. An overview of the proposed model

data to learn effectively. Consequently, they may not perform well in applications where input data is scarce or represented as 1D signals. To address this limitation, 1D CNNs have recently been proposed in the literature and have demonstrated impressive performance in several applications. Unlike 2D CNNs, 1D CNNs perform convolutions only along one dimension, resulting in a simpler and more compact architecture. This design enables real-time and low-cost hardware implementation. In addition, 1D CNNs demonstrate greater computational efficiency than their 2D counterparts when dealing with time-series data, since they use fewer parameters and process data along a single dimension. This leads to faster training times and better suitability for deployment on resource-constrained devices.

When working with time-series data, where inputs are represented as one-dimensional sequences of values (e.g., sensor readings over time), 1D CNNs are particularly well-suited. Their use of weight sharing along one dimension allows the model to capture important temporal patterns without an excessive number of parameters. A comprehensive review of the general architecture and principles of 1D CNNs, along with their major engineering applications, can be found in [43].

In this study, to achieve a lightweight model, we adopt a simple architecture for the 1D CNN. Since the output vector length depends on the input length and the kernel size, padding is applied to ensure the output has the same dimension as the input after convolution. Consistent input-output dimensions improve the evaluation of the CNN block's influence on overall model performance. Using scalar multiplications and additions, the 1D convolution layer processes the raw data and learns to extract features that are then passed to the ReLU activation function. ReLU is chosen to avoid the vanishing gradient problem and to accelerate training compared to other activation functions, such as sigmoid or tanh, since its gradient computation is faster. The full architecture of the 1D CNN layer designed in this study is illustrated in Fig. 2.

3.3. Transformer Encoder layer

The important features extracted from the 1D CNN block are then passed through the transformer encoder layer to leverage the transformer's ability to capture extensive dependencies within time series data. The self-attention mechanism of the transformer allows the model to precisely focus on crucial parts of the input sequence during encoding, effectively managing intricate temporal relationships. The integration of 1D CNN into the transformer extends the model's capacity to learn complex dependencies efficiently and temporal data concurrently, which is pivotal for enhancing the model's performance in predicting RUL.

As mentioned above, we utilize only the encoder block in this study, instead of a standard transformer model. A transformer encoder layer consists of a positional encoding layer, a multi-head attention layer, and a position-wise feed-forward network.

Positional Encoding The first step of using a transformer encoder is to add relative positional information to the input sequences. Based on the features of the RUL data sent by the 1D CNN, this task is performed by using Positional Encoding (PE) functions. The formulae of these PE functions are defined as

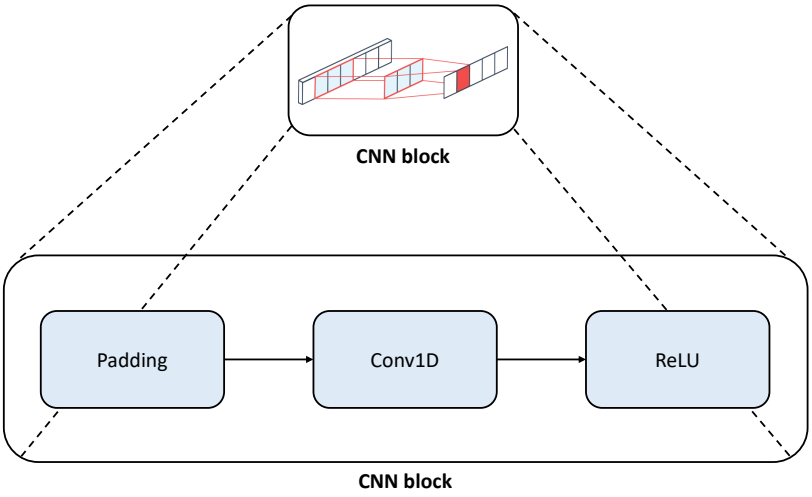

Figure 2. The structure of a CNN layer using 1D CNN

$$PE_{(p,2k)} = \sin\left(\frac{p}{l_{max}^{2k/d_{\text{model}}}}\right), \quad (1)$$

$$PE_{(p,2k+1)} = \cos\left(\frac{p}{l_{max}^{2k/d_{\text{model}}}}\right), \quad (2)$$

where p is the position of an element in the code sequence, d_{model} stands for the dimensionality of the model,l_{max} is the user-defined s calar, and k is the dimension of the index. The positional information enables the model to understand the order and relationships between elements in the sequence, enhancing its ability to capture sequential dependencies and make accurate predictions.

Multi-head self-attention After adding the positional information from the first s tep, t he e xtracted features from the RUL data are passed through the multi-head attention layer. For RUL prediction, it is key to pay more attention to the crucial features that contain more degradation information. The self-attention mechanism is an effective method to learn the dependencies among sequential inputs. The output from each self-attention function is called a head, and multihead self-attention refers to the application of multiple self-attention mechanisms.

In the working process, the self-attention function first transforms the input sequence into the matrices of Queries (Q), Keys (K), and Values (V) representations by the formulas

$$K = \alpha * W_K + \beta_K, \quad (3)$$

$$Q = \alpha * W_Q + \beta_Q, \quad (4)$$

$$V = \alpha * W_V + \beta_V, \quad (5)$$

where $\alpha = x_i' + PE$ is the input sequence with x_i' is the output from the positional encoding layer, $(W_K, \beta_K), (W_Q, \beta_K)$ and (W_V, β_K) are the trained weight and bias matrices. Then, the scaled dot-product attention sublayer calculates the dot product of two matrices Q and K (scaled by $\sqrt{d_{model}}$) to obtain a score of the Values V. The weights assigned to the values are obtained by applying a softmax function

$$Attention(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{model}}}\right)V. \quad (6)$$

where K^T denotes the transpose of the key matrix K. After that, the outputs from each attention head are concatenated and linearly transformed by multiplying by the weight matrix W_O to produce the final output of the Multi-Head Attention following the formula

$$\text{Multi-Head}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_h)W_O,$$

where head_i represents an individual head, and h is the number of heads.

Position-wise feed-forward This layer plays a crucial role in transforming and processing the intermediate representations of the data. It consists of two linear layers followed by ReLU activation functions, allowing the model to learn complex relationships and non-linear transformations in the data. The formula applied in the layer is

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (7)$$

where x is the input vector, W_1 and W_2 are the weight matrices of the two Fully Connected layers, b_1 and b_2 are the bias vectors corresponding to the two Fully Connected layers, and max(0,x) represents the element-wise ReLU activation function applied to the vector.

3.4. Fourier transform

To improve the RUL prediction performance, in the proposed model, we first applied two consecutive transformer encoder blocks. However, it would take more time for the training due to the higher complexity of the attention mechanism compared to the use of only one block. Therefore, rather than using the standard transformer encoder as in the first block, we explored the ability of a non-parameterized Fourier Transform, aiming to reduce the training time for the model.

Technically, the Fourier Transform decomposes a function into its fundamental frequencies. The formula to define a discrete Fourier Transform (DFT) is

$$X_l = \sum_{j=0}^{M-1} x_j \cdot e^{-\frac{i2\pi}{M} lj}, \tag{8}$$

where:

- x_j is the amplitude value of the signal after it is sampled,
- X_l with $l \in [0, M-1]$ represents each sampled signal that has been converted to the frequency domain,
- i denotes the imaginary unit,
- M is the total number of signal samples.

As discussed in [13], replacing the self-attention sublayer in a Transformer encoder with a standard, non-parameterized Fourier Transform can reduce training time by approximately 20% compared to the standard Transformer model, while still maintaining an accuracy level of 92-97%. Furthermore, this model demands less memory and operates efficiently with smaller model sizes, making it a more efficient choice compared to Transformer encoder models with equivalent speed and accuracy constraints. In the design of this study, the DFT does not replace self-attention throughout the whole architecture. Instead, self-attention is retained in the first encoder to learn adaptive contextual dependencies, while the second encoder uses a parameter-free DFT as an efficient refinement mechanism. Since degradation signals often exhibit long-range temporal trends and frequency-related characteristics, transforming the contextualized representation into the frequency domain helps mix global information at a lower computational cost than applying a second self-attention block. The main structure of the Fourier Transform sublayer is depicted in Figure 3.

