Remaining-Useful-Life Prediction Using Machine Learning and Explainable AI

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Computer Science(Specialization in Data Science)

by

JYOTHI K C

19BDS0144

Under the guidance of

Prof. Dr. SWATHI J N

SCOPE

VIT, Vellore.

May, 2023

DECLARATION

I hereby declare that the thesis entitled "**Remaining-Useful-Life Prediction Using Machine Learning and Explainable AI**" submitted by me, for the award of the degree of *Bachelor of Technology in Computer Science(Specialization in Data Science)* to VIT is a record of bonafide work carried out by me under the supervision of Prof./ Dr. Swathi J N.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place : Vellore

Date: 20.05.2023

Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled "Remaining-Useful-Life Prediction of Aircraft Engines Using Machine Learning and Explainable AI" submitted by Jyothi K C 19BDS0144, SCOPE, VIT, for the award of the degree of Bachelor of Technology in Programme, is a record of bona fide work carried out by him / her under my supervision during the period, 01. 07. 2022 to 30.04.2023, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date :

Signature of the Guide

Internal Examiner **External Examiner**

Prof. Parveen Sultana Computer Science

(Specialization in Data Science)

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to all those who have supported me throughout the journey of completing this capstone project. First and foremost, I am immensely thankful to my project advisor, Prof. Swathi J N for their unwavering guidance, expertise, and encouragement. Their invaluable insights and constructive feedback have played a pivotal role in shaping the direction and quality of this project.

I would also like to extend my heartfelt thanks to my professors and mentors who have contributed to my overall academic growth. Their dedication, knowledge, and passion for their respective fields have been a constant source of inspiration for me. I am grateful for the opportunities they have provided and the skills they have instilled in me.

Last but not the least, I would like to express my heartfelt appreciation to my family. Their unwavering love, encouragement, and belief in my abilities have been a constant source of strength and motivation throughout this endeavor. Their understanding and support during the ups and downs of this project have been invaluable.

Jyothi K C

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
No.	Number
LSTM	Long Short-term Memory
xAI	Explainable AI

EXECUTIVE SUMMARY

Remaining useful life (RUL) can be predicted by using model based, data driven and hybrid methods. Model-based methods are challenging, expensive, and time-consuming to develop in complex equipment due to the need for a lot of prior system knowledge. Data driven methods, i.e., predicting RUL using machine learning (ML), on the other hand are cheaper, require less historical data and less complex. The rise of Industry 5.0 has popularized interpretable data-driven prediction methods. Existing literature has proven that deep learning (DL) models such as multi-layer perceptron (MLP), and long short term memory (LSTM) have performed well in RUL prediction. This thesis proposes a novel explainable framework TCN-LSTM-xAI, using TCN-LSTM, a hybrid of temporal convolutional network (TCN) and Long Short-Term Memory (LSTM) as the prediction model. Also, an interpretation of the model's results is provided using the post-hoc explainable artificial intelligence (xAI) algorithm, Shapley Additive Explanations (SHAP). Furthermore, the proposed model is evaluated against two state-of-the-art models, CNN-LSTM and TCN on the publicly available NASA CMAPPS turbofan dataset. The proposed methodology shows comparable results to the existing model in current literature with an RMSE 12.772 and a detailed insight of the proposed model's predictions is provided. A visualization dashboard web application was designed.

1. INTRODUCTION

1.1. THEORETICAL BACKGROUND

Condition based monitoring (CBM) schedules and performs maintenance plans for components and machines based on their needs, in the manufacturing industry. Prognostic and health management (PHM) implements CBM by monitoring the wear of components via indicators. Predictive maintenance (PdM) plays a crucial role in PHM of machines in the era of industry 4.0. A machinery prognostic program generally consists of four technical processes, i.e., data acquisition, health indicator (HI) construction, health stage (HS) division, and Remaining-Useful-Life (RUL) prediction. Remaining useful life refers to operational lifetime of a component after the initial degradation of the machinery.[1]The final objective of PdM is to forecast RUL of the product or equipment as it decreases the unplanned downtime of the process and takes cost-effective maintenance decisions. The methods for RUL prediction are broadly divided into three types: (i)Knowledge-based method (experience-based),(ii) physical model-based method, (iii) and data-driven approach.[2]
The knowledge-based model uses historical failure data for prediction using methods such as fuzzy logic, Weibull distribution and Bayesian approach. However, these methods are suitable for simple processes or systems and not very accurate. In the physics based-model approach based on internal mechanism, a physical-mathematical model is developed, reflecting performance degradation of the system. This method is suitable for dynamic modelling and does not require collecting a lot of data, but it required expert knowledge. This model is further classified as a mathematical model, Hidden Markov model, Probability distribution model and filter models like Kalman and particle filter. It is challenging to construct a perfect model in a complex process or machine by considering all degradation mechanisms.[3]
Data driven method uses data collected from the equipment or components for prediction of failure. It does not require a separate performance degradation process[4]This data-driven model is further categorized as supervised, unsupervised, and semi-supervised models. Supervised models need the labelled dataset to train the model while unsupervised model uses the unlabeled data set to train the model. Supervised models are further classified as classification and regression models. Regression algorithms like simple linear regression,

random forest regression, artificial neural network, support vector regression etc. are used where continuous values are needed to predict the RUL.

Prior work done by researchers in data-driven RUL prediction is broadly segregated into three industrial sectors: Bearings, Aircraft and Batteries. Lithium-ion batteries are used as energy sources in electric vehicles (EVs) and portable electronic devices due to their conducive properties such as long shelf-life, low costs and high energy densities. Accurate estimation of the capacity degradation process and RUL of battery prevents potential degradation in the system's performance or fends off operational hazards.[5] Similarly, evaluating the RUL of an aircraft's engine can optimize maintenance schedules, thereby avoiding disasters. Due to the broad scope of applications, this project focuses on the RUL prediction in the aircraft sector.

The advances in deep learning algorithms have improved the prediction results significantly. However, interpreting the results and understanding which features a model takes into consideration remains obscure in these models, hence giving rise to their alternative name "black-box model". Explainable AI(XAI) is a set of techniques which are implemented to make the decision -making process in a machine learning model transparent. [6]

In this paper, Shapley Additive exPlanations(SHAP) , a model-agnostic XAI technique will be used to identify the significant parameters which contribute to the RUL prediction by the model in the NASA C-MAPSS dataset.

1.2. MOTIVATION

The advent of Industry 5.0 in recent times inspired me to base my project on predictive maintenance in the manufacturing sector and making the results interpretable using xAI, thereby enabling users to understand the high-performance results of black-box models, which are obscured otherwise. Furthermore, through reviewing past literature the lack of application of a TCN-LSTM hybrid for the purpose of predicting remaining useful life in turbofan jet engines, motivated me to implement an explainable regression framework, TCN-LSTM-xAI.

1.3. AIM

The aim of this project is to implement a novel TCN-LSTM xAI regression framework for predicting the remaining-useful-life in the NASA C-MAPPS dataset and evaluate its performance using other state-of-the-art models such as Conv-LSTM, a hybrid of CNN and LSTM, and TCN.

1.4. OBJECTIVES

The following are the objectives of this project:

- Implement a novel deep learning model, temporal convolutional network integrated with LSTM on the NASA turbofan dataset for predicting the remaining-useful-life.
- Evaluate and compare the results by implementing two other deep learning models, TCN and Conv-LSTM respectively.
- Make the results interpretable, i.e., make transparent the features that the black box model gives priority for predictions.
- Develop a dashboard providing insights for the data and prediction results.

2. LITERATURE SURVEY

2.1. SURVEY OF THE EXISTING MODELS/WORK

2.1.1. Predictive Maintenance

Predictive Maintenance mainly consists of three processes: (i) data acquisition, (ii) data processing, (iii) maintenance decision making, and the research is concentrated in these aspects of the domain. The technological advances in IoT have led to data collection via sensors, which record the health status of a part or machinery for its conditional monitoring.[7], [8]Predictive maintenance techniques can be divided into three categories based on the types of sensors: (i) existing sensor-based maintenance, (ii) test-sensor-based maintenance, and (iii) test signal-based maintenance techniques. [9]

There are several frameworks introduced for the data processing and decision-making processes in literature. For instance, [10] provides sampling and data reduction techniques as well as a framework integrating edge computing paradigm with machine learning models. Complex Morlet wavelet-based envelope has been introduced as a feature extraction technique for sensor-based data in the early years[11]. [12] has explored the challenges involved in analyzing multiple-sourced heterogenous data and predicting the remaining useful life while preserving the spatiotemporal property of data.

With the evolution of cloud technology, automating decision-making processes, such as scheduling maintenance plans in industries, is one of the most explored topics. Estimating the remaining-useful-life or the state-of-health (SoH) or the state-of-charge(SoC) of a component is the main task of predictive maintenance, as these are used as metrics to decide when a machinery needs to be replaced.[13] There have been several innovative methods proposed for predicting the RUL of a component over the years. Initial solutions are centered around statistical and machine learning algorithms like hidden Markov models, Random forest and Support Vector Machines (SVM).[14] For instance, [15] relied on empirical mode decomposition (EMD) for decomposing the raw time series data of Lithium-ion batteries into sub-layers, after which RUL was prediction using Autoregressive integrated moving average(ARIMA) model. Other algorithms commonly used include fuzzy logic and decision trees. [16]used the predictive models, regression trees (CART) and adaptive neuro-fuzzy inference system (ANFIS) on the vibration data from accelerometers to calculate the remaining-useful-life of the bearings.

Recently, the trend has shifted from conventional machine learning algorithms to adopting neural networks such as convolutional neural network(CNN), multi-layer perceptron (MLP) and long short-term memory(LSTM).[17], [18] For instance, [19] proposed a deep adversarial network using CNN for feature extraction and then subjecting the data(from the publicly available PRONOSTIA and NASA C-Mapps datasets) to adversarial training to make the model robust to potential sensor malfunctions. RNN models such as LSTM and GRU are more suitable for analyzing sequential or time-series data. [20]Also, attention mechanisms have been implemented with the deep neural networks to obtain better performance as demonstrated in [17], [21]

2.1.2. Explainable AI

Over the last few years, more focus is put on a model's interpretability rather than its predictive performance in various sectors including defense, healthcare, education and manufacturing. [22] Explainability/ interpretability in artificial intelligence (xAI) refers to comprehending the factors behind a black-box model's predictions. This method can be implemented using interpretable machine learning algorithms/glass-box models or model-specific explainable AI(xAI) techniques. However, this leads to a trade-off between performance and interpretation. [23] Therefore, majority of the proposed architectures resort to model-agnostic explanation of AI models.[24] A few popular post-hoc algorithms LIME , SHAP, Grad-CAM and DiCE algorithms.[25], [26]. Two perspectives are used in existing research to categorize xAI methods: (1) local interpretations (2) global interpretations. [27] The former explains the behavioral patterns, thereby, explaining the whole logic of the model. The latter focuses on the decision-making aspects at specific instances, i.e., understanding the model's predictions. [28] used Shapley Additive Explanation (SHAP), a global interpretability algorithm, to interpret the results of the proposed 1DCNN- LSTM-BiLSTM model's RUL predictions. Authors of [29] implemented an explainable BiLSTM framework for intrusion detection by incorporating local as well as global interpretations via Local Interpretable Model-agnostic Explanation (LIME) and SHAP xAI techniques.

