Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences

LSTM Neural Networks for Remaining Useful Life Estimation of Turbofan Engines

A thesis presented by

Luisa Fernanda Montoya Herrera

Submitted to the
School of Engineering and Sciences
in partial fulfillment of the requirements for the degree of

Master of Science In

Manufacturing Systems

Monterrey, Nuevo León, December 4th 2020

Declaration of Authorship

I, Luisa Fernanda Montoya Herrera, declare that this dissertation titled *LSTM Neural Networks for Remaining Useful Life Estimation of Turbofan Engines* and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- • If this dissertation has previously been submitted at this University or other institution, it has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for quotations, this dissertation is entirely my work.
- I have acknowledged all the primary sources of help.

Luisa Fernanda Montoya Herrera
Monterrey, Nuevo León
December 4th 2020

@2020 by Luisa Fernanda Montoya Herrera
All rights reserved

Dedication

To

I thank my parents for their unconditional support in each of the things that I have proposed in life. This achievement is both theirs and mine. In particular, I am grateful to my mother for teaching me, through her example, that striving always brings a good reward. I thank my friends and all the people who could see this process during my stay at Tecnológico de Monterrey; this experience was much more rewarding thanks to them.

Acknowledgments

I want to thank my thesis advisor, Dr. Rubén Morales Menéndez, and co-advisors Adriana Vargas Martínez and Jorge de Jesús Lozoya Santos, for their essential suggestions, guidance, and patience. Their support has been critical to achieving the goals of this work.

I thank Tecnológico de Monterrey for giving me a scholarship during my studies and CONACyT for financial support.

Thank you.

Thank..

LSTM Neural Networks for Remaining Useful Life Estimation of Turbofan Engines

By

Luisa Fernanda Montoya Herrera

Condition-based Maintenance is a maintenance strategy that monitors the actual condition of a system to make predictive decisions whit respect to it. This type of maintenance includes detection, diagnosis, and prediction of system failures. It has become increasingly important because it generates the least losses, reducing total maintenance costs in a business by 5In general, the *Remaining Useful Life* estimation allows making failure predictions. The complexity of failure prediction in mechanical systems has led to a significant amount of literature. Different solutions have been proposed; however, this still a real problem.*Remaining Useful Life* estimation can be done from other approaches, for example, using physical models, knowledge-based models, or data-driven models. Extracting relevant features from raw data using physical or knowledge-based techniques alone, in most cases, is not enough due to the complexity of the characteristics present in the data. Literature shows that data-driven approaches are the most used for prediction.

In recent years, *Deep Learning* models for different applications have been used, including failure detection, diagnosis, and prediction. The Deep Learning model's advantage is that an in-depth knowledge of the system is not required, and due to its robustness, complex learning results are satisfactory. For Remaining Useful Life estimation, *Long Short Term Memory* neural networks are a viable option since they can adequately handle the time series needed for failure predictions using *Remaining Useful Life* estimation.

The three main stages for developing this method based on *Long Short Term Memory* neural networks were data pre-processing, model training, and model performance evaluation. The methodology uses two datasets of turbofan engines with different operational conditions and faults for its validation. The process evaluates signals obtained from sensors located along with a turbofan engine simulated through a Simulink-based program.

This methodology presents a reasonably acceptable performance in terms of Root Mean Squared Error of 2.85 with a standard deviation of 0.39. It means that on average for the engines, the failure prediction will have an error of 3 cycles; and a Score function of 7.26 with a standard deviation of 1.76, which is an asymmetric algorithm where late predictions are more penalized than early predictions, increasing exponentially with the error. The proposed methodology has the advantage of being more straightforward than other methods found in the literature. Besides, the obtained values of the predictions are conservative.

Contents

1 Introduction **4**
- 1.1 Motivation . . . 5
- 1.2 Problem Description . . . 5
- 1.3 Research Question . . . 6
- 1.4 Solution Overview . . . 7
- 1.5 Main Contribution . . . 7
- 1.6 Organizations . . . 7

2 State of the Art **9**
- 2.1 Introduction . . . 9
- 2.2 Data-driven based approaches . . . 10
- 2.3 *LSTM* neural networks . . . 12
- 2.4 Comparison . . . 14

3 Simulation System **17**
- 3.1 Turbofan Engines . . . 17
- 3.2 Turbofan engine degradation . . . 18
- 3.3 Simulation conditions . . . 20
  - 3.3.1 Damage propagation modeling in the simulation system . . . 21
  - 3.3.2 Simulation databases . . . 22
- 3.4 Remaining Useful Life Estimation . . . 23
  - 3.4.1 RUL labeling . . . 24

4 Proposal **26**
- 4.1 Methodology . . . 26
  - 4.1.1 Data pre-processing . . . 26
  - 4.1.2 Training/testing relation . . . 27
  - 4.1.3 Key Performance Indicators . . . 29
  - 4.1.4 Flow diagram of the proposed methodology . . . 31

4.2 LSTM Neural Network design and structure 31

5 **Results** **37**
5.1 Data pre-processing 37
5.1.1 Data normalization 38
5.2 LSTM neural network design 40
5.2.1 Neural Network Train/Test 43
5.3 *LSTM* Neural network performance 45
5.4 Discussion 47
5.5 Comparison 49

6 **Conclusions** **55**
6.1 Contributions 55
6.2 Future work 56

Bibliography **58**

A Acronyms and Variables Descriptions **66**
A.1 Acronyms 67
A.2 Variable description 68

B Results of real vs predicted *RUL* **69**
B.1 Engine 3 69
B.2 Engine 4 70
B.3 Engine 5 70
B.4 Engine 61 71
B.5 Engine 190 71

Curriculim Vitae **73**

List of Figures

3.1 Scheme of a Turbofan and its components . . . 18

3.2 Subroutines of the engine simulation . . . 20

3.3 Piece-wise linear degradation model . . . 25

4.1 (a) represents a sensor with discrete constant values, (b) represents a sensor with continuous and inconsistent values, and (c) represents a sensor with continuous and monotonic values. . . . 27

4.2 Correlation matrix of the sensors in the same cluster. . . . 29

4.3 (a) represents the split of the database and, (b) represents the cross-validation procedure's split. . . . 29

4.4 Scheme of *RUL* estimation methodology using LSTM . . . 32

4.5 Representation of a Long-Short Term Memory Neural Network . . . 33

4.6 Representation of the input shape of the LSTM Neural Network . . . 34

4.7 Basic Structure of a model using *LSTM* for *RUL* Estimation . . . 36

5.1 Operating Regimes resulting from the combination of the operational settings . . . 38

5.2 It represents the behavior of all the input sensor for the *LSTM* model. . . . 39

5.3 Figure (a) represents the loss function in terms of MAE for 200 epochs, and Figure (b) represents the loss function in terms of MAE for 100 epochs. . . . 41

5.4 it is a box plot of the RMSE for the training set for 16, 32, and 64 batch size . . . 42

5.5 It is the box plot of the RMSE for the test set for 16, 32, and 64 batch size . . . 43

5.6 (a) it is the loss function for the PHM008' dataset, and (b) it is the loss function for the CMAPS dataset. . . . 44

5.7 It represents Real Rul vs. Predicted Rul in train and test set for PHM08' and CMAPS databases. . . . 45

5.8 (a) it is the error distribution for PHM008' dataset and (b) it is the error distribution for CMAPS dataset. . . . 46

5.9 ((a) it is the box plot in the train set for the RMSE, and (b) it is the box plot in the test set for for the RMSE. . . . 49

5.10 (a) it is the box plot in the test set for the Score function, and (b) it is the box plot in the test set for the score function. 50

5.11 (a) it is Real RUL vs Predicted RUL for a threshold of 50 in motor 1, (b) it is Real RUL vs Predicted RUL for a threshold of 60 in motor 1 and (c) it is Real RUL vs Predicted RUL for a threshold of 70 in motor 1. 51

B.1 Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 3 for PHM08’ dataset. 69

B.2 Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 4 for PHM08’ dataset. 70

B.3 Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 5 for PHM08’ dataset. 70

B.4 Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 61 for PHM08’ dataset. 71

B.5 Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 190 for PHM08’ dataset. 71

List of Tables

2.1 Comparison between different investigations on RUL estimation using Deep Learning techniques . . . 15

3.1 PHM08' dataset characteristic . . . 23
3.2 Turbofan Engine dataset characteristic . . . 23

4.1 CMAPS outputs to measure system response . . . 28
4.2 LSTM Neural Network Structure . . . 35

5.1 RMSE values and training time for different learning rate values . . . 40
5.2 RMSE values for 100 and 200 epochs . . . 40
5.3 RMSE values and Std for 16, 32 y 64 batch size . . . 42
5.4 Neurons configurations . . . 42
5.5 Established Hyper-parameters for *LSTM* model . . . 44
5.6 RMSE and standard deviation values for the PHM08', and the CMAPS dataset . . 46
5.7 RMSE and Score function values for threshold 50, 60, and 70 for PHM08' and CMAPS dataset . . . 48
5.8 Comparison of the proposed methodology with other authors . . . 52

A.1 Acronyms Definitions . . . 67
A.2 Variables Description . . . 68

Chapter 1

Introduction

The demand for faster and more efficient manufacturing processes has increased in recent years, with the rise of the Industry 4.0 revolution. As sensors and indicators fully monitor most manufacturing processes, a considerable amount of data has become available to analyze and use as support for decision making. Conventional maintenance strategies are insufficient for manufacturers to stay competitive against companies using emerging technologies, such as *Smart Manufacturing (SM)*, where manufacturing systems are real-time monitored using *Artificial Intelligence (AI)* approaches. Statistics show that most companies using SM technologies have increased efficiency and customer satisfaction [1]. Researches developed the concept of *Condition-Based Maintenance (CBM)*, which focuses on the fault detection, diagnosis, and prognosis of machines and systems. CBM is also known as *Prognosis and Health Management (PHM)*.

The concept of *CBM* was first introduced by the Rio Grande Railway Company in the late 1940s [2]. *CBM* is a maintenance strategy that seeks for the optimization of the repairing times of a machine. To achieve optimization is necessary to monitor the equipment continuously. Thus, maintenance actions are only taken when there is evidence of abnormal behaviors. One of the most notable advantages of this type of maintenance is that it can significantly reduce unnecessary stops. To achieve a good *CBM* program, it is necessary to acquire the data, process it, and make the required decisions based on the obtained information

As [3] said, *CBM* is conducted based on the observation that systems usually suffer a degradation process before failure. The maintenance staff could observe the degradation process by indicators such as temperature, pressure, voltage, and vibration to take advantage of the information obtained by monitoring these sensors. The correct instrumentation of the systems is crucial for the development of this type of strategy.

1.1 Motivation

Predictive analytic, also called prognosis, is one of the main topics nowadays, forecasting a 20% to 50% market penetration in about 2 to 5 years, according to Gartner Inc. [4]. Since fault prediction is less developed than fault diagnosis, early development of fault prognosis was once regarded as a complement to fault diagnosis [5]. Experts can see the prediction of failure in mechanical systems from two different points. First, as the forecast of the remaining time a system has before it fails, *Remaining Useful Life Estimation (RUL)*; second, as the possibility that a machine or system works without failure taking into account the current state of the engine and the historical behavior it has.

Creating a RUL monitoring system will improve downtime, which generates reductions in productivity and, therefore, economic losses. According to AltexSoft, a company specialized in technology applied to industry, using a CBM program may reduce the time invested by making maintenance plans between 20%-50%, increase equipment availability and uptime by 10%-20% and reduce total maintenance costs in a business by 5%-10Ẇhen the aim is to achieve maximum reliability, an appropriate CBM system with monitoring capabilities must be adopted, gathering and combining all kinds of useful sources of information simultaneously and providing the prognostics needed to assure the assets' correct operation [6]. Besides, it will help maintenance programs migrate *CBM* using *Deep Learning (DL)* tools that will facilitate decision-making regarding the machines' maintenance times.

1.2 Problem Description

There are different techniques and methodologies to implement *CBM* today in companies. These methods can be distinguished from those carried out offline, such as all visual inspection methods, vibration analysis, ultrasound, thermography, and lubricants, among others, and those done online, generally using physical, knowledge, and data approaches. Direct or offline measurement methods have the advantage of being precise; however, they are vulnerable to various disturbances found in the field and can interrupt the normal operations of the analyzed system, which leads to system stoppages that cause economic losses.

On the contrary, in indirect methods, The maintenance personnel can obtain through the sensors the measurements of the system's behavior directly related to its useful life, which represents a significant advantage since they can be easily installed and allow constant monitoring of the conditions of the system. It indicates a need to implement intelligent systems that can help in decision making regarding the reading and interpretation of said data; These systems serve as a tool for proper *CBM*. Low-cost monitoring systems to collect data from machines for detection, diagnosis, and failures are prevalent. To predict failures, data-driven approaches such as similarity modes or Remaining Useful Life estimation are good options. The *RUL* of an asset or system is defined as the length from the current time to the end of the useful life [7], which is considered the core and always a significant *PHM* challenge. Then, *RUL* calculation depends on the data available, the lifetime data for similar machines, the run-to-failure histories of other devices, and a known threshold value of a condition indicator that detects the failure.

Turbofans or jet engines are the most critical system on an aircraft. The maintenance carried out to the airplane engines is planned by hours or phases, from an oil level check to an inspection for turbine damage; nothing should be left unverified. By correctly maintaining turbofans, technical life can be prolonged thanks to less mechanical stress and reduced vibrations. Since this work requires a lot of time and effort from all maintenance personnel, it is necessary to do this type of maintenance only when it is needed. In this sense, the estimation of the *RUL* plays an essential role in turbofan engine maintenance. Being able to predict when an engine is going to fail or at what time it is more appropriate to repair so that it continues to operate correctly will avoid unnecessary stoppages of the machines, which is directly related to maintenance costs.

It is possible to use the instrumentation of mechanical systems to predict the most suitable time to perform maintenance, that is, estimate *RUL* using the data from the sensors and analyzing its variation over time to establish when it is appropriate to schedule engine maintenance.

1.3 Research Question

The complexity of estimating the useful life of systems has led to a significant amount of research related to this problem. Several techniques have been proposed; however, predicting the system's damage remains a real and complex problem to solve due to its constant change in the process variables.

Literature shows that data-based approaches have gained much ground for the classification, diagnosis, and prediction of failures in recent years. Estimating systems' lifetime is not always an easy task because some critical characteristics are related to time series and cannot be directly seen. In this case, *DL* is a tool with a powerful capacity to learn features and predict the relationship between data and the life of the systems by estimating the RUL of a fleet of aircraft engines to establish how much time the system has left before it fails.

1.4 Solution Overview

Develop a methodology for RUL estimation, using *Long-Short Term Memory (LSTM)* neural networks that avoid the vanishing problem and allow a better selection of the information stored within the network, therefore presenting better results when working with time series. The methodology consists of 4 phases: 1. data acquisition, 2. preprocessing of signals to choose the parameters and features to be entered into the model, 3. training and validation of the *LSTM* neural network using cross-validation and, finally 4. *RUL* estimation. As *DL* approaches can handle the signal's noise, the preprocessing phase will only normalize the data and select the system's representative sensor. The *LSTM* neural network trains the model until reaching a good performance so that the *RUL* results are the most accurate according to the system's behavior.

This solution is general for any system with time dependency; however, it is necessary to modify different systems. In this case, a fleet of aircraft will serve to validate the methodology, observe the system's degradation, and estimate the *RUL* of the system.

1.5 Main Contribution

This thesis's main contribution is developing a simple methodology using a *DL* approach for the estimation of the remaining useful life of turbines of a fleet of aircraft that have similar operating and capacity characteristics. The *DL* technique used is an *LSTM* neural network, which is selected based on its features of being suitable for managing time series. This methodology can be used to monitor the system and make decisions regarding the maintenance of the turbines. Applying this model to the industry represents an advantage with the incentive that this methodology uses a neural network with reduced computing time. It does not take up much computational cost for its development.

The entire coding process of the methodology is developed using python 3.7 in the Google collaborative work environment, using the TensorFlow, Keras, and sklearn libraries for the preprocessing of the data and the training of the model. This code is adaptable to the work environment of jupyter or any other that is required. Multiple tests with various configurations are detailed to determine a fair value for each of the parameters and thus obtain an adequate and robust model.

1.6 Organizations

This research work is organized as follows:

- Chapter 2 presents the state of the art of different approaches of *PHM* for *RUL* estimation. Also, the areas of opportunity to develop this research are identified.
- Chapter 3 includes the theoretical background of turbofan engine degradation, simulation conditions and, description of the datasets used and a *RUL* estimation overview.
- Chapter 4 present the proposed methodology.
- Chapter 5 presents the results of the evaluation of the methodology and a comparison with similar research projects.
- Chapter 6 presents the conclusions, the contributions and, future work of this research.
- Bibliography
- The appendix which is composed of the acronyms and variables definition, additional results of the real vs. predicted *RUL*, and the code with which all the research work was developed

Chapter 2

State of the Art

This chapter covers the most relevant work related to the data-driven approaches for machine health monitoring, especially for *RUL* estimation of mechanical elements as turbofan engines using *DL*.

2.1 Introduction

Prognosis is the capability to use available observations to predict upcoming states of a machine or forecast the fault before it occurs [8]; The prediction or prognosis of failures in mechanical systems refers to the life that a machine or system has before it fails. Using the current condition and historical data is necessary to predict how much time remains before a failure occurs, also known as *RUL*. Monitoring the fault propagation process using a forecast or trend model for certain system variables is the most common. Researches were developing many techniques and research works related to failure analysis and RUL estimation. In recent years, with the rise of *AI,* a tendency towards the applications of *DL* systems can be observed. These algorithms allow developing robust monitoring systems focused on *PHM*. A wide variety of research has been conducted regarding *DL* to detect and diagnose failures in systems. However, studies have been more limited to the lack of available run to failure data about failure prognosis.

For *CBM*, companies are using three different approaches [9] states that they are all framed within the same three categories: Model-based, Knowledge-based, and Data-driven based methods. Model-based approaches require an accurate mathematical model to be developed and use residuals as features, where residuals are the outcomes of consistency checks between the sensed measurements of a real system and the outputs of a mathematical model [10]. Knowledge-based approaches use symbolic representations to solve problems, which can be very difficult when dealing with complex systems. Data-driven based techniques are used when system models are not available, unknown, or are too complex to model, but instead, monitoring systems are available

[9]. Most common data-driven based approaches are based on AI, which refers to techniques that fit into Machine Learning (ML) with examples of use as *Support Vector Machine (SVM)*, Fuzzy Logic, and *DL* [11; 12; 13; 14]. However, there are also many other applications such as Wavelets and Fourier transform.

2.2 Data-driven based approaches

DL is a subset of *ML* that allows computational models composed of multiple processing layers to learn representations of data with various abstraction levels using the back-propagation algorithm [15]. In recent years, *DL* has been of great interest for researchers, especially in object recognition, image segmentation, speech recognition, and machine translation. As a ML branch, *DL* attempts to model hierarchical representations behind data and classify or predict patterns via stacking multiple layers of information processing modules in hierarchical architectures [16]. Although *DL* is not something new, its recent popularity is mostly due to the increase in computers' computing power and the increase in the data available from systems or machines. There are different types of models within the *DL*, such as *Auto Encoders (AE)*, *Deep Belief Network (DBN)*, *Deep Boltzmann Machines (DBM)*, *Convolutional Neural Networks (CNN)* and, *Recurrent Neural Networks (RNN)*, adapting the model's characteristics to different real problems. Currently, *DL* can be used in manufacturing systems to monitor machines' status; this is known as *Machine Health Monitoring Systems (MHMS)*.

Authors in [17] formulated a generic framework of structural health prognostics composed of a health index, offline learning based on sparse Bayes learning techniques, a generic online prediction scheme using the similarity-based interpolation and uncertainty propagation map for the prognostic uncertainty management. The methodology applies to different engineered systems, and with two cases of study, its possible to demonstrate its effectiveness. In [18], the authors used a semi-supervised *RBM* model with the aim of pre-train the model and saw the effect on the *RUL* estimations results. Additionally, a *Genetic Algorithm (GA)* approach is applied to tune the hyper-parameters in the training process. The results obtained in this research show that the proposed method outperforms other models like *CNN* and *LSTM* models. In In [19], the authors propose a *Deep separable convolutional network (DSCN)* for *RUL* estimation of turbofan engines. The proposed model uses the sensors' data as inputs, introduce separable convolutions to increase the sensitivity of the *DSCN*. A unit is constructed behind the detachable convolutional layer to perform adaptive feature response re-calibrations. The *RUL* is finally estimated with a fully-connected layer. The result shows that the proposed method outperforms the other methods, and comparing it with a *LSTM*, emphCNN, and *Bidirectional Long-Short Term Memory (Bi-LSTM)* is better.