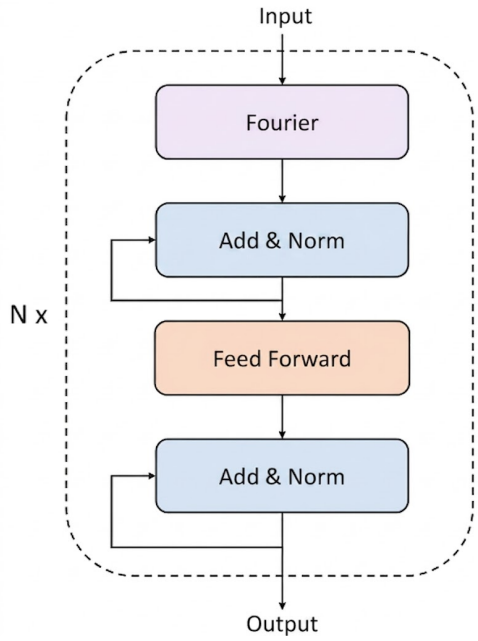

Figure 3. Fourier Transform layer

3.5. Bayesian optimization-based hyperparameter selection

Besides the model's architecture, the hyperparameters are also of interest in our proposed model. While many existing studies rely on predefined hyperparameters or grid search methods that take a considerable amount of time, we propose using the Bayesian optimization (BO) algorithm to discover optimal hyperparameters for the encoder in our transformer-based RUL prediction model. The fundamental concept of BO involves utilizing previous evaluations of the loss function f (objective function) to identify the next optimal point for sampling f. The optimization problem is defined as

$$\phi^* = \arg\max_{\phi \in \Phi} g_\phi(x), \tag{9}$$

where $g_\phi(x)$ is a black-box function parameterized by $\phi \in \Phi$, representing the space of all possible hyperparameters. The BO algorithm constructs an approximation function, known as a surrogate function, to model the behavior of the true objective function. Given the limited information available from the black-box function, the surrogate function aims to replicate the behavior of the true function. The surrogate function is initially chosen based on the objective function's domain and is continuously updated using Bayesian principles with observed data. The updated surrogate function, or the posterior, is then used to build the acquisition function to select the next evaluation point. As new data is accumulated, the iterative process is repeated. By incorporating them into the analysis, the model can adapt its parameters accordingly, enhancing its performance over time.

The BO algorithm comprises two main components: a statistical model to provide the probability distribution of the objective function and an acquisition function to guide the selection of evaluation points. In this study, a

Gaussian process is utilized for the surrogate function and the Expected Improvement (EI), defined by

$$EI(\phi) = \mathbb{E}[max(g_\phi - g^*), 0], \tag{10}$$

is applied for the acquisition function, where g^* represents the current best observed outcome.

4. Experiments and Results

4.1. Datasets

C-MAPSS dataset C-MAPSS, or Commercial Modular Aero-Propulsion Simulation System, is a model that generates realistic, comprehensive commercial turbofan engine data. Taking the input of actual flight conditions recorded on board a commercial jet, the model simulates run-to-failure trajectories of engines where the flights o perated u nder v arious c onditions. The engines can change from one condition to another within a realistic transition period. During the simulated flights, t he f ault w as i njected a t a given time and persisted throughout the remaining flights, affecting the performance of the engines. The simulated dataset is often called the C-MAPSS dataset for convenience. Based on different o perating conditions, it encompasses four subsets labeled FD001, FD002, FD003, and FD004. Each is then split further into the training set, which consists of run-to-failure data, and the testing set, which contains data that ends before a fault is detected. Among four subsets of the C-MAPSS, FD004 is considered the most challenging as the corresponding flights experienced the most diverse operating conditions with the most fault modes. FD004 is followed by FD002 and then FD003 in terms of complexity. The details of each dataset are shown in Table 2. In the literature, the dataset is considered a well-known benchmark for evaluating the performance of many studies related to predictive maintenance, where the primary goal is to accurately identify the RUL value for each engine in the test sets.

N-CMAPSS dataset. Introduced in [44], the N-CMAPSS is a further improvement of the C-MAPSS dataset, aiming to facilitate the development of deep learning-based models for predictive maintenance applications. Compared to the original C-MAPSS dataset, the N-CMAPSS includes comprehensive flight conditions and the health status (i.e., healthy or faulty) from different commercial flight r outes. T he fi delity of degradation modeling in generating the dataset is improved by relating the onset of the degradation process to the operation history.

There are eight subsets of data in the N-CMAPSS, which is twice the number of subsets in the C-MAPSS. These subsets are derived from 128 units and seven possible failures, from DS01 to DS08. An overview of the new dataset is presented in Table 3. Due to the large size of the entire dataset, only the DS02 dataset was chosen for the experiment. This dataset also reflects the most common failure modes. In particular, the DS02 dataset includes the complete run-to-failure profiles of 9 engines, comprising 6.5 million data samples. These nine engines exhibit two distinct types of failure modes as follows:

- Engines 2, 5, and 10 experience a degradation pattern characterized by abnormal deterioration in the efficiency of the high-pressure turbine (denoted as HPT_E).
- Engines 11, 14, 15, 16, 18, and 20 face a more intricate failure mode involving abnormal degradation of both the low-pressure turbine efficiency and flow (denoted as LPT_E + LPT_f), as well as the efficiency degradation in the high-pressure turbine (denoted as HPT_E + LPT_E + LPT_f).

The anomalous degradation of system components follows a stochastic process, initially starting with gradual, normal degradation before transitioning to a rapid, abnormal deterioration. This shift from normal to abnormal degradation is generally stable and occurs at various cycles for each engine. Furthermore, the initial degradation state of each engine is distinct, highlighting variations among the engines. Table 4 presents an overview of the DS02 dataset, specifying the unit of measurement as the engine number. The length of the data is denoted by m_i, the total number of cycles is denoted as t_{EOL}, and the initiation time of the fault is indicated as t_s.

4.2. Data pre-processing

In both C-MAPSS and N-CMAPSS datasets, some sensors provide constant data throughout the entire life cycle, and these data have no effect on the prediction of RUL. Therefore, in data preprocessing, we first remove these data from the original dataset. Then, to handle the wide range of data from various measurements of different sensors, we normalize the data by applying the min-max normalization technique. That is, $s^{(j)}$, which represents the data for the j-th sensor, is normalized by the following formula

$$s_t^{(j)} \leftarrow \frac{s_t^{(j)} - \min(s^{(j)})}{\max(s^{(j)}) - \min(s^{(j)})}. \tag{11}$$

After normalization, all values within the dataset lie within the $[0, 1]$ range, preparing them for further analysis. In our study, all extensive experiments on both C-MAPSS and N-CMAPSS datasets were conducted utilizing a Dell Precision 3640 Tower workstation equipped with an NVIDIA Quadro P2200 GPU and an Intel Core i7-10700K CPU, with 32GB of RAM.

