applied sciences

MDPI

Article

Remaining Useful Life Estimation of Turbofan Engines with Deep Learning Using Change-Point Detection Based Labeling and Feature Engineering

Kıymet Ensarioğlu [1,2,*], Tülin İnkaya [1] and Erdal Emel [1]

[1] Department of Industrial Engineering, Faculty of Engineering, Bursa Uludağ University, Bursa 16059, Türkiye; tinkaya@uludag.edu.tr (T.İ.); erdal@uludag.edu.tr (E.E.)
[2] Department of Industrial Engineering, Faculty of Engineering, Yalova University, Yalova 77200, Türkiye
* Correspondence: kiymet.ensarioglu@yalova.edu.tr

Abstract: Accurate remaining useful life (RUL) prediction is one of the most challenging problems in the prognostics of turbofan engines. Recently, RUL prediction methods for turbofan engines mainly involve data-driven models. Preprocessing the sensor data is essential for the performance of the prognostic models. Most studies on turbofan engines use piecewise linear (PwL) labeling, which starts with a constant initial RUL value in normal/healthy operating time. In this study, we designed a prognostic procedure that includes difference-based feature construction, change-point-detection-based PwL labeling, and a 1D-CNN-LSTM (one-dimensional–convolutional neural network–long short-term memory) hybrid neural network model for RUL prediction. The procedure was evaluated on the subset FD001 of the C-MAPSS dataset. The proposed procedure was compared with machine learning and deep learning models with and without the new difference feature. Also, the results were compared with the studies that used similar labeling approaches. Our analysis of the numerical results underscores the clear superiority of the proposed 1D-CNN-LSTM model with the difference feature in RUL prediction, with a score of 437.2 and an RMSE value of 16.1. This result illustrates the superior predictive capability of the 1D-CNN-LSTM model, which outperformed traditional machine learning methods and one of the earliest deep learning methods. These findings emphasize the superior predictive capability of the 1D-CNN-LSTM model and underline the potential of the feature engineering process for more accurate and robust RUL prediction in the context of turbofan engine prognostics.

Keywords: remaining useful life; prognostics and health management; feature engineering; turbofan engines; change point detection; convolutional neural network; long short-term memory network

check for updates

Citation: Ensarioğlu, K.; İnkaya, T.; Emel, E. Remaining Useful Life Estimation of Turbofan Engines with Deep Learning Using Change-Point Detection Based Labeling and Feature Engineering. *Appl. Sci.* **2023**, *13*, 11893. https://doi.org/10.3390/app132111893

Academic Editor: Satoru Okamoto

Received: 2 October 2023
Revised: 22 October 2023
Accepted: 25 October 2023
Published: 30 October 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, with technological progress, the complexity of turbofan engines, which are core components of an aircraft, has increased. Nearly 60% of total breakdowns pertain to the turbofan engines of an aircraft [1]. The varying operating hours and conditions affect the performance of turbofan engines. Therefore, it is necessary to determine and implement strategies to ensure safety and reliability. Maintenance strategies like corrective or preventive maintenance have been insufficient to meet the reliability and safety issues requirements. The prognostics and health management (PHM) concept, which is also named condition-based maintenance (CBM) and predictive maintenance (PdM), has gained more attention thanks to its ability to overcome emerging problems [2]. PHM ensures systems work optimally as designed, prevents failure, reduces maintenance costs, and monitors and manages the health of the systems and equipment [3]. As stated in the 2004 International Organization for Standardization [4], prognostics mainly focuses on providing the health state of the system and predicting failure modes and the RUL of components within that system [5]. The RUL is the time between the current time and

when a component or a system no longer maintains its healthy state and useful life [6]. RUL prediction allows for reducing costs, scheduling appropriate maintenance and replacement plans, and preventing occupational accidents.

PHM plays a critical role in industries like aviation, where reliability is paramount due to the potentially devastating impact of accidents in terms of human life and economic losses. In this study, we focused on turbofan engines because a turbofan engine is the most critical component of an aircraft. It is one of the most complex components that require high reliability and quality [7]. RUL prediction is essential for developing maintenance strategies to reduce the cost and reliability aspects of turbofan engines [8].

Existing RUL prediction methods can generally be categorized into model-based, data-driven, and hybrid approaches [9,10]. Model-based approaches are helpful methods that accurately describe system degradation using mathematical models [11]. However, building mathematical models in complex systems can be difficult and costly, requiring expertise and domain knowledge about physical systems. Data-driven approaches can reveal the characteristics of system degradation by converting historical sensor data into useful information. With advanced sensor and computer technologies, data-driven approaches have been widely used to learn about complex system degradation with less expertise in industry and academia. Hybrid approaches are methods in which the two aforementioned approaches are used together to take advantage of model-based and data-driven approaches. However, hybrid approaches also have limitations due to both methods, making the use of hybrid approaches challenging. In this study, data-driven approaches were considered in the RUL prediction of turbofan engines.

Data-driven approaches are based on statistical techniques and artificial intelligence [12]. Statistical methods use models with various assumptions to probabilistically predict the RUL [6]. Statistical techniques, such as the Wiener process [13] and gamma process [14] are used for degradation modeling, and Markovian-based models [15] are used as an RUL prediction model for turbofan engines.

Artificial intelligence utilizes machine learning methods, mainly deep learning techniques [16]. For supervised learning, signal data needs target values that indicate the degradation process. The PwL function is mainly used to construct a degradation model [17]. This model assumes that the RUL is constant until a certain point of failure. After this point of failure, it decreases linearly. For generalization and reducing the complexity of RUL prediction problems, all engines use the same constant RUL value during the normal/healthy operation of the engines [7,9,18,19].

Deep-learning-based RUL prediction methods have produced significant results for turbofan engines. Several studies on turbofan engines used deep learning techniques, such as recurrent neural networks (RNNs) to analyze time series data. An RNN, which is a deep learning technique for time series data, can successfully obtain short-term dependencies but has problems learning long-term dependencies. A long short-term memory (LSTM) network controls the flow of time-dependent information through gates to overcome the long-term dependency problem. Zheng et al. [11] proposed an LSTM model to make RUL predictions using the time-sequence data from the sensors. They achieved better RUL prediction accuracy compared with the hidden Markov and traditional RNN models. Wang et al. [20] used sensor data sequence with the bidirectional long short-term memory (BiLSTM) network for RUL prediction and stated that the BiLSTM network outperformed the traditional machine learning approaches. Wu et al. [21] used sensor smoothing and selection and proposed a deep LSTM network for RUL prediction. They tested the proposed approach using the C-MAPSS sub-datasets FD001 and FD003 [22]. To solve the high-dimensionality problem of C-MAPSS data, an autoencoder (AE) was used to compress the sensor measurements, and a BiLSTM model was proposed with the aim of better prediction [6]. Dual-channel LSTM [19] was used with the first-order difference of sensor measurements and extracted more information about the degradation process of the engines. By classifying the degradation phases, multi-scale LSTM was applied to predict the RUL

after the constant value degradation phase [23]. They achieved a 40% reduction in the root-mean-square error (RMSE) for the FD001 dataset.

Another deep learning technique, namely, CNN, has been broadly used in turbofan RUL prediction due to its superiority in feature extraction. A deep CNN approach with a time window [9] was applied to normalized raw C-MAPSS data. Muneer et al. [24] combined deep CNN with an attention mechanism to extract highly abstract degradation and trend features. Temporal convolution expanded the receptive field for long sequences to improve the prediction performance [25]. Tan et al. [26] also used temporal convolution with an attention mechanism for selecting relevant information from a series of sensor measurements. The multi-scale deep CNN method with different-sized filters was proposed to achieve complex features for the degradation process and had a high prognostic performance [27]. Also, the spatio-temporal attention mechanism with position encoding was applied to capture the temporal relation between features [28].

Various types of CNNs and RNNs have been used in the literature in a hybrid fashion to improve generalization and consider temporal and spatial features of sensor measurements. A CNN and LSTM were combined within a deep network in a parallel manner for complicated prognostic scenarios [29]. The health index (HI), which was calculated from raw sensor measurements, was used as a new time series and fed to the network, which consisted of serial CNN-LSTM layers [30]. Furthermore, a double-channel hybrid deep neural network containing CNN and BiLSTM layers was applied for better prediction performance [18]. A lower-dimensional projection of the sensor measurements obtained from a CNN-based autoencoder method was used as input for CNN-LSTM serial layers in [31]. The features from a one-dimensional fully convolutional neural network and LSTM network were fused and then fed into the following CNN network to improve the effectiveness of the prediction model [1].

A group of studies used an attention mechanism, which converts the original input into weighted input to represent the features better. For instance, an attention mechanism was used in the bidirectional gated recurrent units (BiGRUs)-CNN hybrid neural network in [32]. Gong et al. [33] used an attention-based temporal convolutional network (TCN) and the BiGRUs for RUL prediction. A random forest (RF) algorithm was used to rate the importance of the features and implement variational mode decomposition on these features.

Another group of studies focused on constructing new features to contribute to the learning of degradation processes. By taking the first-order differences for the time series of sensor measurements, new features were created and added to the dataset [19,34]. Correlation-based degradation features were used in [35], while mean and trend coefficients were used in [36]. A complete ensemble empirical mode decomposition method was used to create sensor measurement trend features [37]. Bae and Xi [38] transformed the time-based cycle feature into a physical health time step to provide meaningful features.

The studies presented above used a PwL function for target RUL labeling. The constant and maximum part of the piecewise RUL, mostly named RUL_{max}, is common for all the engines in the dataset. This approach is helpful for generalization but does not accurately reflect the situation of turbofan engines in the real world. The degradation process of each engine differs from the other. The studies on engine-specific PwL target labeling are less frequent than studies using a common RUL_{max} for all engines. The main reason is the lack of sufficient data to differentiate engines from each other while predicting the RUL in the healthy state of the engine. Also, incomplete sensor data in the test dataset hinders precise change point determination. Most studies using the engine-specific RUL_{max} focused only on the training dataset. In general, health state (HS)-division-based specific RUL labeling was studied in a dual-task manner for both RUL prediction and HS evaluation simultaneously [39,40]. A dual-LSTM framework was used for change point detection in RUL labeling and HI-based RUL prediction [41]. The RUL prediction in the healthy state is difficult due to the lack of information about the degradation process, and thus, the RUL prediction was made using the sensor measurements after the change point [42,43].

The challenge of predicting the RUL is not confined solely to turbofan engines but spans across diverse domains, encompassing applications in areas such as Li-ion batteries and bearings, where RUL prediction plays a prominent role. In [44], an LSTM and gradient boosting machine (GBM) were utilized to analyze Li-ion batteries, combined with explainable artificial intelligence techniques for feature selection. The study in [45] introduces a novel approach for improving the accuracy of bearings' RUL prediction, combining a multi-branch convolutional network (MBCNN) with global attention and a BiLSTM network, utilizing both spatial and timing features from vibration signals, which were ultimately tested on a public bearing degradation dataset. Also, the significance of RUL labeling on RUL prediction is demonstrated in the study by [46] with load calculations, which assessed bearings.

When considering the studies conducted not only within the realm of RUL prediction for turbofan engines but also in various other applications, it becomes evident that there is an apparent necessity for comprehensive and pioneering research in the domains of target labeling approaches, feature engineering processes, and the development of effective network models.

This study primarily investigated data-driven approaches, specifically deep learning techniques, for turbofan engine RUL prediction. Various deep learning methods, including long short-term memory (LSTM) and convolutional neural networks (CNNs), were examined, and this paper proposes a novel approach that combines change-point-detection-based target labeling and feature construction for RUL prediction. This research aimed to improve the accuracy of RUL prediction in different stages of the engine's life and used engine-specific RUL labels for a more realistic representation of engine behavior. This paper presents a comprehensive experimental study using turbofan data and the results were compared with existing methods to demonstrate the proposed method's effectiveness.