2.2. GAPS IDENTIFIED IN LITERATURE

Temporal convolutional network (TCN) integrated with LSTM has shown good performance e results [30], [31] however, this hybrid model has not been implemented for predicting performance in the turbofan engine prediction with interpretable results. Moreover, there is a lack of interpretable TCN-LSTM algorithms in current literature. Thus, this thesis introduces a novel TCN-LSTM-SHAP explainable regression framework and the TCN-LSTM model's results are compared with Conv-LSTM (hybrid of CNN and LSTM) and TCN models for comparison of the prediction results.

3. PROPOSED SYSTEM OVERVIEW

3.1. DATASET INFORMATION

To evaluate the performance of our TCN- LSTM model, we design computational experiments based on the NASA turbofan C-MAPPS dataset. In this section, we will go into

detail about the dataset, its features and the metrics commonly used to evaluate the prognosis dataset.

The NASA C-MAPPS datasets consists of a fleet of engines' run-to-failure data containing multi-variate time series of sensor readings. The synthetic failure data is obtained via run-to-failure simulations carried out using C-MAPSS, a widely used simulation software for the transitory operation of modern commercial turbofan engines. The machine under analysis mimics a high-bypass, twin-spool commercial turbofan engine. [32] The main components of the engine are: fan, Low-Pressure Compressor(LPC) , High-Pressure Compressor(HPC) , combustor or burner, High-Pressure Turbine(HPT) and Low-Pressure Turbine(LPT). The high-speed or core shaft connects the HPC to the HPT while the fan or low-speed shaft connects the fan, LPC and LPT. [33] [34]

The sensor readings mainly comprise fuel flow ,speed , pressures , and temperatures at the different turbofan stages. The initial health conditions of each unit are unknown and the degradation occurs over time, with sensory signal length varying from unit to unit. Notably, all the rotating sub-components of the engine might be affected by flow and efficiency deterioration. The simulation is such that the flight conditions follow the real-time trends of the operational aircrafts. Data augmentation techniques such as dynamic time warping for run-to-failure data scarcity issues are unnecessary due to the sufficient degradation trajectories provided in the dataset.[34]

Table 1 lists the names of the sensor columns which are considered as the features and their variation w.r.t to the target column, RUL obtained via exploratory data analysis (EDA). The ↑ of a sensor implies that the RUL is directly proportional to the sensor, while ↓ the indicates that the sensor is inversely proportional to the RUL, i.e., the RUL value decreases as the sensor value increases. The ~ is used to notate the redundant sensors which remain constant or show no particular trend w.r.t to the RUL. Correlation analysis was also performed as part of the EDA process. Figure 1 depicts the correlation heatmap of the sensor features and operational settings. Figure 2 represents some of the sensor trends w.r.t RUL. The average, maximum and minimum values for a sensor of a particular engine unit are plotted against the RUL(measured in cycles) of that engine. The short-term variance fluctuations are due to the sensor noise which will not be removed during pre-processing to make the model robust to real-time operating conditions.

The primary objective of this NASA Prognostics Data challenge[32], where the dataset was released, is to devise a prognostic model that provides an accurate RUL estimate on the test dataset. The dataset contains three flight classes based on the duration of flight (i.e., short,

medium, and long length). Each flight cycle covers climb, cruise, and descent conditions which correspond to the different flight routes and differing records lengths.

The NASA C-MAPPS repository consists of four datasets, each configured distinctly in terms of number of failure modes ,number of engines and operational conditions. Each sub-dataset contains separate files for training , testing and ground truth(RUL) data. Table 2 summarizes the sub-datasets in the C-MAPPS dataset. This paper utilizes the FD003 dataset. To evaluate the performance of our TCN- LSTM model, we design computational experiments based on the NASA turbofan C-MAPPS dataset. In this section, we will go into detail about the dataset, its features and the metrics commonly used to evaluate the prognosis dataset.

The NASA C-MAPPS datasets consists of a fleet of engines' run-to-failure data containing multi-variate time series of sensor readings. The synthetic failure data is obtained via run-to-failure simulations carried out using C-MAPSS, a widely used simulation software for the transitory operation of modern commercial turbofan engines. The machine under analysis mimics a high-bypass, twin-spool commercial turbofan engine. [32] The main components of the engine are: fan, Low-Pressure Compressor(LPC) , High-Pressure Compressor(HPC) , combustor or burner, High-Pressure Turbine(HPT) and Low-Pressure Turbine(LPT). The high-speed or core shaft connects the HPC to the HPT while the fan or low-speed shaft connects the fan, LPC and LPT. [33] [34]

The sensor readings mainly comprise fuel flow ,speed , pressures , and temperatures at the different turbofan stages. The initial health conditions of each unit are unknown and the degradation occurs over time, with sensory signal length varying from unit to unit. Notably, all the rotating sub-components of the engine might be affected by flow and efficiency deterioration. The simulation is such that the flight conditions follow the real-time trends of the operational aircrafts. Data augmentation techniques such as dynamic time warping for run-to-failure data scarcity issues are unnecessary due to the sufficient degradation trajectories provided in the dataset.[34]

Table 1 lists the names of the sensor columns which are considered as the features and their variation w.r.t to the target column, RUL obtained via exploratory data analysis (EDA). The ↑ of a sensor implies that the RUL is directly proportional to the sensor, while ↓ the indicates that the sensor is inversely proportional to the RUL, i.e., the RUL value decreases as the sensor value increases. The ~ is used to notate the redundant sensors which remain constant or show no particular trend w.r.t to the RUL. Correlation analysis was also performed as part of the EDA process. Figure 1 depicts the correlation heatmap of the sensor features and operational settings. Figure 2 represents some of the sensor trends w.r.t RUL.

The average, maximum and minimum values for a sensor of a particular engine unit are plotted against the RUL(measured in cycles) of that engine. The short-term variance fluctuations are due to the sensor noise which will not be removed during pre-processing as in order to make the model robust to real-time operating conditions.

The primary objective of this NASA Prognostics Data challenge[32], where the dataset was released, is to devise a prognostic model that provides an accurate RUL estimate on the test dataset. The dataset contains three flight classes based on the duration of flight (i.e., short, medium, and long length). Each flight cycle covers climb, cruise, and descent conditions which correspond to the different flight routes and differing records lengths.

The NASA C-MAPPS repository consists of four datasets, each configured distinctly in terms of number of failure modes ,number of engines and operational conditions. Each sub-dataset contains separate files for training , testing and ground truth(RUL) data. Table 2 summarizes the sub-datasets in the C-MAPPS dataset. This paper utilizes the FD003 dataset.

Table 1 List of sensor features and their variation w.r.t to RUL for FD003 N-CMAPPS dataset

Sensor Number	Symbol	Description	Units	Trend
1	T2	Total temperature at fan inlet	ºR	~
2	T24	Total temperature at LPC outlet	ºR	↓
3	T30	Total temperature at HPC outlet	ºR	↓
4	T50	Total temperature at LPT outlet	ºR	↓
5	P2	Pressure fan inlet	psia	~
6	P15	Total pressure in bypass-duct	psia	~
7	P30	Total Pressure at HPC outlet	psia	↑
8	Nf	Physical fan speed	rpm	↑
9	Nc	Physical core speed	rpm	↑
10	epr	Engine pressure ratio	-	~
11	Ps30	Static pressure at HPC outlet	psia	↓
12	Phi	Ratio of fuel flow to Ps30	pps/psi	↑
13	NRf	Corrected fan speed	rpm	↓
14	NRc	Corrected core speed	rpm	↓
15	BPR	Bypass ratio	-	↑
16	farB	Burner fuel-air ratio	-	~
17	htBleed	Bleed enthalpy	-	↓
18	Nf_dmd	Demanded fan speed	rpm	~
19	PCNfR_dmd	Demanded corrected fan speed	rpm	~
20	W31	HPT coolant bleed	lbm/s	↑
21	W32	LPT coolant bleed	lbm/s	↑