Table 2. A description of the C-MAPSS dataset

C-MAPSS Dataset	FD001	FD002	FD003	FD004
Operating conditions	1	6	1	6
Fault modes	1	1	2	2
Training set	20631	53759	24720	61249
Training engines	100	260	100	249
Test set	100	259	100	248
Testing engines	100	259	100	249

Table 3. Overview of the N-CMAPSS dataset

Name	Units	Flight Classes	Failure	Modes Size
DS01	10	1,2,3	1	7.6 M
DS02	9	1,2,3	2	6.5 M
DS03	15	1,2,3	1	9.8 M
DS04	10	2,3	1	10.0 M
DS05	10	1,2,3	1	6.9 M
DS06	10	1,2,3	1	6.8 M
DS07	10	1,2,3	1	7.2 M
DS08	54	1,2,3	1	35.6 M

Table 4. Description of the DS02 dataset

Unit	m_i	t_{EOL}	t_s	Fault mode
2	8531	75	17	HPT_E
5	10 334	89	17	HPT_E
10	9527	82	17	HPT_E
16	7652	63	16	$HPT_E + LPT_E + LPT_f$
18	8907	71	17	$HPT_E + LPT_E + LPT_f$
20	7681	66	17	$HPT_E + LPT_E + LPT_f$
11	6634	59	19	$HPT_E + LPT_E + LPT_f$
14	1567	76	36	$HPT_E + LPT_E + LPT_f$
15	4334	67	24	$HPT_E + LPT_E + LPT_f$

4.3. Evaluation metrics

In this study, the performance of the proposed model is evaluated using two well-established metrics: the root mean square error (RMSE) and the scoring function (score). The first metric is defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{y_i})^2}, \tag{12}$$

and the scoring function is defined as

$$score = \begin{cases} \sum_{i=1}^{N} e^{\frac{\widehat{y_i} - y_i}{a_1}} - 1, & \text{if } \widehat{y_i} \geq y_i, \\ \sum_{i=1}^{N} e^{\frac{y_i - \widehat{y_i}}{a_2}} - 1, & \text{if } \widehat{y_i} < y_i, \end{cases} \tag{13}$$

where:

- N is the number of samples,
- y_i is the true RUL value,
- $\widehat{y_i}$ is the predicted RUL value.

The use of two controlled parameters a_1 and a_2 in the scoring function enables a conservative estimation of RUL. More specifically, the underestimation of RUL (i.e., the predicted RUL is less than the actual RUL) is preferred over the overestimation (i.e., the predicted RUL is greater than the actual RUL) since a too-optimistic estimation is more likely to cause more serious consequences. Therefore, the overestimation is more heavily penalized than the underestimation. This is achieved by using a larger value of a_2 compared to a_1. For example, the values $a_1 = 10$ and $a_2 = 13$ are typically chosen as introduced in [45]. In addition, the

penalty from the scoring function grows exponentially with increasing error. Meanwhile, the RMSE assigns equal weights to both late and early predictions. As a result, it can be more sensitive to data with large deviations. In the literature, these two evaluation indicators are often used in conjunction with each other, preventing the model from underestimating to achieve a low score and avoiding sensitivity to data variability. Based on their definition, the lower the values of the two metrics, the higher the model's accuracy.

4.4. Performance of the proposed method on the CMAPSS dataset

The proposed model includes nine essential hyperparameters, which are the number of heads in the multi-head attention (n_head), the number of stacked Transformer layers (num_layers), and the feed-forward layer dimension (dff) in the Transformer, regularization (weight_decay), the learning rate (*lr*), the number of epochs (n_epochs), the sample size (l_win), the kernel size (kernel_size) of the CNN, and the dropout rate (dropout). These hyperparameters notably affect the model's performance, leading to the use of the BO for their automated optimization. Following the approach outlined in [12], log-uniform distributions are utilized for the prior of continuous hyperparameters. In contrast, discrete uniform distributions are employed to initialize discrete hyperparameters. The optimal values of these hyperparameters for each subset in the C-MAPSS and the DS02 in the N-CMAPSS after training are displayed in Table 5.

The performance of the proposed model corresponding to these hyperparameters evaluated on the C-MAPSS dataset is presented in Table 6. For comparison purposes, the performance of several other recent models operating on the same dataset and employing the same metrics is reproduced. The best results are highlighted in bold. As can be seen from the table, our proposed model provides the best performance on all C-MAPSS subsets for both RMSE and Score metrics, achieving an impressive accuracy in predicting the RUL values compared to other models. Indeed, for FD001, our model leads to the smallest RMSE value of 6.97, while the RMSE values from other models are all greater than 8.4. This marks an improvement of at least 17%. When it comes to the Score, the obtained result is even more impressive: it is reduced from 164 given by the second-best model, i.e., the GA in [31], to only 81, which corresponds to a decrease of about 50%. Similar results are observed in the FD003 dataset, where the RMSE is reduced from the second-best model (i.e., the Trans-Lighter in [12]) of 9.03 to 6.52, and the Score from [31] is reduced from 183 to 91, which is a decrease of 55%.

Among the four subsets of the C-MAPSS, FD002 and FD004 are considered more challenging than FD001 and FD003 as they come from engines operating in harsher environments with more operating conditions and fault modes. However, our proposed model has produced a significant RUL prediction performance on the FD002 dataset, yielding an RMSE of 8.03 and a Score of 178. These values are much smaller than the corresponding values of the other models. They are even better than the performance of competing models on the two less challenging datasets, i.e., FD001 and FD003. For the FD004 subset, it achieves an RMSE of 12.24, somewhat better than the 12.41 from both the DA-Transformer [10] and Trans-Lighter [12] models. More impressively, it provides a Score below 850, significantly outperforming other models (typically exceeding 900, with many even surpassing 2000).

Another notable advantage of the proposed model, as shown by the obtained results, is its balanced performance across both RMSE and Score metrics, unlike some competing methods that excel in only one. For instance, compared to the ARMAGCN-GRU model in [38], the DA-Transformer model in [10] achieves lower RMSE but higher Score values. This consistent effectiveness underscores the model's reliability for RUL prediction, making it a robust choice for practical applications.

The performance of the proposed model is visualized by showing the discrepancy between predicted and actual RUL values across the four C-MAPSS subsets, as shown in Figure 4-7. The closely aligned lines in these figures highlight its accuracy in predicting RUL based on time-series data from the C-MAPSS.

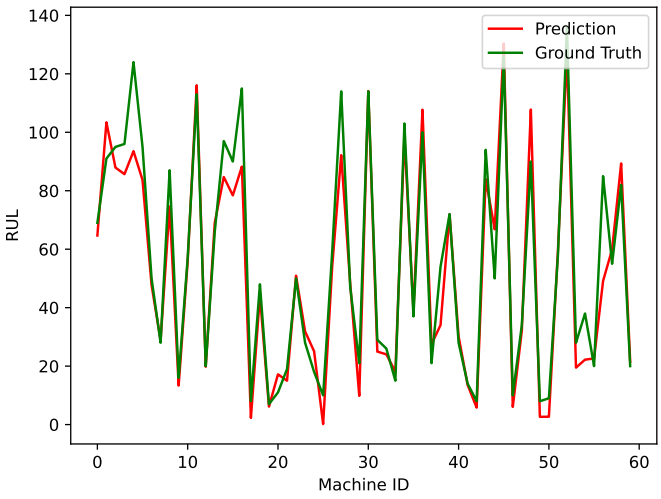

Figure 4. The ground truth RUL vs. the predicted RUL on the FD001 dataset

Table 5. The optimal hyperparameters of the proposed model

Hyperparameters	FD001	FD002	FD003	FD004	DS02
weight_decay	0.0018	0.0015	0.0020	0.0016	0.00034
lr	0.0011	0.0015	0.0032	0.0040	0.0011
dropout	0.072	0.082	0.064	0.13	0.072
n_head	4	23	18	1	2
kernel_size	5	5	7	5	7
num_layers	2	1	3	2	2
l_win	123	123	123	124	92
n_epoch	102	104	82	98	52
dff	256	128	128	128	64

Table 6. RUL prediction performance of the proposed model vs. some recent models on the C-MAPSS dataset