The main contributions of this paper are as follows:

1. To the best of our knowledge, this study was the first to combine change-point-detection-based target labeling and feature construction in RUL prediction.
2. In line with the current trend in RUL prediction, we embraced a flexible target-labeling approach, employed innovative feature engineering strategies, and introduced an efficient hybrid network to enhance prediction accuracy while minimizing computational complexity, as validated through comparisons.
3. Most studies used PwL target labels for RUL prediction with a constant RUL_{max} value like 120, 125, and 130. However, this study used an engine-specific PwL target label for each engine.
4. Previous studies used the original training dataset for both training and testing by splitting it into two disjoint sets. Different from them, this study focused on predicting the RUL using the original test dataset to ensure a fair evaluation.
5. We fitted continuous PwL functions on the one-dimensional data that was fused using an autoencoder-based feature extraction method for engine-specific target RUL labeling. We use the Python library pwlf, in which the unknown breakpoints are determined with the differential evolution-based optimization algorithm. An increasing number of line segments were used until there was no significant improvement in finding the first change point. The earliest first change point was selected for piecewise target RUL labeling.
6. A handcrafted feature was constructed, which includes the difference between the sensor measurements in every cycle and the first sensor measurement, and an added feature to the dataset. Thus, RUL prediction was improved in the middle- and early-life stages.

The remainder of this paper starts with the theoretical background of the deep learning methods and feature engineering approaches in Section 2, along with the proposed architecture of the neural networks used in this study. The experimental study was performed using the FD001 sub-dataset of the C-MAPSS dataset. The results are compared with the studies using similar labeling methods in Section 3. Finally, Section 4 concludes the paper and proposes future work.

2. Methodology

This section introduces the theoretical background of the proposed study for RUL prediction. First, the deep learning techniques CNN and LSTM are defined. Next, the conceptual framework of the autoencoders employed in the feature engineering process is elucidated. Next, the feature engineering approaches are explained. Finally, the proposed deep network structure is given.

2.1. Convolutional Neural Networks

A CNN is a deep learning neural network commonly used for image classification, object detection, and other computer vision tasks [47]. The main component of a CNN is the convolution layer, which applies multiple filters/kernels to the input image to produce a set of feature maps. The convolution operation slides the filter over the input data and computes the dot product between the filter weights and the input data at each step. This operation produces a feature map. The size of the output feature map depends on the size of the input data, the size of the filter, and the stride and padding used during the convolution. After the convolution operation, it applies an activation function like rectified linear unit (ReLU) or sigmoid. This step introduces nonlinearity to the model. The feature maps obtained are then fed into other layers within the CNN, such as pooling and fully connected layers. These layers utilize the extracted features to perform classification or regression. The following equation is used to perform a convolution operation:

$$y_i = (X * k_i) + b_i \tag{1}$$

where X is the input tensor, k_i is the ith convolution kernel, b_i is the bias vector, y_i is the ith obtained feature map, and $*$ denotes the convolution operation. These feature maps are passed through an activation function σ, as expressed

$$m_i = \sigma(y_i) \tag{2}$$

A subsequent pooling operation downsamples the feature maps produced by the convolutional layers, reducing the spatial size of the data and the number of parameters in the model. Max pooling is one of the most commonly used pooling types in CNNs and is functionally denoted as

$$m_i^l = max\left(m_i^{l-1}, p, s\right) \tag{3}$$

yielding m_i^l as the ith feature map in the pooling layer l, where its parameter m_i^{l-1} is the ith feature map in layer $l-1$, p is the pooling size, and s is the stride. The fully connected layers connect every neuron in one layer to every neuron in the next layer, and they are typically used toward the end of the CNN to perform classification or regression on the extracted features.

One-dimensional CNNs, which are variations of CNNs, are designed to process sequential data, such as time series [48]. The number of dimensions of the CNN refers to the number of dimensions over which the filter/kernel slides. The input and output of a 1D-CNN are two-dimensional. The first dimension is timesteps and the other is features. The basic architecture of a 1D-CNN is similar to a regular CNN, but the filters are applied only along the timesteps dimension as stated in [49]. The 1D convolution operation for multivariate time series data is depicted in Figure 1, where the height represents the number

of time steps, its width is one, and the number of input features is equivalent to the depth or the number of channels. The lines represent filtering operations with 1D kernels.

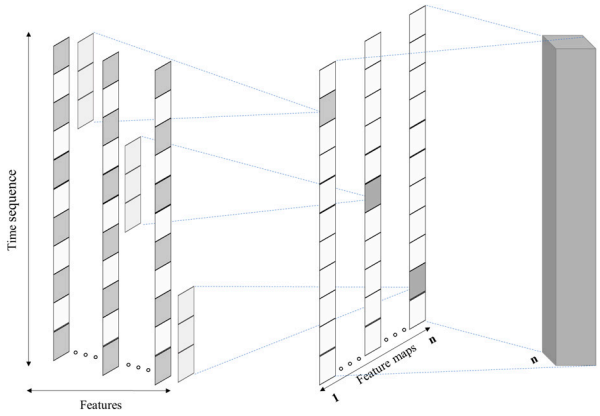


Figure 1. One-dimensional convolution operation for multivariate time series [49].

One of the main advantages of using 1D-CNNs is that they can capture local patterns and features within the sequential data, which can be helpful for tasks such as detecting time-dependent anomalies. Additionally, 1D-CNNs require less computational resources than other types of deep learning models, which can be helpful in applications with limited resources [48,50].

Overall, CNNs and 1D-CNNs are powerful deep learning models that have been shown to achieve successful results on a wide range of computer vision and sequential data tasks.

2.2. Long Short-Term Memory Network

An LSTM network is a type of RNN architecture introduced by [51]. LSTM was designed to handle the vanishing gradients issue for long-time sequences in traditional RNNs. These vanishing gradients make it difficult for the RNN to learn long-term dependencies effectively.

LSTMs can keep information over a longer period by incorporating a memory cell. An LSTM controls the short-term and long-term cell states via three gates. These are the forget, input, and output gates, operating in the specified order. The basic structure of an LSTM cell is shown in Figure 2. The input of the current cell is x_t and the short-term and long-term states of the previous cell are h_{t-1} and c_{t-1}, respectively. During the operation of the gates, three signals are generated from the gates and denoted as f_t, i_t, g_t, and o_t, respectively.

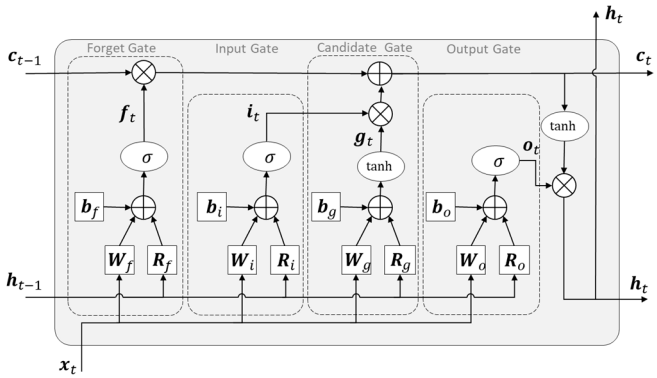


Figure 2. The basic structure of an LSTM cell.

The forget gate controls how much of the previous long-term cell state c_{t-1}'s information is retained or forgotten. The forget gate signal is generated through a sigmoid function $\sigma(.)$ using the short-term state of the previous cell, namely, h_{t-1}, and input of the current cell, namely, x_t:

$$f_t = \sigma\left(W_f x_t + R_f h_{t-1} + b_f\right) \tag{4}$$

where W_f and R_f are the weights and b_f is the bias.

The input gate controls the influence of the new information on the current cell with Equation (5):

$$i_t = \sigma(W_i x_i + R_i h_{t-1} + b_i) \tag{5}$$

where W_i and R_i are the weights and b_i is the bias. The current cell state candidate values g_t are created, and the previous long-term cell state is updated into a new state using the following equations:

$$g_t = tanh\left(W_g x_t + R_g h_{t-1} + b_g\right) \tag{6}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes g_t \tag{7}$$

where W_g and R_g are the weights, b_g is the bias, $\otimes$ is element-wise multiplication, $tanh$ is the tangent hyperbolic activation function, and c_t is the updated long-term cell state. The output of the LSTM cell is obtained with the following two equations:

$$o_t = \sigma(W_o x_t + R_o h_{t-1}, b_o) \tag{8}$$

$$h_t = o_t \otimes tanh(c_t) \tag{9}$$

where W_o and R_o are the weights and b_o is the bias.

The updated long-term cell state c_t and the short-term cell state h_t are then passed to the next time step as input. This process of updating the cell and hidden state is repeated at each time step of the input sequence.

2.3. Autoencoders

Autoencoders are a type of neural network architecture, where the output is the same as the input, which can be used for unsupervised learning tasks, such as data compression, feature extraction, and dimensionality reduction [7,52]. The main components of an autoencoder include an encoder network, a decoder network, and a bottleneck layer, as shown in Figure 3.

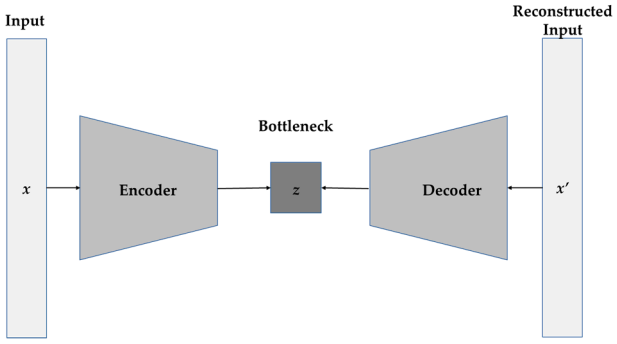


Figure 3. Main components of an autoencoder.

The encoder takes the condition monitoring input data x and maps it to a lower-dimensional latent space representation z. The output of the encoder is

$$z = f(Wx + b) \tag{10}$$

Encoding is typically done through a series of hidden layers that gradually reduce the number of nodes in the network. The bottleneck is the layer in the middle of the network where the input data is compressed into the lower-dimensional latent space representation. The size of the bottleneck layer determines the dimensionality of the latent space representation. The decoder takes the latent space representation z and maps it back to the reconstructed input data x':

$$x' = f(W'z + b') \tag{11}$$

Like the encoder, the decoder typically consists of a series of hidden layers that gradually increase the number of nodes in the network. The autoencoder is trained to minimize the difference between the original input data and the reconstructed input data by the decoder. A loss function, such as the mean squared error (MSE) or binary cross-entropy, is minimized.

Autoencoders can capture complex nonlinear relationships between variables, making them suitable for representing high-dimensional data with nonlinear structures [7].

2.4. Feature Engineering

Feature engineering comprises selecting, processing, and transforming raw data into features that can be used as inputs to a machine learning algorithm [53]. It is a crucial step in the machine learning pipeline, as the quality of features can significantly impact the accuracy and performance of the model.

Feature selection, scaling, extraction, and encoding are the standard techniques used in feature engineering [53]. Feature selection involves selecting a subset of the most relevant features from the available data. Feature scaling helps to normalize or standardize the values of features to ensure they are on a similar scale. One of the scaling operations is min–max normalization:

$$x^*_{i,j} = \frac{x_{i,j} - x^{min}_j}{x^{max}_j - x^{min}_j}, \qquad \forall i, j \tag{12}$$

where $x^*_{i,j}$ represents the normalized value of the ith data point for the jth feature. $x_{i,j}$ represents the raw value of the data before normalization, and x^{max}_j and x^{min}_j are the maximum and minimum values of the jth feature, respectively.

Feature extraction involves constructing and transforming raw data into meaningful features to be used as inputs of a machine learning model. Dimensionality reduction and decomposing are some of the feature extraction techniques. An autoencoder-based dimensionality reduction method is employed for feature extraction.

When the relationships between the features and the target variables are not straightforward, the feature construction technique is valuable for creating new features from the existing data. In order to improve the prediction performance, this work hypothesized the engine's degradation tendency to represent a new feature. The difference between the first sensor measurement when the engine starts to operate and the last measurement is calculated, and the process is illustrated in Algorithm 1.

