UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MECÁNICA

A FRAMEWORK FOR DEGRADATION START DETECTION AND RUL PROGNOSIS OF PHYSICAL ASSETS

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN MECÁNICA

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MECÁNICO

CRISTIÁN ISMAEL SCHAAD CONCHA

PROFESOR GUÍA:
ENRIQUE LÓPEZ DROGUETT

MIEMBROS DE LA COMISIÓN:
VIVIANA MERUANE NARANJO
RODRIGO PASCUAL JIMÉNEZ

SANTIAGO DE CHILE
2020

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN MECÁNICA
Y AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: CRISTIÁN ISMAEL SCHAAD CONCHA
FECHA: 2020
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

A FRAMEWORK FOR DEGRADATION START DETECTION AND RUL PROGNOSIS OF PHYSICAL ASSETS

Data-driven approaches for Prognostics and Health Management (PHM) of physical assets have increased in popularity in the recent decade due to the large amounts of data produced by the asset sensor systems, the availability to high computational power and the open access to self-learning algorithms, like artificial neural networks (ANN). Among the various PHM objectives is the prediction of remaining useful lifetime (RUL), which with the development of data-driven approaches is causing a paradigm shift from standard corrective and preventive maintenance to a predictive one.

Typical RUL prognostic approaches have been based on the asset's sensor data to perform predictions, as these should correlate to the current health state of the asset. ANNs implemented for these tasks have obtained remarkable results, proving its capabilities on the field. However, a typical problem found in these implementations is that sensor data in the early stages of operation does not show noticeable degradation pattern, resulting in predictions with significant errors in comparison to the true value. This problem has partially been avoided by arbitrarily setting a maximum RUL value that can be predicted or by setting a statistical threshold after which predictions can actually be made. These adaptations have made improvements in RUL accuracy, implying that added considerations and restrictions to the prognostic method can be a subject of study in itself.

Considering this matter, for the present Thesis work is proposed a 2-stage RUL prognostic Framework in which the first stage performs a degradation assessment of the asset with a following stage to make RUL predictions after a certain threshold is crossed. The first stage builds a degradation index based on the Mahalanobis distance (MD) between current operational data and known healthy data, thus quantifying the advance of an on going degradative process, if any. The second stage uses an ANN to make the RUL predictions only after a degradative process is detected. This proposed Framework is tested on both a turbofan engine degradation dataset (C-MAPSS) and a rolling bearing degradation dataset (FEMTO).

The implementation done with proposed Framework proves that healthy data does not provide meaningful information for accurate RUL predictions, and therefore, can be discarded without negative effects on RUL accuracy. Moreover, by starting with RUL predictions only after the degradation threshold is crossed, RUL inaccuracies at early stage of operation are avoided.

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN MECÁNICA
Y AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: CRISTIÁN ISMAEL SCHAAD CONCHA
FECHA: 2020
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

A FRAMEWORK FOR DEGRADATION START DETECTION AND RUL PROGNOSIS OF PHYSICAL ASSETS

Los enfoques basados en datos para el pronóstico y la gestión de la salud (PHM) de activos físicos se han popularizado en la última década debido al alto volumen de datos obtenible por los sensores, la disponibilidad para una significativa capacidad de cómputo y el acceso libre a algoritmos de aprendizaje automático, como las redes neuronales artificiales (ANN). Entre los diversos objetivos dentro del área de PHM, se encuentra la predicción de la vida remanente útil (RUL), que junto con el desarrollo de los enfoques basados en datos, está provocando un cambio de paradigma en las gestiones de mantenimiento, pasando de uno correctivo y preventivo a uno de carácter predictivo.

Típicamente los enfoques para el pronóstico de RUL se han basado en datos sensoriales del activo, ya que estos debiesen correlacionarse con su actual estado de salud. Las ANN implementadas para estas tareas han obtenido resultados notables, demostrando su alta capacidad dentro de este campo. Sin embargo, un problema típico que se encuentra en ellas es que los datos sensoriales en las etapas iniciales, al comienzo de la vida operacional, no muestran un patrón de degradación distinguible; obteniéndose predicciones con errores significativos en comparación al valor de verdad. Este problema se ha evitado parcialmente estableciendo arbitrariamente un valor máximo de RUL que se puede predecir, o estableciendo un umbral estadístico después del cual se pueden realizar predicciones. Estas adaptaciones han mejorado la precisión del RUL estimado, lo que implica que las consideraciones y restricciones adicionales al método de pronóstico pueden ser un tema de estudio en sí mismo.

Considerando este último punto, para el presente trabajo de Tesis, se propone un marco de trabajo para el pronóstico de RUL divido en 2 etapas; la primera realiza una evaluación y cuantificación de la degradación del activo. seguida por una etapa de predicción de RUL. Dentro de esta primera etapa se construye un índice de degradación basado en la distancia de Mahalanobis (MD), la cual toma la distancia entre la distribución de datos de operación actual y la distribución datos en un estado saludable previamente conocido, cuantificando así el avance de un proceso degradativo en caso de estar presente. La segunda etapa del esquema utiliza una ANN para realizar las predicciones de RUL, pero sólo después de que se ha detectado un proceso degradativo en la primera etapa. Este esquema propuesto se pone a prueba en un dataset de degradación de motores de aeronaves tipo turbofan (C-MAPSS) y en un dataset de degradación en rodamientos (FEMTO).