Dataset	FD001		FD002		FD003		FD004	
Model & Refs.	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
DA-Transformer [10]	8.67	343	12.42	740	9.37	571	12.41	991
Trans-Lighter [12]	8.40	–	9.85	–	9.03	–	12.41	–
DARNN [14]	12.04	261	19.24	933	10.18	247	18.02	2587
DATCN [15]	11.78	229	16.95	1842	11.56	257	18.23	2317
NAS-Transformer [21]	11.50	202	16.14	1131	11.35	227	20.00	2299
AGCNN [23]	12.42	225	19.43	1492	13.39	227	21.50	3392
DAST [27]	11.43	203	15.25	924	11.32	154	18.36	1490
FeaR-STAT [28]	12.01	200	15.5	1023	10.9	199	15.03	1587
Trans-TCNN [29]	12.31	252	15.35	1267	12.32	296	18.35	2120
Kamei et al. [30]	13.52	287	19.32	1436	13.44	263	20.38	2784
GA [31]	10.81	164	15.66	988	10.97	183	15.63	1090
GCN-TCN [35]	11.50	187.2	16.92	1132.2	11.05	196.0	19.33	1443.4
GT [32]	12.09	262	11.46	550	10.16	197	13.89	963
MCDAN [34]	–	–	13.11	769	–	–	14.6	1082
ARMAGCN-GRU [38]	11.59	191	13.63	704	11.40	203	14.47	927
RMTF Transformer [33]	11.17	212	14.02	970	11.48	238	17.07	1500
GCU-Transformer [46]	11.27	–	22.81	–	11.42	–	24.86	–
Proposed model	**6.97**	**81**	**8.03**	**178**	**6.52**	**91**	**12.24**	**841**

The bold results indicate the best performance

4.5. Performance of the proposed method on the N-CMAPSS dataset

Similar experiments were conducted on the N-CMAPSS dataset, specifically to evaluate the validity and robustness of the proposed model. As mentioned above, the evaluation was confined to the DS02 subset rather than covering all N-CMAPSS subsets. The optimal hyperparameters of the model, trained on the DS02

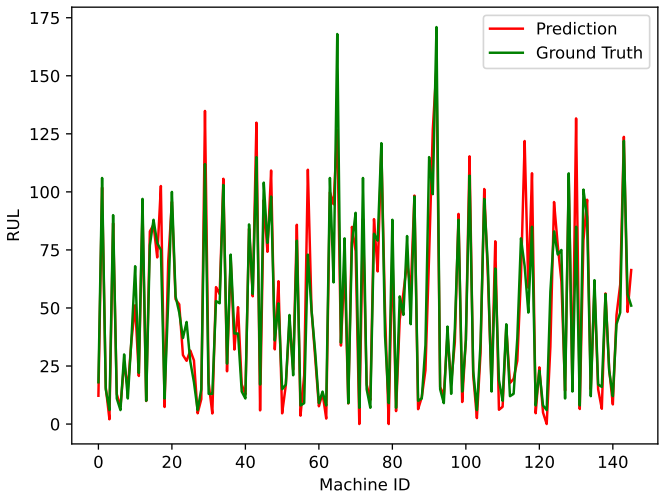

Figure 5. The ground truth RUL vs. the predicted RUL on the FD002 dataset

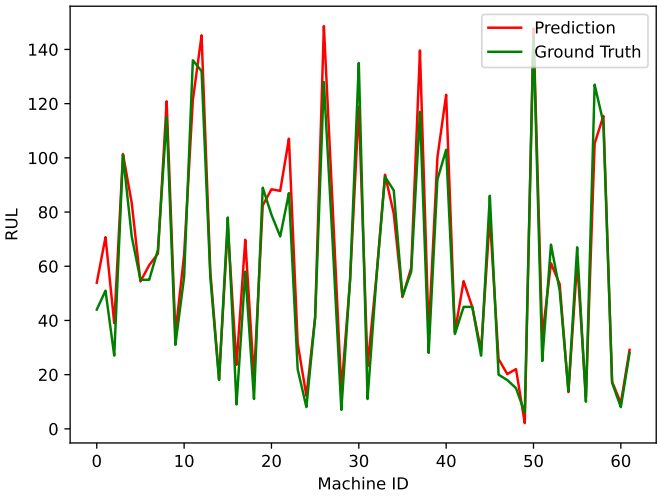

Figure 6. The ground truth RUL vs. the predicted RUL on the FD003 dataset

subset, are presented in Table 5 (last column, titled DS02). Its performance was evaluated using RMSE and Score metrics, as in previous experiments. The obtained results are presented and compared with those of recent studies in Table 7. As can be seen from this table, the proposed model achieves the smallest RMSE of 3.71, while other competing methods yield RMSEs all greater than 4. For the Score metric, it provides the second-lowest value, outperformed only by the GT model in [32]. Figure 8 illustrates the discrepancy between predicted and actual RUL values for the DS02 subset,

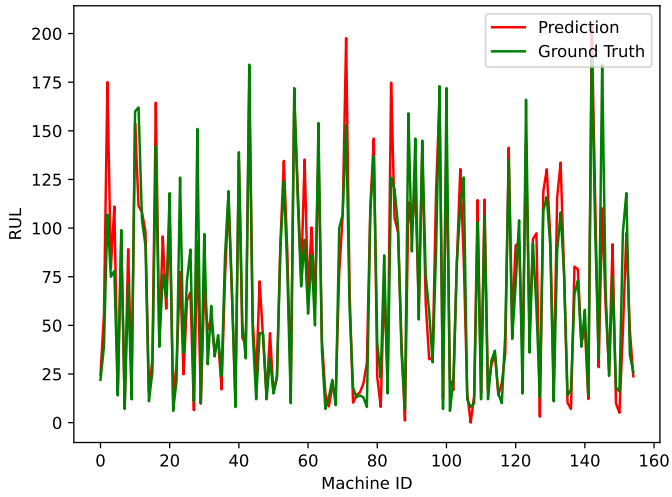

Figure 7. The ground truth RUL vs. the predicted RUL on the FD004 dataset

with closely aligned values demonstrating its accuracy in RUL prediction.

Table 7. The RUL estimation performance of the proposed model vs. recent models on the DS02 dataset

Model & Refs.	RMSE	Score
SCTA-LSTM [47]	4.15	4489
GT [32]	6.86	**142**
MCD[16]	7.31	-
DGP [20]	6.17	5577
Input-Attn-LSTM [15]	4.32	4655
Proposed model	**3.71**	3981

The performance of a DL-based model is reflected by not only its prediction accuracy but also its training time. In Figure 9, we compare the performance of the proposed model with other models in terms of training time for the DS02 subset, using the workstation configuration described above and the established hyperparameters. One can see that the Trans-Lighter model [12] performs best thanks to its lightweight architecture with a single encoder layer. With the integration of more architectures like 1D CNN and Fourier layers, the proposed model required slightly longer training time. This can be considered a trade-off for achieving higher RUL prediction accuracy. However, compared to other models like GCU-Transformer [46] and LSTM-CNN [48], its training time is significantly shorter, taking 1613 seconds compared to 1783 seconds for GCU-Transformer (10% longer) and 2872 seconds