In order to show the relationship between the feature $diff_t$ and the hypothesized engine's degradation tendency, the correlation coefficient for the difference and the target value is calculated with the following formula:

$$\rho_{x,\,y} = \frac{\sum (x_i - x)(y_i - y)}{\sqrt{\sum (x_i - x)^2 (y_i - y)^2}} \tag{13}$$

where x_i and y_i are the individual sample points for the input feature and the engine's degradation, respectively, and x and y are the mean values of those samples, respectively. Thus, based on the hypothesized correlated results, a new difference feature is intended to be added to the training dataset for RUL prediction.

Algorithm 1 Constructing and adding new features

input: Data sequence $X = (x_1, x_2, \ldots, x_T)$
parameters: $T \leftarrow$ length of the sequence, $t \leftarrow$ current time step $t \in [1, T]$
output: Augmented data sequence X'
begin
    D $\leftarrow$ []
    for $t \leftarrow 1$ to T **do**
    $diff_t \leftarrow x_t - x_1$
        $D \leftarrow D + [diff_t]$
    end
$X' \leftarrow X$+D
end

2.5. Change Point Detection

Change point detection is crucial in detecting the early signs of deterioration to prevent industrial equipment from unexpected disruptions [54].

The PwL fitting method [55] is used for change point detection in generating the target labels. It fits PwL functions to predict nonlinear or multiple trends in the data. By optimizing segment positions and associated slopes, PwL fitting provides a broad perspective and capacity to model changes in data patterns. This makes it particularly valuable in applications such as change point detection and trend analysis.

The Python library pwlf [55] performs a least squares fit, which solves for the β parameters that minimize the sum-of-squares error of the residuals for any given set of breakpoint locations b if the change points are known. The sum of the squares of the residuals can be expressed as a function dependent on the change point locations $SSR(b)$, and the optimization problem is as follows:

$$\begin{gathered} minimize\ \ SSR(b),\ \ b = \left[b_2,\ \ldots,\ b_{n_b-1}\right]^T \\ subject\ to\ x_1 \leq b_k \leq x_n,\ \ k = 1,\ 2,\ \ldots,\ n_b \end{gathered}$$

where x is the 1D data, n_b is the number of change points, and b_{n_b} is the last breakpoint. $n_b - 1$ is the number of line segments. The library assumes that b_1 is the first value and b_{n_b} is the last value of the one-dimensional dataset. Differential-evolution-based global optimization is used to find the best change point locations when the change points are unknown but the desired number of segments is known.

Although PwL fitting is not a change point detection algorithm per se, it has the same principle as the PwL target labeling approach. It is helpful in the early detection of the first change point with the practical usage of a line segment parameter. The change point detection method based on pwlf offers the advantage of the early detection of breakpoints through its line segment parameter. It also possesses the potential to address the challenges related to the HS division problem of engine degradation processes.

2.6. Proposed Network Structure

The structure of the deep network proposed in this study is illustrated in Figure 4. First, a dimension-reduction-based autoencoder is used as a preprocessing step that helps to detect the change point by reducing the multi-sensor turbofan engines to a single signal. The input sample is shaped into a 2-dimensional tensor as $N_{tw} \times N_{ft}$, where N_{tw} and N_{ft} denote the time sequence length and the number of the features, respectively. The details of the input data preparation are described in Section 3.1.

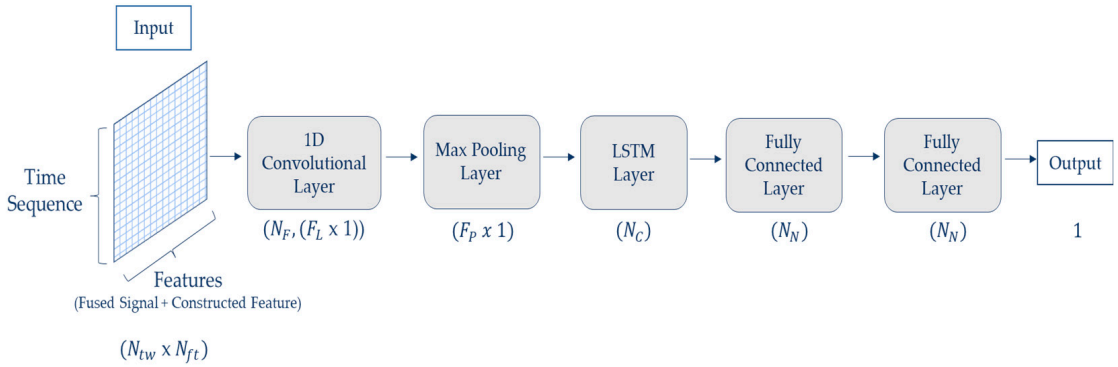


Figure 4. Proposed network structure.

A 1D-CNN is employed in the first layer to extract spatial features from the fused sensor measurement and the newly added feature. Then, a max-pooling layer is used for reduced complexity and overfitting. The 1D-CNN layer configuration is $(N_F, (F_L \times 1))$, where N_F is the number of filters and F_L is the filter size. In order to keep the size of the feature map fixed, zero-padding is used. The pooling layer resizes each feature map independently by utilizing the max operation and F_P is the pooling size. An LSTM is used for revealing temporal information from the extracted features and N_C denotes the number of LSTM cells. Next, a fully connected layer smooths the feature matrix and maps these obtained feature vectors to the target labels of the samples. N_N is used to represent the number of neurons used in the fully connected layer. In the end, a fully connected output layer with one neuron is placed to make RUL predictions.

ReLU is the activation function, and the Adam algorithm is the optimization algorithm for the training network.

3. Experimental Study

This section aims to demonstrate the effectiveness of the feature engineering approach and 1D-CNN-LSTM hybrid neural network for RUL prediction. Experiments were conducted on the subset FD001 of the C-MAPSS datasets injected with additive noise. In model training, the MSE of the cross-validation set was used to evaluate the performance of the trained neural networks. Tests were conducted on a separate dataset to analyze the results using the two evaluation metrics, i.e., the RMSE and score function. Also, prediction results were compared with similar works of other researchers.

3.1. Experimental Setup

3.1.1. Dataset Overview

This study adopted the publicly available C-MAPSS turbofan engine sub-dataset FD001 generated by the Commercial Modular Aero-Propulsion System Simulation software (version 2) created by NASA [22]. The dataset consists of simulated sensor signals of hundreds of motors operating until system failure due to varying degrees of deterioration. In this study, the FD001 sub-dataset was selected to verify the effectiveness of the proposed approach.

Detailed information on the FD001 sub-dataset is presented in Table 1. While each engine was set to operate under stationary conditions, the FD001 sub-dataset was intentionally contaminated with noise.

Table 1. Details of the FD001 sub-dataset of the C-MAPSS dataset.

FD001	
Train Trajectories	100
Test Trajectories	100
Operating Conditions	1
Fault Modes	1

The time series data were acquired from 21 sensors for each engine in the dataset. The description of the 21 sensors in the dataset is presented in Table 2. Each engine unit had a certain level of initial wear that was considered normal. Furthermore, engines were healthy in the early stages of operations and tended to degrade over time until a failure occurred.

Table 2. Description of 21 sensors in the C-MAPSS dataset.

Sensor Number	Sensor Description	Units
1	Fan inlet temperature	°R
2	LPC outlet temperature	°R
3	HPC outlet temperature	°R
4	LPT outlet temperature	°R
5	Fan inlet pressure	psia
6	Bypass duct pressure	psia
7	HPC outlet pressure	psia
8	Physical fan speed	rpm
9	Physical core speed	rpm
10	Engine pressure ratio P50/P2	-
11	HPC outlet static pressure	psia
12	Ratio of fuel flow to Ps30	pps/psia
13	Corrected fan speed	rpm
14	Corrected core speed	rpm
15	Bypass ratio	-
16	Burner fuel–air ratio	-
17	Bleed enthalpy	-
18	Required fan speed	rpm
19	Required fan conversion speed	rpm
20	High-pressure turbines cool airflow	lb/s
21	Low-pressure turbines cool airflow	lb/s

The FD001 sub-dataset included training and test sets. In the training set, the time series of 21 sensor values, which were collected periodically at each operation cycle, included the whole lifetime of the engines. In contrast, in the test set, engine sensor readings were terminated at an arbitrary operation cycle before the system failure. The FD001 sub-dataset had 20,631 cycles for the training dataset and 13,096 for the test dataset. The last point of the cycles in the test dataset was the point at which the RUL was predicted. The primary purpose of RUL prediction is to create a model based on the training dataset and to predict the RUL for the endpoint in the test dataset using the proposed model.

3.1.2. Data Preprocessing and Feature Selection

Each training and test dataset of turbofan engines contained outputs from 21 sensors. The characteristics of sensors, such as prognosability, monotonicity, trendability, and robustness, along with detailed analyses, can be found in [33,56–58].

Some sensor measurements did not provide degradation information for RUL prediction since they had constant values throughout the engine's lifetime [8,56]. To choose a subset of informative sensors from the FD001 dataset, the sensors that exhibited trending and constant values as time progressed are categorized in Table 3. As a result, sensors with constant values were discarded and only 14 of them were selected for further analysis as in [8,9,17].

Table 3. Trends of the sensor data in FD001 [17].

Trend	Sensors
Ascending	2, 3, 4, 8, 9, 11, 13, 15, 17
Descending	7, 12, 14, 20, 21
Constant	1, 5, 6, 10, 16, 18, 19

In the C-MAPSS dataset, measurements from different sensors varied in magnitudes and units. In order to suppress the different effects of low and high magnitudes in the model, the sensor data needed to be normalized. The sensor measurements were normalized to the [0, 1] range using the min–max normalization Formula (12), as outlined in Section 2.5, which represents the feature scaling step in the feature engineering process. As an example, the normalized life-cycle data of 14 sensors of engine #1 is illustrated in Figure 5.

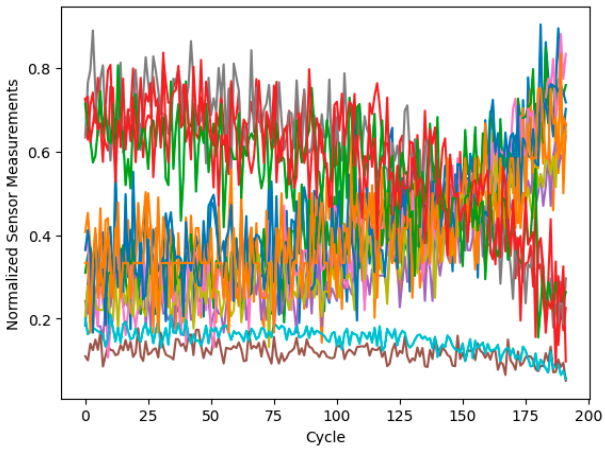


Figure 5. Normalized time series data of 14 sensors of engine #1 in the training set.

As mentioned, the simulated sensor signals were mixed with noise, which could be removed using a low-pass filter to smooth the sensor data. A low-pass filter passes signals with lower frequencies and attenuates signals with higher frequencies, thereby preserving long-term trends while removing small fluctuations and outliers [59]. The sampling frequency of data recordings was 1 Hz. The designed low-pass filter was a first-order filter with a 0.08 Hz critical frequency. The signal of a randomly selected engine sensor from the FD001 sub-dataset is shown in Figure 6 with and without filtering.