In [20] the authors used a *Stacked Denoising Autoencoder (SDA)* for health state identification in three main steps, dividing it into training and testing groups for the *SDA* model. A deep hierarchical structure is set with a transmitting rule of greedy training. Finally, a sparsity representation was applied to obtain high-order characteristics with better robustness in iteration learning. Other authors have taken advantage of CNN networks' remarkable capacity to process images and have used them for the diagnosis or classification of faults. Authors in [21] used *Deep Convolutional Networks (DCNN)* through the combination of wavelet packet transform and space reconstruction to rebuild a 2-D wavelet packet energy image of the frequency subspaces. Also, in [22], the author used 1-CNN for fault detection, which does not need a separated feature extraction algorithm because the input is the raw data resulting in more efficient systems in terms of both speed and hardware. Different approaches have been used to study the detection and diagnosis of failures in systems using, *AE* and *CNN*

The authors in [23], used an *Adaptive Neuro-Fuzzy Inference System (ANFIS)* model for machine fault prognosis of a bearings dataset provides by the University of Cincinnati, the model used to consist of an off-line phase where the model is trained. In an on-line phase where predictions are made, the proposed model outperforms the traditional *ANFIS* models for different prediction horizons. Authors in [24] proposed a combination of Data-driven and knowledge-based prognostics equipment, and a Weibull proportional hazard model is used to establish the relationship between failure rate and state parameters. The least-squares nonlinear regression is used to obtain the failure rate trend. The results obtained show that the model successfully predicts with a standard deviation of 0.42. The authors in [25] proposed a data-driven prognostic approach combining Principal Component Analysis (PCA) with an exponential degradation model using an ideal health indicator to predict the *RUL* of a rotating shaft. Feature selection was made by a feature importance ranking, reducing the model's number of inputs for better performance. In[26] the authors propose a correlation method to reduce the complexity of the model to estimate RUL in turbofan engines. This approach shows better efficiency and high-performance thanks to excluding the low correlated inputs of the dataset.

In the same way, the authors in [27] proposed an ensemble data-driven prognostic approach which combines an accuracy-based weighting, a diversity-based weighting, and an optimization-based weighting. Then they used a k-fold *cross validation (CV)* to estimate the prediction error required. The results obtained in this research for three different cases suggest that this approach is better to estimate *RUL* compared with any other algorithm. Authors in [28] used a combination of a stochastic Wiener process with Principal Component analysis to estimate RUL. This research shows the relevance of probabilistic approaches using a stochastic process instead of probabilistic approaches with lifetime models and similarity-based approaches with the same health indicator. In [29], the authors present a hybrid method of a mixture of Gaussian hidden Markov model, and fixed-size least squares support vector regression for fault prognostic. First, the models are trained; then, the system recognizes the unknown samples. Finally, the forward variables are calculated and serve as inputs for the regression to compute the RUL of the unknown sample.

In recent years, the use of *DL* techniques has increased considerably. However, there are other data-based approaches that, as seen above, can be used. For example, to extract the data characteristics, the authors use *Principal Component Analysis*, *Independent Component Analysis*, *ANOVA*, *Self-organizing Maps*, and *Fast Fourier Transform*, among others. For classification and diagnosis, the authors use *SVM*, *K-Nearest Neighbors*, and *Decision Trees*; However, *Artificial Neural Networks* have been preferred by many engineers and widely applied to fault diagnostics of various engineering systems. Different probabilistic and regression models are used in prognoses, such as *Proportional Hazard Model*, *Markovian Process-Based Models*, *Wiener Process*, *Threshold Regression Model*, and *Bayesian Models*. However, for *CBM*, *DL* models that require less knowledge of the systems and are much easier to use are preferred [30].

2.3 *LSTM* neural networks

LSTM has been widely used in speech recognition, machine translation, sentiment analysis, video activity recognition, music generation, and handwriting generation. However, these networks' use for predicting mechanical systems failures has not been as explored as to its use in other areas. [31] used a *LSTM* to predict the *RUL* using multiple sensor time-series signals and apply this method to the CMAPS turbofan engine data set from NASA. After completing the model, 10 engines are used to validate the trained *LSTM* and compare against various models such as *Support Vector Machine (SVM)*, *SVR*, *CNN*, *Multi Layer Perceptron (MLP)*. Results showed that the proposed model gave the best performance in terms of the Score, R-score, and Error range 655, 18.33, and [-47, 56], respectively. The authors [32] proposed a novel *Bi-LSTM* and *FNN* architecture on the same data set to predict the *RUL* of the components while taking into account the system's operational conditions using different sequence lengths and determining the impact of adding auxiliary input to try and increase accuracy. Results were evaluated using the mentioned prediction *Root Mean Squared Error (RMSE)* and Score obtaining 25.9 and 4882, respectively; the authors used a CNN model to compare this result.

In [33], authors use an *LSTM* neural network to estimate the RUL of a fleet of engines identifying the initial useful life with an Euclidean distance-based method to make the estimations more accurate; the authors observe that this method is better compared with *MLP*, *SVR* and *CNN*. Authors in [34] opted for a *Gated Recurrent Units (GRU)* approach, reporting an RMSE of 35.0 and taking 0.34 hours for training, showing that they obtained better results than using an *LSTM*. [35]

used used a *BLSTM* based *AE* scheme to predict *RUL* using the same *CMAPS* model to create six simulated data sets. The results using the already mentioned model for prediction score, accuracy, and RMSE were 1098, 47%, 19.5, respectively.

The authors use a Vanilla LSTM [36] to estimate the RUL of four sets of turbofan aircraft simulation from the NASA dataset that were normalized, comparing its results against a standard *RNN* and a *GRU-RNN*. The proposed model obtained the lowest *Mean Squared Error (MSE)* for the most complex fault modes using 32 nodes. Results showed 3485.1 and 578.1 *MSE* for a hybrid fault and a single operating mode, 2710.7 and 1205.4 *MSE* for a hybrid fault and multiple operational modes, on train and validation sets, respectively. Also, [37] used a *Bi-LSTM* neural network to predict *RUL* in a turbofan jet engine from the NASA dataset. The methodology was carried out in two stages, online and offline, where the sensor data is preprocessed. The signals feed the proposed model with the *RUL* labels via a Back-propagation algorithm. In the online phase, the auxiliary data and the sensor data are fed to the training model to obtain real RUL values based on real-time monitoring. Prognosis metrics are the prediction *Score* and *RMSE*, obtaining 25.1 and 4793 for the FD002 dataset and 26.6 and 4971 for the FD004 dataset for *RMSE* and prediction score. The proposed model outperformed models such as *CNN*, and a *Multi-objective Deep Belief Network Ensemble* proposed in [38].

In [39], the authors used an *LSTM* neural network to predict excess vibrations in aircraft engines, using a data set recorded via the Flight Data Recorder system. The authors evaluated three different architectures and results showed that even when one of the three architectures had the best performance, it was computationally more expensive and took more time series as input. Predictions were evaluated through *MSE* and *Mean Absolute Error (MAE)* for 5, 10, and seconds. [40] Proposed a *DL* method to estimate the *RUL* of aero-propulsion engines using an *LSTM* structure with two layers and 64 neurons per layer. The results of the proposed method, compared with other methods, is better; the *LSTM* neural network results in terms of *RMSE* and the scoring function value were 16.7372 and 3.88 × 102.

Results show that, compared with other structures, LSTM Neural Networks has better results of *MAPE*, *RMSE* and R^2. In 2008 there was a competition called PHM08', one of the winners were [41] using an *RNN* with 24 inputs has three layers of feed-forward connections and recurrent connections and also used an evolutionary algorithm for updating weights and bias, obtaining an error of 519.8. The other winner was [42], who presented a Similarity-based approach for *RUL*; first, a performance assessment is made. Then, an *RUL* estimation based on a *Health Index (HI)*, A total score of 5636.06 is achieved, which is the overall best in the competition. The authors in [43] proposed a Hybrid method combining an *LSTM* neural network and a *CNN*, which is validated with the turbofan engine dataset from NASA. The input layer is a two-dimensional matrix that includes operational settings and sensor measurements in its columns and snapshots of each time cycle in its rows. This layer's output serves as the input of an *LSTM* layer, and the output layer is a dense layer used for *RUL* estimation purposes. The results obtained with this hybrid method outperform the ones obtained with an *LSTM* neural network in FD001, FD002, and FD003 datasets.

In [44], the authors developed a methodology that includes a hybrid model for *RUL* estimation using a *CNN* and an *LSTM* neural network. The *CNN* network is used to extract characteristics and reduce the network's dimensionality. The *LSTM* network is used to analyze the time series, and finally, a fully connected layer is used for the network output. The results obtained of RMSE and Score are equal to 7.81 and 88.66, respectively. The authors in [45] proposed an *LSTM* neural network to diagnose and predict complicated operations, hybrid fault, and intense noise. The input layer is a 3D array with shape (sample number, time steps, feature). There is a hidden layer, and finally, two outputs layer, one for diagnosis and the other for *RUL* estimation. The results obtained with this model show the deep neural network can predict good results under complex operations modes and hybrid degradation.

In [46], the authors used a convolutional *Bi-LSTM* with *Multiple Time Windows (MTW) (MTW CNN-BLSTM)* for *RUL* estimation. In the training phase, multiple *CNN-BLSTM* base models with different time window sizes are trained, and then, in the test phase, test units are classified, and suitable base models are applied to predict the *RUL* of the system finally. The model is validated with the turbofan engine dataset from NASA; the results obtained show better results in *RMSE* and standard deviation than other models as *CNN*, *Bi-LSTM* *CNN-BiLSTM*. The authors in [47] proposed an LSTM based Encoder-Decoder scheme to obtain an unsupervised health index for *RUL* estimation using multi-sensor time-series data. The results obtained in terms of *RMSE* and *Score function* were 12.08 and 256, respectively.

2.4 Comparison

Table 2.1 shows a comparison between different *DL* methods found in the literature in recent years concerning *RMSE* and *Score function*, which are the most used key performance indicators to predict failures in turbofan engines.

Table 2.1: Comparison between different investigations on RUL estimation using Deep Learning techniques

References year	*DNN Architecture*	*Case Study*	*Results*	*Additional Information*
[41] 2008	*RNN*	C-MAPSS Dataset	*Score*=519.8	Winner of the PHM08' competition
[46] 2020	*CNN, BiLSTM*	C-MAPSS Dataset	*RMSE*=12.66 *Score*=304.29	Moving time window implementation
[48] 2018	*BD-LSTM*	C-MAPSS Dataset	*RMSE*=15.42	Better performance compared with SVR, MLP, LSTM
[49] 2019	*ResCNN*	C-MAPSS Dataset	*RMSE*=24.97 *Score*=3400.44	Better performance compared with MLP, SVM, DBN, LSTM, CNN
[50] 2018	*LightGBM*	C-MAPSS Dataset	*RMSE*=13.45 *RMSE*=250.4	Better performance compared with XGBoost, CNN, RF
[51] 2018	*TW, ELM*	C-MAPSS Dataset	*RMSE*=13.78 *Score*=267.31	Better performance compared with SVM, SVR, LSTM
[52] 2008	*Kalman Filtering, MLP, RBF*	PHM08 Dataset	*MSE*=984	Normalization is made based on 6 failures modes
[53] 2017	*GBT, CNN* *SEM*	C-MAPSS Dataset	*GBT RMSE*=31.13 *SEM RMSE*=26.76	Normalization is made based on 6 failures modes
[54] 2019	*RNN, LSTM, GRU*	C-MAPSS Dataset	*RMSE*=18.82 *Score*=699.9	Energy of coefficients
[44] 2020	*CNN-ULSTM*	C-MAPSS Dataset	*RMSE*=7.81 *Score*=88.66	Better performance compared with SVR, MLP, LSTM
[55] 2017	*LSTM*	PHM08 Dataset	*RMSE*=17.84	Better performance compared with Naive Bayesian regression model
[56] 2019	*RNN with statical recurrent unit*	C-MAPSS Dataset	*RMSE*=19.63 *Score*=3200	Better performance compared with ANN, SRNN, LSTM
[57] 2018	*MLPNN Infer,KF, and MLPNN Project*	PHM08 Dataset	Does not apply	HI estimation for RUL
[58] 2019	*Similarity based model, ANN*	PHM08 Dataset	*Score*=5530.12	*DPE*
[59] 2017	*RNN*	C-MAPSS Dataset	*MSE*=466	Does not apply
[60] 2016	*PCA, WED*	Turbofan engine Dataset	Does not apply	HI estimation for RUL
[61] 2019	*BLSTM*	C-Mapss Dataset	*RMSE*=26.61 *Score*= 4971	Better performance compared with DCNN, SKF, MODBNE
[62] 2019	*CNN-RNN*	C-MAPSS Dataset	*RMSE*=29.73 *Score*=7212.2	Better performance compared with MLP, SVR, CNN

Table 2.1: Comparisson between some machining researches using wavelet approach (Continued)

References	*Defects*	*Case Study*	*Technique*	*Additional Information*
[63] 2019	*LiRUL*	C-MAPSS Dataset	*RMSE*=20.72	Better performance compared with others LSTM architectures

When looking through the literature, a tendency can be seen related to prognosis and health monitor-systems. Data-driven methodologies are the most used in the works reviewed. Specifically, *DL* techniques have gained significant strength in recent years due to the increased processing capacity of the equipment to analyze the data available from the systems. There is also a noticeable evolution from models with simple networks such as *FNN* and their variants to the use of *AE*, *CNN*, or *LSTM* networks that represent greater complexity in their models but at the same time produce better results with much shorter processing times. New *DL* tools have emerged; *LSTM* neural networks have the advantage that they can handle the complexity of working with time series, which represents an advantage for RUL estimation of systems; thus, there is an area of opportunity to apply these tools. It is observed from the literature that to date, there are models that represent the behavior of the systems properly; however, there is a lack in the results obtained and therefore an opportunity to improve the performance of these models and make them less complicated and more conservative. In this research work, LSTM networks' use to estimate *RUL* of mechanical systems, specifically the turbofan engine, will be studied.

A review of the most relevant works of recent years related to the prediction of failures using *DL* was presented. In the next chapter, the information related to the concepts necessary to understand the rest of the work in a better context is shown.

Chapter 3

Simulation System

This chapter presents all the necessary information regarding turbofan engines and *RUL*, in addition to all the necessary background to understand the proposed methodology.

3.1 Turbofan Engines

Turbofan engines are a generation of jet engines that replaced turbojets, which produces thrust using a combination of exhaust flow and bypass air accelerated by a driven fan driven by the jet core. It could be low or high bypass. The incoming air is divided into two paths at the front of the engine: bypass or secondary air and primary air. They have several advantages: they consume less fuel, which makes them cheaper, produce less pollution, and reduce environmental noise [64]. It is usually interesting to maintain high bypass degrees since they reduce noise, pollution, specific fuel consumption, and increase performance. However, an increase in bypass reduces the specific thrust at speeds close to or higher than the speed of sound; thus, low bypass turbofan engines are used in military aircraft.

A turbofan is a machine that works using thermodynamic and mechanical cycles, producing work from the latent energy supply [65]. Its operation is based on the Joule- Brayton cycle; the ideal cycle efficiency is obtained by dividing the net useful work by the energy used, as shown in equation (3.1). if it is considered that the fluid is an ideal gas where $\gamma = \frac{C_p}{C_v}$, the efficiency of the cycle is represented by the equations (3.1) and (3.2) [66].

$$\eta_i = \frac{W_{out} - W_{in}}{Q_{in}} \tag{3.1}$$

$$\eta_i = 1 - \frac{1}{\frac{p2}{p1}^{\frac{\gamma-1}{\gamma}}} \tag{3.2}$$

Turbofans engine is made up of a *fan* where the propulsion begins, a *compressors* where the secondary air is compressed to increase the pressure and air temperature. Later, the air passes to the *combustion chamber*, where the air is mixed with the fuel and burns the mixture, which then passes to the *turbines* where the air rotates the various axes. Once the hot air has passed through the turbines, it exits through a *nozzle* at the engine's rear. The nozzle walls force the air to accelerate, and the air weight combined with this acceleration produces part of the total thrust [67], due to the principle of action and reaction, also known as Newton's third law. The degradation of turbofan engines occurs over time, which is usually measured in hours of operation. In Figure 3.1, a diagram of the turbofan is observed, whit the sensors placed to give an idea of the engine aging. It means that the parameters that indicate the motor's health state must be chosen. For example, a permissible value of efficiency, temperature, or pressure of any engine components can be taken as a reference, from which an engine overhaul should be done [68].

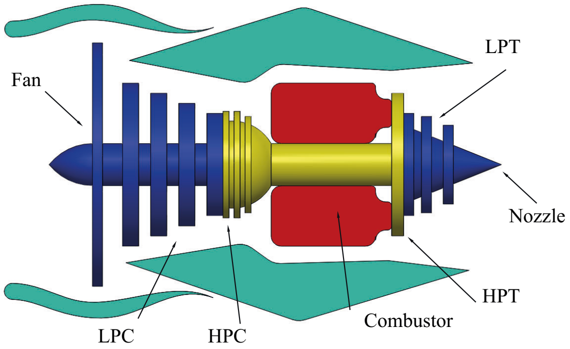


Figure 3.1: Scheme of a Turbofan and its components

3.2 Turbofan engine degradation

The most common degradation can be defined as the blade (airfoil) pollution, pitting on the blade surface, the rising of the blade clearances, the rotor imbalance caused by blade clearances, nozzle hoarse, unstable airflow, low speeds, and higher ambient temperatures [69; 70; 71]. Gas turbine degradation could be classified as fouling, corrosion, oxidation, hot corrosion, erosion and wear, and particles' coalescence and mechanical degradation [72]. According to [67], mechanical wear due to regular use is reflected as changes in internal flow characteristics, components efficiency, and at its optimum point of operation, causing a decrease in efficiency or safe engine margins. Internal engine damage may reduce the component's strength, and its accumulation also causes the initiation of flaws, which may lead to cracking and component failure [73]. Usually, high-pressure shaft components are more susceptible to wear due to higher speed and pressure operation conditions [74]. Below, equation (3.3) to equation (3.6) shown different types of efficiency of a turbofan engine; when these efficiencies fall below their normal operating points, it can be considered that there is a failure in the system.

Propulsive efficiency:

$$\eta_p = \frac{Thrust\ Power}{Power\ imparted\ to\ engine\ airflow} \tag{3.3}$$

Thermal Efficiency:

$$\eta_{th} = \frac{Power\ imparted\ to\ engine\ airflow}{Rate\ of\ energy\ supplied\ in\ the\ fuel} \tag{3.4}$$

Overall Efficiency:

$$\eta_o = \frac{T * u}{\dot{m}_f * Q_r} \tag{3.5}$$

Where T is the thrust, u is the aircraft speed, $\dot{m}_f$ is the fuel flow, and Q_r the fuel calorific value.

Specific Fuel Consumption:

$$TSFC = \frac{\dot{m}_f}{T} \tag{3.6}$$

Turbofan degradation has a direct consequence of failure occurrence at some point in the jet's operation. In this case, *RUL* estimation of the system becomes essential to monitor the system's status continually and make appropriate decisions according to said monitoring. One approach used to analyze the data obtained from the monitoring system is the *DL* models. The faults to be considered for this investigation are related to the fan and the high-pressure compressor.

3.3 Simulation conditions

CMAPS is a flexible turbofan engine simulation environment that provides easy access to health, control, and engine parameters in an extensive and realistic commercial turbofan engine range. The simulation has different simulation modes, Open-Loop Engine and Generation of Linear Models, Controller Design With the Model-Matching Algorithm, and Simulation of the Controlled Engine. CMAPS simulates an engine model of the 90,000 lb thrust class. The package includes an atmospheric model capable of simulating operations at altitudes ranging from sea level to 40,000 ft, Mach numbers from to 0.90, and sea-level temperatures from -60 to 103. The package also includes a power management system that allows the engine's operation over a wide range of thrust levels throughout the full range of flight conditions [75].

It also contains fan speed controllers and a set of regulators and limits that prevent the engine from exceeding its design limits. Figure 3.1 shows significant engine components, and Figure 3.2 shows some subroutines of the engine simulation, starting with the inlet of the system that corresponds to the atmosphere, which is in contact with the fan, passing through the compressor, the combustion chamber, and the turbine to finally reach the nozzle which represents the outlet of the system.