La implementación realizada con este esquema demuestra que los datos saludables no brindan mayor información para dar con predicciones precisas de RUL y, por lo tanto, pueden descartarse sin tener efectos negativos en su exactitud. Por otro lado, comenzar con las predicciones de RUL una vez cruzado el umbral de degradación definido evita las predicciones imprecisas durante la etapa de operación saludable.

To my siblings Tomás, Francisca and Sofía

Acknowledgments

First of all I would like to thank my professor Dr. Enrique Lopez Droguett for his continuous support and trust during this journey. This was a long and bumpy road, but it surely is something I am profoundly proud to have done. It truly has directed my path into a field for which I have intense attraction and incredible curiosity, the field of data science. For this I could not be more grateful.

My total gratefulness go out to my fellow schoolmates with which I shared this journey. The conversations, advices, questions and answers that we have had has taught me the beauty and importance of a community. In no particular order, my thanks are for: Gabriel, Javier, José, Danilo, Iván, Tomás, Pato, Philip, Cristián and Mohammad. To my lifetime friends as well, Andrés, Felipe y Brian, thank you for all this years of adventures.

To all those people that have made this place a University, my thanks. Professors, administration, gym teachers, workers and any classmates with which a conversation or smile was shared.

And finally, to my family, for providing me with a life filled with simple love and love for the simple. Specially to my Mom and Dad, as they have taught me to appreciate life in a way that nobody else could ever explain how. And to my abuela and abuelos, for teaching me to cherish this world with the eyes of a child. Maybe my abuelo that left this world long ago would have been happy to see my as a fellow colleague, a Mechanical Engineer.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Any physical object that is subject to a repetitive use or stress will undergo a degradation process that, given enough time, will render it incapable of performing its original function. This is valid for all things in nature, obviously including man-made objects; from the simplest tools to the most complex and intricate systems. However, despite this seemingly pessimist point of view, maintenance services are exceedingly useful (and in fact, necessary) for reducing the risk of total failure occurrence in any physical asset, slowing down its degradation process, and, therefore, extending the asset's remaining useful life to periods beyond of that it could have been initially expected to achieve.

Said maintenance services can be divided between into two broad categories: corrective and preventive maintenance. Corrective maintenance correspond to the restoration of an asset to a state in which it can perform its required function once a fault is detected and recognised, while on the other hand preventive maintenance is carried out at predetermined intervals to reduce the probability of failure or further degradation. Preventive maintenance is seen as better option between the two, but nevertheless, it can lead to an overcare for the asset, which, in the end, would be a waste of time and resources. In order to remedy this, other forms of preventive maintenance that are based on the monitoring of the actual state of the assets have appeared: condition based maintenance (CBM) and predictive maintenance (PM) [9].

Both CBM and PM services rely on a monitoring of key performance and operational parameters of the asset, like temperature, vibrations, etc., in order to take further maintenance actions based on this information. The difference between these two lies in the way they use the information; CBM uses it to detect anomalies during operation, while for PM it is used to prognosticate when maintenance will be needed. In regard to this predictive approach, it has been the subject of a growing attention because of the numerous benefits expected from it; like reduction in the number of breakdowns and downtime and an the increased reliability on the asset performance. For studying this type of maintenance and their benefits and applicability is the field of Prognostics and Health Management.

1.1 Prognostics and Health Management

The main goal of the Prognostics and Health Management (PHM) field is to estimate the time to failure of an asset (machine or system) and asses the risk for one or more existing and future failure modes [13]. Therefore, it is intended with a PHM approach to project into the future the behaviour of an asset, identifying its health state at every operational instant, in order to finally take the correct decisions; either to continue with the asset operation or take it to maintenance service. To quantify the risk in the monitored asset, different prognostics measures (metrics) are employed, being the most widely used the *Remaining Useful Life* (RUL). The RUL of an asset indicates its remaining time to the next operational failure, that is. the time left until the equipment can't perform its required function in the way that it is expected.

Current approaches for PHM can be categorised between 3 classes: model-based, data-driven and hybrid approaches, which are a combination of the first two [20]. Model-based approaches, also treated as physics-based models, represent the degradation process through an analytic model or by an stochastic process [23]. Data-driven approaches, on the other hand, take routinely generated operating data to diagnose and predict the future outcome (RUL) of the system by using it to train a predictive model. Both approaches have their advantages and disadvantages. Model-based approaches offer an interpretability of the problem, but are only useful when there is prior knowledge on the behaviour and degradation of the system and extremely difficult to apply in complex systems with multiple variables affecting it. Data-driven models main advantage is that they do not require an understanding of the degradation process itself, therefore is a simple solution to employ in complex systems, but because of this, there is little to no interpretability of the model (and parameters) obtained. Nevertheless, due to the ongoing growth on computing power, and because manufactured goods have exponentially increased in complexity, data-driven approaches have taken the leading role within PHM approaches.