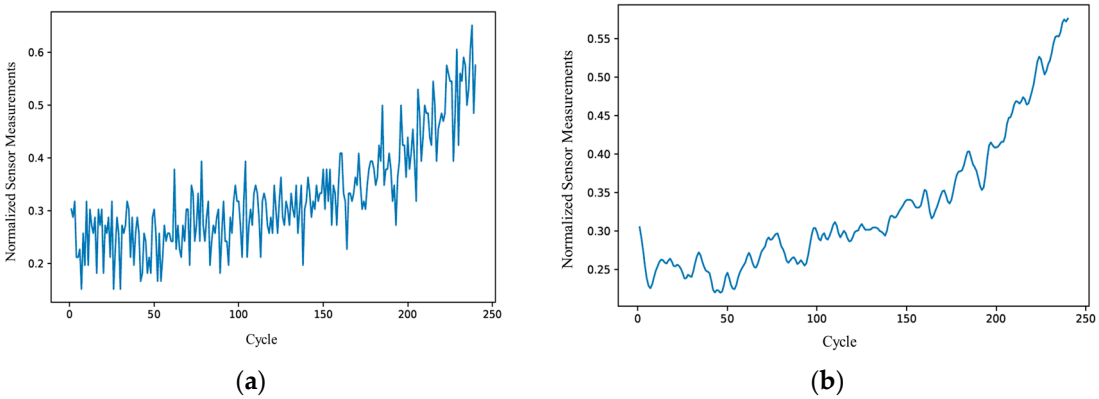


Figure 6. Filtering dataset: (**a**) before filtering; (**b**) after filtering.

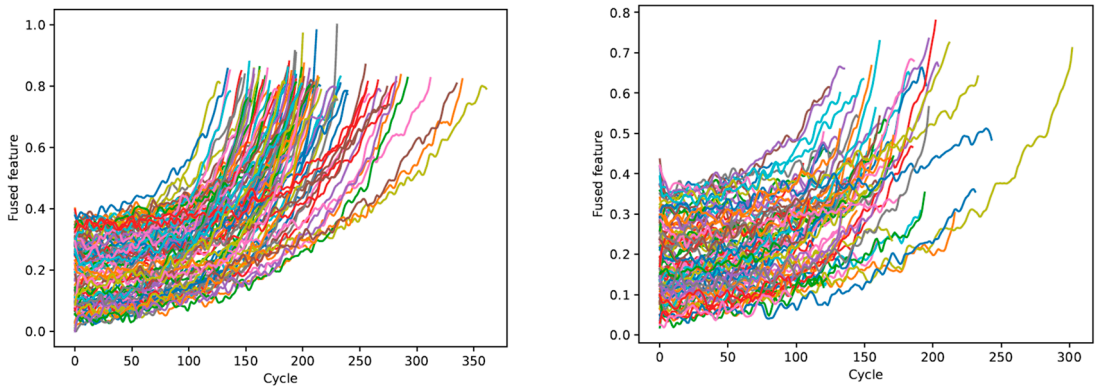
3.1.3. Autoencoder-Based Feature Extraction

In this work, data from 14 selected sensors were processed to extract a single signal via a three-layer autoencoder. Hyperparameters of the autoencoder were evaluated based on the ranges in Table 4, and the parameters that performed best were chosen for optimal performance.

Table 4. Hyperparameter values evaluated in the autoencoder network.

Hyperparameters	Range
Number of epochs	(20, 25, 30, 40)
Batch size	(64, 128, 256)
Number of nodes in hidden layer	(5, 7, 10)

The number of epochs was set to 30, and the batch size was set to 128. The number of neurons in the input layer and the output layer was equal to the number of selected features and was 14. The number of the nodes in the hidden layers, which is shown in Figure 3 as "Encoder" and "Decoder", was set to 7 and the bottleneck size, which was also the desired reduction size of the input, was 1. The MSE was utilized as the loss function, and adaptive moment estimation (Adam) was used to minimize the loss function. A total of 80% of the training dataset was dedicated to the training process, with the remaining 20% used for validation purposes. The conversion process was applied to both the training and test datasets. The sensor signals after the feature extraction process are shown in Figure 7. Signals in different colors represent each engine in the dataset.



(**a**) (**b**)

Figure 7. Extracted sensor signals (**a**) 100 engines for training dataset; (**b**) 100 engines for test dataset.

3.1.4. Sliding Time Window Processing

In time-series-based prognostic problems, temporal information is essential for RUL prediction. Since temporal sequence data are hypothesized to include more information about the current degradation state than single-time step data, this study adopted a sliding time window strategy to exploit temporal information.

The time window was created using a single step sliding over the data with a specific time window length. If L is the number of engine cycles until failure and tw is the time window length, then $L - tw + 1$ samples can be extracted from engine data. The sample at time t can be written as follows:

$$x_t = [x_{t-tw+1}, x_{t-tw+2}, \ldots, x_t] \text{ where } t = tw, \ldots, L \quad (14)$$

As stated in [9,36], a longer time window yields better performance by including more degradation information. However, a longer time window increases the computational complexity. Here, the window size was set to 30 to not exceed the cycle length of the engine with the shortest cycle length in the test dataset. The time windows were designed to overlap, and the step size was set to 1. Sliding time window processing is shown in Figure 8. Using the given window length and step size, 17,731 training samples were obtained. During the testing and training phases, the last sensor measurement data of the last time window was used for the RUL prediction of each engine. As the number of engines in the test data set for FD001 was 100, we had 100 samples of 30-step time windows to test the prediction model.

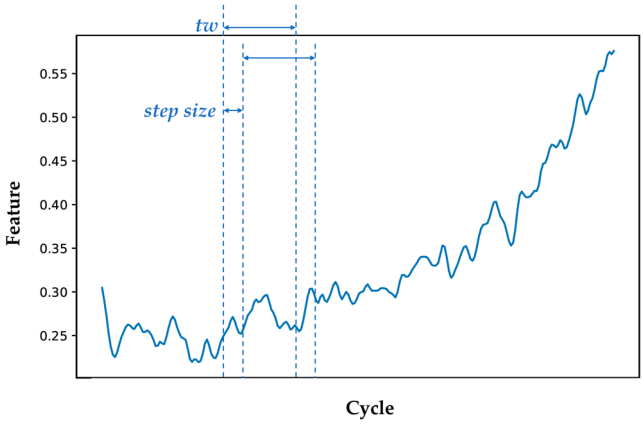


Figure 8. Illustration of sliding time window.

3.1.5. Change-Point-Detection-Based Target Label Construction

Several labeling methods were used for the target RUL labeling because there was no given information in the dataset for RUL labeling. One of the most used models for target RUL labeling is the PwL degradation model proposed by [17]. In PwL-based RUL labeling, target RUL is considered constant due to normal operating conditions for the initial period and linearly decreases after the system begins to degrade. Generally, most studies use around 120–130 cycles as the maximum RUL for any cycle of the constant part of the PwL function [12,17,28]. Although a single constant maximum RUL value for all engines seems convenient for generalization, each engine has a different life-cycle length and degradation process. We used the Python library pwlf to fit continuous PwL functions to one-dimensional time series data for change point detection [55].

To find the earliest change point in the sensor data of each engine, we increased the desired number of line segments until there was no meaningful difference for the subsequent segmentation.

Figure 9 shows this process for a randomly selected engine. Blue lines represent extracted sensor signals, red lines represent fitted linear lines, and dashed yellow lines represent the first breakpoint. The last point of the first line segment gives the first change point of the sensor data. In Figure 9, the line segments are increased stepwise from two to five. It is obvious that when there are two segments, the detection of the first change point becomes late. The number of desired line segment parameters was set to "5" because there was no significant improvement in detecting the first change point. When this process was applied to all engines for two PwL segments, the target RUL labels of these engines are observable in Figure 10.

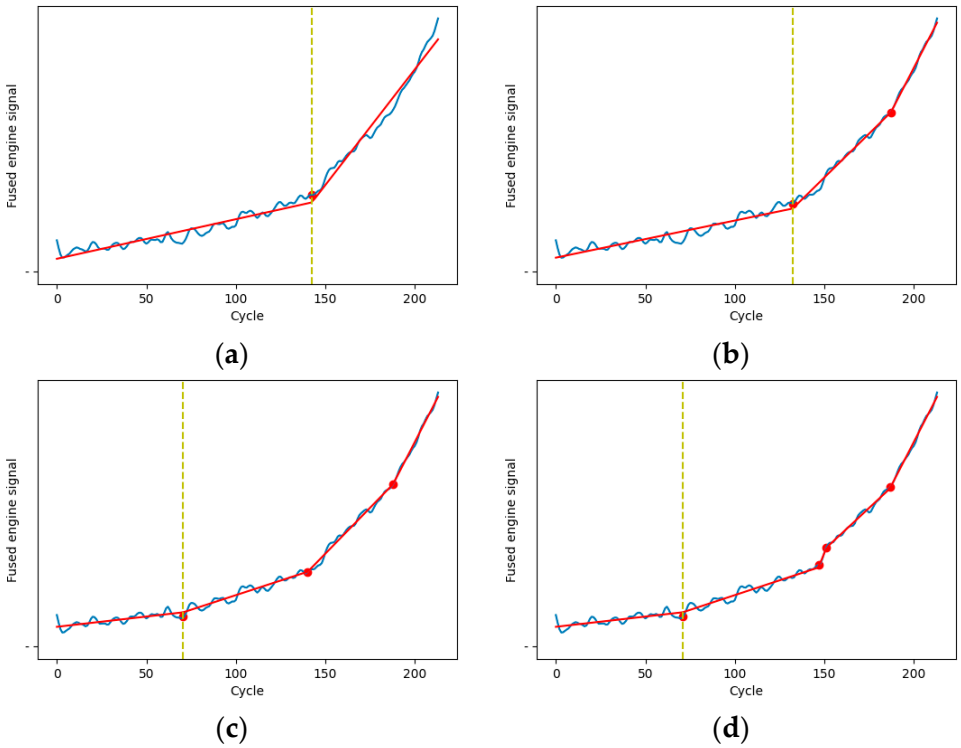


Figure 9. Change point detection process of a randomly selected engine from FD001 by changing line segment parameter: (**a**) 2 line segments; (**b**) 3 line segments; (**c**) 4 line segments; (**d**) 5 line segments.

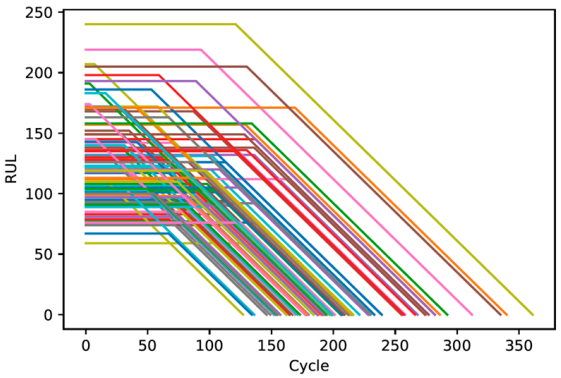


Figure 10. RUL labels for all engines in the training dataset with maximum RUL specific to each engine.

3.1.6. Feature Construction

The entirety of the dataset comprised the sensor readings obtained from all engines. In order to find out the relationship between the sensor data and the RUL, we investigated the correlation of the new difference feature, i.e., the difference between the last and the first data point in sensor signals of the training set with the given RUL values. We truncated the signals of engines at random cycles in the training dataset as in the test dataset to ensure the relationship was valid for both datasets. The scatter diagram in Figure 11 indicates the existence of such a relationship.

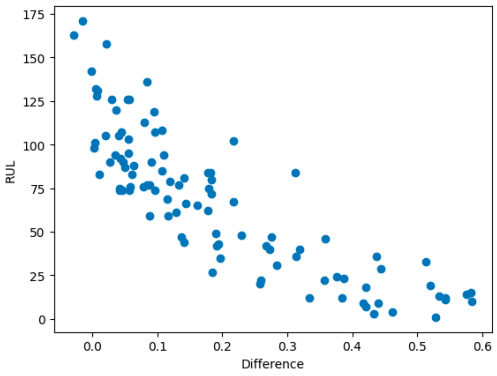


Figure 11. Relationship between the RUL versus the difference between the last and first data points for all engines in training set.

After performing the correlation analysis between the RUL and the difference of the last and first points in the sensor signals using Equation (13), the correlation coefficient was found to be $\rho = -0.85$ with a p-value < 0.00. Based on this, we concluded that there was a robust negative relationship between the RUL and the difference between the points mentioned above.