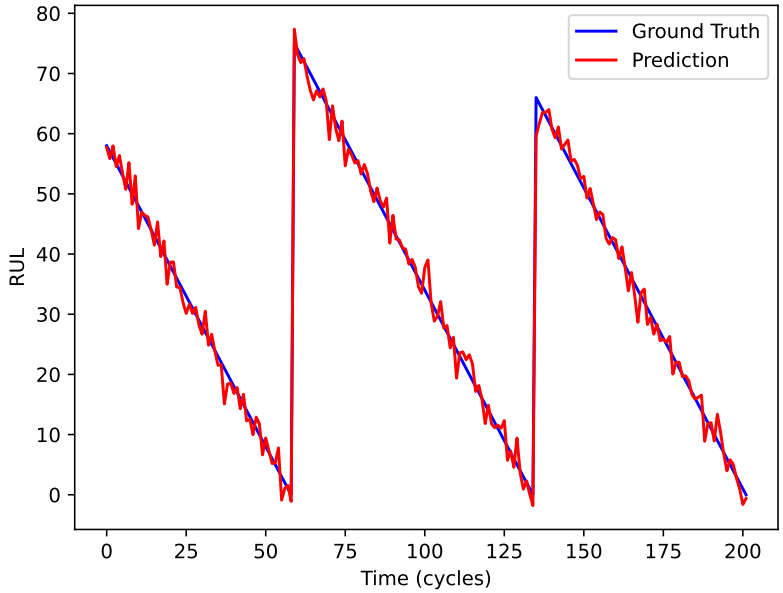

Figure 8. The ground truth RUL vs. the predicted RUL on the DS02 dataset

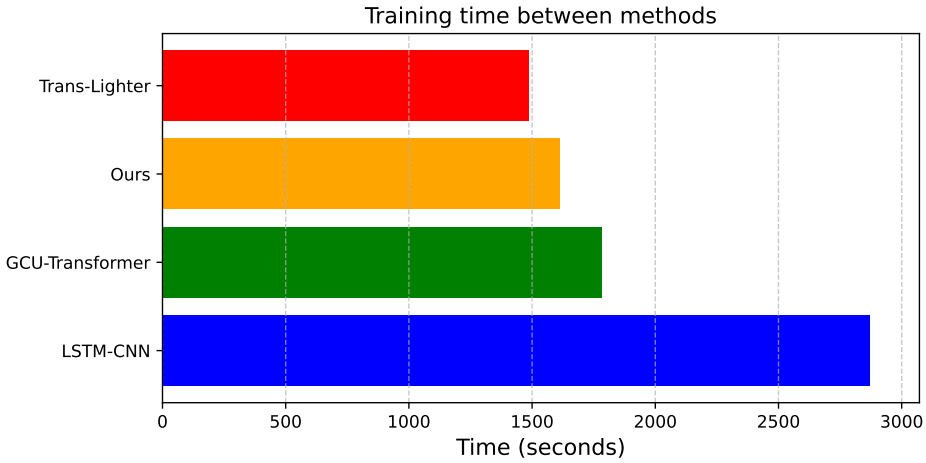

Figure 9. Comparison of training time among methods

for LSTM-CNN (nearly 80% longer). This result further confirms its effectiveness for predicting RUL values.

4.6. Effectiveness of hyperparameters and the use of the BO algorithm

In the design of any DL model, various hyperparameters are selected to govern the model's behavior. The choice of these hyperparameters affects the model's performance and the efficiency of the optimization process. As the number of hyperparameters increases, the computational burden of the BO algorithm increases, leading to reduced effectiveness. From this point of view, assessing the correlation between hyperparameters and the RMSE values is an efficient way to determine which hyperparameters should be prioritized for optimization. The formula to calculate these correlation coefficients is as follows

$$r_{\kappa\hat{y}} = \frac{n_{obs}\sum \kappa\hat{y} - \sum \kappa \sum \hat{y}}{\sqrt{n_{obs}\sum \kappa^2 - (\sum \kappa)^2}\sqrt{n_{obs}\sum \hat{y}^2 - (\sum \hat{y})^2}} \quad (14)$$

where:

- $r_{\kappa\hat{y}}$ represents the correlation coefficient between the hyperparameters κ and the RMSE $\hat{y}$,
- n_{obs} denotes the number of observations,

- $\sum \kappa\hat{y}$ is the sum of the products of corresponding values of κ and $\hat{y}$,
- $\sum \kappa$ and $\sum \hat{y}$ are the sums of all κ and $\hat{y}$,
- $\sum \kappa^2$ and $\sum \hat{y}^2$ represent the sums of the squares of all κ and $\hat{y}$ values, respectively.

After the optimization process, a set of correlation coefficients is obtained, revealing the impact of each hyperparameter on the model's RUL prediction accuracy. In Figure 10, these correlation coefficients are presented for the FD002 subset. They show the relationship between hyperparameters and the resulting RMSE obtained from the BO algorithm, with dotted bars indicating negative correlations and diagonal dashed bars indicating positive correlations. In our experiments, hyperparameters with non-zero correlation coefficients were selected. When the model complexity increases and the number of hyperparameters grows, those with low correlations with RMSE may be discarded to ensure that the desired performance of the BO algorithm is maintained.

To visualize the correlation coefficients, Figure 11 illustrates the model's RUL prediction accuracy across various hyperparameter configurations (each line connecting the hyperparameter values represents a specific training iteration). As shown in Figure 10, the correlation coefficient between the hyperparameter 'l_win' and RMSE is -0.413. Figure 11 shows that the model achieves higher RUL prediction accuracy with lower RMSE for larger 'l_win' values, resulting in denser clustering of iterations. Similarly, the hyperparameter 'weight_decay' has a correlation coefficient of 0.071. Lower 'weight_decay' values show a higher frequency of training iterations due to their positive correlation with RMSE.

Figure 12 illustrates the use of the BO algorithm, examining model's performance and training time. The y-axis displays the RMSE obtained from training iterations, while the x-axis denotes the elapsed time in training intervals, with each unit representing approximately 33 minutes. Each plotted point corresponds to a distinct training iteration. From the figure, using the BO algorithm requires approximately 60 training iterations (about 680 minutes) to reach the optimal hyperparameter settings for the FD002 subset, providing the smallest RMSE of 8.03. Investigating the performance of the model also involves manually adjusting different hyperparameter values. For instance, modifying the 'l_win' parameter from 123 to 115 within the optimal hyperparameter set found by the BO for FD002 notably increases the RMSE from the best value of 8.03 to 10.81.

To highlight the effectiveness of the BO algorithm, an optimization process was executed using the Random method for the FD002 subset, as presented in Figure 13. Utilizing random methods for 80 training iterations, the model yields a minimum RMSE of 10.58, which is 32% higher than the RMSE from the BO. Comparing the graphs from both methods, the RMSE values from the Random method show complete randomness, while those from the BO algorithm show a decreasing trend with more training iterations. This increases the likelihood of discovering optimal hyperparameter settings compared to the Random method.

4.7. Ablation Study

To further analyze the contribution of each component in the proposed framework, an ablation study was conducted by selectively removing the CNN front-end, Transformer encoder, and DFT-based encoder. All experiments were performed under identical training settings, including the same optimizer, learning rate, batch size, training epochs, and hyperparameter configuration across all C-MAPSS subsets. Each configuration was evaluated using three random seeds, and the reported results correspond to the average performance across all runs.

Table 8 summarizes the evaluated architectures. The proposed model consists of a CNN feature extractor, one Transformer encoder, and two DFT-based encoders. Three simplified variants were constructed by removing either the DFT encoder, the Transformer encoder, or the CNN front-end to isolate the contribution of each module. Furthermore, all compared configurations maintain approximately similar encoder depth to ensure a fair comparison between attention-based and frequency-domain token mixing mechanisms.

Table 8. Ablation model configurations

Model	CNN	Transformer	DFT
Proposed (M0)	✓	1	2
M1	✓	1	0
M2	✓	0	3
M3	0	1	2

The quantitative results are presented in Table 9. Among all evaluated variants, removing the CNN front-end (M3) caused the most significant performance degradation across all datasets. Compared with the proposed model, M3 increased the RMSE from 15.06 to 18.20 on FD001, from 18.94 to 31.78 on FD002, from 19.26 to 33.75 on FD003, and from 31.51 to 41.62 on FD004. The performance gap becomes particularly noticeable on the more complex FD002 and FD004 subsets, which contain multiple operating conditions and fault modes. These results indicate that the CNN module plays a critical role in extracting local degradation patterns and low-level temporal features from multivariate sensor signals.