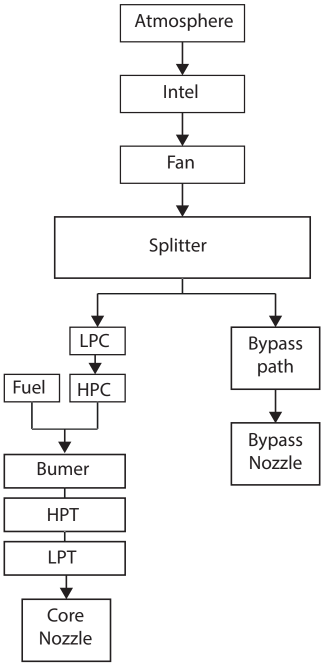


Figure 3.2: Subroutines of the engine simulation

The simulator has two types of inputs, control inputs, and health parameters, which its manipulation is direct. The system inputs constitute a set of health parameters associated with pressure, flow, and efficiency characteristics of the fan, *low-pressure compressor (LPC)*, *low-pressure turbine (LPT)*, *high-pressure compressor (HPC)*, and *high-pressure turbine (HPT)* of the turbofan. The system outputs include various sensor response surfaces and operability margins. Table 4.1 shows 21 of the 58 outputs available for the system used.

There is a fleet of aircraft whose engines are subjected to different flight conditions through simulation. Depending on different factors, the engines' damage will be different between each cycle of the engines. The sensors located in the turbofans are counted to predict the remaining useful life of engines throughout the cycles measured every ten minutes until failure. Further, the effects of between-flight maintenance have not been explicitly modeled but have been incorporated as the process noise [75]. Additionally, to simulate this scenario, it is necessary to add initial wear, which is very common in real systems and is considered normal but unknown at the beginning of each cycle. Also, an additional noise is simulated that serves the model as the factors that are not taken into account in the simulation as stoppages due to maintenance between each flight. The two types of failures observed in the simulation are failures in the fan and failures in the high-pressure compressor; however, there is not enough information to classify these failures with the data available.

3.3.1 Damage propagation modeling in the simulation system

One of the most important steps in simulating the data is the correct propagation of the system's damage. Common models used across different application domains include the Arrhenius model, the Coffin-Manson mechanical crack growth model, and the Eyring model [75]. What all these models have in common is the exponential evolution of the faults, in [75] they used a generalized equation for wear, which ignores micro-level processes but retains macro-level degradation characteristics as shown in equation (3.7):

$$w = Ae^{B(t)} \tag{3.7}$$

Assuming an upper wear threshold, th_w, which denotes an operational limit beyond which the component/subsystem cannot be used. The generalized wear equation is written as a time-varying health index, but it is not discussed here because the main interest in this work is to detect when the element will fail, but not how it will fail. This dataset was used for the prognostics challenge competition at the International Conference on Prognostics and Health Management (PHM08). The winners of the competition were [41; 42; 76].

Two different databases are resulting from the simulation made with CMAPS. The first dataset is used in the Prognostics Health Management PHM08' competition carried out in 2008. Six different failure modes are used; this dataset is considered the most standard for calculating the degradation of engines by *RUL*. The second, Turbofan Engine Degradation dataset or CMAPS, is divided into four subsets representing different failure modes. These two databases are used to evaluate the robustness and effectiveness of the proposed method.

3.3.2 Simulation databases

PHM08' Database

PHM08's dataset consists of multiple multivariate time series. The dataset is divided into training and test sets and could be considered from a fleet of engines of the same type. As mentioned before, each engine starts with different degrees of initial wear and manufacturing variation, which is unknown to the user but can be considered healthy, which means it is not considered a fault condition. In addition to the data from the 21 sensors, data from 3 operational settings, altitude, mach number, and the throttle resolve angle, which significantly affect the sensors, are included. The motors are operating normally at the beginning of each time series, and they begin to degraded through cycles. In the training set, each motor's time series ends when the RUL is zero; that is, the motor is no longer in a suitable state of operability. However, for the test set, the time series ends for a specific time before. It must be determined how many cycles remain before the engine is no longer operable or the respective maintenance must be done.

In this dataset, six different operating conditions resulting from the combinations of the three operational settings are identified; each operating condition presents a different trend in the sensors' behavior over time. In addition to this, the dataset shows faults in both the fan and the high-pressure compressor. Table 3.1 shows the characteristics of the PHM08' dataset, including the number of engines in the training, test, and validation set, the number of conditions, and the fault modes.

Turbofan engine degradation database

This dataset was obtained under similar conditions to those of the PHM08' dataset; however, the data set is divided into four different subsets. FD001 dataset contains 100 train and test engines with one operation condition and one fault which occurs in the high-pressure compressor. FD002 has 260 engines in the train set and 259 in the test set, with six operation conditions and one failure in the high-pressure compressor. FD003 has 100 train and test engines with one operation condition and two faults, which occurs in the high-pressure compressor and fan. FD004 has 248

Table 3.1: PHM08' dataset characteristic

Data Set: PHM08'
Train trajectories: 218
Test trajectories: 218
Validation Trajectories: 435
Conditions: SIX
Fault Modes: ONE (HPC Degradation)

engines in the train set and 249 in the test set, with six operation conditions and two failures in the high-pressure compressor and fan. These characteristics are shown in Table 3.2

Table 3.2: Turbofan Engine dataset characteristic

Data Set: FD001	**Data Set: FD002**
Train trajectories: 100 Test trajectories: 100 Conditions: ONE (Sea Level) Fault Modes: ONE (HPC Degradation)	Train trajectories: 260 Test trajectories: 259 Conditions: SIX Fault Modes: ONE (HPC Degradation)
Data Set: FD003	**Data Set: FD004**
Train trajectories: 100 Test trajectories: 100 Conditions: ONE (Sea Level) Fault Modes: TWO (HPC Degradation, Fan Degradation)	Train trajectories: 248 Test trajectories: 249 Conditions: SIX Fault Modes: TWO (HPC Degradation, Fan Degradation)

3.4 Remaining Useful Life Estimation

The *RUL* estimation of a system or a component is defined as the length from the current time to the end of its useful life, and it can be used to characterize the system's current health status [77]. As for *CBM*, it is possible to make *RUL* estimation through three main approaches, model-based, knowledge-based, and data-driven. The *RUL* of a system is a random variable, and it depends on the current age, the operation environment, and the observed condition monitoring [77]. For

RUL estimation, an instrumented system must monitor the system's environmental conditions and parameters, which give indications of the status in real-time. These estimations are subject to uncertainties inherent to the system. Besides, *RUL* is a technique used to estimate the system's degradation, which occurs due to wear and damage from its use. It means that it can not predict sudden failures that are generally due to human errors, cracks in elements, defects in materials, among others.

For a turbofan engine system, *RUL* is the number of cycles remaining for each engine before it must undergo a repair or schedule a maintenance shutdown. This process will be monitored in line with the in-flight data of the turbofan sensors. The maintenance team will have access to data to verify the operation. The monitoring system will then present the necessary alerts at the adequated cycles that depend on each aircraft cluster's requirements, for example, between 10 and 8 hours before the engine will fail. For this case, each cycle is equivalent to 10 minutes. In this way, the first alert of system degradation could occur at 70, 60, or 50 remaining cycles equivalent to a range of 12 to 8 hours of remaining operation.

3.4.1 RUL labeling

Since the system degradation initially can be considered constant, a piece-wise linear function is used to label the *RUL* of the data, the *RUL* value of 120 is set, according to [41]. This value could change in terms of the minimum run the length of the databases. Then, at a certain point of operation, the motor begins to degrade linearly with a slope of -1. Whit equations (3.8) to (3.12), it is possible to obtain the operating cycle where each motor starts to degrade:

$$120 = cut + b \tag{3.8}$$

$$0 = max_{cycle} - b \tag{3.9}$$

From equations (3.8) and (3.9):

$$120 + cut = max_{cycle} \tag{3.10}$$

$$cut = max_{cycle} - 120 \tag{3.11}$$

Figure 3.3 shows the *cut* value for the maximum cycle of the motor, where the model is divided, forming a piece-wise linear equation, then:

$$y = \begin{cases} 120 \rightarrow 0 < x \leq cut \\ -x + (120 + cut) \rightarrow cut < x \leq max_{cycle}) \end{cases} \tag{3.12}$$

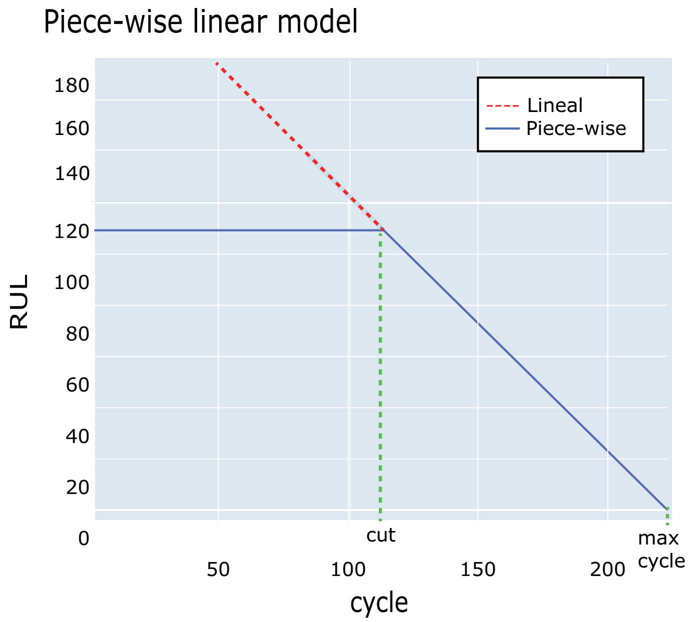


Figure 3.3: Piece-wise linear degradation model

Using piece-wise linear functions is more realistic than using a linear function. In mechanical systems, machines are in good condition from the start of the operation until a specific cycle, which means that its degradation does not begin with the first moment of use.

In this chapter, all the theoretical background necessary to fully understand this research work is presented. The description of the methodology used in this research work is developed in the following chapter.

Chapter 4

Proposal

This chapter presents the proposed methodology to address RUL estimation in turbofan engines and the description of the Turbofan Engine Degradation database [75] provided by the National Aeronautics and Space Administration (*NASA*).

4.1 Methodology

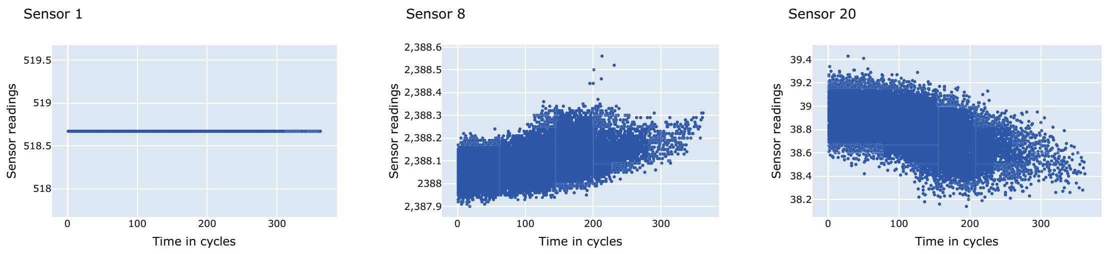
LSTM neural networks mainly focus on fault diagnosis and classification [78; 79; 80]. However, their primary use in *MHM* is on predicting the *RUL*, which is an essential part of *PHM*. Using data-driven approaches helps learn the relationship between monitored data, and the corresponding *RUL* gradually becomes prosperous [81]. Thanks to the structure of *LSTM* networks, they focus on solving problems in terms of *CBM*, especially for failure prediction using *RUL*. One of the main advantages of these networks is that the preprocessing of the data is not extensive. In most cases, it will be enough to normalize the data and identify the most suitable features for the network, according to their monotonicity, prognosability, and trendability. Figure 4.7 shows *RUL's* estimation in systems with multiple failure modes such as turbofan engines using *LSTM* neural networks. In this work, we developed a methodology for estimating *RUL* in turbofan engines based on *LSTM* using signals from sensors located throughout the entire turbofan. This section describes the methodology step-by-step and the training procedure.

4.1.1 Data pre-processing

The pre-processing stage consists of observing the sensors' behavior and how they are correlated to select those that best adjust to the system's degradation and then normalize them.

Sensor selection starts from observation in each operating regime. Sensors present discrete or continuous values. Sensors with discrete values remain constant throughout all cycles, which does not represent useful information for system degradation. Sensors with continuous values show a positive or negative trend and could have inconsistent values towards the end of the cycles for the engines or have a monotonic trend. Figure 4.1 show the different trend in the data. With this information, sensors with constant values over time will be discarded, and only sensors with continuous values will be used. However, to observe the dependence between sensors is made a correlation of the sensors. Figure 4.2 shows the correlation of sensors; y-axis and x-axis are the sensors, from 1 to 21 and operating conditions from 1 to 3 of the system and, the color bar goes from -0.2 for sensors which its correlation is negative, to 1 which its correlation is positive, through 0 which are non-correlated sensors.

It is necessary to emphasize that*DL* models can handle the sensor data that are not representative but would take more time to train them. The preprocessing and selection of the sensors are made first. Figure 4.1 shows the different scales that the sensor records have; that is why the data must be normalized. Table 4.1 shows the turbofan engine system's inputs. The model's input variables are time series with data from the different sensors located in the fan, compressors, turbines, and turbofan nozzle. The data from these sensors have different value scales; therefore, they will be normalized from 0 to 1 to have all the same numerical range values. Beyond selecting the most representative sensors and normalizing the data, it is unnecessary to make additional pre-processing since the model could handle the raw data.



(a) (b) (c)

Figure 4.1: (a) represents a sensor with discrete constant values, (b) represents a sensor with continuous and inconsistent values, and (c) represents a sensor with continuous and monotonic values.

4.1.2 Training/testing relation

In DL models, it is essential to separate the database in training and testing. Typically the data should be shuffled to assure that the testing and training sets share homogeneity in their characteristics. However, when dealing with time series is not that simple because the data is dependent on time. For preserving the time series, it is necessary to choose 70% of the engines for training

Table 4.1: CMAPS outputs to measure system response

Input	**Name**	**Units**
T2	Total Temperature at Fan inlet	R
T24	Total Temperature at LPC outlet	R
T30	Total Temperature at HPC outlet	R
T50	Total Temperature at LPT outlet	R
P2	Pressure at Fan inlet	psia
P15	Total Pressure in bypass-duct	psia
P30	Total Pressure at HPC outlet	psia
Nf	Physical Fan Speed	rpm
Nc	Physical Core Speed	rpm
epr	engine Pressure Ratio	—
Ps30	Static Pressure at HPC outlet	psia
ϕ	Ratio of Fuel flow at Ps30	pps/psi
NRf	Corrected Fan Speed	rpm
NRc	Corrected core Speed	rpm
BPR	Bypass Ratio	–
farB	Burner Fuel-air Ratio	–
htBleed	Bleed Enthalpy	–
Nf_{dmd}	Demanded Fan Speed	rpm
$PCNfR_{dmd}$	Demanded Corrected Fan Speed	rpm
W31	HPT Coolant Bleed	lbm/s
W32	LPT Coolant Bleed	lbm/s

and 30% for testing. Then, 20% of the training set is used to validate it, as shown in Figure 4.3a. Then, doing cross-validation verifies that the model is independent of the training data. Figure 4.3b shows the data splitting in training and testing for different percentages of the dataset. 70, 80, 90 and, 100% of the database is split for cross-validation to see the variance of the results obtained with the model.

The failures treated with this methodology are failures due to the engines' regular use with different operational conditions. These failures can occur mainly in the fan, the high-pressure compressor, or both. The *RUL* estimation is valid only for failures whose degradation evolves in time and has an identifiable behavior but not for sudden failure predictions.

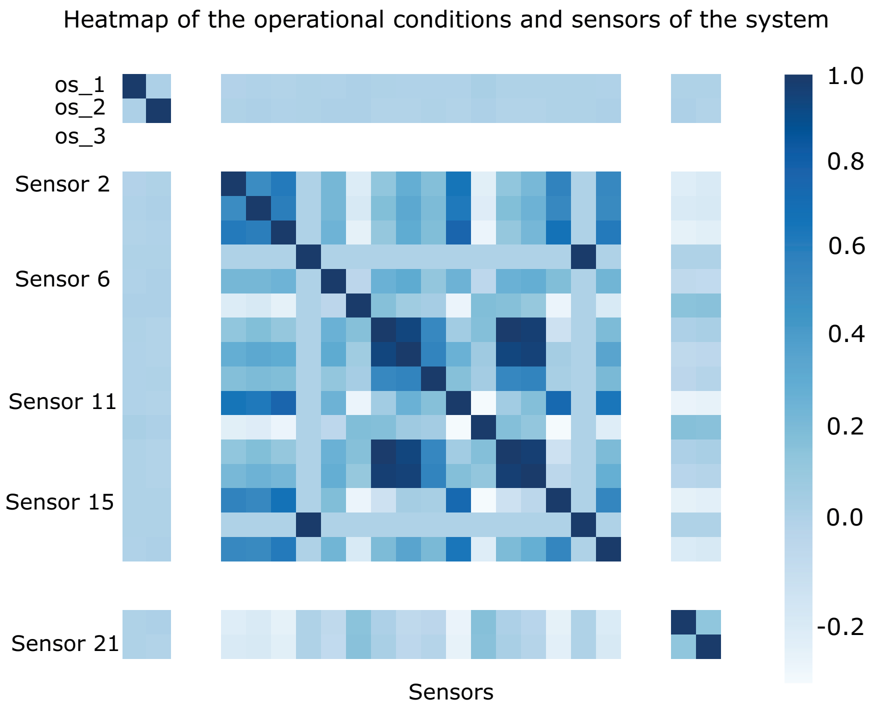
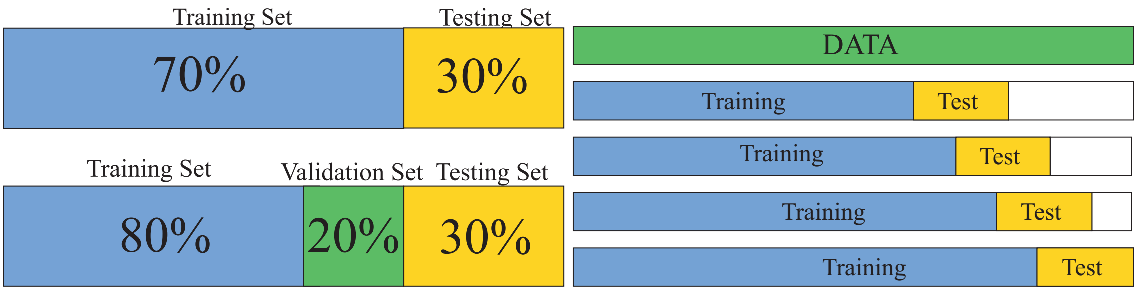


Figure 4.2: Correlation matrix of the sensors in the same cluster.



(a) Training/ testing split

(b) Cross Validation

Figure 4.3: (a) represents the split of the database and, (b) represents the cross-validation procedure's split.

4.1.3 Key Performance Indicators

When working with *DL* models, it is essential to define the appropriate metrics to measure the model's behavior. The most commonly used metrics are *MAE* and *RMSE* for prediction problems.

MAE is the sum of the difference between the real and predicted values over the number of values. It does not consider the direction of the errors. The difference between real and predicted values is the error:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}|}{n} \tag{4.1}$$

RMSE is the square root of the sum of the squared error over the number of values. RMSE gives a higher weight to large errors. Due to that, *RMSE* is more useful when large errors are particularly undesirable [82]. *RMSE* is a frequently used measure of the differences between values predicted by a model and the values observed. The *RMSE* for training and test sets should be very similar if a suitable model has been built. If the *RMSE* value for the test set is higher than the value for the training set, there is probably an overfit of the data:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2} \tag{4.2}$$

In turbofan engine degradation, the model will use *MAE, MSE* and, *RMSE* error . Also, the model will use another metric called *Score function*. The *Score Function* is a performance measure that the authors in [75] designed to measure system performance for prognosis and health management competition in 2018. In *PHM*, the critical aspect is to avoid failures. Therefore it seeks to predict failures early instead of predicting them late. The scoring algorithm that the authors developed for this challenge was asymmetric around the actual time to failure. The penalization of the late predictions was heavier than early predictions. This penalty grows exponentially with an increasing error. Equation 4.3 shows how the parameters $a_1 = 10$ and $a_2 = 13$ control the asymmetric preference: :

$$Score = \begin{cases} \sum_{i=1}^{N} e^{-\frac{d}{10}} - 1 \rightarrow for d < 1 \\ \sum_{i=1}^{N} e^{\frac{d}{13}} - 1 \rightarrow for d \geq 1 \end{cases} \tag{4.3}$$

where $d = \hat{y}_{RUL} - y_{RUL}$ and N is the number of units or motors in the dataset.