Another conclusion from this relationship is that it provides valuable information for the prediction model to differentiate the engines with different maximum RUL levels from the starting point to the change point despite utilizing the same sensor measurements.

As in Algorithm 1, construction of the feature from a given engine's fused sensor time series data was carried out at each time step by taking the difference to the first time series sample of the measurements, thereby achieving an equal sample size with fused sensor data; this feature was then used with the sensor data as a new dataset for normalization and sliding time window processing.

3.1.7. Evaluation Metrics

In this study, two performance metrics were utilized to evaluate the performance of the proposed approach for RUL prediction, namely, the scoring function and RMSE. Both metrics utilize the definition of $d_i = RUL_{predicted} - RUL_{actual}$, which denotes the difference between the predicted RUL and the actual RUL

The scoring function proposed by the International Conference on Prognostics and Health Management (PHM08) Data Challenge [60] is defined by

$$score = \sum_{i=1}^{N} S_i, \quad S_i = \begin{cases} e^{\frac{-d_i}{13}} - 1, & d_i < 0 \\ e^{\frac{d_i}{10}} - 1, & d_i \geq 0 \end{cases} \tag{15}$$

where N is the number of testing samples. The scoring function is asymmetric and gives a larger penalty to late predictions than early predictions due to the severe consequences of system failure.

The *RMSE*, which is a standard evaluation metric for prediction, gives equal penalties for both early and late predictions, and it is calculated as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} d_i^2} \tag{16}$$

The lower the scoring function and *RMSE* values, the better the prediction result.

3.1.8. Network Configuration and Hyperparameter Setting

The proposed RUL prediction approach contains deep neural networks. The hyperparameters for network design affect the prediction performance of deep neural networks.

Grid search [31,33,61]; random search [24,62]; and heuristic algorithms, such as genetic algorithm [49], are used for selecting the best hyperparameter combination.

In this paper, the proposed neural network structure 1D-CNN-LSTM is given in Section 2.6. The proposed hybrid neural network structure was compared with the base versions of a 1D-CNN and the LSTM network. The structures of the base 1D-CNN and base LSTM network are presented in Figures 12 and 13, respectively.

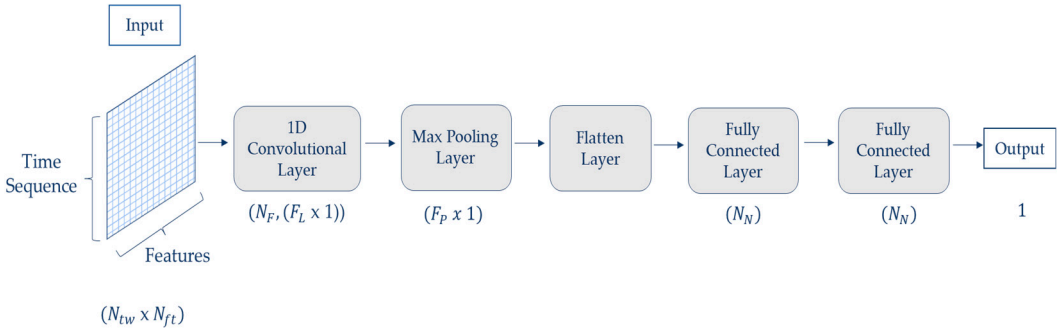


Figure 12. Base 1D-CNN structure.

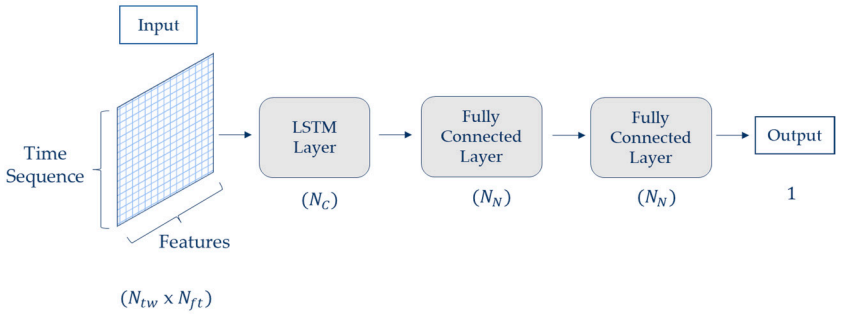


Figure 13. Base LSTM structure.

The hyperparameters evaluated were the number of layers, batch size, number of epochs, filter size, and number of units in the first layer of the network. In this study, a general procedure was created to design the two-layer neural networks. For clarity, in the two-layer network, the filter size or the number of units in the first layer started with a value, and it was halved in the second layer. This was the general procedure used to design the two-stage neural networks in this study. It also gave helpful insight into the effects of the low and high values of the filter size and the number of units used in these neural networks.

The values of hyperparameters are shown in Table 5. For the 1D-CNN-LSTM hybrid neural network, the value of 1 in the number of layers meant one 1D-CNN layer followed by one LSTM layer. Similarly, the value of 2 means two consecutive 1D-CNN layers and two consecutive LSTM layers.

Table 5. Hyperparameter values evaluated in the proposed approach.

Hyperparameter	Range Values
Number of layers	{1, 2}
Number of epochs	{20, 25, 30, 40, 50, 60, 80, 100}
Batch size	{64, 128, 256}
Filter size of the first layer	{32, 64, 128, 256}
Number of units of the first layer	{32, 64, 128, 256}

Cross-validation (CV) is an appropriate method for measuring a model's performance [63]. In k-fold cross-validation, the dataset is split into k non-overlapping folds, where k-1 folds are used for training and one-fold is used for testing. RMSE is the performance measure used in the k-fold CV process.

The Adam algorithm was employed for the optimization of network weights. For the 1D-CNN, the kernel size was set to three, and max pooling with a pool size of three was used for the pooling operations. The activation functions used in all network layers except the output layer were ReLU. The number of fully connected layers was two, and the number of nodes used for those two layers was set to 50 for each neural network.

Experiments were run on a personal computer with Intel Core i5-8250U (1.60 GHz) CPU, 8 GB memory, and Microsoft Windows 11 operating system. The programming language was Python, and the libraries used for building the neural networks were Tensorflow [64] and Keras [65].

The best hyperparameter combination was determined according to the main effects plots. First, we created a hyperparameter space by considering the full range of values for each hyperparameter. In this space, there existed 192 unique combinations resulting from various hyperparameter value permutations (two discrete values for the first hyperparameter, eight for the second, three for the third, and four for the fourth). We then ran each combination through ten statistically independent experiments to thoroughly evaluate the overall effects of each one. After calculating the mean performance of each hyperparameter value, main effects plots helped to examine the impact of a hyperparameter on the performance. Figure 14 shows the main effects plots for the hyperparameters of the base and hybrid networks. For instance, the impact of the number of layers parameter, which signifies the depth and complexity of the network model, can be observed in Figure 14a.

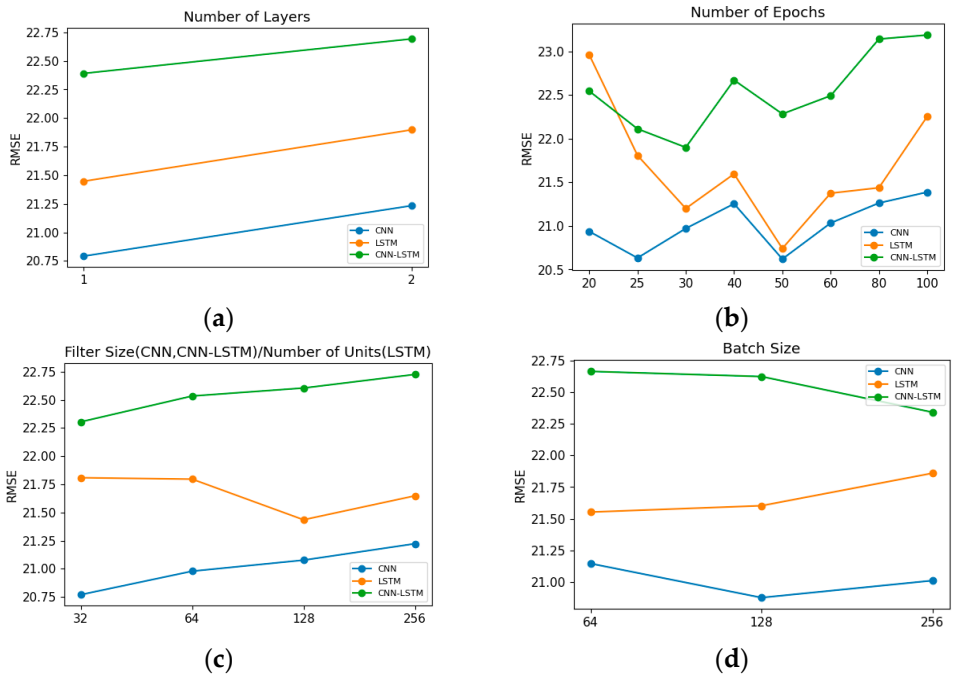


Figure 14. Results of the experimental analysis on hyperparameters: (**a**) the number of layers; (**b**) the number of epochs; (**c**) the filter size/units; (**d**) batch size.

After analyzing the experimental results of the main effects, the hyperparameter values yielding the lowest RMSE values were selected for the final network configuration and training, as shown in Table 6.

Table 6. Selected hyperparameter values.

Hyperparameters	CNN	LSTM	CNN-LSTM
Number of layers	1	1	1
Number of epochs	50	50	30
Batch size	128	64	256
Filter size in the first layer	32	-	32
Number of units in the first layer	-	128	-

3.1.9. Prognostic Procedure

The overall framework proposed for the RUL prediction of turbofan engines using target label construction based on change point detection and feature engineering is presented in Figure 15. Within the framework, data preprocessing is the first critical step in reducing the complexity of calculations required for RUL prediction. Feature selection was performed by selecting sensors whose values changed over time to reduce the complexity. Since sensors have measurement values in different units, min–max normalization was performed per sensor to ensure comparability and eliminate bias. Filtering was applied to uncover the main trend in sensor values due to noise in the dataset. Feature extraction was achieved using an autoencoder-based dimensionality reduction approach to create an encoded signal from the selected sensor data. Also, the encoded signal was used to create the target label value based on the change point on this single signal. A new feature was constructed from the dataset and added to the existing dataset to enhance the prediction accuracy.

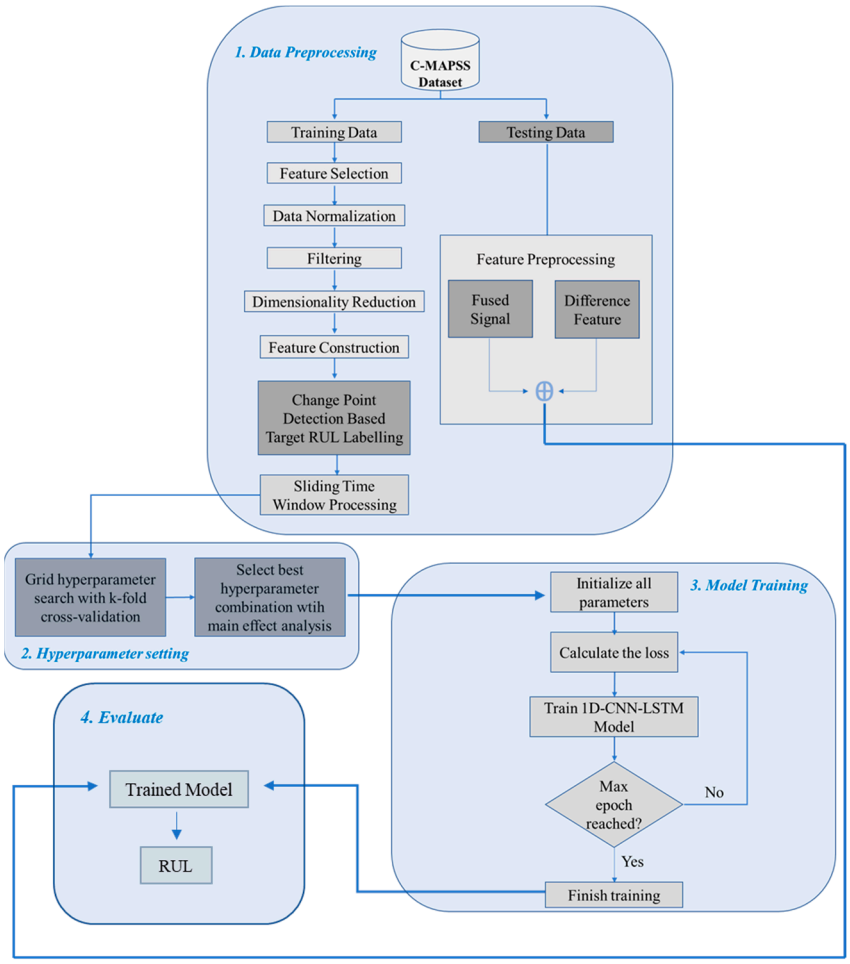


Figure 15. The prognostic procedure of the proposed model.