And among data-driven approaches for predictive estimation over asset future outcomes, there is a wide variety of techniques and models [34] that can be further divided between statistical/stochastic and Machine Learning (ML) approaches. Statistical algorithms estimate RUL through a probabilistic model based on a probability density function that is fitted with the available data [30]. Wiener process [44], Gamma Process [39] and Markovian based models [4] are examples of statistical data-driven methods used for prognostics. On the other hand, ML algorithms learn by itself to represent the available (*training*) data to produce the desired output. ML algorithms have shown improved performance over conventional approaches in diverse areas of PHM [15], and currently are extensively researched in the field of Computer Science.

1.2 Machine Learning and Deep Learning

Machine Learning (ML) is a field of Computer Science (CS) that has taken great relevance in the last decade mainly due to the explosive increase in computation power and due to the current tendency towards *big-data*, i.e. the production of complex data, in large volumes, and at high speeds. The ML field studies and develops novel algorithms capable of an automatic learning on how to represent raw data by finding underlying patterns in it.

A significant amount of proficient algorithms are present within ML, being the subfield of Deep Learning (DL) on of the most remarkable because it can represent data with a high degree of abstraction, meaning that it represents data over other more simpler representations [45], [17]. Artificial Neural Networks (ANN) are the basic building blocks in DL.

ANN were initially inspired by the neural connection and synapses inside the brain, where sets of neurons activate depending on the activation of other interconnected neurons and so on. Even though ANN were devised in the late 1940's, they started to be increasingly employed in the last two decades thanks to the rise of computational power. New ANN architectures and optimization techniques are being continuously proposed and developed, making of this a very active research field within CS.

Due to the use of non-linear activation functions on each of the neurons in an ANN, these methods are universal function aproximators [8], and because of this, the applications that can be found with these models are tasks of regression and classification, which can be severely complex when multiple dimensions and variables are influencing it. Because regression and classification (i.e. prognostics and diagnosis) are precisely the tasks required to achieve the objectives of PHM, and because these tasks are based on multi-dimensional data, the use of DL offers an option for data-driven approaches to the PHM of physical assets. And even though ANN models require large amounts of data to be trained, it is an appropriate tool to be applied in PHM related tasks as nowadays large sets of data can be collected through simulation, testing and, of course, monitoring.

1.3 Motivation

Deep Learning algorithms have already been successfully implemented in tasks of RUL prediction of physical assets, having an extensive scientific literature that backs their improved performance and accuracy over other prognostic methods. Despite the differences on data processing that these models have, they typically predict RUL values directly from sensor data.

Nevertheless, during the early stages of life of physical assets, sensor data typically does not show noticeable trends or patterns that would indicate the presence of an ongoing degradative process on the equipment. This common occurrence leaves all sort of algorithms, despite their complexity, unable to make accurate predictions. This phenomenon is sometimes overlooked in both industry and research, leaving out room for further improvement on the accuracy and applicability of said RUL prognostics algorithms. An issue like this is extremely necessary to address to further improve the capabilities of predictive maintenance and increase industrial interest on these solutions.

Due to the problem explained above, it is in the motivation of the present thesis work to establish a practical Framework for quantifying the degradation of an asset and performing a corresponding RUL prediction when the quantified degradation surpasses a defined threshold. Therefore, this Framework consist of 2 main sections: the first for identifying if the operating condition is under a degradation process or not, and for the second stage, if the asset is under one, a corresponding regression algorithm is implemented to make the actual RUL predictions.

For the first section of this devised Framework, a single value measure for quantifying degradation is needed. For this, the Mahalanobis distance (MD) seems to be the best candidate, which gives a single dimensional measure on multi-variate systems given by the multiple sensors in use by the asset. The MD is actually a distance measure between the points in the multivariate system, but unlike the Euclidean distance, it considers the distribution of each variable and the correlations between them.

The following stage of the Framework is implemented if the calculated degradation index (MD) surpasses a defined threshold for healthy data. After crossing it, an artificial neural network is used to prognosticate RUL values based on the multi-sensor data of the asset under degradation. The election of the ANN type is open, depending to the specific implementation case and the available data format. Recurrent neural networks (RNN), convolutional neural networks (CNN) and combinations are suitable candidates for this stage.

For the deployment of the proposed Framework, both of its stages need to be trained. The first one, to establish a corresponding threshold, and the second one, to learn the internal parameters of the ANN for processing the sensor data.

1.4 Objectives

According to the motivation behind this Thesis work, the objectives can be divided into a general and specific points to accomplish.

1.4.1 General Objective

The main purpose of this work is to develop a 2-staged Framework for a data-driven Remaining Useful Life (RUL) prognosis that first calculates a degradation index for the asset and then performs RUL predictions after a certain degradation threshold is crossed.

1.4.2 Specific Objectives

The following are the specific points necessary to achieve the main objective:

- Development of a degradation index based on the Mahalanobis distance (MD) from the multi-sensor system of an asset.
- Develop an unsupervised algorithm for healthy data space definition and the degradation starting point detection through confidence intervals.
- Test the proposed Framework on a simulated case, C-MAPSS, a turbo fan degradation dataset and compare the implementation without the degradation assessment stage.
- Validate the proposed Framework on an experimental case: FEMTO, a ball bearing degradation dataset.
- Conclude about the advantages and disadvantages of the proposed Framework and discuss about its practical applicability.