When evaluating the results, the authors who developed the Score Function equation found that this forecasting metric must be improved. Predicting further into the future is more complicated than predicting closer to the end of life. Also, it is more important to weigh the *RUL's* accuracy when it is closer to the end of the useful life. Considering it, more weight should be assigned to cases where the motors have a shorter *RUL*. Another feature of the data is that the algorithm's performance is evaluated from multiple engines, for example, in fleet applications. Since the metric is a combined aggregate of performance for individual engines, an additional correlation metric must be employed to ensure that an algorithm consistently predicts well for all cases rather than well for some and bad for others. Due to this function's characteristics, it will be used to complement the results obtained with *MAE* and *RMSE*.

4.1.4 Flow diagram of the proposed methodology

Figure 4.4 shows a diagram of the methodology implemented to estimate the remaining useful life in turbofans engines using *LSTM* neural networks.

4.2 LSTM Neural Network design and structure

The idea of *RNN* first emerged in 1974. The Hopfield Newark introduced this concept in 1982 [83], but the idea was described shortly in 1974 by [84]. RNN are networks with loops in them. They have short-term memory that gives them the possibility to remember and apply that knowledge in the forward part. The main problem with *RNN* approaches is the vanishing gradient problem. With the use of certain activation functions, the gradients of the loss function approach zero, making the network difficult to train or even not train at all. The lower the gradient is, the harder it is for the network to update the weights and, the longer it takes to get to the final result. Other architectures as Gated Recurrent Unit *GRU*, *LSTM* and its variants or even *Convolutional LSTM (ConvLSTM)* were proposed to overcome this problem.

The simplest RNN is the Elman networks, which take the output from hidden layers as inputs of the next layer [85] and Jordan networks, feeding the next layer from the output layer instead of the hidden layer [86]. The equations of these networks are bellowed:

- Elman Networks

$$h_t = \sigma_h(W_h x_t + u_h h_{t-1} + b_h) \tag{4.4}$$

$$y_t = \sigma_y(W_y h_t + b_y) \tag{4.5}$$

- Jordan Networks

$$h_t = \sigma_h(W_h x_t + u_h y_{t-1} + b_h) \tag{4.6}$$

$$y_t = \sigma_y(W_y h_t + b_y) \tag{4.7}$$

where x_t is a vector of inputs, h_t are hidden layer vectors, y_t are the output vectors, σ_h and σ_y are the activation functions, W and u are weight matrices and, b is the bias vector.

LSTM neural networks are a type of *RNN* capable of learning long-term dependencies. The first to talk about *LSTM* neural networks were Hochreiter and Schmidhuber. The critical insight in

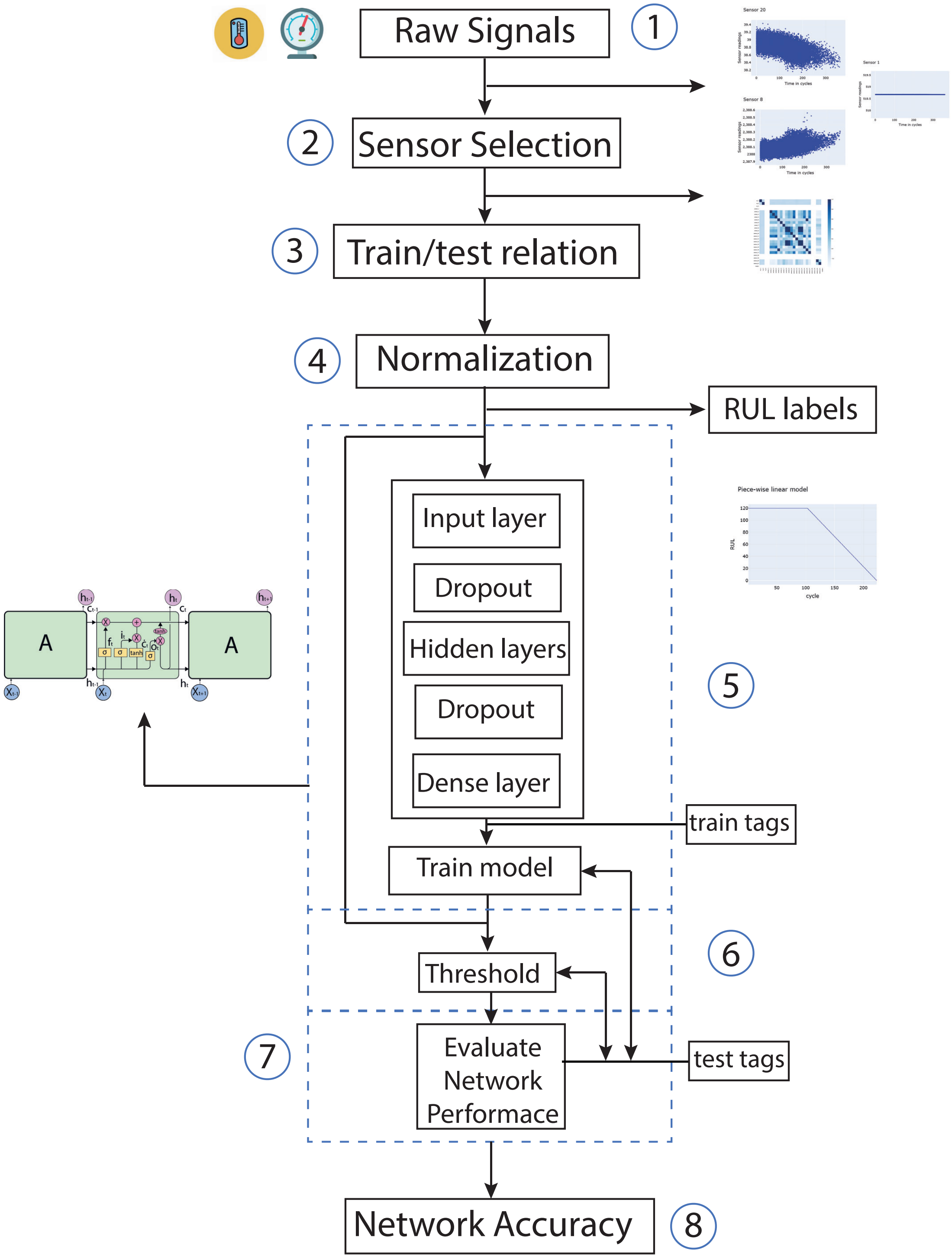


Figure 4.4: Scheme of *RUL* estimation methodology using LSTM

the *LSTM* design was to incorporate nonlinear, data-dependent controls into the *RNN* cell, which can be trained to ensure that the gradient of the objective function concerning the state signal does not vanish [87]. *LSTM* removes or adds information to the cell states called gates: input gate i_t forget gate f_t and output gate o_t can be defined as in [88].

Neural networks have an input layer, an output layer, and hidden layers, which are the layers between the input and the neural network's output. Hyper-parameters are the parameters configured before training the model; the most important are: batch size, epochs, learning rate, time steps, and activation functions. This model's *LSTM* neural network is a simple *DL* model with a 1input layer, 3-hidden layers, and 1-output layer with 128, 64, 32, 16, and 1 neuron, respectively. Figure 4.5 shows the general scheme at time t of the neural network. For *LSTM* networks, the input shape parameter defines the network's input and must be a 3D matrix of the form batch size, time steps, and input data, as shown in Figure 4.6. The unit parameter only corresponds to the number of neurons that will have the input layer's output. It has no direct relationship with the network's input, so it can be any number that will contain the results of having updated the weights and biases in the first layer.

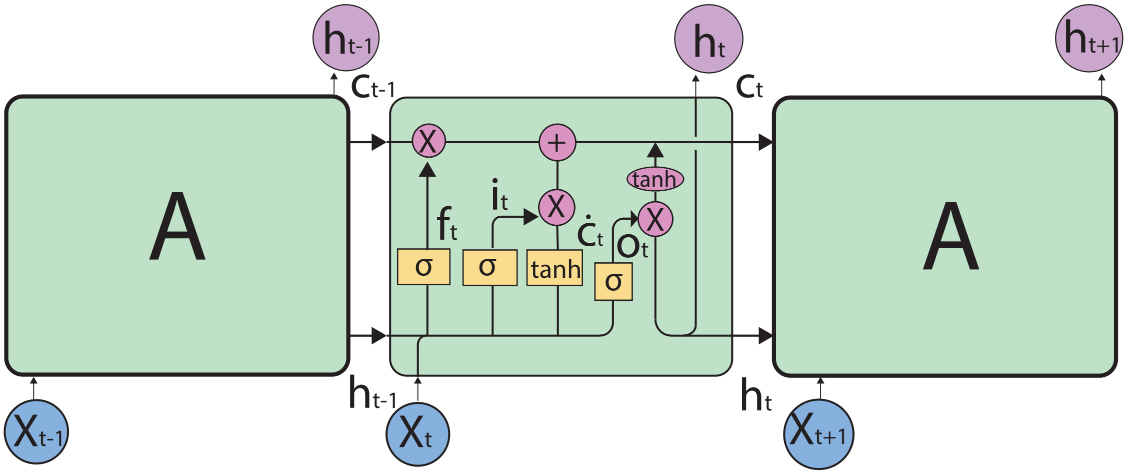


Figure 4.5: Representation of a Long-Short Term Memory Neural Network

The core idea behind *LSTM* neural networks is their memory cell. It can maintain its state over time through the cell state vector, representing the memory of the *LSTM*, and it changes, forgetting old memory and adding new memory through its gates. The *LSTM* gates are divided into three: the input gate, which identifies the information that must be used to modify the network memory, the *sigmoid* function decides which values to let through 0.1, and the *tanh* function gives weightage to the values which are passed, deciding their level of importance ranging from-1 to 1; forget gate, which identifies the information that should be forgotten, this is done through the activation functions, 0 means that the information should be omitted and 1 that the information should be kept for each number in the cell state; the output gate, where the input and the memory of the block

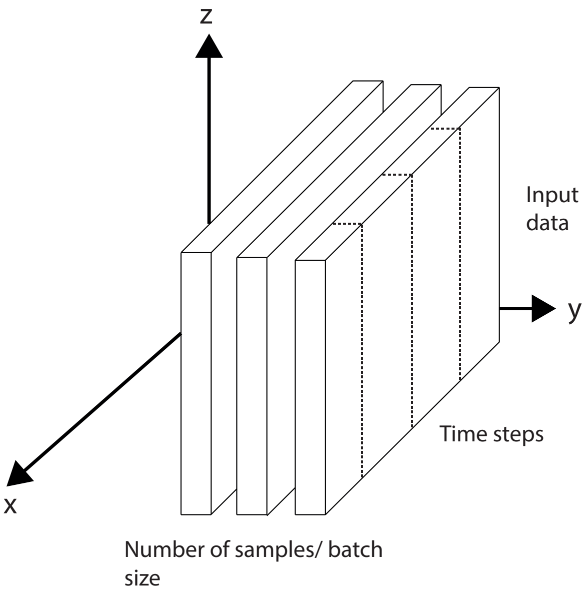


Figure 4.6: Representation of the input shape of the LSTM Neural Network

are used to decide the output, *sigmoid* function decides which values to let through 0.1 and *tanh* function gives weightage to the values which are passed deciding their level of importance ranging from-1 to 1 and multiplied with the output of *sigmoid*.

The first step in the *LSTM* is to decide which of the weights W, biases b, and the values h_{t-1}, which correspond to the network output in $t-1$ that enters the actual cell through the input gate is to be discarded and which information is to be stored via 4.8. The next step is to decide what new information will be stored in the cell state, this procedure is done in two stages; first a sigmoid layer decides which values are going to be updated, see equation (4.9), then a tanh layer creates a vector of new candidate values, see equation (4.10):

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{4.8}$$

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{4.9}$$

$$\widetilde{C}_t = tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{4.10}$$

Later the value of C_{t-1} is updated to C_t. the old state is multiplied by ft, forgetting what sigmoid and than layer decided to forget earlier, then $i_t\widetilde{C}_t$ is added as in equation (4.11). These are the new candidate values:

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{4.11}$$

Finally, based on the cell state, after passing the information through the forget gate and updating Ct's values, it is decided what the output will be. See equations (4.2) and (4.13).

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{4.12}$$

$$h_t = o_t * tanh(C_t) \tag{4.13}$$

The X_t inputs will be the data from the sensors. The entire internal process is done to update the weights and bias, taking into account the network's activation functions and other parameters. At this point, the network identifies which data is useful and what data can be discarded. Table 4.2 shows the general structure of the proposed *LSTM* neural network with its respective properties. Although there is no established methodology in the literature to select the networks' hyperparameters, some recommendations can be followed. For example, activation functions depend on the type of data available. However, it is recommended to use non-linear functions such as sigmoid, relu, tanh, or softmax; finally, the use of these functions is established for the model's performance. When training a neural network, the model might fit perfectly to the training data but not to the test data; this is known as overfitting. To prevent this problem, a parameter called dropout is used.

Table 4.2: LSTM Neural Network Structure

# Layer	Layer type	Units	Activation function	Dropout
1	LSTM	128	Softmax	0.2
2	LSTM	64	ReLu	0.2
3	LSTM	32	ReLu	0.2
4	LSTM	16	ReLu	0.2
5	Dense	1	ReLu	-

Figure 4.7 shows a general methodology to estimate RUL with*DL* approaches using LSTM neural networks. First, an off-line phase where the historical data obtained from the sensors is used as inputs to the model and identify the behavior of the system degradation. Then, a preprocessing is done by normalizing the data for a better performance of the model. The trend of the sensors is analyzed to select which are best related to the system's behavior. Finally, a model is made with

an *LSTM* neural network in which the data is divided into training and test data. Once the desired performance is achieved, the model is saved and used in the second phase. In the online phase, real-time data is also pre-processed, and the trained model is applied. Finally, an estimation of how much useful life the system has left is done, and with this, *CBM* decisions can be made. Although this methodology can be used with any other *DL* model, the idea of using *LSTM* neural networks arises from their ability to handle time series in such an efficient way. That is why these networks are the most suitable for making predictions because changes in system data are directly related to operating time.

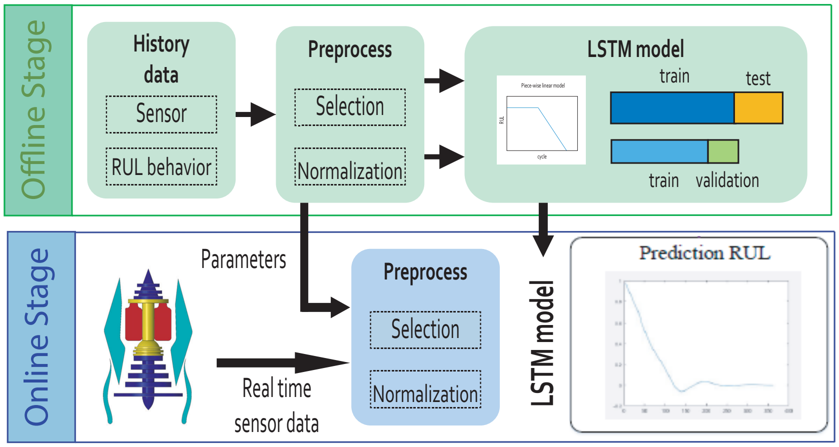


Figure 4.7: Basic Structure of a model using *LSTM* for *RUL* Estimation

In this chapter, a detailed description of the methodology used to develop the research work was made. In the next chapter, the results obtained when developing it will be shown.

Chapter 5

Results

This chapter presents the results obtained when applying the proposed methodology to the PHM08' and CMAPS datasets. It also explains the selection of sensors such as time steps, activation functions, and the number of training epochs for feeding the *LSTM* neural network. In addition to evaluating the network with two datasets. Then, the methodology is compared with other works focused on *DL* techniques.

5.1 Data pre-processing

As previously mentioned, the two databases have similar characteristics; however, it is necessary to make a slightly different pre-processing due to the operational conditions and the different failures that occur.

Sensor selection

Case I

In PHM08' database are six operation regimens resulting from the operational conditions that exist. Also, two different failures occur in the fan and the high-pressure compressor. Figure 5.1 shows the different clusters that exist. Then, when applying K-means, an unsupervised classification algorithm that groups object into k groups based on their characteristics, it is included in the dataset the six resulting clusters as another feature of the system.

For each regime, there is similar behavior of the sensors. By observing the sensors' behavior through all the cycles for each motor, sensors with constant values are eliminated. These sensors were 1, 5, 6, 10 16, 17, 18, and 19. A correlation of the sensors to identify which ones are more correlated, either positively or negatively, is made. The results show that sensors have a correlation value of 0.7 or more; therefore, the model uses all remaining sensors as input.

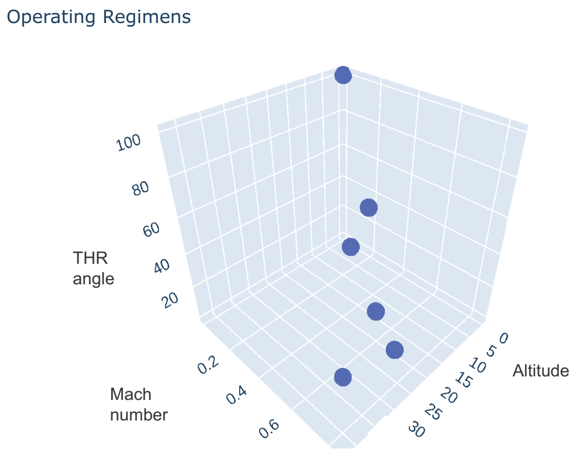


Figure 5.1: Operating Regimes resulting from the combination of the operational settings

Case II

In the CMAPS database, it is unnecessary to apply k-means since there is just one operating regime and a single fault in the high-pressure compressor. However, the rest of the methodology uses the same procedure as in PHM08' database.

Figure 5.2 shows the behavior of the 21 sensors for the two databases. For PHM08' dataset, this behavior is for each cluster, and for CMAPS dataset is the system's general behavior through the cycles. Sensors 1, 2, 5, 6, 10 16, 17, 18, and 19 are constant throughout the cycles for each engine; that is why they are not considered since they do not have any relevant information regarding system behavior degradation.

5.1.1 Data normalization

Normalization of the data is necessary to ensure that all the model features are on the same scale; in this case, normalization of the data ranges from 0 to 1. However, the use of other ranges is also valid. Equation (5.1) shows the normalization of the features. X is the value to be normalized, and X_{min} and X_{max} are the minima and maximum values in the data to be normalized, respectively. In this way, each of the system's inputs will be on a scale from 0 to 1.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{5.1}$$

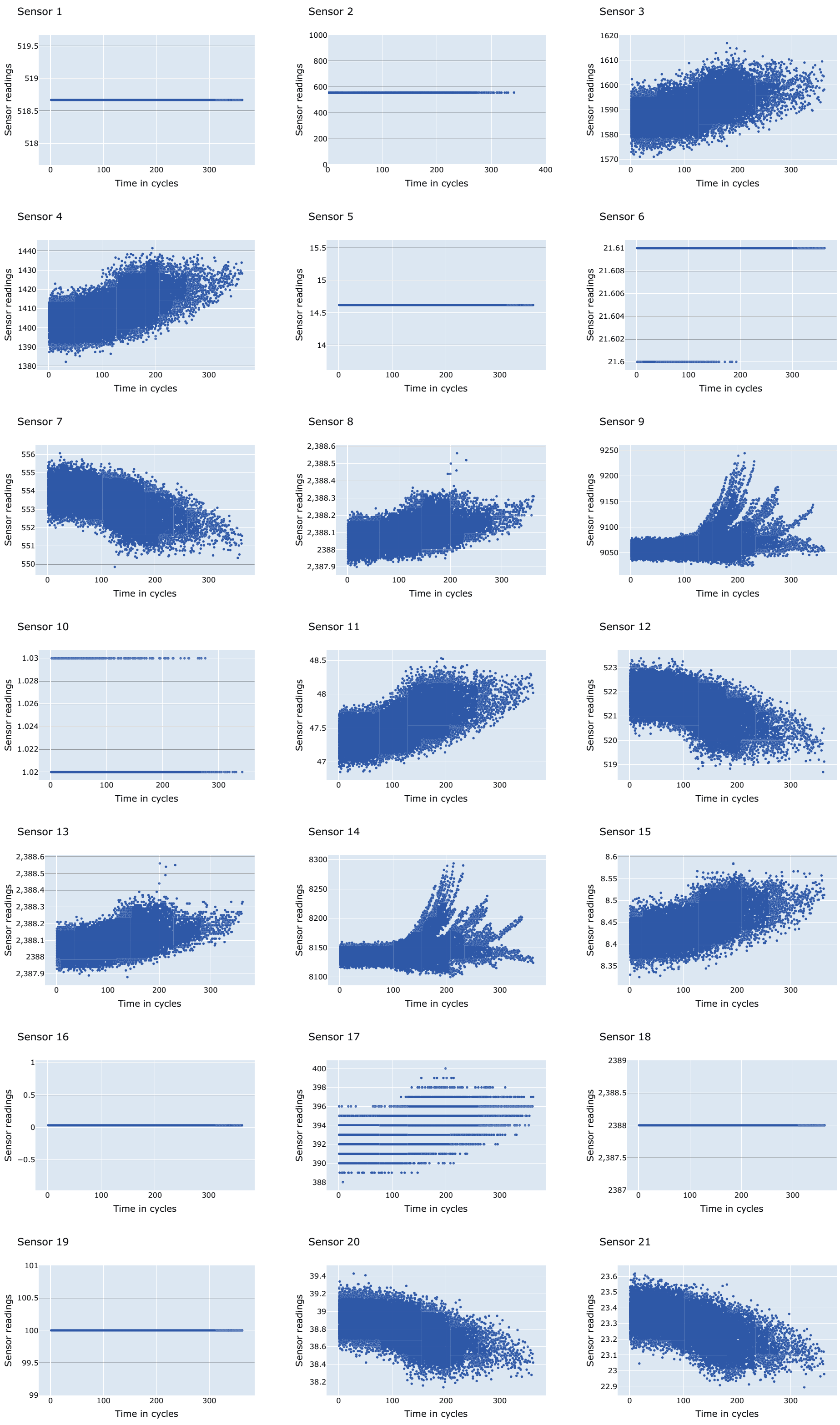


Figure 5.2: It represents the behavior of all the input sensor for the *LSTM* model.