The target RUL labeling is a crucial factor that affects the prediction accuracy in supervised learning. Since predefined target labels did not exist in the dataset, the PwL degradation models, which are commonly used in the literature, were also adopted here. Although there were multiple sensor signals for each engine in the dataset, a one-dimensional encoded signal was obtained using an autoencoder network to determine a single change point for each engine. Two PwL functions were fitted to the encoded signal using a convex optimization algorithm, yielding a breakpoint where the signal decreased linearly after its breakpoint value. After determining the breakpoints, target RUL labels specific to each motor were created. The final input data sample preparation was conducted using the sliding window technique after all the data preprocessing steps. The input data was finally shaped to a size of 30×2, which consisted of a fused sensor signal and the new difference feature for each time step of the sliding window.

The RUL prediction was performed using the 1D-CNN-LSTM hybrid network and compared separately with the base 1D-CNN, base LSTM networks, and other traditional machine learning methods. In the training phase, samples prepared using sliding window methods were used in the k-fold cross-validation approach to find the best neural network hyperparameters. Predictions were made for all combinations of hyperparameter value ranges using the main effect analysis in the training phase. The hyperparameters yielding the best results in the cross-validation were used in the test phase for the RUL prediction.

3.2. Experimental Results and Discussion

In this subsection, we present the experimental results of the proposed approach for RUL prediction. First, the performances of several models are evaluated using RMSE and score function in Section 3.2.1. The effectiveness of the change point detection approach is assessed in Section 3.2.2. In Section 3.2.3, the effect of the feature construction is analyzed, and an ablation study is also presented. Finally, a comparison with the related methods in the literature is provided in Section 3.2.4.

3.2.1. Prediction Performance Comparison of Different Neural Network Architectures

The prediction results of the 1D-CNN-LSTM hybrid network were compared with its base versions of a 1D-CNN and LSTM network separately. Also, the proposed hybrid network was compared with the well-known traditional machine learning methods, such as k-NN [66], RF [33], and GBM [44]. Additionally, one of the earliest deep learning methods, namely, the group method of data handling (GMDH) [67–69], was used for comparison. Each experiment was repeated ten times to alleviate the impact of randomization, and the mean values were reported. The trained model was used to predict the RUL values for test engines. The performances of the proposed 1D-CNN-LSTM hybrid model and its base versions, namely, a 1D-CNN and LSTM network, for the same engine on the training set are depicted in Figure 16.

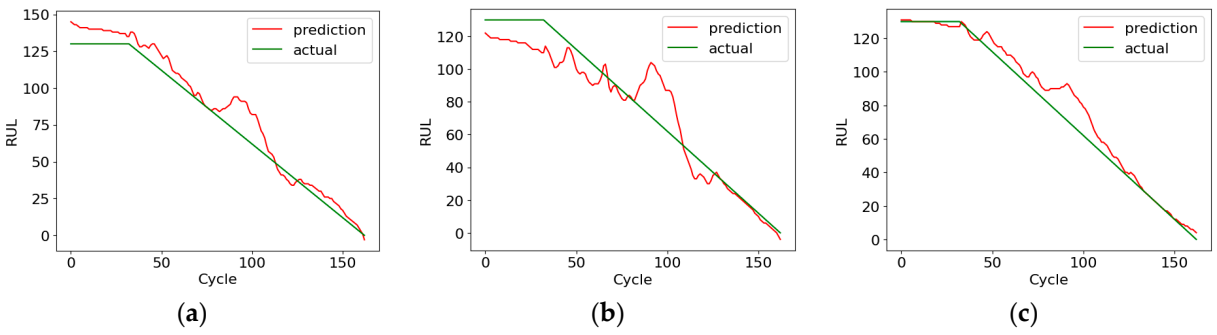


Figure 16. Training results of a randomly selected engine: (**a**) 1D-CNN; (**b**) LSTM; (**c**) 1D-CNN-LSTM.

From Figure 16, it can be observed that the prediction errors were minor when the engine was closer to failure because the system accumulated more degradation information as time passed. In addition, we deduce from Figure 16 that each neural network model successfully detected the change point of the engines because the breakpoints of the actual and predicted signals were very close to each other.

The RUL prediction results using the test data for the eight-network architectures are displayed in Table 7 with and without the new difference feature. The best performances for RMSE and score function are presented in bold. The results shown are the average of the ten repeated experiments and their standard deviation. In the results, first, only the fused sensor values of the relevant network are included, and just below, the values of the newly added difference feature combined with the fused sensor values are included.

Table 7. Comparison of RUL prediction performances on the test set with and without the difference feature.

Methods	RMSE	Score
k-NN (without difference feature)	19.2 ± 0.4	1180.6 ± 360.7
k-NN (with difference feature)	20.9 ± 0.2	825.2 ± 22.8
RF (without difference feature)	21.5 ± 0.2	2447.9 ± 383.2
RF (with difference feature)	19.5 ± 0.2	830.2 ± 105.2
GBM (without difference feature)	19.2 ± 0.3	1454.7 ± 395.6
GBM (with difference feature)	18.3 ± 0.5	714.2 ± 195.9
GMDH (without difference feature)	18.0 ± 0.2	651.5 ± 90.5
GMDH (with difference feature)	17.7 ± 0.2	542.1 ± 70.4
LSTM (without difference feature)	17.8 ± 0.7	2476.3 ± 1607.1
LSTM (with difference feature)	17.0 ± 1.3	1176.5 ± 1368.6
1D-CNN (without difference feature)	17.2 ± 0.2	716.1 ± 129.8
1D-CNN (with difference feature)	**15.5 ± 0.5**	498.9 ± 170
1D-CNN-LSTM (without difference feature)	19.1 ± 2.1	1680.9 ± 1284.1
1D-CNN-LSTM (with difference feature) (proposed)	16.1 ± 0.5	**437.2 ± 126.4**

The RUL prediction performances of the proposed 1D-CNN-LSTM hybrid network using the base variants 1D-CNN and LSTM network with the difference feature in terms of score function were 437.2, 498.9, and 1176.5, respectively. The average score of the 1D-CNN-LSTM model with the difference feature was 14% better than that of the 1D-CNN model with the difference feature added, and 169% better than that of the LSTM model with the difference feature added. Additionally, the standard deviation of the 1D-CNN-LSTM model was 126.4, that of the 1D-CNN model was 170, and that of the LSTM model was 1368.6. The standard deviation of the proposed 1D-CNN-LSTM model was the smallest among them, providing more robust prediction results. Also, the 1D-CNN increased the network's performance by providing helpful information to LSTM thanks to its robust feature extraction capability within the 1D-CNN-LSTM hybrid network for both cases.

Compared with one of the pioneers of deep learning methods, namely, GMDH, the proposed hybrid 1D-CNN-LSTM network demonstrated superior performance, outperforming GMDH's score of 542.1, and it also excelled in terms of the RMSE.

When comparing deep learning methods and traditional machine learning methods for solving the turbofan RUL prediction problem, it was observed that the performance of the 1D-CNN-LSTM hybrid network was notably strong in comparison with the LSTM network, which exhibited lower performance.

The test data set of the FD001 input into the trained 1D-CNN-LSTM network and the positive and negative difference between the actual and the predicted RUL values in time steps for each engine are shown in Figure 17. Negative values indicate early predictions, while positive values indicate late predictions.

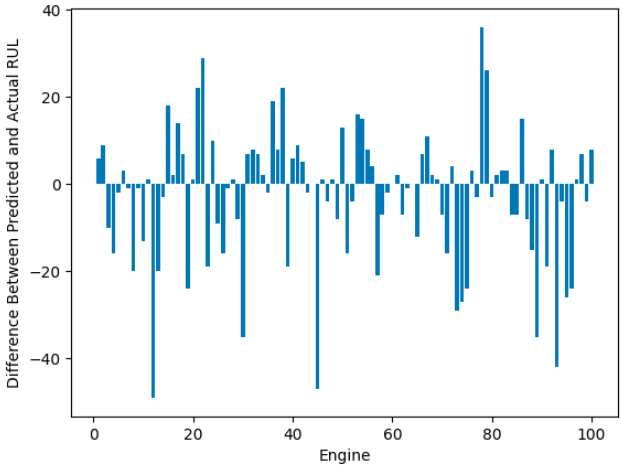


Figure 17. Difference between actual and predicted RUL values on the test set.

3.2.2. Evaluation of Change Point Detection Methods

Change point detection is crucial in engine-specific target labeling for RUL prediction. To demonstrate the effectiveness of our proposed pwlf-based change point detection method, we conducted evaluations using the widely recognized PELT algorithm [70] for the change point detection and various parameter settings of pwlf. The relevant evaluations are presented in Table 8 and best performances are presented in bold.

Table 8. Evaluations of change point detection methods.

Methods	RMSE	Score
PELT	35.8 ± 1.6	3576.5 ± 726.0
pwlf with 2 lines	34.1 ± 1.1	3092.5 ± 395.7
pwlf with 3 lines	17.0 ± 1.0	503.7 ± 118.6
pwlf with 4 lines	16.4 ± 0.6	522.2 ± 128.8
pwlf with 5 lines (proposed)	$\mathbf{16.1 \pm 0.4}$	$\mathbf{437.2 \pm 119.9}$

In the evaluation of the change point detection, we applied the proposed 1D-CNN-LSTM network with the difference feature within the context of this research. Ten repetitions were performed for each method, and the results are presented with their respective average and standard deviation values. Notably, the outcomes obtained with the complete set of parameters for the pwlf method exhibited superior performance compared with the PELT method. Furthermore, when we examined the pwlf method while adjusting the line-segment parameter, an increase in the parameter values resulted in decreased prediction errors and improved prediction performance. The suggested approach for change point detection, which encompassed a 5-line segment configuration, yielded a substantial improvement in motor-specific target labeling for RUL prediction.

3.2.3. Ablation Study of the Proposed Feature-Engineering-Based Approach

In order to evaluate the effectiveness of the proposed feature-engineering-based approach, an ablation study was performed on the aforementioned neural networks in this study. First, original networks, only with the encoded data of fused sensors, were used for the RUL prediction. After that, the difference feature was added to the fused sensor data, and the RUL prediction was performed. The results of the ablation study can also be found in Table 7.

The results demonstrate that all networks, when augmented with the difference feature in the fused sensor data, outperformed the original networks lacking this feature, highlighting the effectiveness of the feature construction approach. Furthermore, it can be concluded that the 1D-CNN-LSTM exhibited superior performance compared with the

base 1D-CNN, base LSTM network, and other related networks, whether with or without additional features, in terms of the RMSE and score function. The predicted results of the proposed 1D-CNN-LSTM neural network compared with the actual RUL values are shown in Figure 18 with and without the difference feature for all engines in the data set. As is noticeable in Figure 18b, the variation of prediction results with the added difference feature was much smaller than using only the fused sensor signals for the task of the RUL prediction.