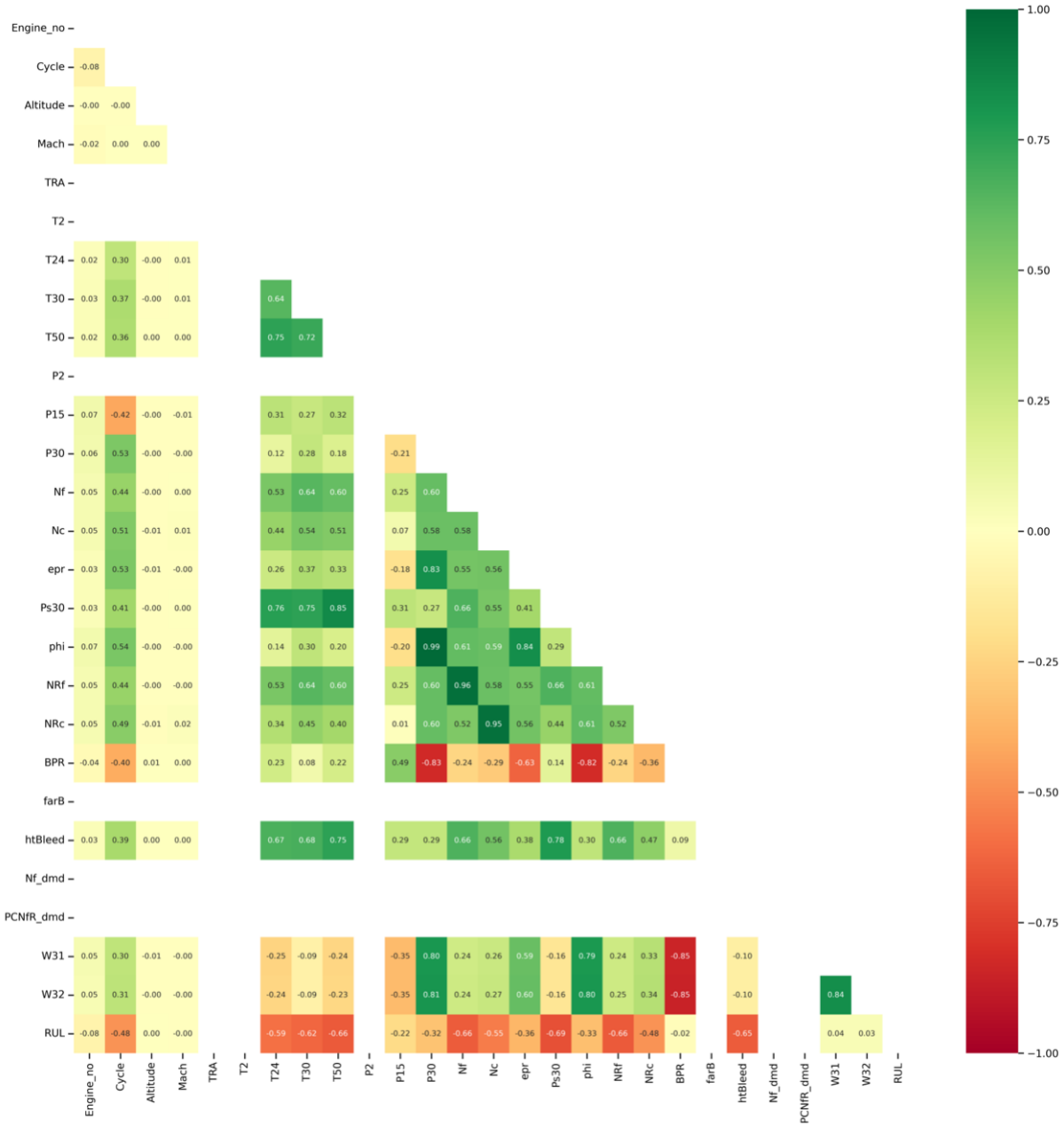

Figure 3-1 Correlation heatmap

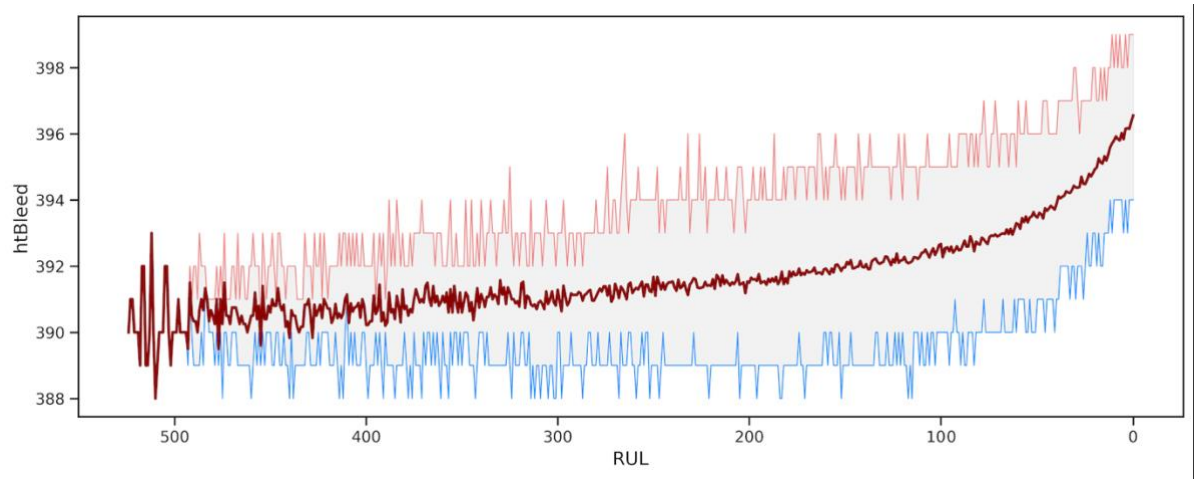

Figure 3-1 Variation of Bleed Enthalpy with RUL

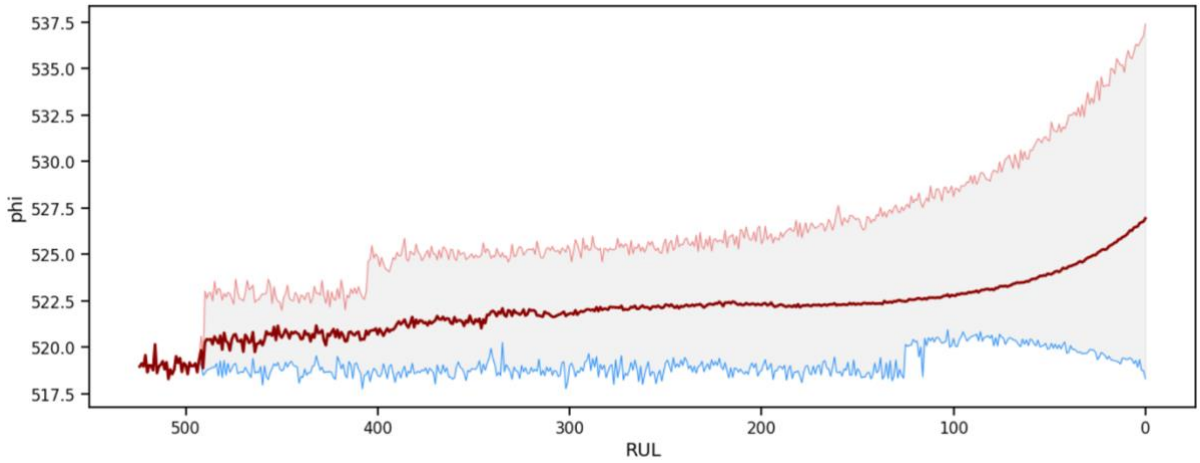

Figure 3-2 Variation of Ratio of fuel flow to Ps30 with RUL

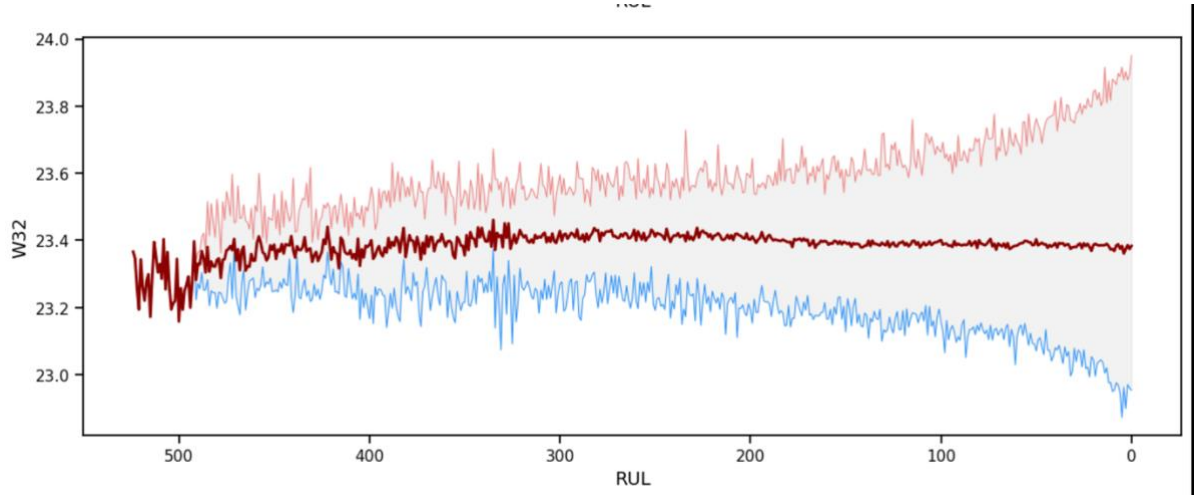

Figure 3-3 Variation of LPT Coolant Bleed with RUL

Minimum , maximum and mean values variation of a few sensors w.r.t RUL.

Dataset	FD001	FD002	FD003	FD004
Number of engines	100	260	100	249
Number of training samples	20,631	53,579	24,270	61,249
Number of test samples	100	259	100	248
Number of columns	26	26	26	26
Average life span(cycles)	206	206	247	245
Operating conditions	1	6	1	6
Failure modes	1	1	2	2

Table 2 Overview of the sub datasets in C-MAPPS NASA repository

3.2. DATA PRE-PROCESSING

3.2.1 Piecewise Linear Degradation

The value of the target column, remaining useful life(RUL) in this case, needs to be fed during model training for data-driven prediction methods. The RUL column corresponds to the duration from the time at which the input data columns were recorded to the time of engine failure. This approach views the simulated engine's deterioration in performance as a linear drop. For an actual engine, it is said to be in a healthy state at the beginning of operations with degradation occurring after a certain operational cycle. Thus, to mimic this process, this paper adopts a piecewise linear function with piecewise linear degradation curve starting at 125(initial RUL value), stays at 125 for some time and then linearly degrades to 0 for each engine in the N-CMAPPS FD003 and FD004 datasets. It was observed that the degradation usually occurred at the late 120-130 cycles.[30] Formula 1 shows the function and Figure 2 depicts the piecewise linear degradation model used for RUL calculation.

$$\text{RUL}=\begin{cases} 125, RUL \geq 125 \\ RUL, 0 \leq RUL \leq 125 \end{cases} \quad (1)$$

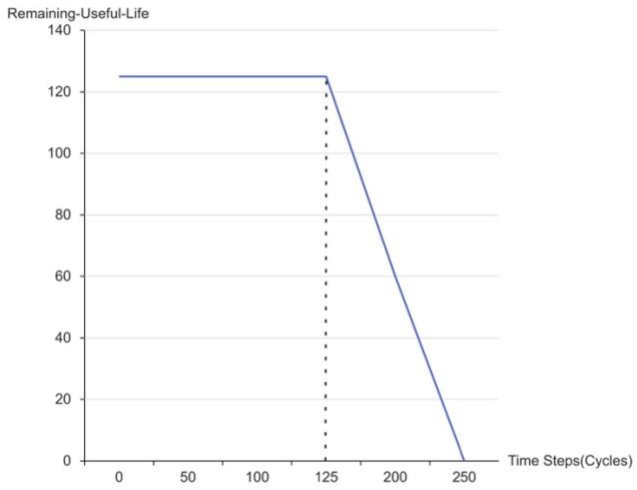

Figure 3-4 Piecewise linear degradation model for RUL prediction

3.2.2. Sliding time window algorithm

The dataset consists of multivariate time series thus, to find patterns which may not be visible while analyzing the entire dataset, we split the time series data into smaller, fixed-length segments or windows, with each window overlapping the previous one by a certain amount. This pre-processing technique is known as sliding window or lag method. Adding lag to the data, reduces the effect of outliers apart from reshaping the data as input required for training neural network models. [35]

In this paper, the sliding time window is used to divide the time series data into batches of 2-dimensional matrices with time steps equal to the size of lag or window length. Let the original sequence be a matrix $X(n) = [x^1,x^2,..,x^{Ls}]$ where x^i represents the data collected by k sensors at time i, n represents the number of engines and L_s represents the running time of the engine's failure point. Assuming the window length is m and the input sequence being X(n) each division generates a sample of size m x k . The shift or step size represents the overlap of the data rows. In this paper, we set the shift to 1. Thus, the total number of samples are L_s – m + 1 with each batch containing samples of size m x k. The diagrammatic representation of the sliding window process is shown in Figure 3.

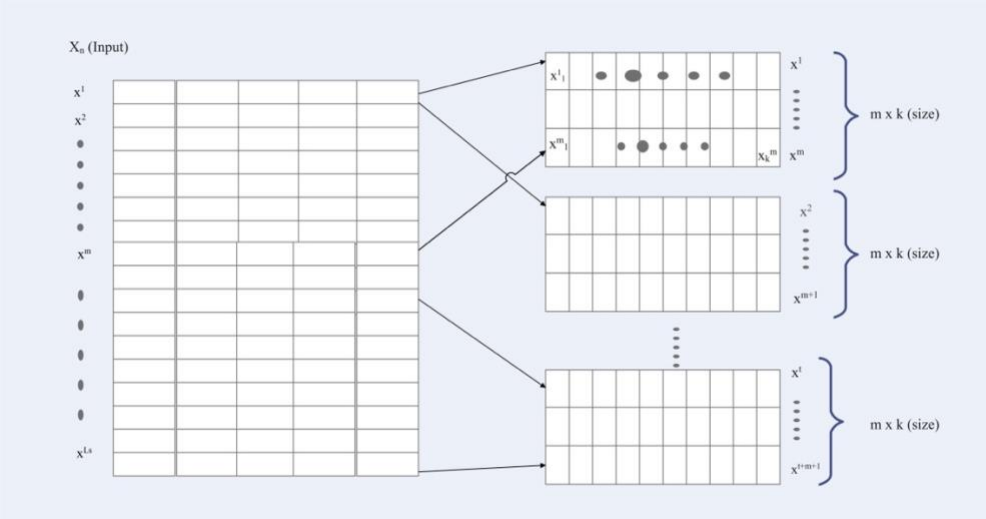

Figure 3-5A diagrammatic representation of the sliding window process for an input sequence Xn .