5.2 LSTM neural network design

The parameters that cause the most significant effect during network training when using *LSTM* neural networks are learning rate, training epochs, batch size, and time steps. The convergence of the network is directly affected by the learning rate. A typical value as $1 \times 10^{-}3$ accelerates the convergence and improves the network's precision, but with a value as $1 \times 10^{-}6$, there is a negative effect on the level of precision achieved. A test is performed with different learning rate values to see which one obtains the best *RMSE* value, as shown in Table 5.1. We carried out the tests to define the parameters of the network with the PHM08' database.

Table 5.1: RMSE values and training time for different learning rate values

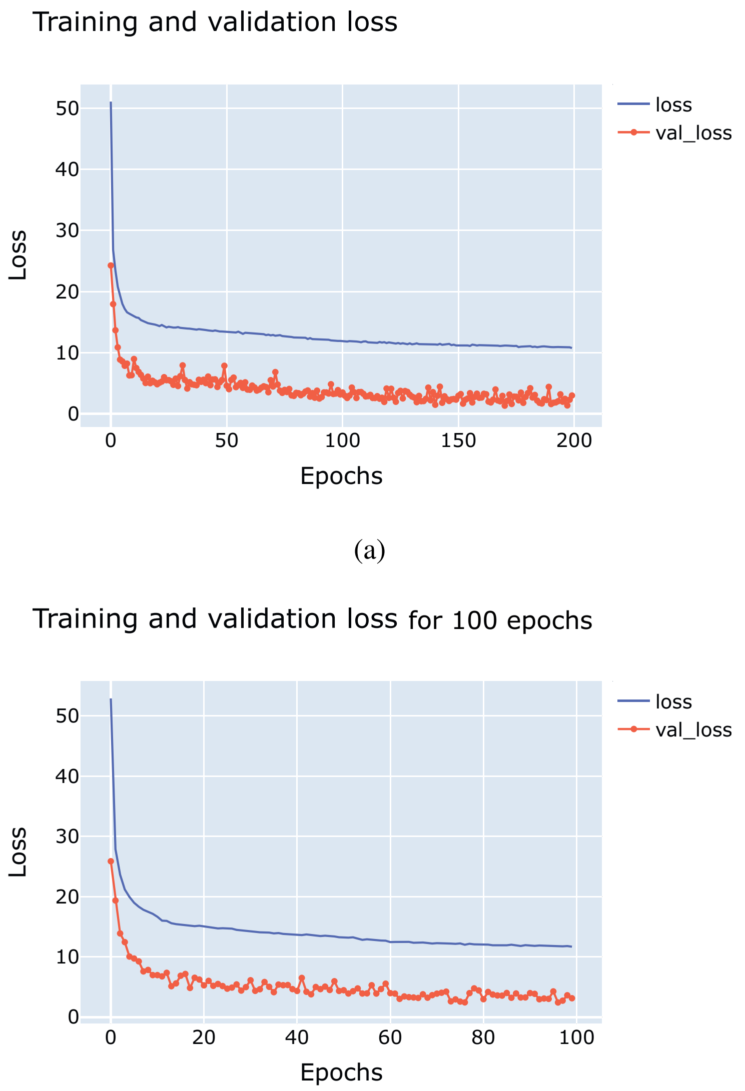
Test	**Batch Size**	**Epochs**	**Time Steps**	**Learning Rate**	**Training Time (s)**	**RMSE**	**Std**
1	32	50	5	$1 \times 10 - 3$	1056	4.03	1.15
2	**32**	**50**	**5**	$1 \times 10 - 4$	**1383**	**3.52**	**0.86**
3	32	50	5	$1 \times 10 - 5$	1563	16.99	1.61
4	32	50	5	$1 \times 10 - 6$	1544	93.34	0.55

For the time steps selection, the model uses values between 3 and 15 time steps, and it obtained the best results with a value of 5, setting it for the rest of the tests. Initially, the model is tuning with 200 epochs; however, this value was reduced to 100 without affecting the results of *RMSE* and improving the training time. Figure 5.3a and Figure 5.3b show the loss function's behavior with 200 and 100 epochs, respectively, and Table 5.2 shows the results of *RMSE* for both epochs values.

Table 5.2: RMSE values for 100 and 200 epochs

Test	**Batch Size**	**Time Steps**	**Learning Rate**	**Training Time (s)**	**RMSE**	**Std**
100 epochs	**32**	**5**	$1 \times 10 - 4$	**2251**	**5.89**	**1.04**
200 epochs	32	5	$1 \times 10 - 4$	3637	5.1	1.35

For batch size selection, the model uses values of 16, 32, and. The tests are carried out with these values since the literature recommends small batch size values, not higher than 100. Table 5.3 shows the results of the three values in terms of RMSE. Besides, Figures 5.4 and 5.5 show the results in box plots to observe the *RMSE* values in train and test set after running the model several times, its standard deviation, and the quartile means location for each batch size.



(a)

(b)

Figure 5.3: Figure (a) represents the loss function in terms of MAE for 200 epochs, and Figure (b) represents the loss function in terms of MAE for 100 epochs.

Finally, the network structure consists of an input layer with 128 units, three hidden layers with 64, 32, 16 units, and an output layer with 1 unit. Table 5.4 shows the runings with different configurations five times for each configuration. First, with an input layer of 20 neurons and increasing neurons until finding the best RMSE value. This configuration of units and activation functions had the best *RMSE* results of the model for both datasets. With this configuration, there was no overfitting of the data due to the train, and the test model's behavior is almost the same, which is reflected in the *RMSE* values obtained.

Table 5.3: RMSE values and Std for 16, 32 y 64 batch size

	TRAIN				**TEST**			
	PHM08’		**CMAPS**		**PHM08’**		**CMAPS**	
Batch size	**RMSE**	**STD**	**RMSE**	**STD**	**RMSE**	**STD**	**RMSE**	**STD**
16	3,63	0,54	6,09	2,43	4,13	0,85	5,846	2,37
32	**3,55**	**0,51**	**7,20**	**2,65**	**3,60**	**0,55**	**7,13**	**2,61**
64	4,22	1,19	6,27	3,30	4,26	1,23	6,20	3,22

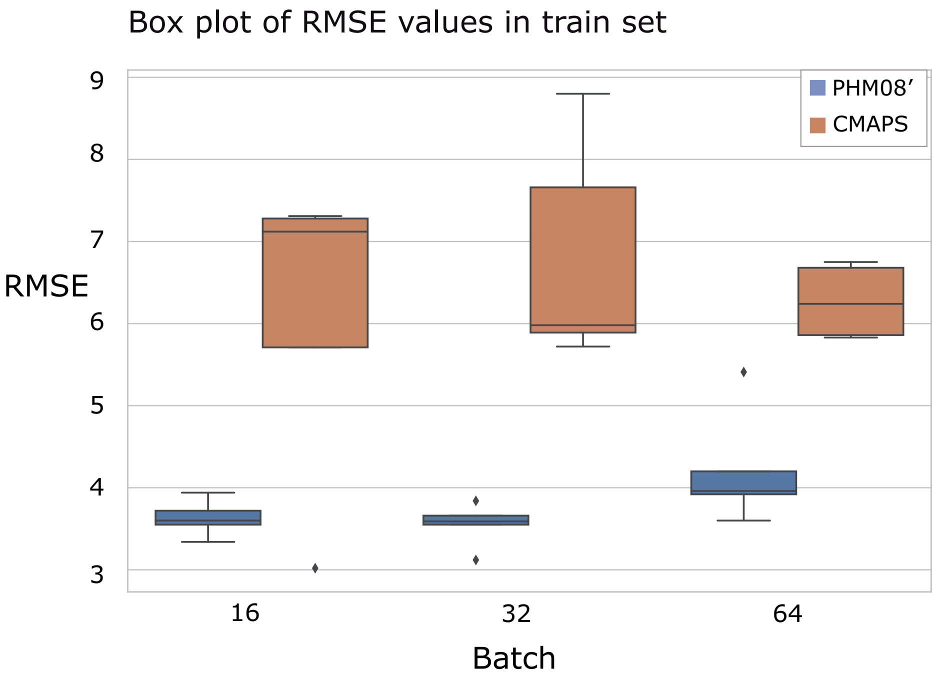


Figure 5.4: it is a box plot of the RMSE for the training set for 16, 32, and 64 batch size

Table 5.4: Neurons configurations

#	**Layer 1**	**Layer 2**	**Layer 3**	**Layer 4**	**Layer 5**	**RMSE**	**Std**
1	21	15	9	3	1	20.15	1.85
2	32	16	9	3	1	15.36	1.53
3	64	32	16	8	1	10.43	1.96
4	120	60	30	15	1	9.62	1.65
5	**128**	**64**	**32**	**16**	**1**	**3.89**	**0.89**
6	256	128	64	32	1	6.23	1.56

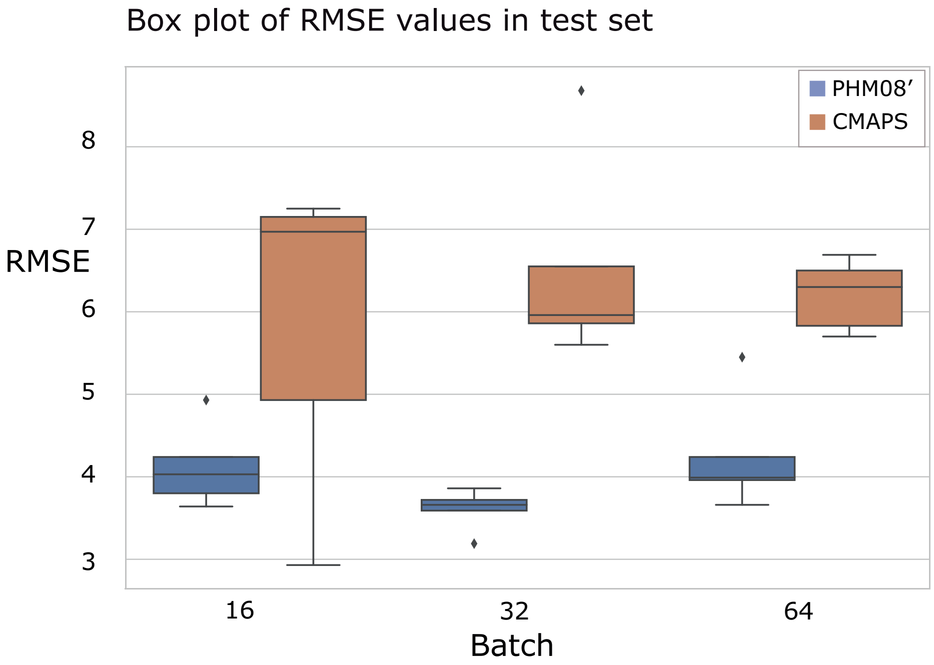


Figure 5.5: It is the box plot of the RMSE for the test set for 16, 32, and 64 batch size

5.2.1 Neural Network Train/Test

Once the model’s structure is defined, cross-validation is performed on the data to ensure that the model results are independent of the data organization. For the PHM08’ datasets, k-fold cross-validation uses 70, 80, 90, and 100% of the data; in this case, k is equal to 4. For the CMAPS database, 100% of the data is used in all cases. Still, different engines are used randomly to train and test the model, this is because the number of engines in this database is less than in PHM08 dataset. When using a percentage of the engines, the model does not have a good performance in *RMSE* and *Score*. Model tuning uses the four different partitions for calculating the RMSE values of the overall system.

Case of study

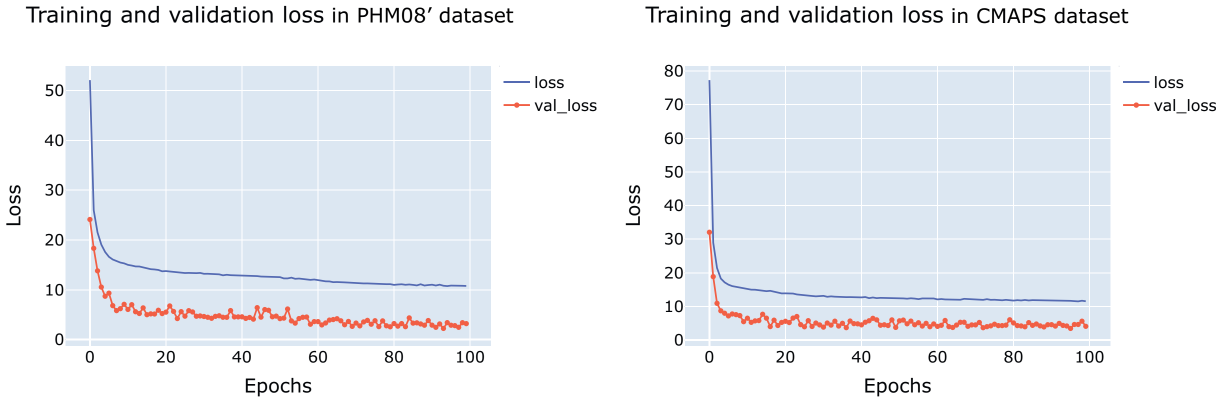
As the hyperparameters and other network values have been established, below is an illustrative example of one of the model’s executions for CMAPS and PHM08 datasets. Table 5.5 shows the final hyperparameters.

The model is trained for 100 epochs with the established hyperparameters. Figure 5.6 shows the loss of the model through the epochs for the two databases. This behavior is typical of a loss function; in this case, it is in terms of *MAE*. Initially, in the first epochs, the loss is high, but the loss begins to decrease exponentially until the last epoch. The model behavior is observed in the

Table 5.5: Established Hyper-parameters for *LSTM* model

Layers	**Epoch**	**Batch Size**	**Time Steps**	**Learning Rate**
4	100	32	5	$1 \times 10 - 4$

test set. This behavior indicates that the model has a good performance with the combination of the hyperparameters.

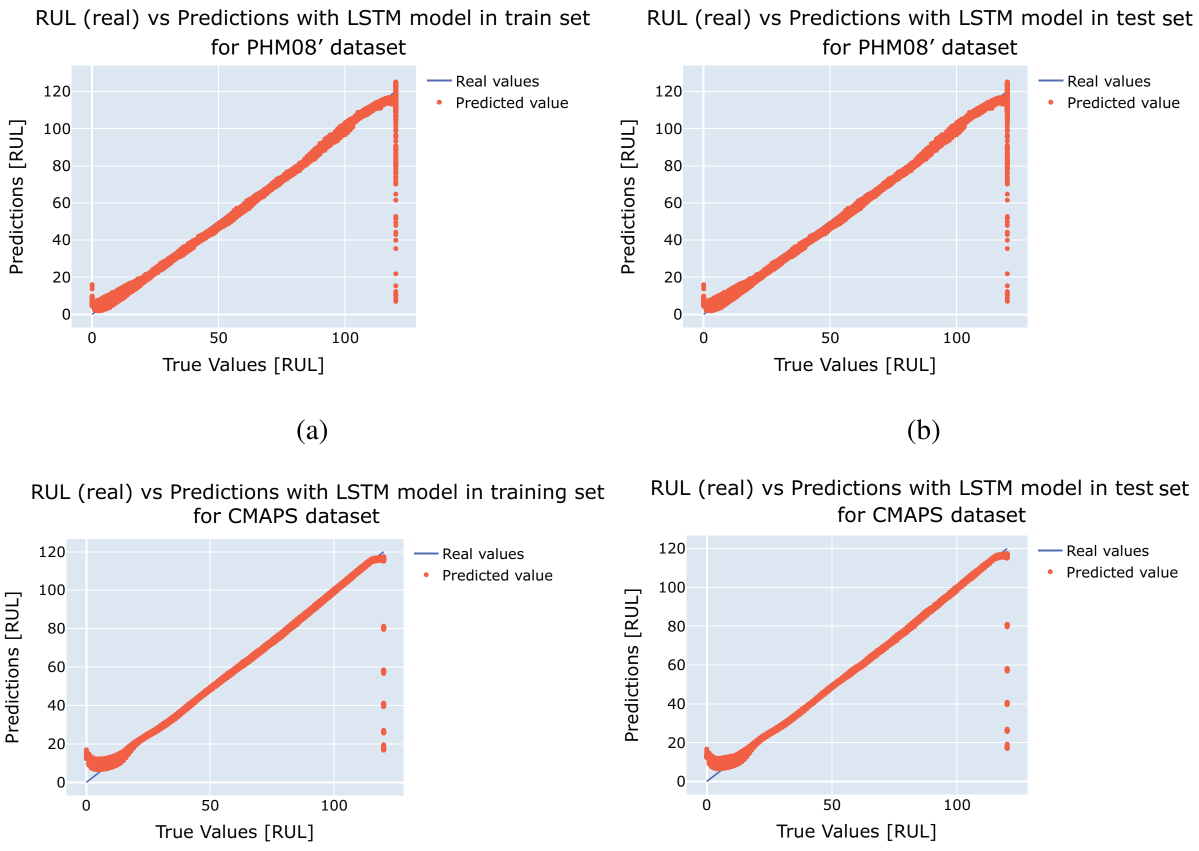


(a) (b)

Figure 5.6: (a) it is the loss function for the PHM008' dataset, and (b) it is the loss function for the CMAPS dataset.

Figure 5.7 shows the behavior for the training and test set for the PHM08' and CMAPS databases, respectively. The test and training set behavior is similar, which indicates that the model can effectively predict the test data, and overfitting was correctly avoided in the model's training. The blue line corresponds to the actual *RUL* values for all engines from cycle 1 to the maximum or final cycle, and the red dots correspond to the RUL predictions for all engines in all cycles. In general, model behavior is right; however, it has significant prediction errors at the end of the cycles, considered outliers.

Additionally, Figure 5.8 shows the distribution of the train set's errors in blue and test set's errors in red for the PHM08' and the CMAPS databases. Error distribution is centered at zero and fits into a normal distribution of the data. Furthermore, the distribution of errors for the test and training set in the two databases has similar behavior, which indicates that the *RMSE* values will be almost the same, which is the predictive model's expectations. The distribution of the errors in the training set and the test presents some outliers. Due to the data distribution, when the model



RUL (real) vs Predictions with LSTM model in train set for PHM08’ dataset

RUL (real) vs Predictions with LSTM model in test set for PHM08’ dataset

RUL (real) vs Predictions with LSTM model in training set for CMAPS dataset

RUL (real) vs Predictions with LSTM model in test set for CMAPS dataset

(a) (b)

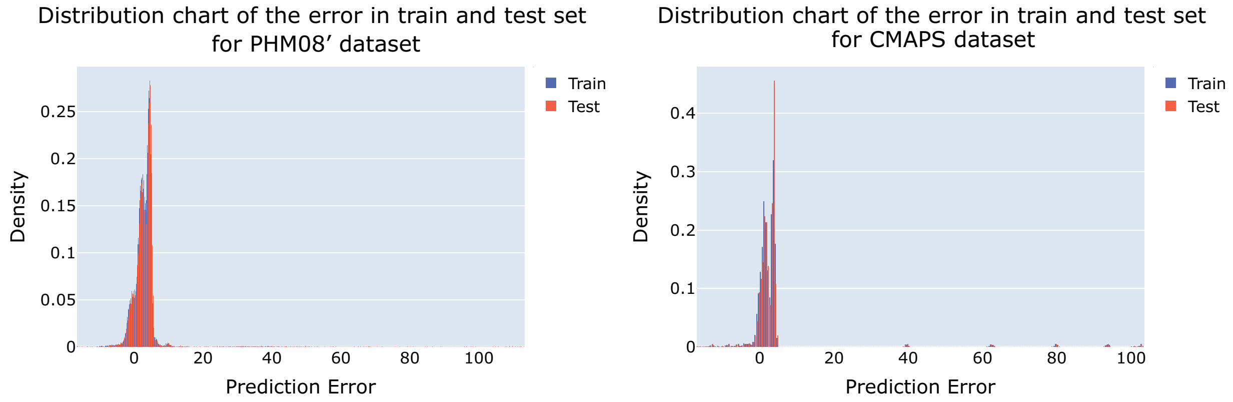
(c) (d)

Figure 5.7: It represents Real Rul vs. Predicted Rul in train and test set for PHM08’ and CMAPS databases.

is passed from one engine to another in training, testing takes one or two cycles to the model to identify that change. These outliers are later discarded for the analysis of the behavior of the model for each engine.