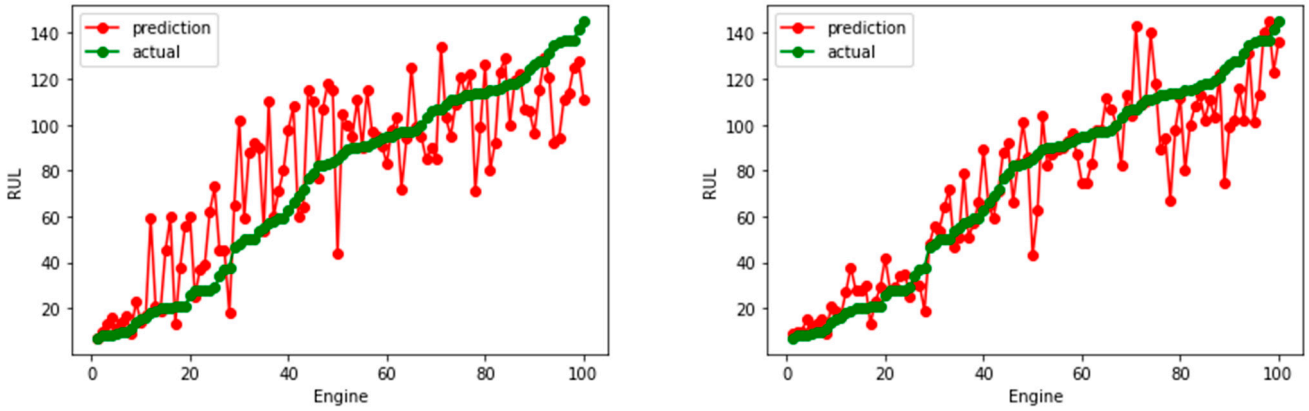


Figure 18. RUL predictions for the 100 testing engines (sorted with increasing RUL): (**a**) without difference feature; (**b**) with difference feature.

3.2.4. Comparison with Related Work

The C-MAPSS dataset is a popular public dataset that has produced numerous state-of-the-art results in prognostic research. This paper proposes a new RUL prediction approach using a change-point-detection-based target labeling and feature engineering. We compared the prognostic performance of the proposed method with the studies that used the same dataset and similar prognostic concepts. The studies shown in Table 9 used engine-specific RUL labels determined with several methodologies, such as LSTM and a cumulative sum control chart. Also, most of the studies using engine-specific RUL labels made predictions on some proportion of the training dataset instead of using the test data set, as mentioned earlier. The "Test Data" column in the comparison below refers to the type of dataset used in the testing phase. "Test" indicates that the test dataset was used for testing, while "Training" indicates that a specific portion of the training dataset was used for testing.

Table 9. Performance comparison of the proposed method and the related papers on the C-MAPSS dataset.

Authors	Approach	RMSE	Score	Test Data
Huang et al. [71]	Bayesian neural network (BNN)	17.92	N/A	Test
Aydemir and Acar [43]	LSTM	17.15	392	Test
Miao et al. [39]	Dual-task LSTM	12.29	N/A	Training
Proposed approach	1D-CNN-LSTM with feat. eng.	16.1	437.2	Test

The test set of FD001 was terminated at an arbitrary operation cycle, and therefore, a few studies, which were based on the targets specific to the engine, used only the test set in the RUL prediction. The RMSE in [39] was the lowest because a part of the training data set, which had all the sensor values from the start to the failure time, was used as the test dataset. Despite [43]'s method yielding a lower score, our proposed method demonstrated the best outcome, as evidenced by the lowest RMSE value among the studies that used only the test dataset for RUL prediction.

4. Conclusions

The effective prediction of the RUL value for turbofan engines is critical to ensure optimal performance and minimize maintenance costs. This paper proposes a prognostic procedure addressing the feature engineering process and a hybrid network based on 1D-CNN-LSTM.

Our approach utilized 1D-CNN architecture to extract spatial information from raw sensor data, while LSTM was used to reveal temporal information from the extracted features. We implemented a range of feature engineering and preprocessing methods. Feature selection, filtering, and normalization were used to improve the data quality and reliability. Also, these methods made it easier for deep learning models to use and interpret the data. In addition, feature engineering approaches, such as dimension reduction and feature creation, not only facilitated the interpretation of existing data but also increased the efficiency of the data and enabled the creation of new features that facilitated the learning of the prediction model. Our proposed prognostic approach demonstrated that the hybrid neural network produced more accurate RUL predictions when combined with practical preprocessing steps. To label the operational life of turbofan engines, we used the PwL target-labeling method, which divides the operational life of the engines into two stages, namely, the stable healthy stage and the linear degradation stage. We leveraged dimensionality reduction for a better representation of the main characteristics and the breakpoints of the sensor measurements. A change point detection method was used to determine the start point of degradation for engine-specific target RUL labeling. Using the maximum RUL specific to each engine presented a challenge due to insufficient degradation information. However, it also brought the solution closer to a realistic scenario. In addition, we constructed a new feature from the existing dataset using the initial sensor measurements to provide more degradation information in the early stage. The proposed methodology provides better prediction results compared with the studies using engine-specific target RUL labels and the actual test dataset.

The prognostic procedure proposed for RUL prediction in turbofan engines essentially demonstrated the effectiveness of a feature engineering technique, specifically the feature construction step, and the 1D-CNN-LSTM network model. The proposed hybrid network was tested on multiple hyperparameter combinations using k-fold cross-validation to obtain the best results. The hybrid model was compared with the 1D-CNN and LSTM networks separately, and our findings demonstrate that the CNN feature extraction capability significantly improved the performance of the LSTM network.

The results obtained provide evidence for the efficacy of the newly created "difference" feature through feature construction. Furthermore, the hyperparameter optimization study revealed a reduced need for deeper and more complex network models in the context of feature engineering processes.

The prediction of RUL has been a longstanding and persistently relevant problem, which has been further invigorated by the advancement of artificial intelligence methods. Considering the current trend in various applications related to turbofan engines and other domains associated with the RUL problem, this study was aligned with contemporary requirements. The findings substantiate the impact of data preprocessing on prediction performance and reveal a reduced dependence on complex network structures. For future work, we aim to achieve higher accuracy in the early stage of the deterioration by combining engine-specific RUL labeling, similarity-based methods, and neural networks. We also aim to adopt more robust approaches to determine the change point for PwL labels. Overall, our proposed approach provides promising results and opens avenues for further research in turbofan engines' prognostics and health management.

Author Contributions: Conceptualization, K.E., T.İ. and E.E.; methodology, K.E., T.İ. and E.E.; software, K.E.; validation, K.E.; formal analysis, K.E.; investigation, K.E.; writing—original draft preparation, K.E.; writing—review and editing, T.İ. and E.E.; visualization, K.E.; supervision, T.İ. and E.E.; project administration, T.İ. and E.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding and the APC was funded by Kıymet Ensarioğlu.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset used in this study is openly available on the NASA repository and it is called the Turbofan Engine Degradation Simulation Dataset (https://data.nasa.gov/Aerospace/CMAPSS-Jet-Engine-Simulated-Data/ff5v-kuh6) (accessed on 1 October 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Peng, C.; Chen, Y.; Chen, Q.; Tang, Z.; Li, L.; Gui, W. A remaining useful life prognosis of turbofan engine using temporal and spatial feature fusion. *Sensors* **2021**, *21*, 418. [CrossRef] [PubMed]
2. Azadeh, A.; Asadzadeh, S.; Salehi, N.; Firoozi, M. Condition-based maintenance effectiveness for series-parallel power generation system—A combined Markovian simulation model. *Reliab. Eng. Syst. Saf.* **2015**, *142*, 357–368. [CrossRef]
3. Ramasso, E. Investigating computational geometry for failure prognostics. *Int. J. Progn. Health Manag.* **2014**, *5*, 005. [CrossRef]
4. *ISO 13381-1 2015*; Condition Monitoring and Diagnostics of Machines—Prognostics—Part 1: General Guidelines. International Organization for Standardization: Geneva, Switzerland, 2015.
5. Leser, P.E. Probabilistic Prognostics and Health Management for Fatigue-Critical Components using High-Fidelity Models. Ph.D. Thesis, North Carolina State University, Raleigh, North Carolina, 2017.
6. Si, X.S.; Wang, W.; Hu, C.H.; Zhou, D.H. Remaining useful life estimation—A review on the statistical data-driven approaches. *Eur. J. Oper. Res.* **2011**, *213*, 1–14. [CrossRef]
7. Song, Y.; Shi, G.; Chen, L.; Huang, X.; Xia, T. Remaining Useful Life Prediction of Turbofan Engine Using Hybrid Model Based on Autoencoder and Bidirectional Long Short-Term Memory. *J. Shanghai Jiaotong Univ.* **2018**, *23*, 85–94. [CrossRef]
8. Wang, T.; Guo, D.; Sun, X.M. Remaining useful life predictions for turbofan engine degradation based on concurrent semi-supervised model. *Neural Comput. Appl.* **2022**, *34*, 5151–5160. [CrossRef]
9. Li, X.; Ding, Q.; Sun, J.Q. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab. Eng. Syst. Saf.* **2018**, *172*, 1–11. [CrossRef]
10. Khelif, R.; Chebel-Morello, B.; Malinowski, S.; Laajili, E.; Fnaiech, F.; Zerhouni, N. Direct Remaining Useful Life Estimation Based on Support Vector Regression. *IEEE Trans. Ind. Electron.* **2017**, *64*, 2276–2285. [CrossRef]
11. Zheng, S.; Ristovski, K.; Farahat, A.; Gupta, C. Long Short-Term Memory Network for Remaining Useful Life Estimation. In Proceedings of the IEEE International Conference on Prognostics and Health Management, Dallas, TX, USA, 19–21 June 2017. [CrossRef]
12. Huang, C.G.; Huang, H.Z.; Peng, W.; Huang, T. Improved trajectory similarity-based approach for turbofan engine prognostics. *J. Mech. Sci. Technol.* **2019**, *33*, 4877–4890. [CrossRef]
13. Le Son, K.; Fouladirad, M.; Barros, A.; Levrat, E.; Iung, B. Remaining useful life estimation based on stochastic deterioration models: A comparative study. *Reliab. Eng. Syst. Saf.* **2013**, *112*, 165–175. [CrossRef]
14. Le Son, K.; Fouladirad, M.; Barros, A. Remaining useful lifetime estimation and noisy gamma deterioration process. *Reliab. Eng. Syst. Saf.* **2016**, *149*, 76–87. [CrossRef]
15. Giantomassi, A.; Ferracuti, F.; Benini, A.; Ippoliti, G.; Longhi, S.; Petrucci, A. Hidden Markov Model for Health Estimation and Prognosis of Turbofan Engines. In Proceedings of the International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Washington, DC, USA, 28–31 August 2011. [CrossRef]
16. Yu, W.; Kim, I.Y.; Mechefske, C. Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. *Mech. Syst. Signal Process.* **2019**, *129*, 764–780. [CrossRef]
17. Heimes, F.O. Recurrent neural networks for remaining useful life estimation. In Proceedings of the International Conference on Prognostics and Health Management, Denver, CO, USA, 6–9 October 2008. [CrossRef]
18. Zhao, C.; Huang, X.; Li, Y.; Iqbal, M.Y. A double-channel hybrid deep neural network based on CNN and BiLSTM for remaining useful life prediction. *Sensors* **2020**, *20*, 7109. [CrossRef]
19. Peng, C.; Wu, J.; Wang, Q.; Gui, W.; Tang, Z. Remaining Useful Life Prediction Using Dual-Channel LSTM with Time Feature and Its Difference. *Entropy* **2022**, *24*, 1818. [CrossRef] [PubMed]
20. Wang, J.; Wen, G.; Yang, S.; Liu, Y. Remaining Useful Life Estimation in Prognostics Using Deep Bidirectional LSTM Neural Network. In Proceedings of the Prognostics and System Health Management Conference, Chongqing, China, 26–28 October 2018. [CrossRef]