In general, a larger time window derives more information from the samples about the trend in the engine's health. However, the complexity increases with the size of the window as a very large window size will require a long data sequence as input which also might inadvertently remove the short-term trend variations. Thus, to ensure data integrity, the window length is dynamically chosen in accordance with the input data. We have fixed the window length to 30, and shift=1, which means there is an overlap of 29 data points between 2 consecutive batches.

3.2.3 Feature Scaling

Some sensor readings in the C-MAPPS dataset contain constant values which are not useful as real engines' sensors record values fluctuating in magnitude and units. Thus, these columns were removed from the input data. The remaining columns were subjected to normalization which involves fixing the values of a dataset to a specific range. This is to standardize the data and thereby, make it easier to compare and analyze. In this paper, min-max normalization was used. This scaling technique sets the values to a range of [-1,1] and is based on the minimum and maximum values of the feature variable. Formula 2 shows the equation for the normalization function. For an input sequence of k sensors, $X= [x^1, x^2, \ldots x^K]$, with $x^i= [x^{1,i}, x^{2,i}, \ldots x^{j,i}]^T$ representing the time series data for *j*-th sensor, the j-th normalized value is calculated as follows:

$$x^i = \frac{2(x^i - min(x^i))}{max(x^i) - min(x^i)} - 1 \qquad (2)$$

3.3. PROPOSED ARCHITECTURE

3.3.1 Temporal Convolutional Network Model

The convolutional neural network (CNN) variant architecture was first proposed by [36] and proven to be optimal for sequence modelling. The following principles were considered while modifying the CNN architecture: (1) ensuring that there is no leakage of data from the future to the past. (2) sequential labelling, i.e., accepting input sequence of any length and returning an output sequence of the same length, like recurrent neural networks (RNN). Thus, the distinguishing characteristics that set apart temporal convolutional network (TCN) are: (1) causal convolutions, i.e., convolving only features at time *t* and earlier(*t-1,t-2…0)* to generate output at time *t.* (2) 1D fully connected CNN architecture , where each hidden layer is the same as input layer. (3) dilated causal convolutions, which enables looking back to exponentially larger receptive field unlike the simple causal connections which only allow look backs to a receptive field with size linear to the depth of the network. (4) residual blocks which are the equivalent of convolutional layers. [37]

Figure 4 and Figure 5 represent a dilated causal operation and the difference between causal and dilated causal convolutions for a 1D CNN layer respectively. The dilation factor (in factors of two, i.e., 2^0, 2^1., etc.) control the depth of the TCN network and is usually specified in factors of two as depicted in the figure. A larger dilation factor allows longer input sequences to be fed into the top layer, thereby, decreasing the depth of the dilated convolution for same sequence length. However, one must ensure that receptive field does not cover all elements of the input sequence. The input sequence is convolved according to the kernel size and the causal property ensures that the output y_t is generated from input sequences (x_t ,x_{t-1},..,x_1).

TCN is composed of residual blocks, which allow the output of a residual block to be added as input of the next block. The activation function passed in the block is ReLU, facilitating non-linear data processing. The activation function for input x of a block is defined as follows:

$$o = Activation\,(x + \mathrm{F}(x)) \qquad (3)$$

Where o is the activation function, x is the block input and $F(x)$ is a series of transformations applied on x. Figure 6 shows the residual block architecture in the temporal convolution network.

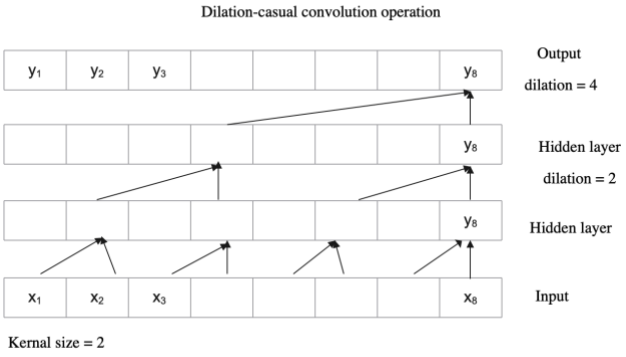

Figure 3-6 Depiction of dilated-causal operation

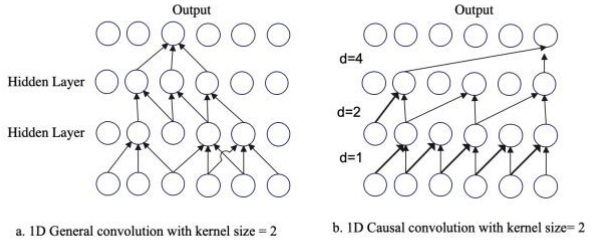

Figure 3-7 1D- General and Causal Convolutions

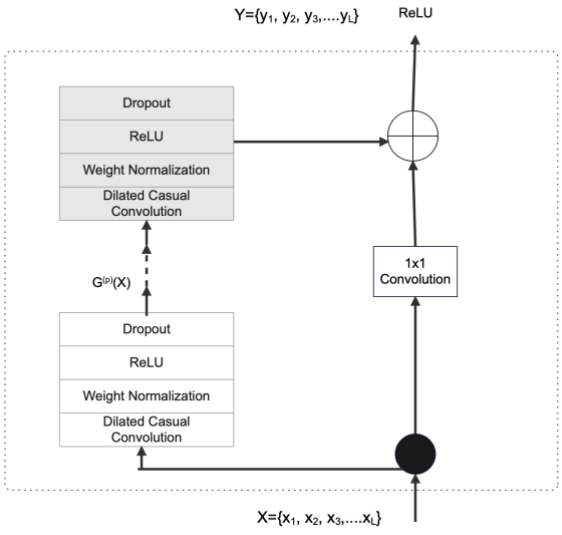

Figure 3-8 Diagrammatic representation of a residual block in TCN

3.3.2 Long Short Term Memory Model

Long Short-Term Memory is an advanced variant of recurrent neural networks(RNNs) .RNNs are popular for forecasting and sequence modelling problems due to their characteristic of adding the output sequence at a previous(t-1) timestep as an additional feature to the input sequence of the current timestep (t) .[38] The LSTM network carries the advantages of the RNN while solving the vanishing gradient problem, thus enabling it to learn long-term dependencies. The architecture comprises of three gates: (i) Input gate (I_g) (ii) Output gate(O_g) (iii) Forget gate(F_g) ,a hidden state(h_t) and a cell state(c_t) . The input gate acts a control for information fed into the cell state while the output gate regulates the outgoing information from the cell state. The forget gate controls the amount of information forgotten from the cell state. Figure 7 shows the architecture of basic cell state.

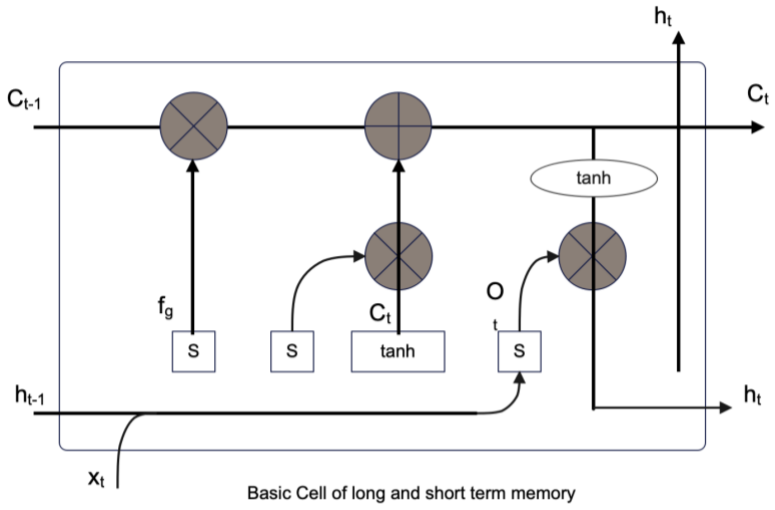

Figure 3-9 Basic cell state representation in LSTM

3.3.3. CNN-LSTM Model

CNN-LSTM , an integration of the convolutional neural network(CNN) and LSTM, has combined the advantages of its predecessors and is found to deliver high performance on time series analysis such as traffic flow prediction and weather forecasting.[39] The spatial feature extraction by the CNN with the temporal feature extraction by LSTM makes CNN-LSTM suitable for sequence labelling.

In this paper, two convolution layers with 128 neurons each and kernel size=3 are used alternatively with two max pooling layers. Dropouts are added for regularization. After the input sequence is processed and returned after reducing the dimensionality and passed to the

first LSTM layer with 128 neurons. The output is returned as full sequences to enhance forecasting and is fed into the second LSTM layer with 128 neurons. Both LSTM layers use Tanh activation, and the penultimate output is passed through one dense layer with 1 neuron.

3.3.4. Proposed TCN- LSTM Model

The prediction model used in this integrates TCN and the LSTM networks. The initial input sequence generated is passed into the TCN to convolve the input data with a greater number of features to a fewer time steps The TCN produces full length temporal output sequences propagated through the LSTM network. The LSTM layer is comprised of two LSTM layers with 128 and 96 neurons respectively. The tanh activation function is implemented in both layers and the output sequence generated is passed through two dense layers of neurons, comprising of 128 neurons and 1 neuron respectively. The second dense layer is added to make the output suitable for the SHAP algorithm. A dropout layer is added with a dropout of 0.2 to prevent over-fitting. Thus, the final output sequence generated is a 1-dimensional array containing the RUL predictions. Figure 7 represents the architecture of the proposed model.

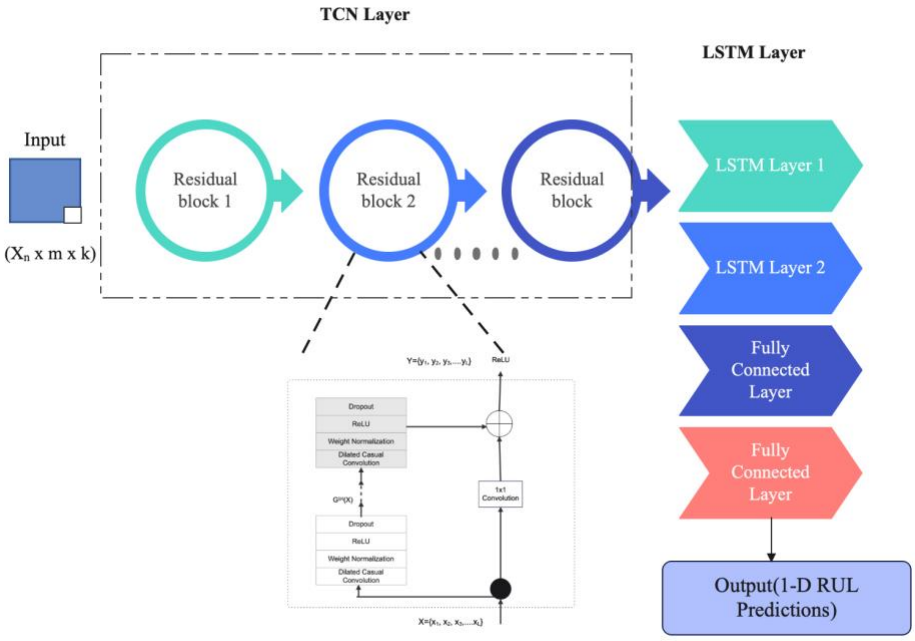

Figure 3-10 Architecture of the Proposed TCN- LSTM

3.3.5 SHAPely Additive Explanations (SHAP)

SHAP is a model agnostic algorithm which provides local explanations, i.e., explanations on individual predictions. The contribution of a feature for the target label's prediction is determined by its Shapely values which is calculated using the coalitional game theory and

have been establish for satisfying properties such as local accuracy, missingness and consistency. These features enable SHAP to provide local and global explanations. This means that one can identify the features which have the most contribution on model's predictions overall as well on an individual sample basis. Each feature is considered as a player and the shapley value for a feature is calculated as the average of its marginal contributions which is computed as follows:

$$\phi_i(v) = \sum S \in N\{i\} \frac{|S|!(n-|S|-1)!}{n!}(v(S \cup \{i\}) - v(S)) \qquad (4)$$

Where $\phi_i, n, S, v(S)\ and\ v(S \cup \{i\})$ are the shapely values for ith data, all participants, all sets except i-th feature in the whole group the contribution coefficient of the remaining subset ,the contribution coefficient of the remaining subset except for the i-th feature, and the total contribution including the i-th feature, respectively. [40]

SHAP can be used for interpreting ML models trained on a range of data types, varying from textual to visual data. has shown the scope of application in several fields such as healthcare, agriculture, and defense. [25][41]

3.3.6 Feature Selection and Hyperparameter Tuning

The 2-dimensional input data is re-formatted into 3-dimensional input sequences by applying the feature sliding window algorithm and other methods mentioned in section B. Additionally, the sensor columns which have constant values and follow no trend w.r.t to RUL were dropped. Table 3 lists the features that were dropped.