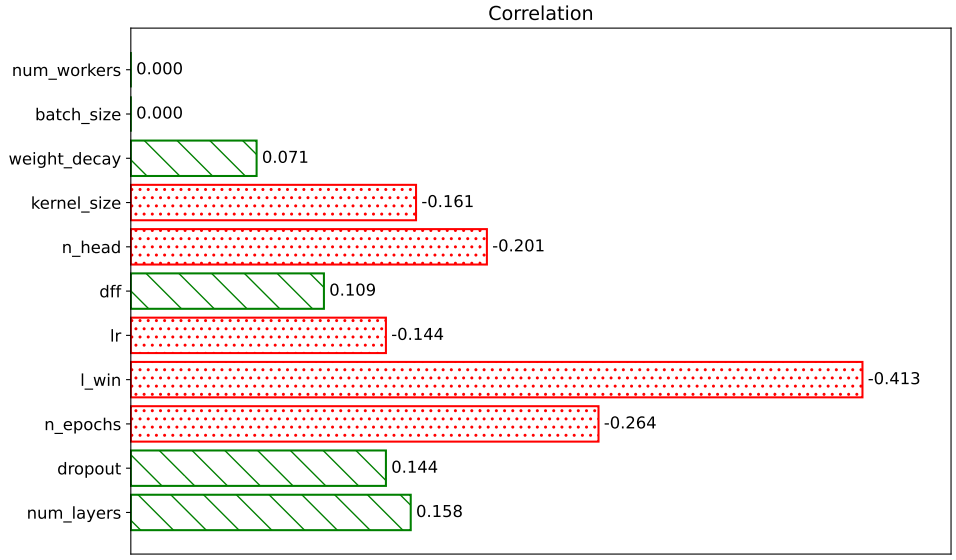

Figure 10. Correlation of the hyperparameters with RUL

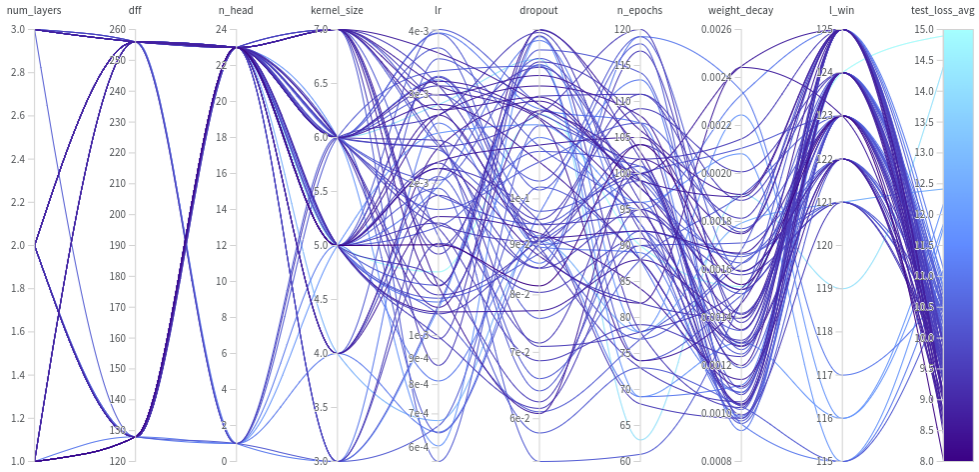

Figure 11. Optimizing process

Comparing M1 and M2 further highlights the contribution of the Transformer and DFT-based encoders. M1, which removes the DFT encoder, achieved RMSE values of 14.68, 17.15, 19.36, and 29.61 on FD001–FD004, respectively. Meanwhile, the pure DFT-based configuration M2 obtained comparable results of 14.57, 18.43, 19.04, and 31.39. The relatively close performance between M1 and M2 suggests that both attention-based modeling and frequency-domain token mixing are capable of capturing global temporal dependencies in turbofan degradation sequences.

Although several simplified configurations achieved lower RMSE values on individual subsets, the proposed hybrid architecture demonstrated more consistent behavior across different operating conditions and fault modes. In particular, the hybrid design combines the local feature extraction capability of CNNs, the sequential dependency modeling of Transformer encoders, and the global frequency-domain representation learning of DFT-based encoders. Overall, the ablation study confirms that the CNN front-end is the dominant feature extraction component, while the Transformer and DFT encoders provide complementary high-level temporal representations that improve the robustness and generalization capability of the proposed framework.

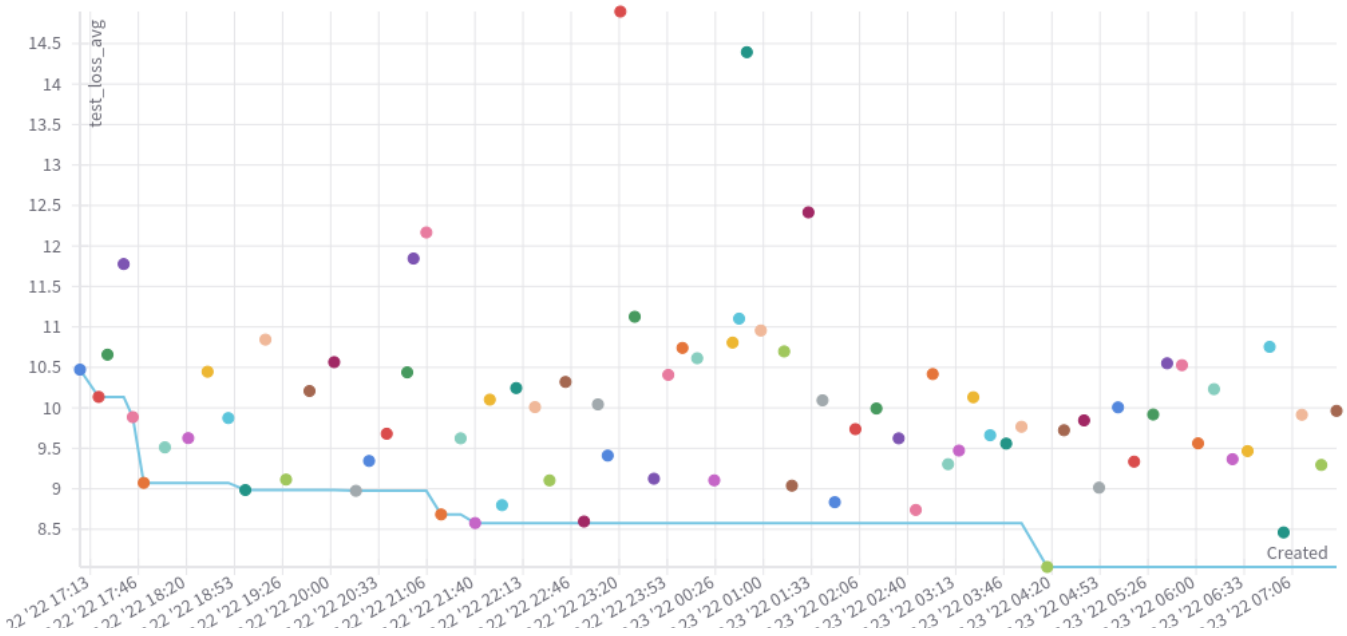

Figure 12. Optimizing hyperparameters using the BO algorithm for the FD002 subset