1.5 Statement and Thesis Scope

The scope for this thesis work is to develop a 2-staged Framework for a data-driven approach in the prognostics of Remaining Useful Life of physical assets. The validation of said framework will be done in two steps, first tested on simulated dataset (C-MAPSS), and then, validated on an experimental dataset (FEMTO).

This complete Framework is computational developed using the `python` language and several open source libraries. All the codes developed for this work will be publicly available at `https://github.com/Cris-Schaad/MD-degradation-detection-for-RUL-prognostics`.

For the C-MAPSS testing, different criterias for defining the MD threshold are going to be analyzed. From this is expected to obtain a general knowledge on how to define the threshold. The FEMTO case study is for validating the Framework and for considering a case when a different type of ANN may be needed.

Chapter 2

Methodology

The methodology followed for accomplishing the objectives set for this Thesis works are described step by step in the paragraphs below. Nevertheless, it should be noted that these steps were not completed in the following order, as this work was research-based and a continuous literature review was performed,

2.1 Literature review

In the first place, a review on scientific literature is performed to assess about the applications of artificial neural networks and machine learning algorithms within the field of prognostics and health management.

This review is focused on the development of artificial neural networks models for remaining useful life prognostics, their advantages and its limitations for practical implementation, as well as on degradation detection and quantification. From this is expected to gather an understanding on the practical limitations these algorithms have, in order to implement a corresponding solution in the Framework to be proposed.

Moreover, with an understanding on the applications of the different neural network and data-driven models involved for RUL prognostics, a theoretical background review is done on those algorithms.

2.2 Framework proposal

Based on the relevant points gathered from the aforementioned review, a practical Framework for RUL estimation is proposed. This proposal has a general scope for an implementation on problems with different types of useful sensor data needing different types of data processing.

The proposal also involves the definition of a data processing scheme which will be the one to be followed for its actual deployment in real industrial circumstances.

2.3 Framework coding and implementation

Once the Framework, its stages and the data processing scheme are defined, it is implemented in a computational environment using `python` language. For the development of the neural networks involves, google's `tensorflow` library is used, and for general machine learning tools, `scikit-learn` library is used (both libraries are deployed in python).

2.4 Case study validation

Having implemented the complete Framework code, it will be tested and validates under two popular case studies intended for RUL prognostics: C-MAPSS and FEMTO datasets. C-MAPPS is a turbofan engine degradation dataset produced in a simulation environment, while FEMTO is a ball-bearing degradation dataset produced experimentally.

For the C-MAPSS case, several configurations for defining a degradation threshold are going to be studied for later implementation on FEMTO.

2.5 Results analysis and concluding remarks

Once tested and validated the proposed Framework on the case studies, the results will be analyzed in order to evaluate its prognostic capabilities and its real-life applicability. From this analysis, the concluding remarks of this thesis will be drawn.

Chapter 3

General background

In this chapter is presented the general background from which this Thesis is developed. First, is presented a scientific journal review and following are described the theoretical basis of the algorithms and techniques involved, those being: artificial neural networks (ANN) and the Mahalanobis distance (MD) measure.

3.1 Journal review

The journal review described below focuses primarily on the application of ANNs in the field of Prognostics and Health Management (PHM), specially on their use for Remaining Useful Life (RUL) estimation. Following comes a review on techniques for measuring health and degradation on physical assets.

3.1.1 ANNs for RUL prognostics

First attempts to implement neural networks in the field came early after the backpropagation algorithm development, in 1989; when McDuff et. al [26] developed a 2-layered MLP for fault diagnostics on F-16 fighter jets. This investigation proved that ANNs have a promising applicability to the field and since then these basic networks have been tested under different model structures with different goal objectives. However, despite their great adaptability, MLPs have been prone to overfitting [17], so different neural networks architectures with a more profound data processing schemes are used before hand to achieve wider generalization.

A neural network with a more complex processing structure is the recurrent neural network (RNN), which is specialised for temporal series processing, and therefore, have been very suitable for RUL prognostics problems. Indeed, Zheng et. al. [46] compared different data-driven approached for RUL prognosis assessing the accuracy increase of RNN based models over other ML approaches like MLP, Support Vector Regression (SVR) and Relevance Vector Regression (RVR) when testing on C-MAPSS aero engine degradation dataset and a milling machine degradation dataset. RNN have also been used for health-based RUL prediction on bearings (Guo et. al [10]) and on C-MAPSS dataset as well (Malhorta et. al [25]).

Within RNN architectures, the Long-Short term Memory (LSTM) has been the most commonly used architecture because it was the first to successfully overcome the vanishing gradient, a major obstacle for training this type of network in the early days. Many adaptation and variations of this architecture have been proposed, of which the Gated Recurrent Unit (GRU) [5] stands out for decreasing the number of operations and trainable parameters needed in the model, requiring fewer computing resources while maintaining a similar performance [6]. However, Yuan et. al. [43] compared LSTM, GRU and other related variations for the RUL estimation on C-MAPSS, proving that the standard LSTM networks outperforms them all on said task.