5.3 *LSTM* Neural network performance

As mentioned above, to ensure that the model’s performance does not depend on the partitioning of the data in training and testing, k-fold cross-validation of the data is performed. In this case, for each k-value, the *RMSE* of the model is obtained in the training and test set. The mean and standard deviation of the validations serve to measure the model’s performance using different partitions for training and testing. Table 5.6 shows the *RMSE* and standard deviation values for the two databases. These values result from making k-fold cross-validation with four folds in each



(a) (b)

Figure 5.8: (a) it is the error distribution for PHM008' dataset and (b) it is the error distribution for CMAPS dataset.

run; that is, each *RMSE* value in the table corresponds to the mean of the *RMSE* in the four-folds. Finally, the average *RMSE* and standard deviation for all tests are shown. The *RMSE* and standard deviation values of the CMAPS database are slightly higher than those of the PHM08 database; this is mainly due to the number of engines in each database, 100 and 218 engines, respectively. However, the values obtained are below compared with the values from other research works. It indicates that the model is validated for its use in the two different databases, obtaining satisfactory results. Finally, the validation results indicate that the motors' data do not depend on the partition that we have of the data.

Table 5.6: RMSE and standard deviation values for the PHM08', and the CMAPS dataset

Train set				**Test set**			
PHM08'		**CMAPS**		**PHM08'**		**CMAPS**	
RMSE	**Std**	**RMSE**	**Std**	**RMSE**	**Std**	**RMSE**	**Std**
3,55	0,29	8,80	0,96	3,59	0,32	8,68	0,96
3,84	0,67	5,72	3,51	3,86	0,70	5,60	3,41
3,94	0,58	5,36	2,97	4,01	0,91	5,36	3,20
3,15	0,65	6,89	2,38	3,11	0,42	5,39	2,73
3,29	1,12	6,15	3,20	3,76	0,89	5,91	3,25
3,37	0,25	8,59	1,98	3,85	0,21	8,42	2,58
3,85	0,38	7,95	1,06	3,29	0,44	7,69	0,92
3,12	0,44	5,89	2,94	3,19	0,55	5,86	2,83
3,66	0,20	9,62	2,6	3,72	0,19	9,55	2,58
3,59	0,94	5,98	3,25	3,66	0,97	5,96	3,28
3,54	**0,55**	**7,10**	**2,49**	**3,60**	**0,56**	**6,842**	**2,57**

The *RMSE* results are a measure of the difference that exists between the real values and the predicted values. This value gives more weight to large errors and is in the same units as the model's target. For the PHM08' dataset, *RMSE* and standard deviation values are 3.54 and 0.55 for the train set and 3.60 and 0.56 for the test set. For the CMAPS dataset, *RMSE* and standard deviation values are 7.10 and 2.49 for the train set and 6.84 and 2.57 for the test sets. These *RMSE* values indicate that the difference between the actual and predicted remaining cycles is approximately 4 for the PHM08 database and 7 for the CMAPS database.

5.4 Discussion

So far, there are predictions for each cycle of all engines. However, for KPI's measurements, it is necessary to establish a threshold at a single point in the cycle. When developing the model, it is necessary to create an alert to observe how many cycles are left until a maintenance stop must be made. Thresholds of 50, 60, and 70 cycles before the engine failure are used; this number will depend on the company's requirements where the failure prediction system is applied by estimating *RUL*. Table 5.7 shows the *RMSE* and Score function values with their respective standard deviations for the two databases. *RMSE* and score function results are satisfactory for the three thresholds. Results variations for the score function are due to its algorithm. This function was developed for the competition in 2008, and the authors [75] suggest this function can be improved. However, lower values are observed than those found in the literature.

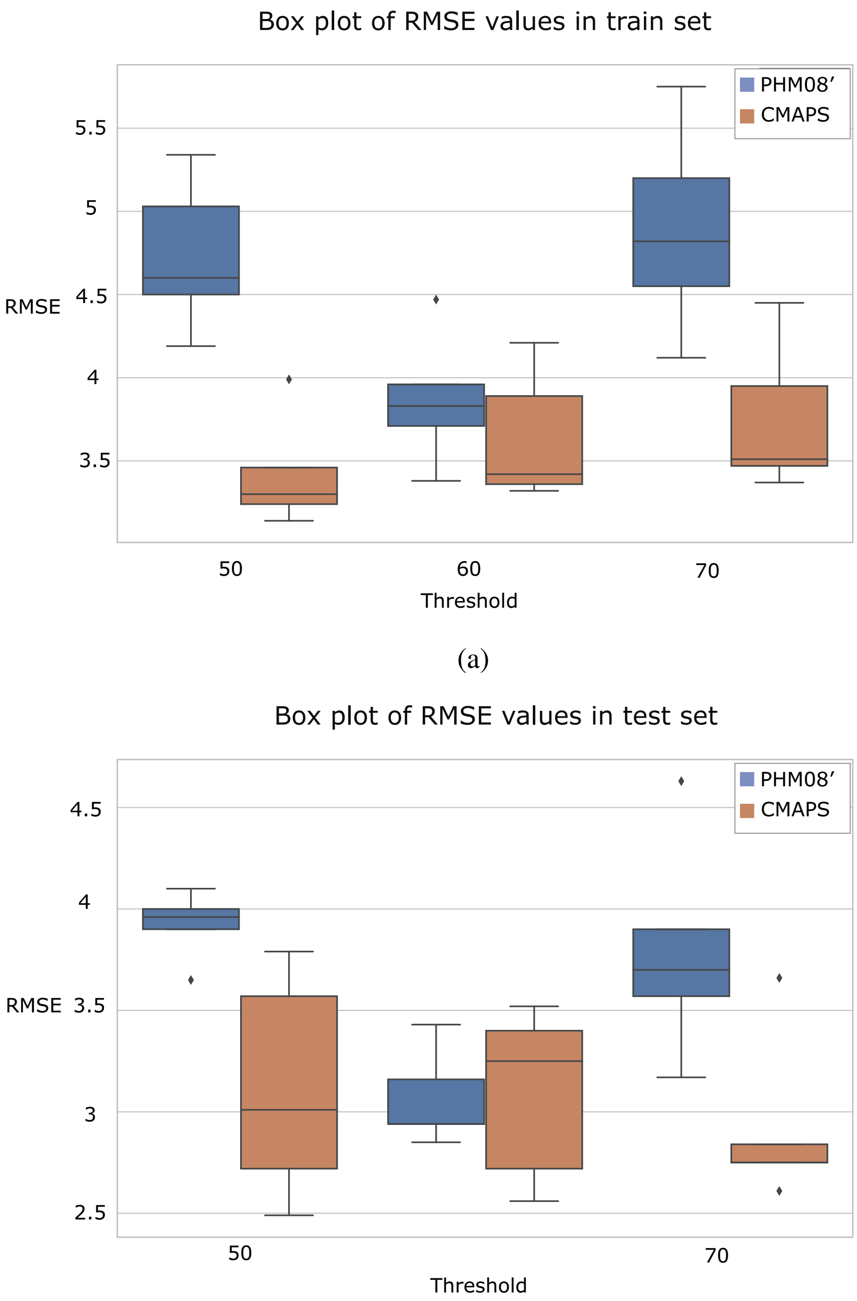
Figure 5.9 shows the box plot for *RMSE* values. Also, the *RMSE* values' variation is more significant for the CMAPS database; however, the values obtained are still below the values found in the literature. In general, the *RMSE* values in the test set are slightly higher than in the training set; however, the behavior is conservative. In the train and test set, *RMSE* values present better performance for the thresholds of 50 and 60 cycles compared with the threshold of 70. Regarding the *RMSE* values, the threshold that offers the best results is 60 cycles. Figure 5.10 shows the box plot of the Score function. In the Score function, the data is more extensive for the PHM08' database because it has more engines than the CMAPS dataset. In general, the Score function values are higher in the training set. It is also because the percentage of data used in training is higher than in the test set (70-30%). For both databases, better Score function results are obtained at the 60 and 70 cycle threshold than at the 50 cycle threshold.

For most of the data, the means are in quartile 2, with a low value of *RMSE* and Score function, which is sought in these two metrics. The standard deviation of data is small for the three thresholds; however, the best results for RMSE and Score function are obtained with a threshold of 60 for both databases.

Table 5.7: RMSE and Score function values for threshold 50, 60, and 70 for PHM08' and CMAPS dataset

Train set								**Test set**							
PHM08'				**CMAPS**				**PHM08'**				**CMAPS**			
RMSE	**Std**	**Score**	**Std**	**RMSE**	**Std**	**Score**	**Std**	**RMSE**	**Std**	**Score**	**Std**	**RMSE**	**Std**	**Score**	**Std**
THRESHOLD 50								**THRESHOLD 50**							
4,00	1,51	43,14	21,32	3,57	1,52	20,13	11,33	4,50	1,77	33,22	21,35	3,99	1,67	9,89	5,01
3,65	0,90	37,17	8,53	3,79	2,07	23,04	16,03	4,60	3,55	45,67	59,38	3,3	1,98	8,51	7,19
3,96	0,83	43,24	12,84	2,72	0,59	12,85	4,32	4,19	1,35	30,95	12,65	3,46	1,68	8,47	4,61
3,90	1,37	39,81	14,40	2,49	0,81	12,23	6,77	5,03	2,10	40,64	28,63	3,24	0,51	7,26	1,76
4,10	1,36	44,35	13,34	3,01	1,69	15,18	11,45	5,34	2,44	49,31	36,76	3,14	1,44	7,47	4,57
3,92	**1,19**	**41,54**	**14,09**	**3,12**	**1,34**	**16,69**	**9,98**	**4,73**	**2,24**	**39,96**	**31,75**	**3,426**	**1,46**	**8,32**	**4,628**
THRESHOLD 60								**THRESHOLD 60**							
3,16	1,41	34,05	22,26	2,72	0,89	14,13	6,19	3,96	1,96	30,09	20,13	4,21	1,78	10,67	5,46
2,85	0,39	29,27	4,69	3,52	1,62	21,19	13,08	3,71	1,74	28,79	23,96	3,42	1,38	8,06	4,07
2,94	0,46	30,19	3,64	3,40	1,18	17,16	8,13	3,38	0,88	24,11	12,35	3,89	1,84	9,84	4,96
2,94	0,86	28,24	6,49	3,25	1,62	17,96	11,20	3,83	1,02	28,16	15,83	3,36	1,15	7,79	3,12
3,43	1,18	36,09	8,99	2,56	1,1	13,75	7,67	4,47	1,50	37,03	21,63	3,32	0,67	7,75	2,56
3,06	**0,86**	**31,57**	**9,21**	**3,09**	**1,29**	**16,84**	**9,25**	**3,87**	**1,42**	**29,64**	**18,78**	**3,64**	**1,36**	**8,822**	**4,03**
THRESHOLD 70								**THRESHOLD 70**							
3,90	1,38	43,57	23,08	2,84	1,05	15,06	6,71	4,82	1,78	35,82	20,01	4,45	1,87	11,40	5,77
3,70	0,67	37,71	10,25	2,75	0,5	13,83	3,58	4,55	2,23	36,22	31,51	3,47	1,81	8,47	5,36
3,57	0,75	35,70	9,91	3,66	2,6	23,39	23,44	5,20	2,24	39,93	21,98	3,37	1,46	7,98	4,15
3,17	0,6	31,70	6,07	2,75	1,3	14,46	7,89	4,12	2,14	31,59	20,18	3,95	2,01	10,07	5,57
4,63	1,84	52,46	26,59	2,61	1,02	13,71	7,22	5,75	4,03	60,96	65,80	3,51	0,62	8,29	1,84
3,79	**1,05**	**40,23**	**15,18**	**2,92**	**1,28**	**16,09**	**9,77**	**4,89**	**2,48**	**40,90**	**31,90**	**3,75**	**1,55**	**9,24**	**4,54**

Finally, Figure 5.11 shows the real *RUL's* behavior and the predicted *RUL* for the three thresholds used in one test engine. In the case of the three thresholds, the model's behavior is conservative; even though for the threshold of 60, the value of the predicted *RUL* is above the real value, the difference between these two values is minimal. It is considered that the model will always predict values that are below the real RUL value; this behavior is the desired one because a model that makes predictions of life more significant than the real ones is not ideal for real-life applications.



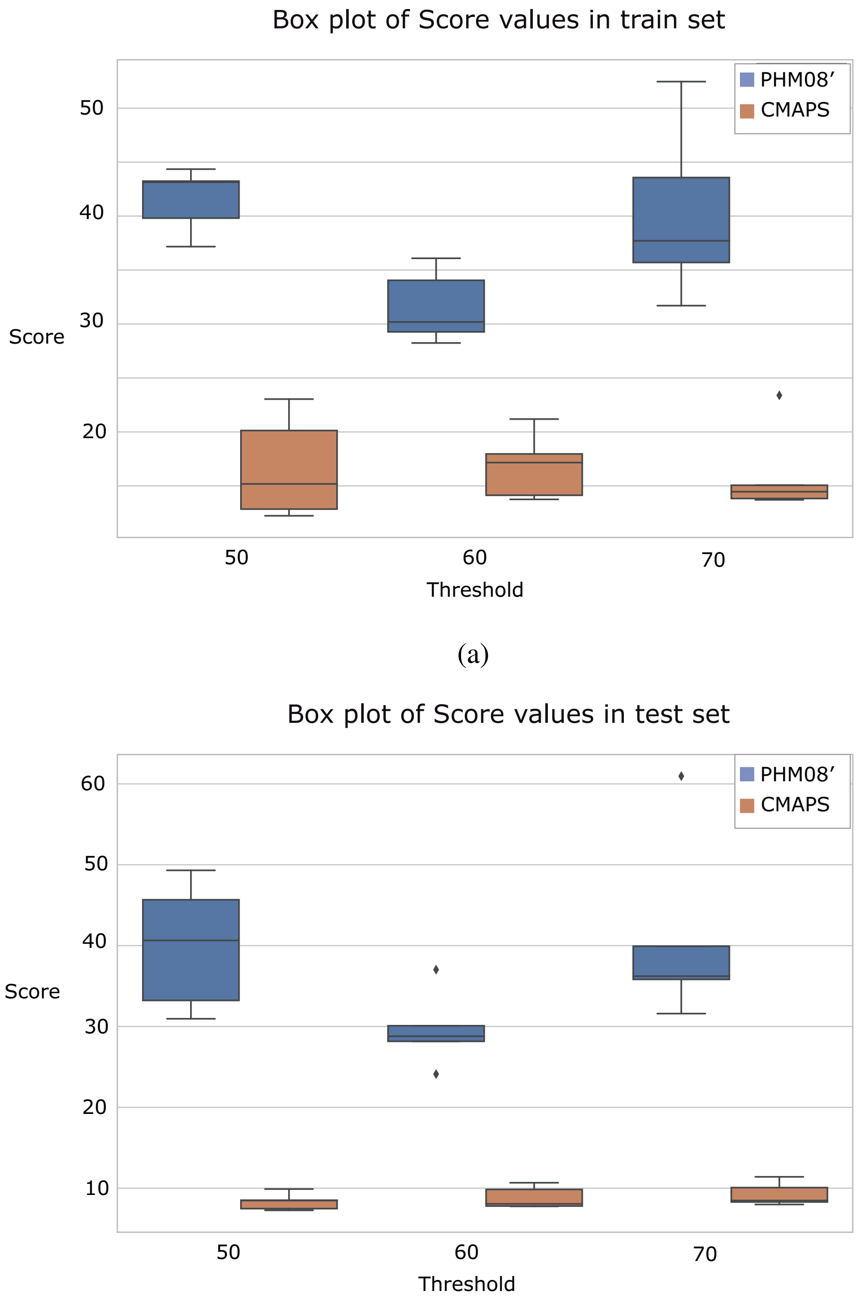
(a)

(b)

Figure 5.9: ((a) it is the box plot in the train set for the RMSE, and (b) it is the box plot in the test set for for the RMSE.

5.5 Comparison

In table 2.1, the authors tend to use hybrid *DL* models, which use LSTM networks combined with other structures such as *CNN*. The models used are complex models that combine different *DL*

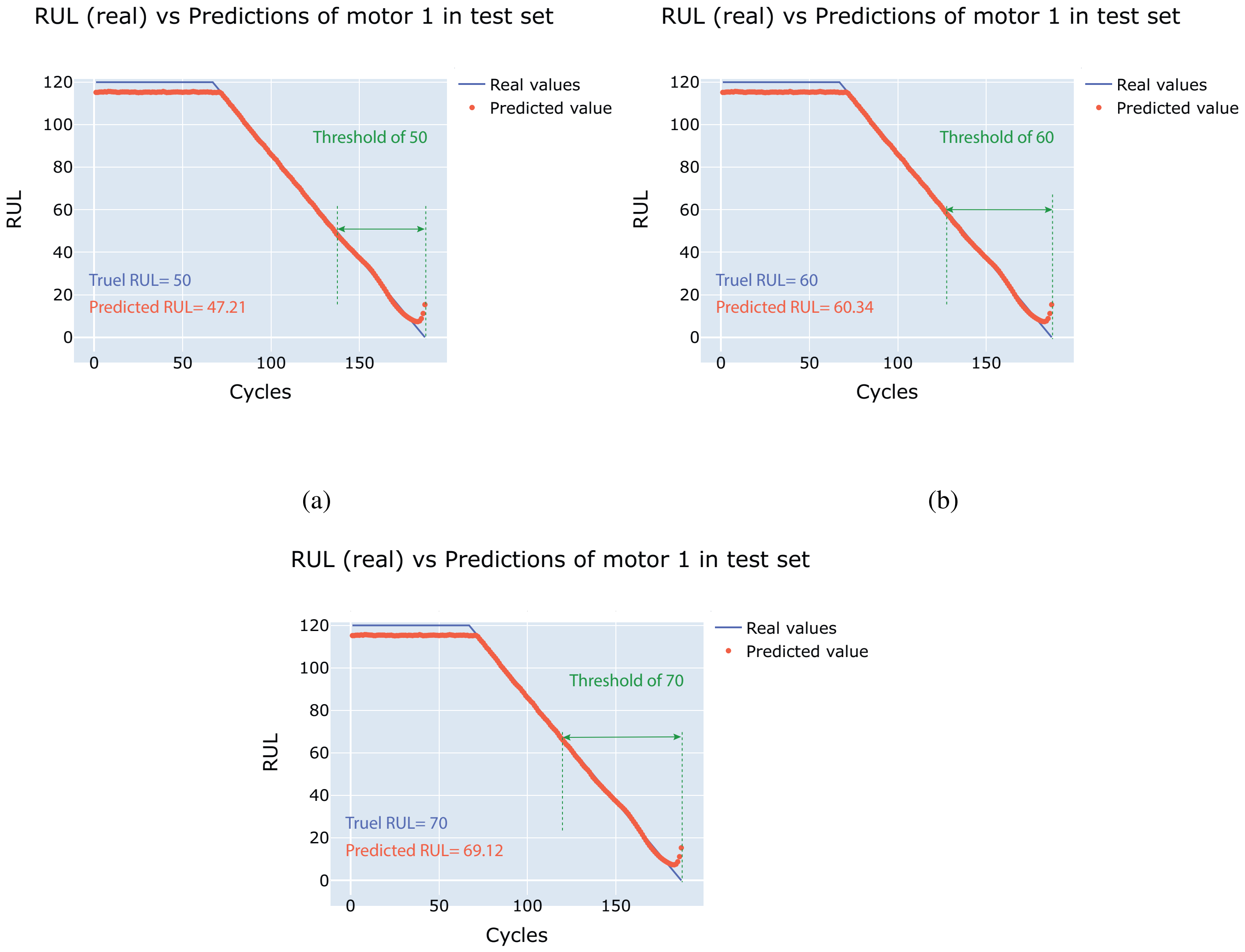


(a)

(b)

Figure 5.10: (a) it is the box plot in the test set for the Score function, and (b) it is the box plot in the test set for the score function.

techniques in the model and techniques for data preprocessing and extraction of characteristics, making it challenging when implementing these models at an industrial level. Table 5.8 compares the proposed methodology and other methodologies used by different authors with different *DL*



(a) (b)

(c)

Figure 5.11: (a) it is Real RUL vs Predicted RUL for a threshold of 50 in motor 1, (b) it is Real RUL vs Predicted RUL for a threshold of 60 in motor 1 and (c) it is Real RUL vs Predicted RUL for a threshold of 70 in motor 1.

approaches.