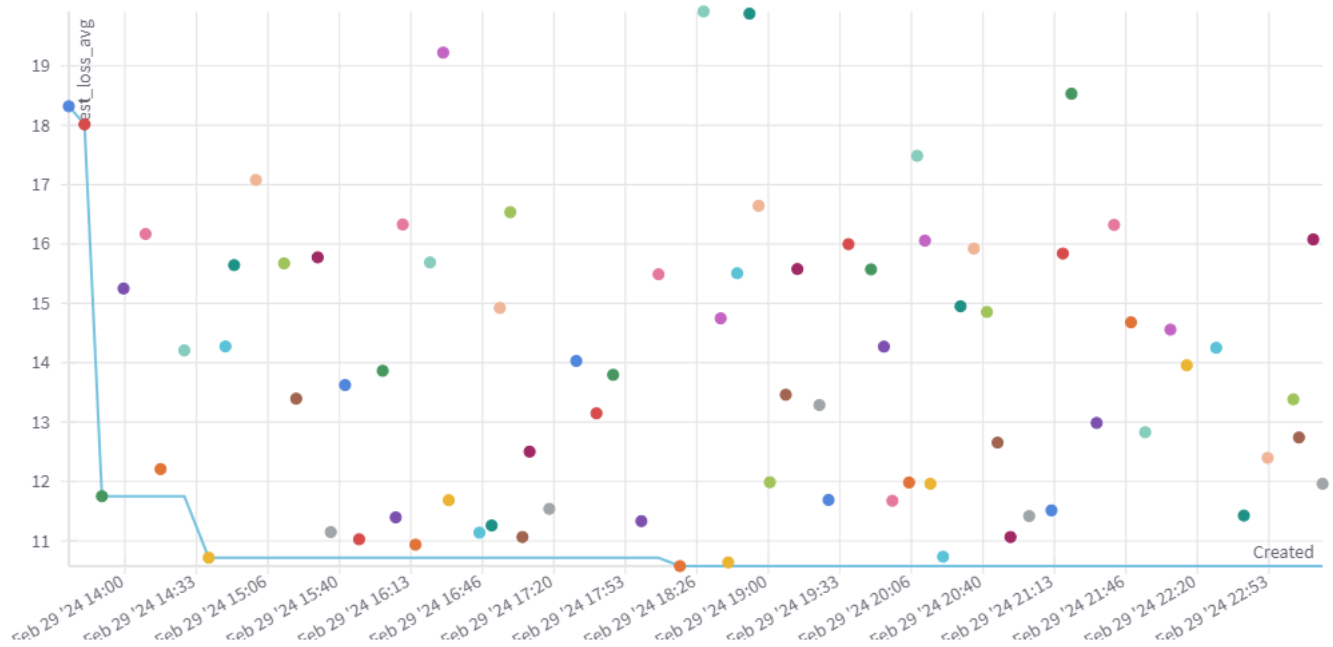

Figure 13. Optimizing hyperparameters using the Random method for the FD002 subset

5. Discussion

While the proposed model has demonstrated strong RUL prediction accuracy compared to other state-of-the-art models, several issues related to its deployment in practice require further exploration. For example, the DL models are usually considered black boxes due to the lack of interpretability. A detailed and transparent explanation will allow clarifying their behavior and highlighting their reliability and usability. In this study, the impact of hyperparameters has been addressed through the use and analysis of the BO algorithm to find optimal hyperparameter settings that ensure its optimal performance. This can be explored further with Explainable Artificial Intelligence (XAI) techniques.

The sensitivity analysis to examine the model's response to changes in input features, operating conditions, and especially prediction uncertainty is an important perspective. Many previous studies relied on stochastic models to characterize the degradation

Table 9. Ablation study results on the C-MAPSS datasets

Model	FD001		FD002		FD003		FD004	
	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
Proposed (M0)	**15.06**	427.3	**18.94**	21,794.7	**19.26**	1,014.9	**31.51**	459,149.0
CNN + Transformer (M1)	14.68	304.5	17.15	29,360.5	19.36	1,610.8	29.61	4,869,938.0
CNN + DFT (Pure FNet) (M2)	14.57	496.4	18.43	10,253.8	19.04	1,080.4	31.39	9,863,812.1
Without CNN (M3)	18.20	1,946.1	31.78	39,364.2	33.75	2,801,927.9	41.62	1,585,318.5

process and estimate the RUL distribution. However, as discussed in [49], stochastic models for complex systems are not straightforward to derive. The authors introduced a data-driven approach called Lognorm-LSTM, which adds a lognormal layer to its output to quantify uncertainty and predict the component's RUL distribution. This approach is promising to apply, offering a potential direction for future research.

Regarding input uncertainty, prognostics for predicting future system behavior face various sources. These uncertainties stem from variability in process behavior due to diverse operating and environmental conditions, model inaccuracies, inherent randomness or noise in sensor data, and imperfect knowledge of system states. Therefore, future investigations should focus on effective uncertainty management in prognostics to ensure accurate RUL prediction.

Finally, a distributed computing environment should be considered, where the model is implemented on edge devices using local data from distributed zones. In this scenario, the Federated Learning technique is applicable due to its ability to preserve privacy and ensure global knowledge for local models. Applying Federated Learning adds a new aspect to studying an effective model, which should be accurate and lightweight.

6. Conclusion

In this study, we propose a hybrid deep transformer-based model for RUL prediction. The model integrates advanced DL components, including a 1D-CNN block and two sequential transformer encoders. The self-attention mechanism in the second encoder is replaced by a DFT. These components leverage their ability to extract local and long-term temporal features and capture dependencies from sensor sequence data. The impact of hyperparameters on the model's performance has been investigated through the BO algorithm. Through extensive experiments on two well-known datasets of C-MAPSS and N-CMAPSS, the model demonstrates strong performance in predicting RUL, achieving superior accuracy and efficiency compared to existing approaches. The findings from this study facilitate PdM, leading to significant improvements in ensuring operational efficiency and reducing maintenance costs for industrial production processes. Regardless of strong benchmark performance, the present study does not yet provide a full interpretability or uncertainty-aware deployment framework. These aspects, together with broader robustness evaluation under real industrial conditions, will be investigated in future work.

Acknowledgment

This research is funded by Hanoi University of Science and Technology (HUST) under Project number T2023-PC-080.

References

[1] M. Xiong, H. Wang, Q. Fu, and Y. Xu, "Digital twin–driven aero-engine intelligent predictive maintenance," *The International Journal of Advanced Manufacturing Technology*, vol. 114, no. 11, pp. 3751–3761, 2021.

[2] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation – a review on the statistical data driven approaches," *European Journal of Operational Research*, vol. 213, no. 1, pp. 1–14, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221710007903

[3] A. Meddaoui, M. Hain, and A. Hachmoud, "The benefits of predictive maintenance in manufacturing excellence: a case study to establish reliable methods for predicting failures," *The International Journal of Advanced Manufacturing Technology*, vol. 128, no. 7, pp. 3685–3690, 2023.

[4] K. Medjaher, D. A. Tobon-Mejia, and N. Zerhouni, "Remaining useful life estimation of critical components with application to bearings," *IEEE Transactions on Reliability*, vol. 61, no. 2, pp. 292–302, 2012.

[5] X. Li, F. Elasha, S. Shanbr, and D. Mba, "Remaining useful life prediction of rolling element bearings using supervised machine learning," *Energies*, vol. 12, no. 14, 2019. [Online]. Available: https://www.mdpi.com/1996-1073/12/14/2705

[6] T. Benkedjouh, K. Medjaher, N. Zerhouni, and S. Rechak, "Remaining useful life estimation based on nonlinear feature reduction and support vector

regression," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 7, pp. 1751–1760, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197613000365

[7] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832017307779

[8] J. Liu, F. Lei, C. Pan, D. Hu, and H. Zuo, "Prediction of remaining useful life of multi-stage aero-engine based on clustering and lstm fusion," *Reliability Engineering & System Safety*, vol. 214, p. 107807, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832021003306

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[10] X. Li, J. Li, L. Zuo, L. Zhu, and H. T. Shen, "Domain adaptive remaining useful life prediction with transformer," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–13, 2022.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[12] H. D. Nguyen, N. H. Long, K. N. Ha, N. V. Hoang, T. T. Huong, and K. P. Tran, "Translighter: A light-weight federated learning-based architecture for remaining useful lifetime prediction," *Computers in Industry*, vol. 148, p. 103888, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361523000386

[13] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon, "Fnet: Mixing tokens with fourier transforms," 2022.