Regarding the more recent CNN, which has gained great popularity in computational visual recognition, has been primarily used for image-based fault diagnostics within the PHM field, proving to achieve superior performances than competing algorithms. As examples; Jansenss et. al [14] showed that CNN outperforms classical engineered-feature extraction for fault detection in rotary machines; Verstraete et al. [36] proposed a deep CNN (i.e. multi-layered CNN) for fault identification in rolling bearings using time-frequency image analysis, outperforming support vector machines (SVM) and multi-layer perceptron (MLP) and Modarres et. al. [27] proposed a deep CNN for damage (crack) detection in structures and bridges that also outperforms ML algorithms such SVM, MLP and Random Forest (RF).

Nevertheless, despite CNN excelling for classification tasks, it has also been tried for prognostics purposes. Babu et. al [1] first developed a CNN model for RUL prognostic based on images formed by sequences of sensor data. Then Li et. al [21], using the same input image format in a Deep CNN model achieved state-of-the-art for the C-MAPSS dataset. The CNN has also been successfully implemented for RUL prediction using time-domain representations (spectrograms) of vibrational data in rolling bearings [22], proving to adapt well to input images containing different types of data.

Many different types of neural networks have been tested and implemented for this RUL estimation purpose. Another one that is under the scope of this Thesis work is the Convolutional-RNN, which is a version of the RNN model that integrates the a convolutional operation in its inner operations. This kind of networks open up the possibility to process sequences of images following on the specialised RNN structure that is able to "remember" past sequences and make a better prediction. Up to date there has not been publications involving purely this kind of networks in the PHM field, however, a modified version was used for RUL prognostics on rolling bearings with overall accurate results [40].

Neural networks have proven to be useful algorithms to implement for RUL prognostics. However, among the publications presented above, all had significant inaccuracies in RUL prediction during the early stages of operation of the asset in question. It has been argued that during these stages there is no noticeable degradation in the equipment, and therefore RUL can not be accurately predicted [11]. Given these, in many case studies, RUL values were limited up until to a certain value before failure [21, 25, 28].

Nonetheless, recently several models have been published that include an algorithm for detecting a degradation threshold after which RUL can be actually predicted. These degradation detection algorithms are based in statistical methods that defines a confidence interval around some key feature or in neural networks for classifying the typo of operation. For ex-

ample, degradation was considered to start on rolling bearings if the kurtosis of the measured vibrational signal was outside of the 2σ confidence interval of the kurtosis in the early stages for 2 consecutive measurements [22]. Similarly, on rolling bearings, Yang et. al [42] developed a CNN to classify vibrational signal between normal and faulty operation, and defined that when 3 out of 5 consecutive classifications were faulty RUL could be actually estimated through neural networks.

3.1.2 Degradation and health estimation on physical assets

The concept of health monitoring in physical asset has usually been referred as an interchangeable term for RUL prognostics. However, there has been attempts to actually quantify the *health state* of an asset by using a health indicator [19] (or from the opposite point of view, a degradation indicator [7]). This task of defining a health measure is an open problem as it can be defined in different ways, and no consensus has been achieved. Nevertheless, proposed health indicators have all been (directly or indirectly) based on sensor data.

For defining the health indicator, different data of sort may be useful, depending on the asset in question. Vibrational data has typically been the most important for equipment where the physical structure and integrity is important [3]. More over, vibrational data can at least enhance monitoring capabilities when using other parameters in more complex assets like in gas turbines [35].

The majority of health indexes proposed have been based on distance between sensor data to a baseline, corresponding to sensor data under known good operational conditions. Primarily the Mahalanobis Distance (MD) has been used to define distance between sensor data points, reducing the dimensionality of a multivariate systems to a single parameter (the distance) and using this as health indicator [7, 19, 31]. This health indicator can be used to detect degradation trends and using it to predict RUL values by projecting its evolution and calculating the time needed until it reaches a known failure threshold. However, MD is restricted to (healthy) datasets following a Gaussian distribution, which may not always be the case. To bypass these problem, clustering algorithms are used instead, the defining the health index as the to the clusters with healthy data [33].

Other ways of defining a health index have been based on the reconstruction error of operational data with AutoEncoder networks. As normally occurring data should be easier to replicate than data under a notorious degradation process, reconstruction errors should increase in end stages of operation, thus being useful for HI definition [25]. In other approaches, sensor readings have been fused together to a single parameter (the HI) through logistic regression [37], to finally project to the failing threshold based on the similarity to known HI curves.

3.2 Artificial Neural Networks

As the name implies, Artificial Neural Networks (ANN) are computational models that generate a network of interconnected "neurons", where each can store a numerical value , that process data to produce a desired output. These networks are structered in stacks of layers formed by grouped neurons; input data is recieved in the input layer and the output is given

by the output layer. Layer in between the input and output, which actually process the input data, are called hidden layers.