The TCN model from [36] is introduced as the first layer in the hybrid structure. Its receptive field was set with dilation factor at (1,2,4) and kernel size=2.The TCN model generates full length output sequences using skip or residual connections. The full sequences of RUL returned by the TCN layer is flattened and then processed by the LSTM layer. The hybrid model introduces two LSTM layers with 128 neurons and 64 neurons respectively. The first LSTM layer is initial to return the entire output sequences while the second layer is configured to return only the last output sequences. The Rectified Linear Unit Function (ReLU) is used as the activation function since it is found to perform well with non-linear time series forecasting problems. [42]

Bayesian Optimization: Bayesian optimization is a hyper-parameter tuning method which runs several trials and selects the hyper-parameters for the tuned model by evaluating the past

model information. This paper has implemented Bayesian Optimization since it is time effective as compared to other methods like Randomized Search and Grid Search.[43] This paper used 10 trials and 2 executions per trial to evaluate the model performance. The hyper-parameters are detailed in Table 4.

To make the predictions more robust we also use multiple test windows where we specify the number of sliding windows per engine to test the model instead of just taking in the last test window of each engine unit. The result is the average of the predictions from each window. In this paper the number of test windows were set to 5. The proposed TCN-LSTM model is also evaluated using the last window of each engine in the test data for comparison with other works in literature.

	Features Dropped
Operational Settings	Altitude Mach TRA
Sensor Features	T2 P2 P15 P30 Epr farB htBleed Nf_dmd PCNfR_dmd

Table 3 List of features that are dropped which were found to downgrade performance

Model	*Hyper-parameters*	
CNN-LSTM	Epochs	150
	Batch size	64
	Filters in 1D CNN	128
	Units in LSTM	128
	Number of LSTM layers	2
	Number of CNN layers	2
	Pooling size(1D pooling)	2
	Dense units	1
	Activation (LSTM)	Tanh
	Activation(dense)	Linear
	Return sequences(LSTM)	Enabled
	Kernel size	3
TCN	Kernel size	2
	Dilations	(1,2,4)
	Skip connections	Enabled
	Filters	64
	Return sequences	Enabled
	Dense number of units and layers	1,1
Proposed TCN-LSTM	Kernel size	2
	Dilations	(1,2,4)
	Skip connections	Enabled
	Filters	64
	Units in LSTM	(128,64)
	Units in Dense	(128,1)
	Dropout Rate	0.2
	Return sequences(TCN and LSTM)	Enabled for TCN layer and LSTM layer
	Activation	ReLU

Table 4 Hyper-parameter settings for the models , TCN, CNN-LSTM and Proposed TCN- LSTM

After the performance evaluation of the deep learning models, a sample of the prediction is fed into the Shapley algorithm to obtain a force plot and summary plot of the contribution of the feature sensors to the predictions. Figure 8 depicts the workflow of the proposed TCN-LSTM-xAI system.

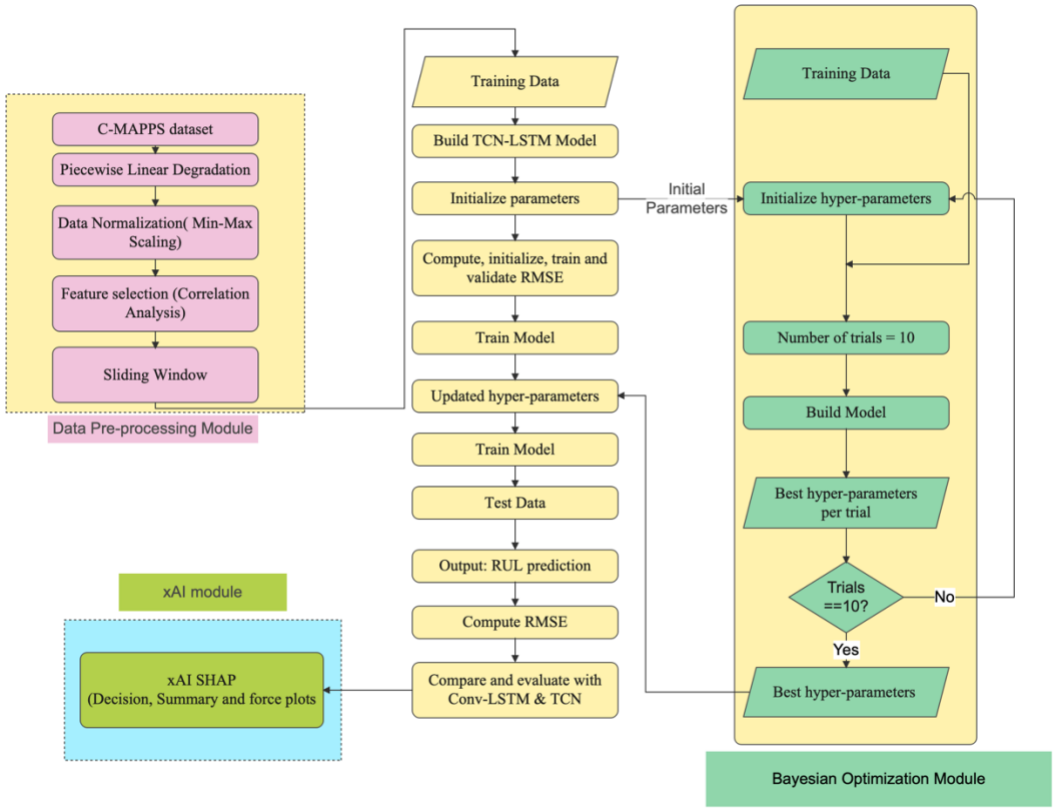

Figure 3-11 Workflow of the proposed TCN-LSTM-xAI framework

4. PROPOSED SYSTEM ANALYSIS AND DESIGN

4.1 INTRODUCTION

The Proposed Temporal convolutional network (TCN) and Long Short Term Memory (LSTM) is evaluated on the NASA FD003 dataset along with the CNN-LSTM and standalone TCN models. The entire work is implemented in Python JupyterLab environment. The proposed system design is scalable and novel, showing scope in industrial applications. The primary implementation is in Python followed by HTML and Dash Bootstrap.

4.2 REQUIREMENT ANALYSIS

4.2.1 FUNCTIONAL REQUIREMENTS

4.2.1.1 Product Perspective

The proposed work has scope for real-world industrial applications. The proposed framework can be used to predict and gain insights into the contribution of features for the model's output results, in this case, operational lifetime i.e., remaining-useful-life (RUL) of turbofan

engines. The explainability offered by the proposed system makes the proposed TCN-LSTM model more transparent.

4.2.1.2 Product Features

The features of the proposed system are listed below:

- Data pre-processing techniques: Pre-processing methods for time series data such as sliding window, piecewise linear degradation are proposed.
- Evaluation of model: The TCN-LSTM model is evaluated on the processed data
- Comparison of models: The proposed method is evaluated against two other state-of-the-art models, CNN-LSTM and LSTM.
- xAI Module: The xAI module provides explanations behind a model's output RUL prediction decisions.
- Predictive Dashboard: The visualizations and trend analysis are presented in an UI to facilitate easy comprehension.

4.2.1.3 User characteristics

A few user characteristics are as follows:

- Technical expertise: Users of this system should have some basic understanding of machine learning and deep learning concepts. Domain knowledge in the aircraft sector will aid in interpreting the prediction results
- Data availability: Users of this system will need to have access to historical data of the turbofan engine failure from past records or simulations.
- Validation dataset: There must be sufficient data for a validation dataset used to reduce over-fitting

4.2.1.4 Assumptions and Dependencies

- The proposed system assumes that the historical training data is sufficient to generate multiple sequences containing 30 windows each. In case of insufficient data, it is recommended that data augmentation or padding is done beforehand.
- The training data is assumed to consist of run-to-failure engine sensor data while the testing data collection is to be stopped prior to engine failure.

4.2.1.5 Domain Requirements

- The web application is hosted on the local server :https://127.0.0.1:8050 using Flask via Dash Plotly

- A GPU runtime environment(local or cloud based) is recommended for training the model.
- The following packages are necessary for deployment: Tensorflow Keras, Pandas, Scikit-learn, Numpy and Flask.

4.2.1.6 User Requirements

- The proposed app can be deployed any naïve user due to the transparency of the deep learning model using SHAP visualizations.
- For detailed analysis of the data, domain knowledge is preferred.

4.2.2. NON-FUNCTIONAL REQUIREMENTS

4.2.1.1 Product Requirements

4.2.1.1.1 Efficiency

Space Complexity analysis:

1. The space complexity for an LSTM network is computed as:

$$\text{space complexity} = O(n * h) \qquad (5)$$

where n is the number of time steps in the sequence and h is the number of hidden units in the network

Thus, the space complexity for the LSTM layer in the proposed system is

$space\ complexity = \Theta(30 \times 128) = \Theta(1)$

2. The space complexity for a CNN is computed as

$$space\ complexity = \Theta(n \times m \times k) \qquad (6)$$

where n is the number of adh is the height of the input image, m is the width of the input image, and k is the number of feature maps in the first convolutional layer.

Thus ,the space complexity for a CNN network is

$$Space\ complexity\ of\ CNN = \Theta(30 \times 15 \times 128) = \Theta(1)$$

3. The space complexity for a TCN is computed as

$$space\ complexity\ of\ TCN = \Theta(n \times m \times k \times l) \qquad (7)$$

where n is the number of time steps in the input sequence, m is the number of features in each time step, k is the number of feature maps in the first convolutional layer, l is the number of output feature maps.