[14] F. Zeng, Y. Li, Y. Jiang, and G. Song, "A deep attention residual neural network-based remaining useful life prediction of machinery," *Measurement*, vol. 181, p. 109642, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224121006114

[15] Y. Song, S. Gao, Y. Li, L. Jia, Q. Li, and F. Pang, "Distributed attention-based temporal convolutional network for remaining useful life prediction," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9594–9602, 2021.

[16] L. Biggio, A. Wieland, M. A. Chao, I. Kastanis, and O. Fink, "Uncertainty-aware prognosis via deep gaussian process," *IEEE Access*, vol. 9, pp. 123 517–123 527, 2021.

[17] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2017, pp. 88–95.

[18] J. Yao, B. Lu, and J. Zhang, "Tool remaining useful life prediction using deep transfer reinforcement learning based on long short-term memory networks," *The International Journal of Advanced Manufacturing Technology*, vol. 118, no. 3, pp. 1077–1086, 2022.

[19] Y. Zhang, Y. Xin, Z. wei Liu, M. Chi, and G. Ma, "Health status assessment and remaining useful life prediction of aero-engine based on bigru and mmoe," *Reliability Engineering & System Safety*, vol. 220, p. 108263, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832021007389

[20] J. Zeng and Z. Liang, "A deep gaussian process approach for predictive maintenance," *IEEE Transactions on Reliability*, 2022.

[21] H. Mo and G. Iacca, "Evolutionary neural architecture search on transformers for rul prediction," *Materials and Manufacturing Processes*, vol. 0, no. 0, pp. 1–18, 2023. [Online]. Available: https://doi.org/10.1080/10426914.2023.2199499

[22] Y. Zhu, Z. Liu, Z. Luo, C. Du, and H. Wang, "Aircraft engine remaining life prediction method with deep learning," in *2022 International Conference on Artificial Intelligence and Computer Information Technology (AICIT)*, 2022, pp. 1–6.

[23] H. Liu, Z. Liu, W. Jia, and X. Lin, "Remaining useful life prediction using a novel feature-attention-based end-to-end approach," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1197–1207, 2021.

[24] D. Kapoor, D. Gupta, S. AgarwaI, M. Uppal, S. Juneja, and M. K. Sharma, "Starnet: Stacked transfer-aware for robust remaining useful life prediction for c-mapss multi-regime engines," *IEEE Access*, 2026.

[25] Y. Mo, Q. Wu, X. Li, and B. Huang, "Remaining useful life estimation via transformer encoder enhanced by a gated convolutional unit," *Journal of Intelligent Manufacturing*, vol. 32, pp. 1997 – 2006, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:233649678

[26] T. Jing, P. Zheng, L. Xia, and T. Liu, "Transformer-based hierarchical latent space vae for interpretable remaining useful life prediction," *Advanced Engineering Informatics*, vol. 54, p. 101781, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034622002397

[27] Z. Zhang, W. Song, and Q. Li, "Dual-aspect self-attention based on transformer for remaining useful life prediction," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022.

[28] G. S. Chadha, S. R. B. Shah, A. Schwung, and S. X. Ding, "Shared temporal attention transformer for remaining useful lifetime estimation," *IEEE Access*, vol. 10, pp. 74 244–74 258, 2022.

[29] H.-K. Wang, Y. Cheng, and K. Song, "Remaining useful life estimation of aircraft engines using a joint deep learning model based on tcnn and transformer," *Computational Intelligence and Neuroscience*, vol. 2021, 2021.

[30] S. Kamei and S. Taghipour, "A comparison study of centralized and decentralized federated learning approaches utilizing the transformer architecture for estimating remaining useful life," *Reliability*

Engineering & System Safety, vol. 233, p. 109130, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832023000455

[31] Z. Xu, Y. Zhang, J. Miao, and Q. Miao, "Global attention mechanism based deep learning for remaining useful life prediction of aero-engine," *Measurement*, vol. 217, p. 113098, 2023.

[32] F. Xiang, Y. Zhang, S. Zhang, Z. Wang, L. Qiu, and J.-H. Choi, "Bayesian gated-transformer model for risk-aware prediction of aero-engine remaining useful life," *Expert Systems with Applications*, p. 121859, 2023.

[33] S. Lv, S. Liu, and H. Li, "New method for remaining useful life prediction based on recurrence multi-information time-frequency transformer networks: Rul prediction with recurrence multi-information tf transformers," *Quality and Reliability Engineering International*, vol. 41, no. 5, pp. 1643–1663, 2025.

[34] C. Zhao, H. Shi, X. Huang, and Y. Zhang, "A multiple conditions dual inputs attention network remaining useful life prediction method," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108160, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095219762400318X

[35] Z. Huang, Y. He, and B. Sick, "Spatio-temporal attention graph neural network for remaining useful life prediction," 2024.

[36] P. R. de O. da Costa, A. Akcay, Y. Zhang, and U. Kaymak, "Remaining useful lifetime prediction via deep domain adaptation," 2019.

[37] H. Li, Y. Li, Z. Wang, and Z. Li, "Remaining useful life prediction of aero-engine based on pca-lstm," in *2021 7th International Conference on Condition Monitoring of Machinery in Non-Stationary Operations (CMMNO)*, 2021, pp. 63–66.

[38] Y. He, H. Su, E. Zio, S. Peng, L. Fan, Z. Yang, Z. Yang, and J. Zhang, "A systematic method of remaining useful life estimation based on physics-informed graph neural networks with multisensor data," *Reliability Engineering & System Safety*, vol. 237, p. 109333, 2023.

[39] J. Chen, Z. Chen, J. Xia, R. Huang, and W. Li, "Multi-granularity cross-domain temporal regression network for remaining useful life estimation of aero engines," in *2022 International Conference on Sensing, Measurement & Data Analytics in the era of Artificial Intelligence (ICSMD)*, 2022, pp. 1–6.

[40] G. Kim, J. G. Choi, and S. Lim, "Using transformer and a reweighting technique to develop a remaining useful life estimation method for turbofan engines," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108475, 2024.

[41] S. Deng and J. Zhou, "Prediction of remaining useful life of aero-engines based on cnn-lstm-attention," *International Journal of Computational Intelligence Systems*, vol. 17, no. 1, p. 232, 2024.

[42] T. Zhang, L. Jiang, R. Huang, and X. Zhang, "A multi-scale cross-channel attention network for remaining useful life prediction with variable sensors," *IEEE Transactions on Instrumentation and Measurement*, 2025.

[43] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *Mechanical systems and signal processing*, vol. 151, p. 107398, 2021.

[44] M. Arias Chao, C. Kulkarni, K. Goebel, and O. Fink, "Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics," *Data*, vol. 6, no. 1, p. 5, 2021.

[45] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 International Conference on Prognostics and Health Management*, 2008, pp. 1–9.

[46] Y. Mo, Q. Wu, X. Li, and B. Huang, "Remaining useful life estimation via transformer encoder enhanced by a gated convolutional unit," *Journal of Intelligent Manufacturing*, vol. 32, pp. 1997–2006, 2021.

[47] H. Tian, L. Yang, and B. Ju, "Spatial correlation and temporal attention-based lstm for remaining useful life prediction of turbofan engine," *Measurement*, vol. 214, p. 112816, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224123003809

[48] H. Du Nguyen, K. P. Tran, P. Castagliola, and F. M. Megahed, "Enabling smart manufacturing with artificial intelligence and big data: a survey and perspective," in *Advanced Manufacturing Methods*. CRC Press, 2022, pp. 1–26.

[49] K. T. Nguyen, K. Medjaher, and C. Gogu, "Probabilistic deep learning methodology for uncertainty quantification of remaining useful lifetime of multi-component systems," *Reliability Engineering & System Safety*, vol. 222, p. 108383, 2022.