Each neuron in network can transmit a "signal" (the numerical value which stores) to the neurons in the neighbouring layer, but always in a forward direction. Its activation, that is, the value that it transmits (storages), depends on the signals send by the lower interconnected neurons in the preceding layer and its actual activation function. Due to this, a neuron may not activate and transmit a signal. Overall, one of the most important characteristics of ANN is that given a large enough neural network, and through the use of non-linear activation functions, the whole network becomes a universal function approximator [8], which makes them employable to a very wide variety of problems.

Due to the last point, ANNs have become the core of Deep Learning as they can autonomously learn to represent data, and moreover, learn representations over other learned representation using multiple layers deep; hence the term Deep.

The quintessential example of an ANN due to its simplicity is the Multi-layer Perceptron, which are just stacks of fully connected neural layers. Other classical DL architectures are aimed at specific types of data formats; for 2D or 3D data, like images, there is the Convolutional Neural Network; for temporal series of data, there is the Recurrent Neural Network. These are further explained below.

3.2.1 Multi-layer Perceptron

A Multi-layer Perceptron (MLP) is an ANN where neurons are stacked in interconnected, vector-like, layers. In the first layer of the network is stacked the input data, after which comes the hidden layers and finally there is the output layer, which outputs the results of the network. In figure 3.1 a simple MLP scheme is presented.

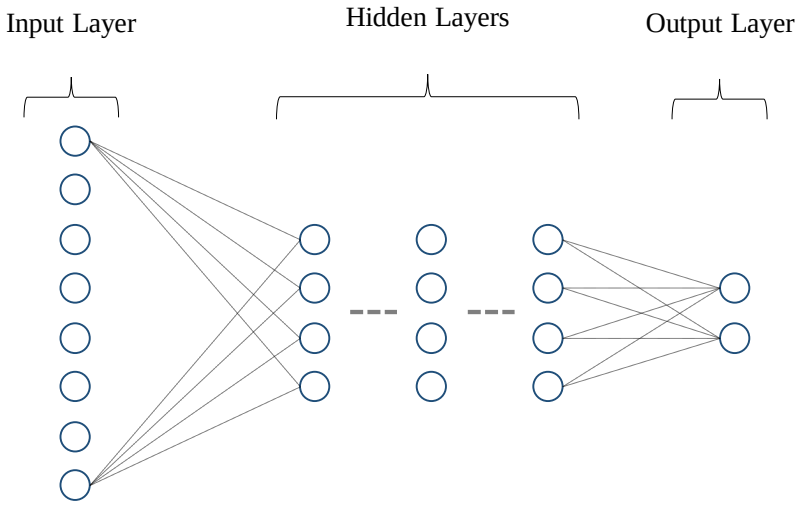


Figure 3.1: MLP scheme

Each neuron in the network outputs a single scalar value (z) from the input vector ($\mathbf{x}$)

given by the connected neurons of the preceding layer. To do this, an affine linear transformation is applied over the input vector, using a set of weights and biases. A further nonlinear activation function (h) is applied over this value. Considering all the neurons in a layer, the output of the layer has a vector form, and the weights ($\mathbf{W}$) and biases ($\mathbf{b}$) are condensed in a matrix and vector form respectively. The output of a layer can be expressed as in equation 3.1.

$$\mathbf{z} = h\left(x \cdot W^T + b\right) \tag{3.1}$$

Network parameters like depth (numbers of layers), number of neurons per layer and their activation function are subject to the criteria of the network developer. The characteristics of the output layer depends on the type of problem trying to solve:

- For **classification problems** an output layer with a quantity of neurons equivalent to the number of classes present on the problem is used. Softmax is applied as activation function due to its capability to translate into probabilities the existence of 1 class over n possible options.
- For **regression problems**, where a scalar estimation is expected, an output layer with one single neuron and a linear activation function is used.

3.2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) are among the most renowned types of ANN because they were specially designed for analyzing time-series data. These networks have succeeded in impressive challenges involving data sequences, such as speech recognition, language processing and event forecasting.

All RNN architectures are composed by a single building unit called "cell" which process the input data in an orderly sequential manner; producing a single output for every time-step, called the hidden state. This hidden state also serves as an extra input for the next time-step, alongside the actual input for that time-step, so the cell has information about the last result predicted. With the hidden state of the past time-step, and the current input, the cell should understand the temporal behaviour of the data, so it can produce an output to the actual time-step according to that development. This process is schematized in Figure 3.2.

Data flow inside the RNN cell is controlled through gates, which are basic single layer perceptrons with an input layer and an output layer and no hidden layers. These gates use non-linear activation functions, and the way the data is manipulated through the cells by the gates, depends on the architecture of the RNN specifically.

The best known RNN architecture is the LSTM (Long-Short Term Memory), introduced by Hockreiter and Schmidhuber in 1997 [12]. There have been several variations to this network through the years, but in 2014 Cho et. al [5] introduced a simpler cell architecture, named the GRU (Gated Recurrent Unit) which performed as well as the LSTM while requiring less parameters.