21. Wu, J.; Hu, K.; Cheng, Y.; Zhu, H.; Shao, X.; Wang, Y. Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network. *ISA Trans.* **2020**, *97*, 241–250. [CrossRef] [PubMed]
22. Saxena, A.; Goebel, K. Turbofan Engine Degradation Simulation Data Set. NASA Ames Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, USA. [Online]. Available online: https://data.nasa.gov/Aerospace/CMAPSS-Jet-Engine-Simulated-Data/ff5v-kuh6/ (accessed on 1 October 2023).
23. Wang, Y.; Zhao, Y. Multi-Scale Remaining Useful Life Prediction Using Long Short-Term Memory. *Sustainability* **2022**, *14*, 15667. [CrossRef]
24. Muneer, A.; Taib, S.M.; Fati, S.M.; Alhussian, H. Deep-learning based prognosis approach for remaining useful life prediction of turbofan engine. *Symmetry* **2021**, *13*, 1861. [CrossRef]
25. Chen, J.; Chen, D.; Liu, G. Using temporal convolution network for remaining useful lifetime prediction. *Eng. Rep.* **2021**, *3*, e12305. [CrossRef]
26. Tan, W.M.; Teo, T.H. Remaining Useful Life Prediction Using Temporal Convolution with Attention. *AI* **2021**, *2*, 48–70. [CrossRef]
27. Li, H.; Zhao, W.; Zhang, Y.; Zio, E. Remaining useful life prediction using multi-scale deep convolutional neural network. *Appl. Soft Comput.* **2020**, *89*, 106–113. [CrossRef]
28. Peng, C.; Wu, J.; Tang, Z.; Yuan, X.; Li, C. A Spatio-Temporal Attention Mechanism-Based Approach for Remaining Useful Life Prediction of Turbofan Engine. *Comput. Intell. Neurosci.* **2022**, *2022*, 1687–5265. [CrossRef] [PubMed]
29. Al-Dulaimi, A.; Zabihi, S.; Asif, A.; Mohammadi, A. A multi-modal and hybrid deep neural network model for remaining useful life estimation. *Comput. Ind.* **2019**, *108*, 186–196. [CrossRef]
30. Kong, Z.; Cui, Y.; Xia, Z.; Lv, H. Convolution and long short-term memory hybrid deep neural networks for remaining useful life prognostics. *Appl. Sci.* **2019**, *9*, 4156. [CrossRef]
31. Wang, Q.; Wu, G. Effective Latent Representation for Prediction of Remaining Useful Life. *Comput. Syst. Sci. Eng.* **2021**, *36*, 225–237. [CrossRef]
32. Liu, H.; Liu, Z.; Jia, W.; Lin, X. Remaining Useful Life Prediction Using a Novel Feature-Attention-Based End-to-End Approach. *IEEE Trans. Ind. Inform.* **2021**, *17*, 1197–1207. [CrossRef]
33. Gong, R.; Li, J.; Wang, C. Remaining Useful Life Prediction Based on Multi-sensor Fusion and Attention TCN-BiGRU Model. *IEEE Sens. J.* **2022**, *22*, 21101–21110. [CrossRef]
34. Wu, Y.; Yuan, M.; Dong, S.; Lin, L.; Liu, Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing* **2018**, *275*, 167–179. [CrossRef]
35. Xiao, L.; Tang, J.; Zhang, X.; Bechhoefer, E.; Ding, S. Remaining useful life prediction based on intentional noise injection and feature reconstruction. *Reliab. Eng. Syst. Saf.* **2021**, *215*, 107871. [CrossRef]
36. Chen, Z.; Wu, M.; Zhao, R.; Guretno, F.; Yan, R.; Li, X. Machine Remaining Useful Life Prediction via an Attention-Based Deep-Learning Approach. *IEEE Trans. Ind. Electron.* **2021**, *68*, 2521–2531. [CrossRef]
37. Zhao, S.; Zhang, Y.; Wang, S.; Zhou, B.; Cheng, C. A recurrent neural network approach for remaining useful life prediction utilizing a novel trend features construction method. *Measurement* **2019**, *146*, 279–288. [CrossRef]
38. Bae, J.; Xi, Z. Learning of physical health timestep using the LSTM network for remaining useful life estimation. *Reliab. Eng. Syst. Saf.* **2022**, *226*, 108717. [CrossRef]
39. Miao, H.; Li, B.; Sun, C.; Liu, J. Joint Learning of Degradation Assessment and RUL Prediction for Aeroengines via Dual-Task Deep LSTM Networks. *IEEE Trans. Ind. Inform.* **2019**, *15*, 5023–5032. [CrossRef]
40. Zhang, Y.; Xin, Y.; Wei Liu, Z.; Chi, M.; Ma, G. Health status assessment and remaining useful life prediction of aero-engine based on BiGRU and MMoE. *Reliab. Eng. Syst. Saf.* **2022**, *220*, 108263. [CrossRef]
41. Shi, Z.; Chehade, A. A dual-LSTM framework combining change point detection and remaining useful life prediction. *Reliab. Eng. Syst. Saf.* **2021**, *205*, 107257. [CrossRef]
42. Baptista, M.L.; Henriques, E.M.; Goebel, K. More effective prognostics with elbow point detection and deep learning. *Mech. Syst. Signal Process.* **2021**, *146*, 106987. [CrossRef]
43. Aydemir, G.; Acar, B. Anomaly monitoring improves remaining useful life estimation of industrial machinery. *J. Manuf. Syst.* **2020**, *56*, 463–469. [CrossRef]
44. Nair, P.; Vakharia, V.; Borade, H.; Shah, M.; Wankhede, V. Predicting Li-Ion Battery Remaining Useful Life: An XDFM-Driven Approach with Explainable AI. *Energies* **2023**, *16*, 5725. [CrossRef]
45. Li, J.; Huang, F.; Qin, H.; Pan, J. Research on Remaining Useful Life Prediction of Bearings Based on MBCNN-BiLSTM. *Appl. Sci.* **2023**, *13*, 7706. [CrossRef]
46. Sova, J.; Kolar, P.; Burian, D.; Vozabal, P. Remaining Useful Life Estimation of Spindle Bearing Based on Bearing Load Calculation and Off-Line Condition Monitoring. *Machines* **2023**, *11*, 586. [CrossRef]
47. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
48. Hong, C.W.; Lee, K.; Ko, M.S.; Kim, J.K.; Oh, K.; Hur, K. Multivariate time series forecasting for remaining useful life of turbofan engine using deep-stacked neural network and correlation analysis. In Proceedings of the IEEE International Conference on Big Data and Smart Computing, Busan, Republic of Korea, 19–22 February 2020. [CrossRef]
49. Ellefsen, A.L.; Ushakov, S.; Aesoy, V.; Zhang, H. Validation of Data-Driven Labeling Approaches Using a Novel Deep Network Structure for Remaining Useful Life Predictions. *IEEE Access* **2019**, *7*, 71563–71575. [CrossRef]

50. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
51. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
52. Al Bataineh, A.; Mairaj, A.; Kaur, D. Autoencoder Based Semi-Supervised Anomaly Detection in Turbofan Engines. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 41–47. [CrossRef]
53. Zheng, A.; Casari, A. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*, 1st ed.; O'Reilly: Sebastopol, CA, USA, 2018; pp. 5–39.
54. Gupta, A.; Masampally, V.S.; Jadhav, V.; Deodhar, A.; Runkana, V. Supervised Operational Change Point Detection using Ensemble Long-Short Term Memory in a Multicomponent Industrial System. In Proceedings of the IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herl'any, Slovakia, 21–23 January 2021. [CrossRef]
55. Jekel, C.; Venter, G. pwlf: A Python Library for Fitting 1D Continuous Piece-Wise Linear Functions. 2019. Available online: https://www.researchgate.net/profile/Charles-Jekel-2/publication/331231072_pwlf_A_Python_Library_for_Fitting_1D_Continuous_Piecewise_Linear_Functions/links/5c6d607aa6fdcc404ec03c53/pwlf-A-Python-Library-for-Fitting-1D-Continuous-Piecewise-Linear-Functions.pdf (accessed on 1 October 2023).
56. Zhang, B.; Zheng, K.; Huang, Q.; Feng, S.; Zhou, S.; Zhang, Y. Aircraft Engine Prognostics Based on Informative Sensor Selection and Adaptive Degradation Modeling with Functional Principal Component Analysis. *Sensors* **2020**, *20*, 920. [CrossRef]
57. Bektas, O.; Jones, J.A.; Sankararaman, S.; Roychoudhury, I.; Goebel, K. A neural network filtering approach for similarity-based remaining useful life estimation. *Int. J. Adv. Manuf. Technol.* **2019**, *101*, 87–103. [CrossRef]
58. Singh, S.K.; Kumar, S.; Dwivedi, J.P. A novel soft computing method for engine RUL prediction. *Multimed. Tools Appl.* **2019**, *78*, 4065–4087. [CrossRef]
59. De Cheveign'e, A.; Nelken, I. Filters: When, Why, and How (Not) to Use Them. *Neuron* **2019**, *102*, 280–293. [CrossRef]
60. Saxena, A.; Goebel, K. PHM08 Challenge Data Set. NASA Ames Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, USA. [Online]. Available online: https://data.nasa.gov/Raw-Data/PHM-2008-Challenge/nk8v-ckry (accessed on 1 October 2023).
61. Huang, C.G.; Huang, H.Z.; Li, Y.F. A Bidirectional LSTM Prognostics Method Under Multiple Operational Conditions. *IEEE Trans. Ind. Electron.* **2019**, *66*, 8792–8802. [CrossRef]
62. Muneer, A.; Taib, S.M.; Naseer, S.; Ali, R.F.; Aziz, I.A. Data-driven deep learning-based attention mechanism for remaining useful life prediction: Case study application to turbofan engine analysis. *Electronics* **2021**, *10*, 2453. [CrossRef]
63. Abiodun, A.; Zhiyu, W.; Wenhai, W.; Weizhong, Q.; Chunhua, Y.; Shenghu, X.; Xinggao, L. Causal augmented ConvNet: A temporal memory dilated convolution model for long-sequence time series prediction. *ISA Trans.* **2022**, *123*, 200–217. [CrossRef]
64. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. *arXiv* **2016**. [CrossRef]
65. Chollet, F.; Allaire, J.J.; Kalinowski, T.; RStudio; Google; Tang, Y.; Bijl, W.; Studer, M.; Keydana, S. Keras. 2015. Available online: https://keras.io (accessed on 1 October 2023).
66. Ramasso, E.; Rombaut, M.; Zerhouni, N. Joint Prediction of Continuous and Discrete States in Time-Series Based on Belief Functions. *IEEE Trans. Cybern.* **2013**, *43*, 37–50. [CrossRef]
67. Najibi, A.; Ghazifard, P.; Torkian, J. On the crashworthiness optimisation of a new multi-corner tube under axial loading. *Ships Offshore Struct.* **2023**, 1–14. [CrossRef]
68. Najibi, A.; Shojaeefard, M.H.; Yeganeh, M. Developing and multi-objective optimization of a combined energy absorber structure using polynomial neural networks and evolutionary algorithms. *Lat. Am. J. Solids Struct.* **2016**, *13*, 2252–2272. [CrossRef]
69. Ivakhnenko, A.G. Polynomial Theory of Complex Systems. *IEEE Trans. Syst. Man Cybern.* **1971**, *4*, 364–378. [CrossRef]
70. Killick, R.; Fearnhead, P.; Eckley, I.A. Optimal detection of changepoints with a linear computational cost. *J. Am. Stat. Assoc.* **2012**, *107*, 1590–1598. [CrossRef]
71. Huang, D.; Bai, R.; Zhao, S.; Wen, P.; Wang, S.; Chen, S. Bayesian neural network based method of remaining useful life prediction and uncertainty quantification for aircraft engine. In Proceedings of the Annual Conference of the Prognostics and Health Management Society, Detroit, MI, USA, 8–10 June 2020. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.