The authors who carry out the different investigations and the proposed method do so in similar contexts, except for the work by [41] in 2008, where it is considered that the available computing capacity was much lower than that currently available. The proposed methodology comparison is made with recent works, from the year 2018 to 2020. Most of the authors use an experimental platform with a windows operating system with Intel Core i5 to i7, 1.60-GHz CPU and 4-8GB RAM,

Table 5.8: Comparison of the proposed methodology with other authors

Author	**Year**	**Method**	**Training time (s)**	**RMSE**	**Score**	**Dataset**
[41]	2008	RNN	N/A	14,99	740,31	PHM08
[51]	2018	Extreme Learning Machine with TW	5,04	13,78	267,31	FD001
[50]	2018	LightGBM- Decision trees TW	50	13,45	250,4	FD001
[48]	2018	Bi-LSTM	N/A	15,42	N/A	FD001
[49]	2019	Residual CNN	N/A	12,16	212,48	FD001
[46]	2020	CNN-BLSTM	200	12,66	304,29	FD001
Proposed Method	***2020***	***LSTM***	***2137***	***3.87***	***29.64***	***PHM08'***
			938	***3.57***	***8.63***	***FD001***

others use Ubuntu Linux with GTX 1080 GPU. This gives an indication that the comparison of the models is being carried out under equal conditions, although they cannot be replicated because there is not enough information.

In [51], they use an Extreme learning algorithm to model the relationship between time-series data and *RUL*, combined with a moving window of time that serves to observe the data's history. This methodology's main advantage is its short processing time and the use of a time-window for sampling the sensor values to integrate the historical information; in this work, the authors use raw data without preprocessing. The results presented in this research reach an *RMSE* value of $13, 78$, and a Score value of 267.31. This is a data-driven approach focused on Deep Learning techniques. Another approach is [50]; authors use *LightGBM- Decision trees*, which works well with high-dimensional inputs and easy to interpret. The methodology's main contribution to the literature is the rapid training time of the model; a time-window is used to observe the data. However, the authors only validate the methodology with FD001 dataset. The results presented in this research reach an RMSE value of $13, 45$, and a Score value of 250.4. This is a data-driven approach focused on Deep Learning techniques

Authors in [49] have shown the efficiency of using *CNN* in the prediction of failures in this type of system. The residual *CNN* applies a residual block, which skips several convolutional layers by using shortcut connections and overcoming vanishing problems. Also, the authors used a k-fold ensemble method that better integrates the cross-validation. However, this methodology has the limitation that the model's tuning time is high because it is done by trial and error. The authors also obtain good results with the FD001 dataset. Still, when they wanted to validate this methodology with the FD004 dataset, the *RMSE* and standard deviation results were not satisfactory. That is, this methodology is valid only for one operational condition and one failure mode. The results obtained were an *RMSE* of 12.16 and a Score value of 212.48; however, they do not report model training times. This is a data-driven approach focused on Deep Learning techniques

On the other hand, the authors in [48] have used an *LSTM* neural network translating the raw sensor data to an interpretable health index to describe the system health better and then track the historical system degradation for accurate prediction of future health conditions. However, they only used the FD001 dataset to validate this model. The results obtained from *RMSE* are 15.42, reducing the prediction error by over two cycles; however, they do not report Score function values or training times. This is a data-driven approach focused on Deep Learning techniques. In [46], a combination of *Bi-LSTM* and *CNN* networks is used to predict the system's RUL values. The authors used multiple *CNN-BLSTM* based models with different time window sizes to learn several temporal dependencies between features, which expands the time window size and reduces the training error compared to other time window approaches. However, even though this method seems more robust, this methodology was validated only with the FD001 dataset for one operational condition and one failure mode. The results obtained from RMSE were 12.66, and the Score values were 304.29 with a training time of 200 seconds. This is a data-driven approach focused on Deep Learning techniques

The research works mentioned above have in common that they use only the FD001 dataset of the CMAPS database. Even though all of them outperform the results they were compared, they are not general methodologies; they only work under specific flight conditions and one failure mode. Contrary to other authors, [41] used the PHM08 database, one of the winners of the competition reaching RMSE values of 14.99 and Score values of 740.31 without reporting training times. This database presented two different failure modes and operational conditions that resulted in 6 operating regimes; this author originally proposed a linear model by parts for the system's degradation and used a simple *RNN* network to develop the methodology proposed in his research work. Also, he finally used a combination of three different models to estimate his Score results function. This is a data-driven approach focused on Machine Learning techniques

In this case, the proposed methodology has the advantage of being a general model used to analyze the two available databases for turbofan engines' simulations. The above methodologies seem robust and deep but have not reported results for the PHM08 database, except for [41], one of the competition winners. The proposed methodology has the advantage of being a simple model that uses a 4-layer *LSTM* neural network and does not require detailed data preprocessing. It is an advantage considering that this research aims to apply them at an industrial level despite being a research work. The *RMSE* and Score results obtained are 3.87 and 9.64, for the PHM08 database and $.57$ and 8.63 for the CMAPS database, with training times of 2137 and 938 seconds, respectively. It means that, for PHM08 and CMAPS datasets, predictions can be made with an error of approximately four-cycle, conservatively.

The proposed methodology has two main types of limitations, the data, and the hyperparameters. First, the system's data is a simulation that represents a restriction related to the type of faults present. In the simulation, failures due to degradation in time are considered, but stochastic failures are not. However, with real data, the model could be adjusted for a combination of wear and stochastic failure. Regarding the simulated data, the other limitation is on the system degradation model, which, although it is not linear, does not adjust to what the degradation of a real process would be. The second limitation of the method is the model and its hyperparameters. The selected hyperparameters for this methodology are trial and error; this means that it was not possible to cover all the hyperparameters' values to optimize them and obtain the best resulting combination. Despite the results obtained exceeding the literature's data, there is no certainty that the hyperparameter combination used was the most optimal.

The proposed methodology is focused on aircraft turbine failures; therefore, the model's training time is not a factor that considerably affects the model if it were to be applied at an industrial level. This methodology aims to monitor the behavior of the system and predict when the aircraft should be stopped for maintenance of the turbines with times that allow the completion of flights, if necessary, for which time is not a determining factor. For the proposed method, RUL estimation is related to degradation over time. However, the model could be improved with real data to predict the remaining cycles to stop the engine for service and detect an anomaly in the sensor's measurements and use it as additional information.

The results shown in this chapter are the most relevant of the research work; then, in chapter 6, the conclusions, contributions, and possible future work are also shown. The appendices with additional information are included in this work.

Chapter 6

Conclusions

Models that use data-driven approaches with *DL* techniques have high computational efficiency and a high degree of precision when working with signals in the time domain without requiring an extensive pre-processing or data adaptation stage. Furthermore, the development of general models for different systems is feasible, representing an advantage if you want to take this type of development to an industrial level. Currently, most companies have integrated systems that serve to collect data from the sensors. The *DL* models offer the possibility of taking advantage of this type of data to the maximum efficiency without higher computational cost than the data analysis that can be performed through these models in the industry.

The methodology developed in this work using *DL* techniques, specifically *LSTM* neural networks, presents a high degree of efficiency when predicting *RUL* values in turbofan engines subjected to different operating conditions. The highest error in terms of RMSE was 9.50, with a standard deviation of 2.55. In terms of the Score function, they were 60.96 with a standard deviation of 65.8; These values demonstrate the model's better performance compared to the research studies found so far.

6.1 Contributions

This work's main contribution is a detailed methodology to estimate the *RUL* of a fleet of turbofan engines. The methodology uses two datasets for its validation, and it can be used for engines with different operating conditions and faults in the system. The specific contributions are described below:

1. A new methodology using *LSTM* neural networks and time-domain signals is presented.

This methodology allows the *RUL* estimation with an average *RMSE* value of 3.87 and an average score function of 29.64 for PHM08' dataset and an average RMSE value of 3.64 and an average score function of 8.82 for CMAPS dataset. It is worth mentioning that with this methodology, the model estimates approximately 10 hours before the system fails. This methodology also works efficiently for different thresholds in which you want to alert that the personnel must repair the system, an industrial level, it is of great value.

2. A detailed test-based study of how hyperparameters affect *LSTM* networks and it best configuration of the network to work with different data domains of turbofan engines, with different modes of operation and different types of failure with time-domain sensor data without the need for extensive preprocessing of the data for use as input to the network.

3. The proposed model has an excellent capacity for generalization or adaptation to different data; even when training with a smaller number of engines, the system can be considered conservative since the *RUL* predictions are below the real values in most cases. Furthermore, reduced computational resources indicate that the developed methodology has sufficient potential to be considered in real-time industrial applications.

6.2 Future work

The present work demonstrated the excellent performance of *LSTM* neural networks in *RUL* estimation of turbofan engine degradation. Even so, there are some points to keep in mind for future work.

- Apply the methodology developed in this work in real-time processes with real data. It would help to understand better what happens in a real process and what kind of problems can arise when applying these models in a system. It to be able to apply this methodology at an industrial level.

- Develop a model in which it is not necessary to pre-process the data so that the neural network can learn the characteristics of the sensors with raw data; this could be achieved by implementing a hybrid model, such as *CNN* or *AE*.

- The failure predictions through RUL that are made in this work assume a system's behavior since there is not enough information about the system; this has the advantage that the model is developed to work with as little information as possible. However, As future work, a more detailed model could be made in which the failure can be detected, diagnosed, and predicted in a more precise way in the system if there is more information about it.

- This methodology is developed using supervised learning techniques; as future work, the field of unsupervised learning could be explored for the diagnosis and prediction of failures using *RUL*. However, it should be taken into account that this type of learning is computationally more expensive and is less accurate and reliable than supervised learning

Bibliography

[1] J Wang, Y Ma, L Zhang, R.X. Gao, and D Wu. Deep Learning for Smart Manufacturing: Methods and Applications. *J of Manufacturing Systems*, 48:144–156, 2018.

[2] Ashok Prajapati, James Bechtel, and Subramaniam Ganesan. Condition Based Maintenance: A Survey. *Journal of Quality in Maintenance Engineering*, 18(4):384–400, 2012.

[3] B Liu, Z Liang, A Kumar Parlikad, M Xie, and W Kuo. Condition-based Maintenance for Systems with Aging and Cumulative Damage Based on Proportional Hazards Model. *Reliability Eng and Sys Safety*, 168:200–209, dec 2017.

[4] Vashisth S., A. Linden, J. Hare, and P. Krensky. Hype Cycle for Data Science and Machine Learning, 2019. *Gartners Inc., Stamford, US*, 2019.

[5] K. Zhong, M. Han, and B. Han. Data-driven Based Fault Prognosis for Industrial Systems: A Concise Overview. *Journal of Automatica Sinica*, 7(2):330–345, 2020.

[6] Alberto Diez-Olivan, Javier Del Ser, Diego Galar, and Basilio Sierra. Data Fusion and Machine Learning for Industrial Prognosis: Trends and Perspectives Towards Industry 4.0. *Information Fusion*, 50:92 – 111, 2019.

[7] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation – A Review on the Statistical Data Driven Approaches. *European Jrnl of Operational Research*, 213(1):1 – 14, 2011.

[8] Van Tran and Bo-Suk Yang. Machine Fault Diagnosis and Prognosis: The State of The Art. *The International Journal of Fluid Machinery and Systems*, 2:61–71, 03 2009.

[9] Sankavaram C.and B. Pattipati, A. Kodali, K. Pattipati, M. Azam, S. Kumar, and M. Pecht. Model-Based and Data-Driven Prognosis of Automotive and Electronic Systems. In *IEEE Int Conf on Auto Science and Eng*, pages 96–101, Aug 2009.

[10] Vachtsevanos G., FL. Lewis, M. Roemer, A. Hess, and B. Wu. *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. John Wiley and Sons, Inc., 2006.

[11] Hack-Eun Kim, Andy C.C. Tan, Joseph Mathew, and Byeong-Keun Choi. Bearing Fault Prognosis Based on Health State Probability Estimation. *Expert Sys with App*, 39(5):5200 – 5213, 2012.

[12] E. Ramasso and R. Gouriveau. Prognostics in Switching Systems: Evidential Markovian Classification of Real-Time Neuro-Fuzzy Predictions. In *Prognostics and System Health Management Conf*, pages 1–10. IEEE, 2010.

[13] Q. Wu, K. Ding, and B. Huang. Approach for Fault Prognosis Using Recurrent Neural Network. *J of Intelligent Manufacturing*, pages 1–13, 2018.

[14] S. Kiakojoori and K. Khorasani. Dynamic Neural Networks for Jet Engine Degradation Prediction and Prognosis. In *Int Joint Conf on Neural Networks*, pages 2531–2538, 2014.

[15] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning, may 2015.

[16] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X. Gao. Deep Learning and its Applications to Machine Health Monitoring. *Mechanical Systems and Signal Processing*, 115:213 – 237, 2019.

[17] Pingfeng Wang, Byeng D. Youn, and Chao Hu. A generic Probabilistic Framework for Structural Health Prognostics and Uncertainty Management. *Mec Sys and Signal Processing*, 28:622 – 637, 2012. Interdisciplinary and Integration Aspects in Structural Health Monitoring.

[18] André Listou Ellefsen, Emil Bjørlykhaug, Vilmar Æsøy, Sergey Ushakov, and Houxiang Zhang. Remaining Useful Life Predictions for Turbofan Engine Degradation Using Semi-Supervised Deep Architecture. *Rel Eng Syst Saf*, 183:240 – 251, 2019.

[19] Biao Wang, Yaguo Lei, Naipeng Li, and Tao Yan. Deep Separable Convolutional Network for Remaining Useful Life Prediction of Machinery. *Mech Sys and Signal Processing*, 134:106330, 2019.

[20] Chen Lu, Zhen-Ya Wang, Wei-Li Qin, and Jian Ma. Fault Diagnosis of Rotary Machinery Components Using a Stacked Denoising Autoencoder-based Health State Identification. *Signal Processing*, 130:377 – 388, 2017.

[21] X. Ding and Q. He. Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis. *IEEE Transactions on Instrumentation and Measurement*.

[22] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj. Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks. *IEEE Transactions on Industrial Electronics*, 63(11):7067–7075, 2016.

[23] F. Cheng, L. Qu, and W. Qiao. A Case-based Data-driven Prediction Framework for Machine Fault Prognostics. In *2015 IEEE Energy Conversion Congress and Exposition*, pages 3957–3963, 2015.

[24] Z. Zhicai, L. Dongfeng, and S. Xinfa. Research on Combination of Data-driven and Probability-based Prognostics Techniques for Equipments. In *2014 Prognostics and Sys Health Management Conf*, pages 323–326, 2014.

[25] M. D. Anis. Towards Remaining Useful Life Prediction in Rotating Machine Fault Prognosis: An Exponential Degradation Model. In *2018 Condition Monitoring and Diagnosis (CMD)*, pages 1–6, 2018.

[26] C. W. Hong, K. Lee, M. Ko, J. Kim, K. Oh, and K. Hur. Multivariate Time Series Forecasting for Remaining Useful Life of Turbofan Engine Using Deep-Stacked Neural Network and Correlation Analysis. In *2020 IEEE Int Conf on Big Data and Smart Computing*, pages 63–70, 2020.

[27] Chao Hu, Byeng D. Youn, Pingfeng Wang, and Joung Taek Yoon. Ensemble of Data-driven Prognostic Algorithms for Robust Prediction of Remaining Useful Life. *Reliability Eng Sys Safety*, 103:120 – 135, 2012.

[28] Khanh Le Son, Mitra Fouladirad, Anne Barros, Eric Levrat, and Benoît Iung. Remaining Useful Life Estimation Based on Stochastic Deterioration models: A Comparative Study. *Reliability Eng Sys Safety*, 112:165 – 175, 2013.

[29] Xiaobin Li, Jiansheng Qian, and Gai-Ge Wang. Fault Prognostic Based on Hybrid Method of State Judgment and Regression. *Advances in Mec Eng*, 5:149562–149562, 01 2015.

[30] Kwok-Leung Tsui, Nan Chen, Qiang Zhou, Yizhen Hai, and Wenbin Wang. prognostics and health management: A review on data driven approaches. pages 1–17, 05 2015.

[31] J. Wu, K. Hu, Y. Cheng, H. Zhu, X. Shao, and Y. Wang. Data-driven Remaining Useful Life Prediction Via Multiple Sensor Signals and Deep Long Short-Term Memory Neural Network. *ISA Trans*, 2019.

[32] H. Cheng-Geng, H. Hong-Zhong, and L. Yan-Feng. A Bi-Directional LSTM Prognostics Method Under Multiple Operational Conditions. *IEEE Trans on Ind Electronics*, 2019.

[33] G. Lan, Q. Li, and N. Cheng. Remaining Useful Life Estimation of Turbofan Engine Using LSTM Neural Networks. In *2018 IEEE CSAA Guidance, Navigation and Control Conf*, pages 1–5, 2018.

[34] C. Jinglong, J. Hongjie, C. Yuanhong, and L. Qian. Gated Recurrent Unit Based Recurrent Neural Network for Remaining Useful Life Prediction of Nonlinear Deterioration Process. *Rel Eng & Sys Safety*, 185:372–382, 2019.

[35] W. Yu, IIY. Kim, and C. Mechefske. Remaining Useful Life Estimation using a Bidirectional Recurrent Neural Network Based Autoencoder Scheme. *Mech Sys and Signal Processing*, 129:764–780, 2019.

[36] Y. Wu, M. Yuan, S. Dong, Li Lin, and Y. Liu. Remaining Useful Life Estimation of Engineered Systems Using Vanilla LSTM Neural Networks. *Neurocomputing*, 275:167 – 179, 2018.

[37] C. Huang, X. Li, H. Huang, and Y. Li. Fault Prognosis of Engineered Systems: A Deep Learning Perspective. In *Annual Rel and Maintainability Symp*, pages 1–7, Jan 2019.

[38] Z. Chong, L. Pin, QA. Kai, and TK. Chen. Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics. *IEEE Trans on Neural Networks and Learning Systems*, 28(10):2306–2318, 2016.

[39] A. ElSaid, B. Wild, J. Higgins, and T. Desell. Using LSTM Recurrent Neural Networks to Predict Excess Vibration Events in Aircraft Engines. In *IEEE* 12^{th} *Int Conf on e-Science*, pages 260–269. IEEE, 2016.

[40] C. Hsu and J. Jiang. Remaining Useful Life Estimation Using Long Short-Term Memory Deep Learning. In *IEEE Int. Conf. on Applied Sys Invention*, pages 58–61, April 2018.

[41] F. O. Heimes. Recurrent Neural Networks for Remaining Useful Life Estimation. In *2008 Int Conf on PHM*, pages 1–6, 2008.

[42] T. Wang, Jianbo Yu, D. Siegel, and J. Lee. A similarity-based Prognostics Approach for Remaining Useful Life Estimation of Engineered Systems. In *2008 Int Conf on PHM*, pages 1–6, 2008.

[43] K. Akkad and D. He. A Hybrid Deep Learning Based Approach for Remaining Useful Life Estimation. In *2019 IEEE Int Conf on PHM*, pages 1–6, 2019.

[44] Qinglong An, Zhengrui Tao, Xingwei Xu, Mohamed El Mansori, and Ming Chen. A Data-driven Model for Milling Tool Remaining Useful Life Prediction with Convolutional and Stacked LSTM Network. *Measurement*, 154:107461, 2020.

[45] M. Yuan, Y. Wu, and L. Lin. Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network. In *2016 IEEE Int Conf on Aircraft Utility Systems*, pages 135–140, 2016.

[46] Tangbin Xia, Ya Song, Yu Zheng, Ershun Pan, and Lifeng Xi. An Ensemble Framework Based on Convolutional Bi-directional LSTM with Multiple Time Windows for Remaining Useful Life Estimation. *Computers in Industry*, 115:103182, 2020.

[47] Pankaj Malhotra, Vishnu Tv, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Multi-Sensor Prognostics using an Unsupervised Health Index based on LSTM Encoder-Decoder. *1st SIGKDD Workshop on Machine Learning for Prognostics and Health Management*, 08 2016.