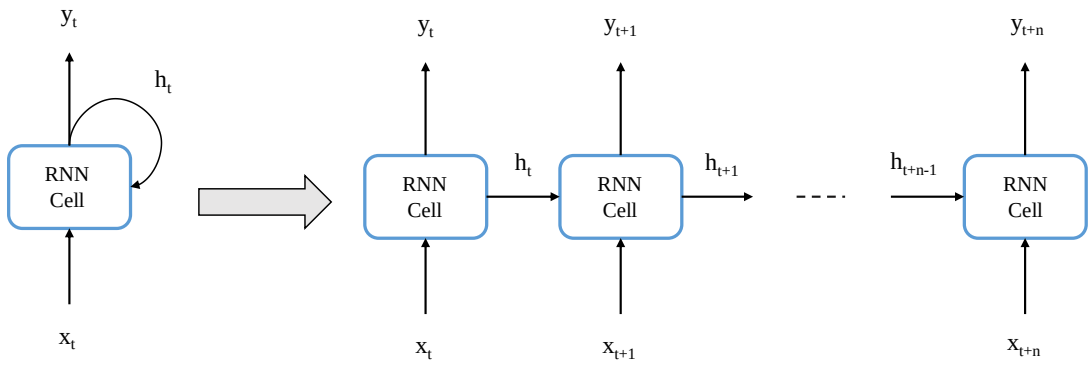


Figure 3.2: RNN sceheme. Right: Unrolled view

Long-Short Term Memory

The Long-Short Term Memory (LSTM) architecture was introduced to solve the vanishing gradient problem, a common issue for RNN when training with data of long time lengths. Formally, the vanishing problem occurs when the error's gradient approximates to zero faster than the weights of the cell can be update through the backpropagation algorithm (see section 3.3). Its most remarkable innovation is the integration of a *cell state*, which is analogous to the hidden state, working in parallel to it. Its purpose is to maintain long term dependencies between data sequences, maintaining a constant error throughout the network when training, preventing its vanishing (to zero).

An LSTM cell contains 3 gates; the forget, input and output gates. The forget gate (f_t) receives the past hidden state (h_{t-1}) and the current input (x_t) to "forget" information that is not relevant of the past cell state using a sigmoid activation function (c_{t-1}); the input gate (i_t) serves to update the cell state according to the current input and past hidden state; and the output gate (o_t) delivers the corresponding time-step output and the new hidden state (h_t), considering as inputs the updated cell state (c_t), the past hidden state and the current time-step input. The cell then outputs the new hidden state and the updated cell state that will be used for the next time-step. In figure 3.3 it is shown the schematic of a LSTM cell and in Equation 3.2 are described all the operations done inside:

$$\begin{aligned} f_t &= \sigma(U_f \cdot h_{t-1} + W_f \cdot x_t + b_f) \\ \mathrm{i}_t &= \sigma(U_\mathrm{i} \cdot h_{t-1} + W_\mathrm{i} \cdot x_t + b_\mathrm{i}) \\ o_t &= \sigma(U_o \cdot h_{t-1} + W_o \cdot x_t + b_o) \\ c_t &= f_t \odot c_{t-1} + \mathrm{i}_t \odot \tanh(U_c \cdot h_{t-1} + W_c \cdot x_t + b_c) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{3.2}$$

where U and W are the weights matrixes associated to the past hidden state and current input respectively and b are the biases. $\odot$ and $\cdot$ are the Hadamard (pointwise) and dot product respectively. All the gates use a sigmoid (σ) activation function and for the current cell state a hyperbolic tangent (tanh) is used.

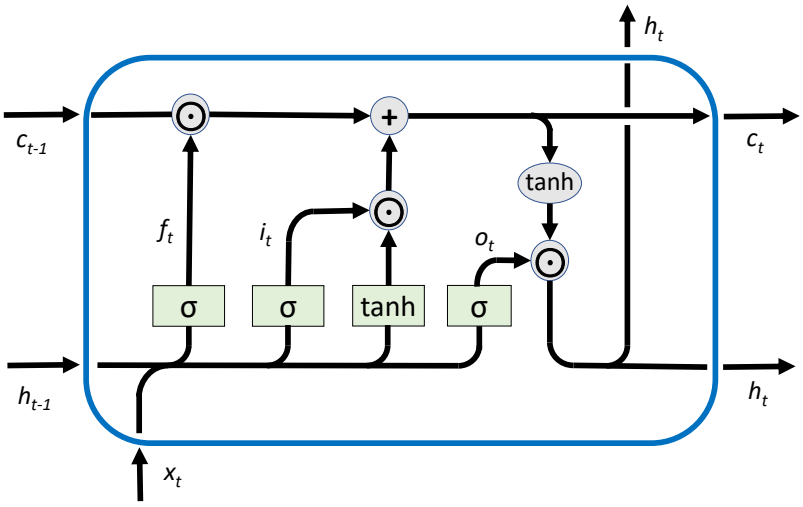


Figure 3.3: Cell structure of the LSTM architecture

3.2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNN) is another renowned ANN algorithm that has taken the leading role in tasks of image classification and pattern recognition due to the high degree of abstraction they can achieve when using multi-layered (deep) networks.