Thus, the space complexity of the TCN network becomes $\Theta(30 \times 15 \times 64 \times (64 \times 3)/0.25) = \Theta(1)$

To sum up the space complexity for the models implemented are:

Proposed TCN- LSTM = $\Theta(1)$

$\text{CNN-LSTM} = \Theta(1)$

$\text{TCN} = \Theta(1)$

Time Complexity analysis:

The time complexity of the deep learning models are measured in the FLOPS (Floating operations per second) and computed for each layer of the network.

The formula for the 1D convoluted layer is

$$FLOPS = 2 * kernel\_size * number\_of\_kernels * output\_shape. \quad (8)$$

Thus, inputting the values of the parameters, we get,

$$FLOPS = 2 \times 3 \times 3 \times (30 \times 128) = 69120\ FLOPS$$

The FLOPS for the pooling layer is computed as:

$$Pooling = heigth \times width \times depth\ of\ the\ image. \quad (9)$$

Thus, we get,

$$FLOPS = 30 \times 15 \times 15 = 13{,}500\ FLOPS$$

The FLOPS for the fully connected (dense) layer is:

$$Dense\ FLOPS = 2 \times Input\ size \times Output\ size. \quad (10)$$

Thus, we obtain,

FLOPS= 2 * 200*1 = 256 + 2(from second dense layer) =258 FLOPS

The FLOPS for an LSTM Layer is computed as follows:

$$LSTM\ FLOPS = 2 * (input\_size * hidden\_size * 4) * num\_layers. \quad (11)$$

Using this formula, we get,

FLOPS for LSTM = 2 * (8 * 128 *128 *4)* 2 = 2, 097.152 = 2M FLOPS

Thus, the time complexity for CNN- LSTM is 2,165,498 FLOPS

The time complexity for TCN is computed as:

$$2 \times no\_input\_channels \times no\_output\_channels \times kernel\_size \times timesteps \quad (12)$$

Thus , the time complexity for TCN = 2*15*2*30 = 27,000 FLOPS

The time complexity of the proposed model is

Proposed TCN- LSTM = 2* (30* 64* 128* 4) *1 + 2*(30*128*4)*1 + 15*30*2*30*2= 2,050,800 FLOPS

4.2.1.1.2. Reliability

The proposed system achieved comparable results and even outperformed several state-of-the-art methods upon evaluation.

4.2.1.1.3. Scalability

Although the training is computationally expensive, the deployment is fast and scalable due to the saved weights and parameters of the proposed model. The project repo contains the keras model file which can be loaded for predictions.

4.2.1.2 Organizational Requirements

4.2.1.2. Implementation Requirements

- The github repository needs to be exported
- Run requirements.txt file to install necessary packages
- Run-time GPU environment required fro model training, the model can be loaded from the trained models for execution.
- The app will be hosted on the Dash server https://127.0.0.1/port:8050

4.2.1.3. Operational Requirements

4.2.1.3.4 Economic

The operational requirements for deep learning rul prediction system economically can be summarized as follows:

- Data: Historical industrial machinery failure data collected over a period of time is required.
- Computational resources: Deep learning models are computationally expensive to train and deploy.

4.2.3 SYSTEM REQUIREMENTS

4.2.3.1 Hardware Requirements

The evaluation of the proposed method is done on Anaconda mini-forge distribution environment. The operating system consists of an 8 cores GPU, 32 GB RAM and M1 Pro chip processor.

4.2.3.2 Software Requirements

The primary library used for the model implementation is the Keras package in Tensorflow. The TCN layer was adapted from [44] Github repository. The data app was built using Dash Plotly, a framework which supports several languages such as Python, React, Flask , HTML and CSS.

5. RESULTS AND DISCUSSION

The evaluation metrics used in this paper are as follows:

RMSE: The root mean squared error is a popular metric for RUL estimation. [45]The formula is:

$$RMSE = \sqrt{\frac{\sum_1^n i \ (Predicted_i - \ Actual_i)^2}{N}}$$

R^2: The formula is as follows:

$$R^2 = \ 1 - \ RSS/\ TSS$$

where RSS is the sum of squares of residuals and TSS is the total sum of squares.

MAE: Mean absolute error

$$MAE = \frac{\sum_1^n i \ |Predicted_i - \ Actual_i|}{n}$$

A. Prediction Results

Table 5 depicts the predictions averaged from 5 test windows for all three algorithms. We can observe from Table 5 that the proposed method shows an 18% decrease approx. from the CNN-LSTM model and around 2% improvement when compared to the TCN algorithm. Although the percentage improvement is minimal in the latter case, the proposed algorithm has achieved comparable results.

Method	**RMSE**	**MAE**	**R-squared**
CNN-LSTM	17.471	12.45	0.821
TCN	14.309	10.505	0.8805
Proposed TCN-LSTM	**14.291**	**10.55**	**0.8807**

Table 5. Experimental Results by averaging the results of the last 5 test windows for each engine

The proposed method is evaluated using only the last test window(of the 5 test windows generated) for comparison with current state-of-the-art models in literature. Figure 6 plots the predicted RUL values against the actual RUL values provided as the ground truth PM challenge. Table 6 shows the metrics for the proposed TCN- LSTM using only the last test window for each engine and Table 7 shows the comparison with other models in literature on the same FD003 sub-dataset.

Proposed TCN-LSTM	
Root Mean Squared Error	12.772101
Mean Absolute Error	8.888184
R^2	0.904805

Table 5 Results of the proposed method using only last test window (out of 5 windows) for the FD003 NASA C-MAPSS dataset.

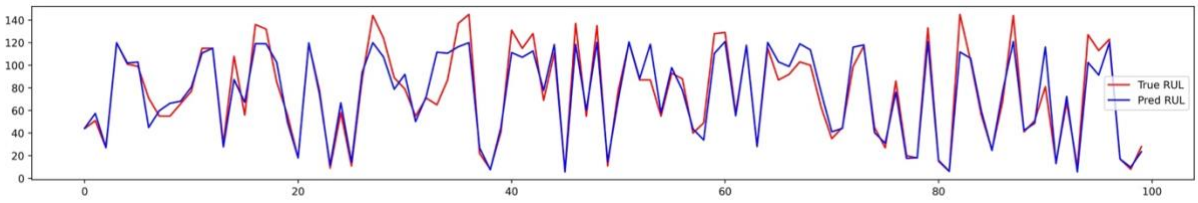

Figure 8. Actual vs Predicted values for RUL by TCN-LSTM (using only last test window for each engine)

Methods	**RMSE**
TCN[21]	13.43
CAE-with BDGRU and BDLSTM[46]	13.41
TCN-BiLSTM[21]	17.85
CNN-RNN[44]	17.8226
Proposed TCN- LSTM	**12.772**

Table 6 Comparison of proposed methodology with existing works

B. xAI SHAP Interpretations

For gaining insight into the proposed model's predictions, the SHAP explainer was trained on 100 samples of the train dataset and computed SHAP values for the first 30 samples of test data (i.e., sample 1's engine unit number is 1, sample 2's is 2 and so on). Generally, the colors of the values represent the contributions of features. Red value means that the feature increases the model's RUL value while the blue denotes that the feature decreases the model's RUL value.

Figure 9 depicts the decision plot for the last window in test engine number 30 in FD004.

- The decision plot shows the contribution of sensors for a prediction in decreasing order of importance. For engine 30, T50, total temperature at LPT outlet, followed by phi (ratio of fuel flow to Ps30), P30(total pressure at HPC outlet) and Ps30(static pressure at HPC outlet) are the sensors with the largest contribution.
- The plot is centred on the base value which is the average of the model over the test dataset. It lies on the x-axis, here, the base RUL value of the TCN-LSTM model's predictions for the first 30 engines in the test dataset is 90.5 cycles approx.
- The predicted RUL value is indicated at the top where the line strikes at i.e., 92.3 cycles approximately and this determines the color of the line, i.e., red line here since the predicted value is higher than the actual model value.
- The shapely values of each sensor from the bottom to the top are added cumulatively to the base value of the model to produce the final output value. We can observe here, that sensor W32 , LPT coolant bleed, is the only sensor that slightly reduces(subtracts) the base RUL value while the others only add on , thereby leading to the increase in the output RUL.

Figure 10 shows a force plot representing the same sample as Figure 9.

- The force plot clearly depicts the range and direction of the impact of contribution (positively or negatively). For example, sensor T50, decreases the value of the output RUL from the base value the most while phi increases the base value the most.
- Another advantage is showing the precise predicted value which is 92.34 cycles.
- A disadvantage of the force plot is that all features are not clearly listed. For instance, like figure 9, figure 10 also shows that P50 followed by phi are the highest contributors. However, it fails to show the lowest contributors (sensor W31 and sensor T30), stopping the plot at sensor Nf, thereby omitting the three least contributing features.

Figure 11 shows the decision plot of the overall predictions for engine 30, including all 30 test data windows that were generated during pre-processing phase as window length. The blue lines depict the windows of data that outputted a lower RUL than the base RUL(90.5 cycles), with the lowest predicted RUL being 85 cycles approx. and the largest predicted RUL value as 96 cycles.

Figure 12 depicts a bar plot depicting the importance of features w.r.t predictions of engine window =30 of Test engine no.29.

Figure 13 represent the summary plot for the overall predictions (all 30 windows) for engine 30, like Figure 11.

- The advantage of summary plots is that we can estimate by how much a particular feature increases or decreases the model output value, i.e., see the range of a sensor's impact on the model output value.
- The values of a feature plotted at 0 implies that the model ignores the sensor value for that sample of prediction. The x-axis value of the feature represents its impact on the output. For example, the lowest value for sensor Nf at a particular window of engine 30 is -2, which means that the sensor decreases the model base output value (90.5 cycles) by 2. Similarily, the maximum value for W32 sensor is 1 from which we can infer that the sensor increases the model's base value to 91.5 cycles.

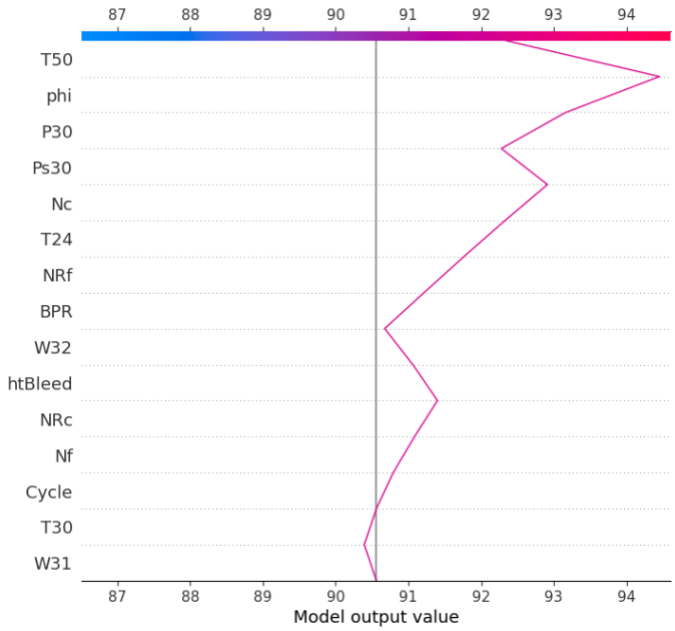

Figure 5-1 Decision Plot of the Proposed TCN-LSTM on the last window of data (window number =30) of the test engine unit no. = 30 with actual RUL value= 89 cycles