[48] Jianjing Z., Peng W., R Yan, and R. X. Gao. Long Short-term Memory for Machine Remaining Life Prediction. *Jrnl of Manufacturing Sys*, 48:78 – 86, 2018. Special Issue on Smart Manufacturing.

[49] L. Wen, Y. Dong, and L. Gao. A New Ensemble Residual Convolutional Neural Network for Remaining Useful Life Estimation. *Mathl Biosc and Eng*, 16:862–880, 01 2019.

[50] F. Li, L. Zhang, B. Chen, D. Gao, Y. Cheng, X. Zhang, Y. Yang, K. Gao, Z. Huang, and J. Peng. A Light Gradient Boosting Machine for Remainning Useful Life Estimation of Aircraft Engines. In *2018 21st Int Confe on Intelligent Transportation Sys*, pages 3562–3567, 2018.

[51] C. Zheng, W. Liu, B. Chen, D. Gao, Y. Cheng, Y. Yang, X. Zhang, S. Li, Z. Huang, and J. Peng. A Data-driven Approach for Remaining Useful Life Prediction of Aircraft Engines. In *2018 21st Int Conf on Intelligent Transportation Sys*, pages 184–189, 2018.

[52] L. Peel. Data Driven Prognostics Using a Kalman Filter Ensemble of Neural Network Models. In *2008 Int Conf on PHM*, pages 1–6, 2008.

[53] Sandip Kumar Singh, Sandeep Kumar, and J.P. Dwivedi. A Novel Soft Computing Method for Engine RUL Prediction. *Multimedia Tools and Applications*, 78, 09 2017.

[54] Jun Wu, Kui Hu, Cheng Yiwei, Ji Wang, Chao Deng, and Yuanhan Wang. Ensemble Recurrent Neural Network-Based Residual Useful Life Prognostics of Aircraft Engines. *Structural Durability Health Monitoring*, 13:317–329, 01 2019.

[55] D. Dong, X. Li, and F. Sun. Life Prediction of Jet Engines Based on LSTM-recurrent Neural Networks. In *2017 Prognostics and Sys Health Management Conf*, pages 1–6, 2017.

[56] A. Ribeiro de Miranda, T. M. G. de Andrade Barbosa, A. G. Scolari Conceição, and S. G. Soares Alcalá. Recurrent Neural Network Based on Statistical Recurrent Unit for Remaining Useful Life Estimation. In *2019 8th Brazilian Conf on Int Sys*, pages 425–430, 2019.

[57] H. Elattar, H. Elminir, and A. el-din Riad. Conception and Implementation of a Data-driven Prognostics Algorithm for Safety–critical Systems. *Soft Computing*, 23, 02 2018.

[58] Oguz Bektas, Jeffrey A. Jones, Shankar Sankararaman, Indranil Roychoudhury, and Kai Goebel. A Neural Network Framework for Similarity-based Prognostics. *MethodsX*, 6:383 – 390, 2019.

[59] N. Gugulothu, Vishnu TV, P. Malhotra, L. Vig, P. Agarwal, and G. Shroff. Predicting Remaining Useful Life using Time Series Embeddings based on Recurrent Neural Networks. *Inte Jornl of PHM*, 9(1), sep 2017.

[60] L. Yongxiang, S. Jianming, W. Gong, and L. Xiaodong. A Data-driven Prognostics Approach for RUL Based on Principle Component and Instance Learning. In *2016 IEEE Int Conf on PHM*, pages 1–7, 2016.

[61] Cheng-Geng Huang, Xiang-Yu Li, Hong-Zhong Huang, and Yan-Feng Li. Fault Prognosis of Engineered Systems: A Deep Learning Perspective. pages 1–7, 01 2019.

[62] X. Zhang, Y. Dong, L. Wen, F. Lu, and W. Li. Remaining Useful Life Estimation Based on a New Convolutional and Recurrent Neural Network. In *2019 IEEE 15th Int Conf on Automation Sci and Eng*, pages 317–322, 2019.

[63] O. Kayode and A. S. Tosun. LiRUL: A Lightweight LSTM Based Model for Remaining Useful Life Estimation at the Edge. In *2019 IEEE 43rd Annual Computer Software and Applications Conference*, volume 2, pages 177–182, 2019.

[64] Various Authors. Turbofan Engine. url https://www.grc.nasa.gov/WWW/K-12/airplane/aturbf.html. accessed 19-06-2020.

[65] J Zambrano Angel, J Barbosa, and P Llanos. Biblioteca Digital Universidad de San Buenaventura Colombia: Model and Desing of a Turbofan Engine JP1 . url http://45.5.172.45/handle/10819/1695. Accessed 2020-06-18.

[66] Y.A. Çengel. *Thermodynamics*. McGraw-Hill Interamericana de España S.L., 2012.

[67] H Richert. *Advanced Control of Turbofan Engines*, volume 1. Springer-Verlag New York, dec 2012.

[68] Angelos T. Kottas, Michail N. Bozoudis, and Michael A. Madas. Turbofan Aero-engine Efficiency Evaluation: An Integrated Approach Using VSBM Two-stage Network DEA. *Omega*, 92:102167, 2020.

[69] Z. S. Spakovszky, J. B. Gertz, O. P. Sharma, J. D. Paduano, A. H. Epstein, and E. M. Greitzer. Influence of Compressor Deterioration on Engine Dynamic Behavior and Transient Stall-Margin. In *Journal of Turbomachinery*, volume 122, pages 477–484. ASME, jul 2000.

[70] Impact of Degradation on the Operational Behaviour of a Stationary Gas Turbine and in Detail on the Associated Compressor. Technical report, 2013.

[71] J Williams. Further Effects of Water Ingestion on Axial Flow Compressors and Aeroengines at Part Speed. In *Proceedings of the ASME Turbo Expo*, volume 6, pages 375–383. American Soc of Mec Eng Digital Collection, aug 2008.

[72] M. Ziya Sogut, Enver Yalcin, and T. Hikmet Karakoc. Assessment of Degradation Effects for an Aircraft Engine Considering Exergy Analysis. *Energy*, 140:1417 – 1426, 2017. Advanced Energy Tec in Aviation.

[73] R. Mishra, M Saravanan, Kavya Srinivasan, and Ll Prakash. Studies On Performance Deterioration of a Low Bypass Turbofan Engine in Service. *2nd National Propulsion Conf*, 68, 11 2016.

[74] H Balaghi Enalou, S Bozhko, M Rashed, and P Kulsangcharoen. A Preliminary Study Into Turbofan Performance with LP-HP Power Exchange. 05 2018.

[75] A. Saxena, K. Goebel, D. Simon, and N. Eklund. Damage Propagation Modeling for Aircraft Engine Run-to-failure Simulation. In *2008 Int Conference on PHM*, pages 1–9, 2008.

[76] L. Peel. Data Driven Prognostics Using a Kalman Filter Ensemble of Neural Network Models. In *2008 Int Conf on PHM*, pages 1–6, 2008.

[77] Chen Xiongzi, Yu Jinsong, Tang Diyin, and Wang Yingxun. Remaining Useful Life Prognostic Estimation for Aircraft Subsystems or Components: A review. In *IEEE 2011 10th Int Conf on Elec Measurement Inst*, volume 2, pages 94–98, 2011.

[78] Jing Yang, Yingqing Guo, and Wanli Zhao. Long short-term Memory Neural Network Based Fault Detection and Isolation for Electro-mechanical Actuators. *Neurocomputing*, 360:85 – 96, 2019.

[79] Georg Helbing and Matthias Ritter. Deep Learning for Fault Detection in Wind Turbines. *Renewable and Sustainable Energy Reviews*, 98:189 – 198, 2018.

[80] R. Zhao, J. Wang, R. Yan, and K. Mao. Machine health monitoring with LSTM networks. In *2016 10th Int Conference on Sensing Technology*, pages 1–6, 2016.

[81] J. Wang, G. Wen, S. Yang, and Y. Liu. Remaining Useful Life Estimation in Prognostics Using Deep Bidirectional LSTM Neural Network. In *2018 Prognostics and Sys Health Management Conf*, pages 1037–1042, 2018.

[82] Tianfeng Chai and R.R. Draxler. Root Mean Square Error or Mean Absolute Error? Arguments Against Avoiding RMSE in the Literature. *Geoscientific Model Development*, 7:1247–1250, 06 2014.

[83] J J Hopfield. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proc of the Nat Academy of Sci*, 79(8):2554–2558, 1982.

[84] W.A. Little. The Existence of Persistent States in the Brain. *Mathematical Biosciences*, 19(1):101 – 120, 1974.

[85] Jeffrey L. Elman. Finding Structure in Time. *Cognitive Science*, 14(2):179 – 211, 1990.

[86] Michael I. Jordan. Chapter 25 - Serial Order: A Parallel Distributed Processing Approach. In John W. Donahoe and V [Packard Dorsel], editors, *Neural-Network Models of Cognition*, volume 121 of *Advances in Psychology*, pages 471 – 495. North-Holland, 1997.

[87] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-term Memory. *Neural computation*, 9:1735–80, 12 1997.

[88] Md Zahangir Alom, Tarek M. Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S. Awwal, and Vijayan K. Asari. The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches. mar 2018.

Appendix A

Acronyms and Variables Descriptions

A.1 Acronyms

Table A.1: Acronyms Definitions

Acronyms	**Description**
AE	Auto Encoder
AI	Artificial Intelligence
BiLSTM	Bidirectional Long-Short Term Memory Neural Networks
CBM	Condition Based Maintenance
CNN	Convolutional Neural Network
ConvLSTM	Convolutional Long Short Term Memory
DBN	Deep Belief Network
DBM	Deep Boltzmann Machine
DL	Deep Learning
DRCNN	Deep Recurrent Convolutional Neural Network
DSCNN	Deep Separable Convolutional Neural Network
FNN	Feed Forward Neural Network
GA	Generic Algorithm
GRU	Gated Recurrent Units
LSTM	Long-Short Term Memory Neural Networks
MAE	Mean Absolute Error
MSE	Mean Squared Error
MHMS	Machine Health Monitoring Systems
ML	Machine Learning
MLP	Multi layer Perceptron
PHM	Prognosis and Health Management
RMSE	Root Mean Squared Error
RRN	Recurrent Neural Network
RUL	Remaining Useful Life
SM	Smart Manufacturing
SVM	Support Vector Machine
SVR	Support Vector Regression

A.2 Variable description

Table A.2: Variables Description

Variable	Description	Units
b_f	Bias	-
c_{t-1}	Long term parameter	-
d	Difference between predicted and real value of RUL	-
f_t	Forget gate	-
h_{t-1}	Short term parameter	-
$\dot{m}_f$	Fuel flow	-
n	Number of samples	-
u	aircraft speed	-
w	Wear of the system	-
$\hat{y}$	Estimation	-
y_i	Real value	-
A	Constant	-
B	Constant that determine acceleration between stress combinations	-
T	Thrust	-
Q_r	Fuel calorific value	-
C_p	Specific heat at constant pressure	$\frac{KJ}{Kg.K}$
C_v	Specific Heat at Constant Volume	$\frac{KJ}{Kg.K}$
P_2	Pressure in 2	KPa
P_1	Pressure in 1	KPa
Q_{in}	Energy investment	KJ
W_{out}	Exit work	KJ
W_{in}	Entrance work	KJ
W_f	Weights	-
γ	Ratio of heat capacity	-
η_i	Ideal efficiency	-
σ	Activation function	-

Appendix B

Results of real vs predicted *RUL*

This appendix shows the actual vs. predicted *RUL* results for other engines used in the case study, so that the different behaviors in the data can be observed.

B.1 Engine 3

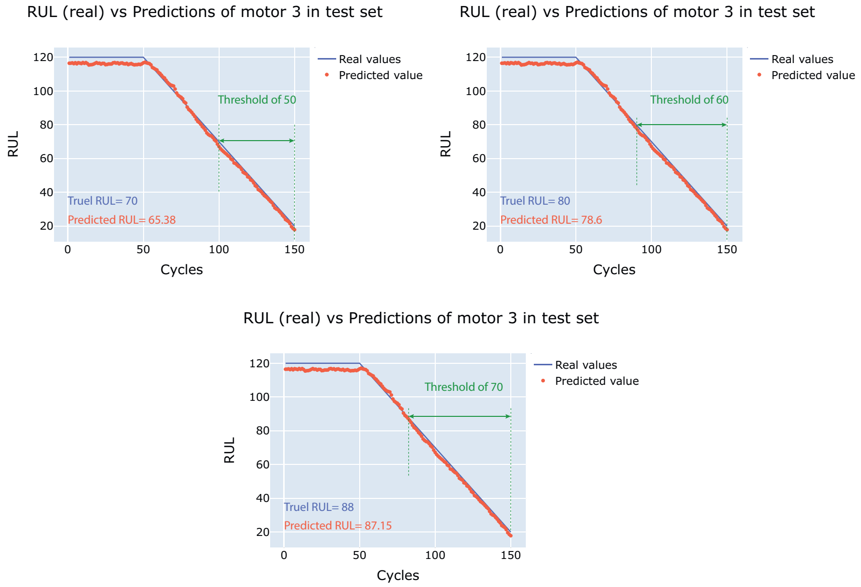


Figure B.1: Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 3 for PHM08' dataset.

B.2 Engine 4

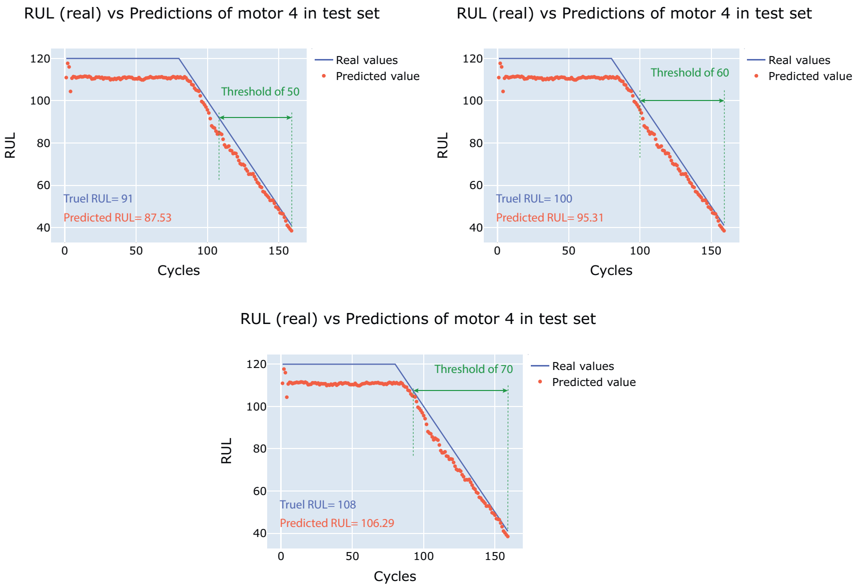


Figure B.2: Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 4 for PHM08' dataset.

B.3 Engine 5

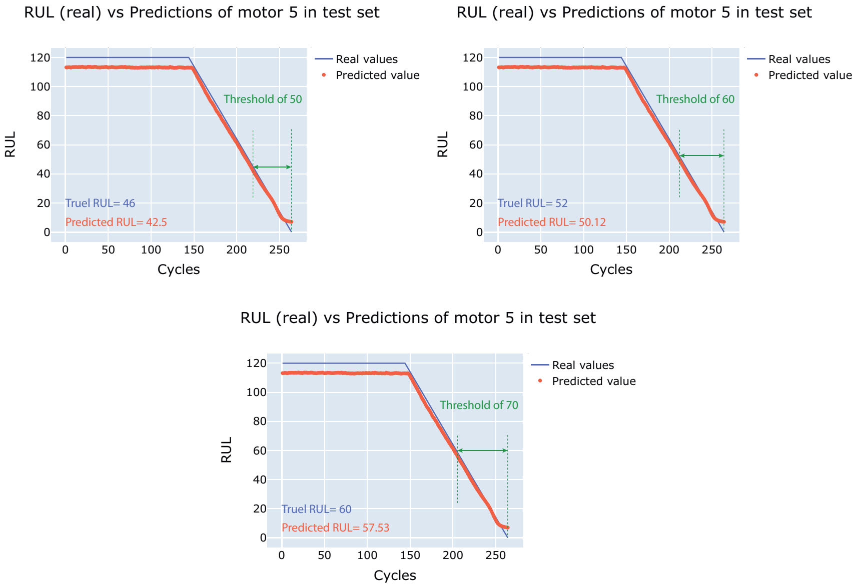
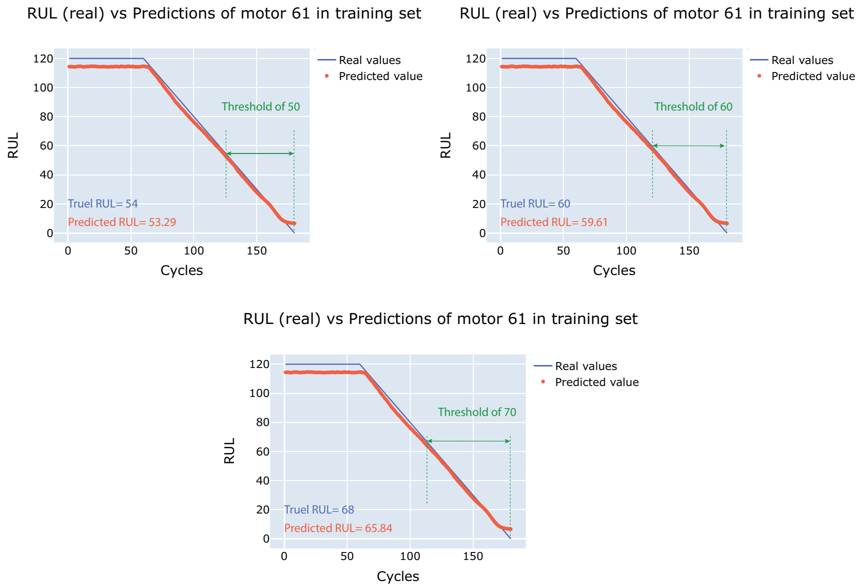


Figure B.3: Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 5 for PHM08' dataset.

B.4 Engine 61



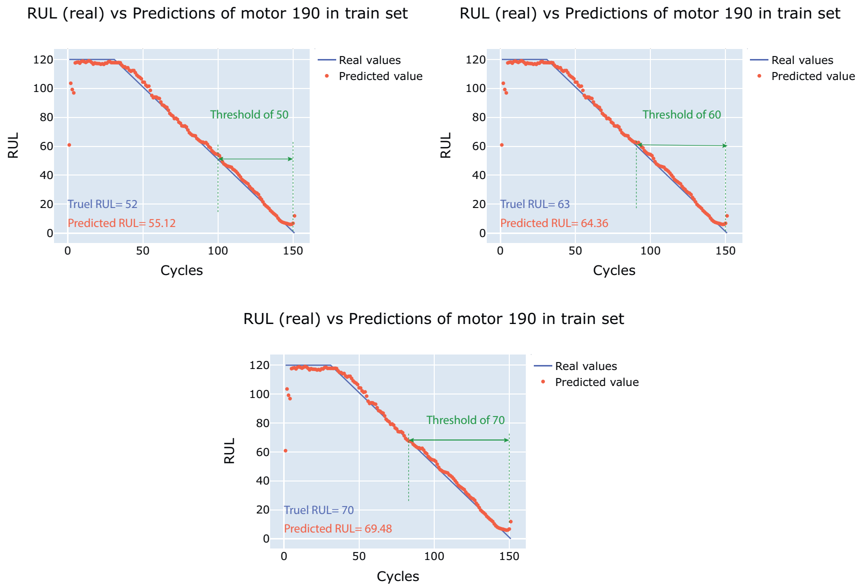
RUL (real) vs Predictions of motor 61 in training set

RUL (real) vs Predictions of motor 61 in training set

RUL (real) vs Predictions of motor 61 in training set

Figure B.4: Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 61 for PHM08' dataset.

B.5 Engine 190



RUL (real) vs Predictions of motor 190 in train set

RUL (real) vs Predictions of motor 190 in train set

RUL (real) vs Predictions of motor 190 in train set

Figure B.5: Real *RUL* vs predicted *RUL* using a threshold of 50, 60, and 70 in motor 190 for PHM08' dataset.

Curriculum Vitae

Luisa Fernanda Montoya Herrera was born in Pereira , Colombia on January 24th, 1994. She received the degree of Mechanical Engineer from the Universidad Tecnológica de Pereira, Pereira, Risaralda, Colombia in December 2017. she was accepted in the Master of Science in Manufacturing Systems program in the Tecnológico de Monterrey, Escuela de Ingeniería y Ciencias, Campus Monterrey.

This document was typed in using LaTeX by Luisa Fernanda Montoya Herrera.