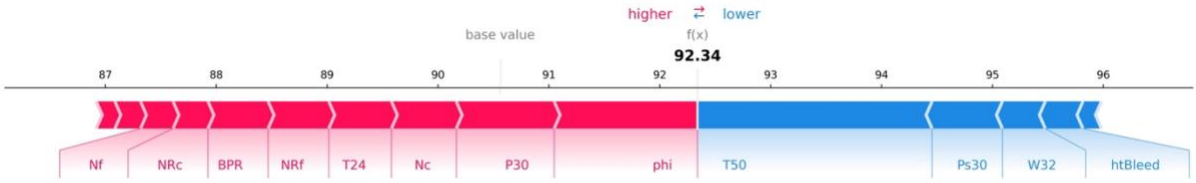

Figure 5-2 Sensor explanations provided by SHAP force plot for the last window of data (window =30) of the test engine unit no. 30 with the actual RUL value at 89 cycles and predicted value is 92.25 cycles

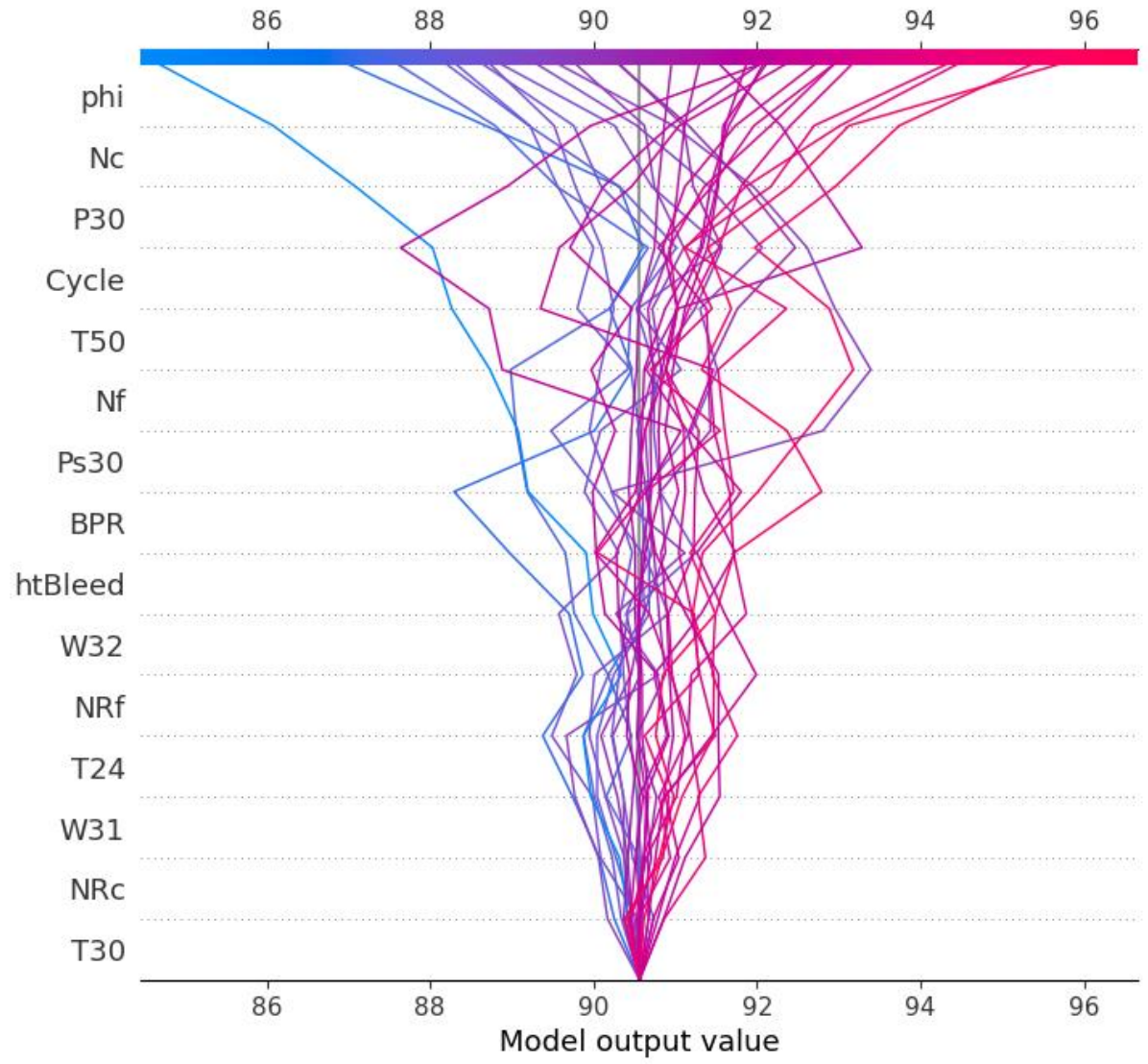

Figure 5-3 Decision plot of the proposed method on the test engine unit no. = 30 for data in all windows (30 in total). The actual RUL is 89 while the model's predicted RUL is 91

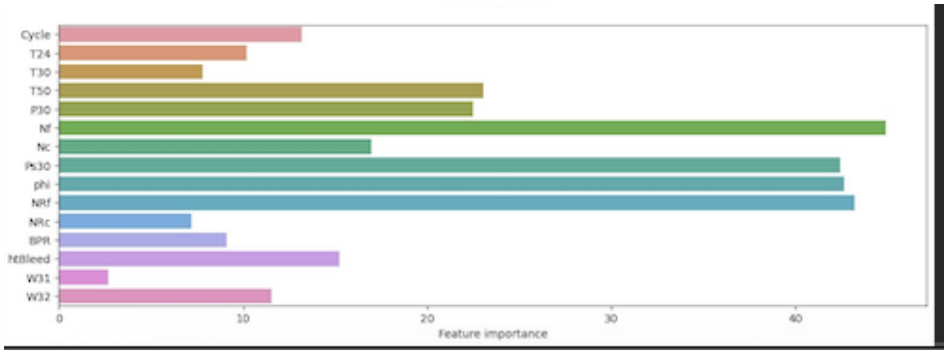

Figure 5-4 Feature importance bar plot for window=30 in test engine unit no. 29

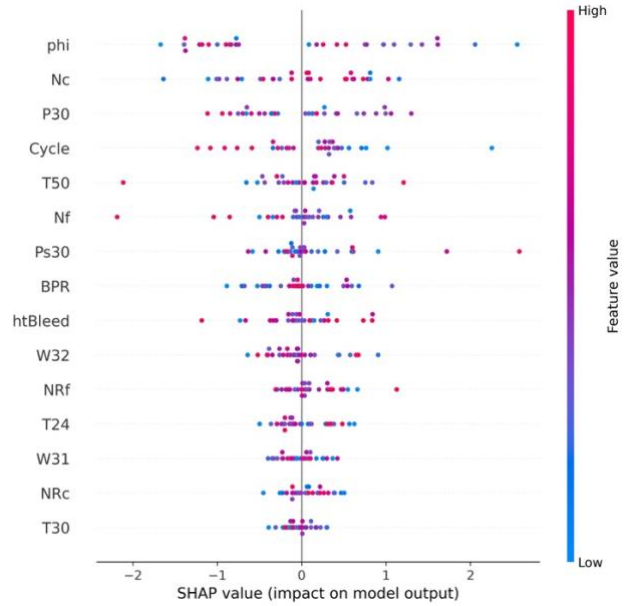

Figure 5-5 Summary plot for engine number 29, considering all 30 windows of data for that unit.

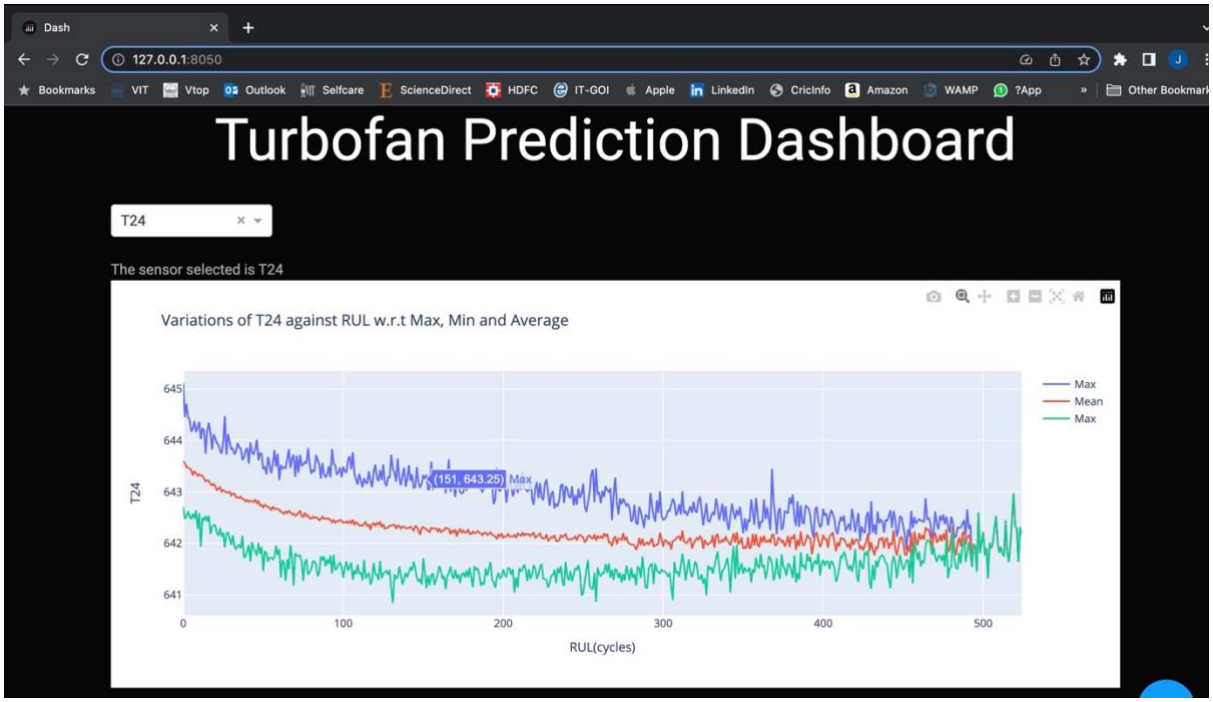

Figure 5-6 Web application using Dash

6. CONCLUSION AND FUTURE WORK

In this paper, a novel TCN- LSTM algorithm was proposed for predicting the remaining useful life in turbofan engines. Further, the model was evaluated against two other algorithms, CNN-LSTM and TCN. The proposed method's performance was compared with existing work in literature. To achieve interpretability of the black-box model, the SHAP algorithm was combined with proposed method to provide visualizations. This could help users gain a deeper understanding of the results, thereby, facilitating the decision-making process.

Future directions are in optimizing the proposed model, extending the implementation to other sub-datasets and exploring other xAI methods such as Local Interpretable Model Explanations (LIME) and GAM (Generalized Additive Models).

7. REFERENCES

[1] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, "Machinery health prognostics: A systematic review from data acquisition to RUL prediction," *Mech Syst Signal Process*, vol. 104, pp. 799–834, May 2018, doi: 10.1016/J.YMSSP.2017.11.016.

[2] W. Luo, T. Hu, Y. Ye, C. Zhang, and Y. Wei, "A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin," *Robot Comput Integr Manuf*, vol. 65, p. 101974, Jan. 2020, doi: 10.1016/j.rcim.2020.101974.

[3] J. Ortiz and R. Carrasco, *Model-based fault detection and diagnosis in ALMA subsystems*. 2016. doi: 10.1117/12.2233204.

[4] M. Baptista, S. Sankararaman, Ivo. P. de Medeiros, C. Nascimento, H. Prendinger, and E. M. P. Henriques, "Forecasting fault events for predictive maintenance using data-driven techniques and ARMA modeling," *Comput Ind Eng*, vol. 115, pp. 41–53, 2018, doi: https://doi.org/10.1016/j.cie.2017.10.033.

[5] L. Ren, L. Zhao, S. Hong, S. Zhao, H. Wang, and L. Zhang, "Remaining Useful Life Prediction for Lithium-Ion Battery: A Deep Learning Approach," *IEEE Access*, vol. 6, pp. 50587–50598, Jul. 2018, doi: 10.1109/ACCESS.2018.2858856.

[6] M. Gashi, B. Mutlu, and S. Thalmann, "Impact of Interdependencies: Multi-Component System Perspective toward Predictive Maintenance Based on Machine Learning and XAI," *Applied Sciences (Switzerland)*, vol. 13, no. 5, Mar. 2023, doi: 10.3390/app13053088.

[7] S. K. Jagatheesaperumal, M. Rahouti, K. Ahmad, A. I. Al-Fuqaha, and M. Guizani, "The Duo of Artificial Intelligence and Big Data for Industry 4.0: Review of Applications, Techniques, Challenges, and Future Research Directions," *CoRR*, vol. abs/2104.02425, 2021, [Online]. Available: https://arxiv.org/abs/2104.02425

[8] Y. Afridi, K. Ahmad, and L. Hassan, "Artificial intelligence based prognostic maintenance of renewable energy systems: A review of techniques, challenges, and future research directions," *Int J Energy Res*, vol. 46, Jul. 2021, doi: 10.1002/er.7100.

[9] T. Zonta, C. A. da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, "Predictive maintenance in the Industry 4.0: A systematic literature review,"

Comput Ind Eng, vol. 150, p. 106889, 2020, doi: https://doi.org/10.1016/j.cie.2020.106889.

[10] T. Hafeez, L. Xu, and G. McArdle, "Edge intelligence for data handling and predictive maintenance in IIoT," *IEEE Access*, vol. 9, pp. 49355–49371, 2021, doi: 10.1109/ACCESS.2021.3069137.

[11] R. Q. Huang, L. F. Xi, C. R. Liu, and J. Lee, "Prognostics for Ball Bearing Based on Neural Networks and Morlet Wavelet," *Materials Science Forum*, vol. 505–507, pp. 1153–1158, Jan. 2006, doi: 10.4028/www.scientific.net/msf.505-507.1153.

[12] J. Yan, Y. Meng, L. Lu, and L. Li, "Industrial Big Data in an Industry 4.0 Environment: Challenges, Schemes and Applications for Predictive Maintenance," *IEEE Access*, vol. PP, p. 1, Oct. 2017, doi: 10.1109/ACCESS.2017.2765544.

[13] R. van Dinter, B. Tekinerdogan, and C. Catal, "Predictive maintenance using digital twins: A systematic literature review," *Information and Software Technology*, vol. 151. Elsevier B.V., Nov. 01, 2022. doi: 10.1016/j.infsof.2022.107008.

[14] H. Liao, W. Zhao, and A. Express Huairui Guo, "Predicting Remaining Useful Life of an Individual Unit Using Proportional Hazards Model and Logistic Regression Model," 2006.

[15] Y. Zhou and M. Huang, "Lithium-ion batteries remaining useful life prediction based on a mixture of empirical mode decomposition and ARIMA model," *Microelectronics Reliability*, vol. 65, pp. 265–273, Oct. 2016, doi: 10.1016/j.microrel.2016.07.151.

[16] V. Tung, A. Chit Chiow, V. Tung Tran, B.-S. Yang, and A. Chit Chiow Tan, "Multi-step ahead direct prediction for machine condition prognosis using regression trees and neuro-fuzzy systems", [Online]. Available: http://eprints.hud.ac.uk/id/eprint/16577/

[17] B. Li, B. Tang, L. Deng, and M. Zhao, "Self-Attention ConvLSTM and Its Application in RUL Prediction of Rolling Bearings," *IEEE Trans Instrum Meas*, vol. 70, 2021, doi: 10.1109/TIM.2021.3086906.

[18] L. Ren, J. Dong, X. Wang, Z. Meng, L. Zhao, and M. J. Deen, "A Data-Driven Auto-CNN-LSTM Prediction Model for Lithium-Ion Battery Remaining Useful Life," *IEEE Trans Industr Inform*, vol. 17, no. 5, pp. 3478–3487, May 2021, doi: 10.1109/TII.2020.3008223.

[19] X. Li, Y. Xu, N. Li, B. Yang, and Y. Lei, "Remaining Useful Life Prediction With Partial Sensor Malfunctions Using Deep Adversarial Networks," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 1, pp. 121–134, 2023, doi: 10.1109/JAS.2022.105935.

[20] B. Zraibi, C. Okar, H. Chaoui, and M. Mansouri, “Remaining Useful Life Assessment for Lithium-Ion Batteries Using CNN-LSTM-DNN Hybrid Method,” *IEEE Trans Veh Technol*, vol. 70, no. 5, pp. 4252–4261, May 2021, doi: 10.1109/TVT.2021.3071622.

[21] L. Nie, S. Xu, and L. Zhang, “Multi-Head Attention Network with Adaptive Feature Selection for RUL Predictions of Gradually Degrading Equipment,” *Actuators*, vol. 12, no. 4, Apr. 2023, doi: 10.3390/act12040158.

[22] A. Holzinger, R. Goebel, R. Fong, T. Moon, · Klaus-Robert Müller, and W. Samek, “xxAI-Beyond Explainable AI,” 2020. [Online]. Available: https://link.springer.com/bookseries/1244

[23] W. Saeed and C. Omlin, “Explainable AI (XAI): A systematic meta-survey of current challenges and future opportunities,” *Knowl Based Syst*, vol. 263, Mar. 2023, doi: 10.1016/j.knosys.2023.110273.

[24] R. Confalonieri, T. Weyde, T. R. Besold, and F. Moscoso del Prado Martín, “Using ontologies to enhance human understandability of global post-hoc explanations of black-box models,” *Artif Intell*, vol. 296, p. 103471, 2021, doi: https://doi.org/10.1016/j.artint.2021.103471.

[25] P. Kang, J. Li, S. Jiang, and P. B. Shull, “Reduce System Redundancy and Optimize Sensor Disposition for EMG-IMU Multimodal Fusion Human-Machine Interfaces With XAI,” *IEEE Trans Instrum Meas*, vol. 72, 2023, doi: 10.1109/TIM.2022.3232159.

[26] Y. Muto *et al.*, “Noise robust automatic charge state recognition in quantum dots by machine learning and pre-processing, and visual explanations of the model with Grad-CAM,” Oct. 2022, [Online]. Available: http://arxiv.org/abs/2210.15070

[27] A. Adadi and M. Berrada, “Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI),” *IEEE Access*, vol. 6, pp. 52138–52160, 2018, doi: 10.1109/ACCESS.2018.2870052.

[28] C. W. Hong, C. Lee, K. Lee, M.-S. Ko, and K. Hur, “Explainable Artificial Intelligence for the Remaining Useful Life Prognosis of the Turbofan Engines,” in *2020 3rd IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, 2020, pp. 144–147. doi: 10.1109/ICKII50300.2020.9318912.

[29] S. Sivamohan and S. S. Sridhar, “An optimized model for network intrusion detection systems in industry 4.0 using XAI based Bi-LSTM framework,” *Neural Comput Appl*, 2023, doi: 10.1007/s00521-023-08319-0.

[30] L. Xiang, J. Liu, X. Yang, A. Hu, and H. Su, “Ultra-short term wind power prediction applying a novel model named SATCN-LSTM,” *Energy Convers Manag*, vol. 252, p. 115036, 2022, doi: https://doi.org/10.1016/j.enconman.2021.115036.

[31] C. Y. Hsu, Y. W. Lu, and J. H. Yan, “Temporal Convolution-Based Long-Short Term Memory Network With Attention Mechanism for Remaining Useful Life Prediction,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 35, no. 2, pp. 220–228, May 2022, doi: 10.1109/TSM.2022.3164578.

[32] “Damage Propagation Modeling”.

[33] “mauooRAPSY 313.” [Online]. Available: https://epubs.siam.org/terms-privacy

[34] J. G. Everett and S. H. Farghal, “Data Representation for Predicting Performance with Learning Curves,” *J Constr Eng Manag*, vol. 123, no. 1, pp. 46–52, 1997, doi: 10.1061/(ASCE)0733-9364(1997)123:1(46).

[35] Y. Zhu, Z. Liu, Z. Luo, C. Du, and H. Wang, “Aircraft engine remaining life prediction method with deep learning,” Institute of Electrical and Electronics Engineers (IEEE), Nov. 2022, pp. 1–6. doi: 10.1109/aicit55386.2022.9930216.

[36] S. Bai, J. Z. Kolter, and V. Koltun, “An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling,” Mar. 2018, [Online]. Available: http://arxiv.org/abs/1803.01271

[37] Julius Richter, “Temporal convolutional networks for sequence modeling,” *Dida Datenschmiede GmbH*, Apr. 05, 2023.

[38] J. Wang, X. Li, J. Li, Q. Sun, and H. Wang, “NGCU: A New RNN Model for Time-Series Data Prediction,” *Big Data Research*, vol. 27, p. 100296, 2022, doi: https://doi.org/10.1016/j.bdr.2021.100296.

[39] H. Tian and J. Chen, “Deep Learning with Spatial Attention-Based CONV-LSTM for SOC Estimation of Lithium-Ion Batteries,” *Processes*, vol. 10, no. 11, Nov. 2022, doi: 10.3390/pr10112185.

[40] D. Yang, E. Yurtsever, V. Renganathan, and K. A. Redmill, “Sensors-21-04608.Pdf,” 2021.

[41] D. Solís-Martín, J. Galán-Páez, and J. Borrego-Díaz, “On the Soundness of XAI in Prognostics and Health Management (PHM),” *Information*, vol. 14, no. 5, p. 256, Apr. 2023, doi: 10.3390/info14050256.

[42] A. Borovykh, S. Bohte, and C. W. Oosterlee, “Conditional Time Series Forecasting with Convolutional Neural Networks,” Mar. 2017, [Online]. Available: http://arxiv.org/abs/1703.04691

[43] K. Pugalenthi, H. Park, S. Hussain, and N. Raghavan, “Remaining Useful Life Prediction of Lithium-Ion Batteries Using Neural Networks with Adaptive Bayesian Learning,” *Sensors*, vol. 22, no. 10, May 2022, doi: 10.3390/s22103803.

[44] X. Zhang, Y. Dong, L. Wen, F. Lu, and W. Li, “Remaining Useful Life Estimation Based on a New Convolutional and Recurrent Neural Network.”

[45] S. Smyl, “A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting,” *Int J Forecast*, vol. 36, no. 1, pp. 75–85, Jan. 2020, doi: 10.1016/j.ijforecast.2019.03.017.

[46] I. Remadna, L. S. Terrissa, S. Ayad, and N. Zerhouni, “Rul estimation enhancement using hybrid deep learning methods,” *Int J Progn Health Manag*, vol. 12, no. 1, 2021, doi: 10.36001/IJPHM.2021.V12I1.2378.