The main purpose of the CNN architecture is to extract spatial features from a 2D data array with 1 or more channels (3 in RGB images) through the use of receptive filters (also referred as kernels) that "look" into every section of the image in search of those particular features. When a receptive filter completes a whole inspection of the image, it produces a **feature map**, which, as the name implies, it highlights the detected features considering their respective position within the image. A single filter produces a single feature map, and more filters can be used to obtain different feature maps, thus revealing more aspects hidden within the image, which may be helpful for the desired objective.

Filters inspect the whole image by taking discretized steps (stride) following a well-defined path; starting from a corner of the image it moves along the horizontal axis in a straight line until it reaches the opposite frame of the image, where it takes a step on the vertical axis and from that point starts covering the whole horizontal axis again. And so, this search continues until the whole image is has been covered by the filter, producing a complete feature map of it. The application of a filter reduces dimensionality as it condenses the values under the filter to a single scalar value. To produce feature maps with the same dimensions of the input image, a padding with zero-valued pixels can be added to the frames of the image and apply the filter considering this padding.

In practice, this feature extraction is done by performing a 2D discrete convolutional (cross-correlation) operation between the filter, a 2D array composed by weights, and the image. In simple words, this operation superimposes the filter over a section of the image, multiplying the kernel weights with the corresponding pixel values of the covered section of the image, returning the sum of all these multiplications. This mathematical operation is shown in equation 3.3, where a filter of height h and width w is convolved ($*$) with an input

array X consisting of only 1 channel, producing the feature map Y.

$$Y_{\mathrm{i},j} = (W * X)_{\mathrm{i},j} = \sum_{a=0}^{h} \sum_{b=0}^{w} (W_{a,b} \times X_{\mathrm{i}+a,j+b}) \tag{3.3}$$

Input images often contain more than one channel, so the actual convolution operation in the CNN is shown in equation 3.4: where X^c is the input image in channel c, $W^{(m,c)}$ is the convolution filter weight matrix of the feature map m for the channel c of the input, b^m is the bias associated to the feature map m and ϕ is the non-linear activation function. These sets of weights and biases used in the convolution are automatically learned through the backpropagation algorithm during the network training process.

$$Y^m = \phi \left(\sum_{c=1}^{C} W^{(m,c)} * X^c + b^m \right) \tag{3.4}$$

Along with the number of filters used and the implementation or not of padding, other user-defined hyperparameters required for the CNN are the kernel shape, the filter stride (that is, the step size used to move along the horizontal and vertical axis of the image) and the activation function to use.

For a further processing of the feature maps, another convolutional operation or max pooling may be applied. For both classification and regression problems, the resulting features maps of the last convolutional layer are flattened and connected to a MLP. In Figure 3.4 a basic CNN architecture is schematised.

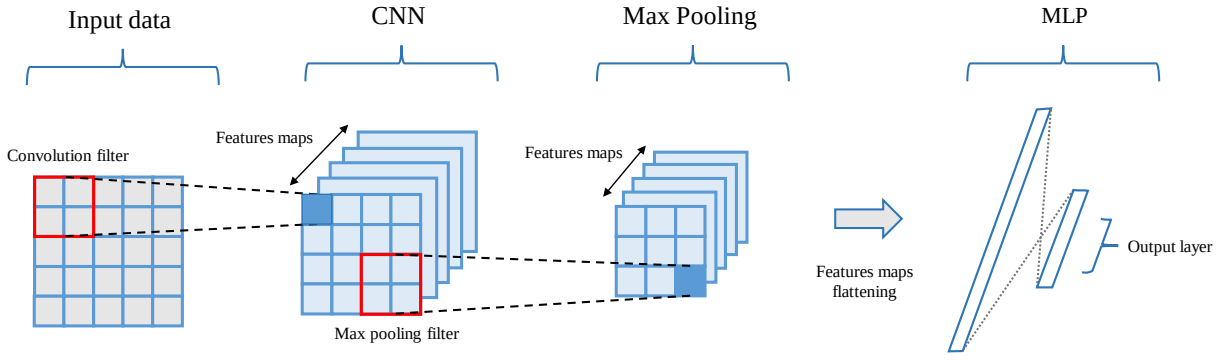


Figure 3.4: Basic CNN architecture with a max pool and MLP layer

Max pooling

Max pooling is a subsampling operation which serves to reduce data dimensionality. Analogously to the convolution operation, a max pooling uses a receptive field that filters pixels by taking only the maximum value present in that field, thus decreasing the image size.

The most significant benefit of applying max pooling is the reduction in the computing power needed, but, as an extended benefit, it also adds a small degree of translational invariance to the model. This property results useful to the network because underlying patterns should be detectable independent of its location within the data.

3.2.4 Convolutional-RNN

A Convolutional Recurrent Neural Network (ConvRNN) is a unique architecture that combines the spatial analysis capabilities of a CNN with the temporal analysis of an RNN for the analysis of sequences of spatial data. This adaptation is done by integrating convolutional operations at the gates of the RNN cell instead of the single layer MLP of the standard structure, which then adds the possibility of processing multi-channel, two-dimensional data formats.

These convolutional operations are performed over both the input and hidden states arriving to the cell, but, because data dimensionality needs to be maintained in order to add up together their information, the convolutions are required to be done with "zero" padding. However, the number of feature maps produced, can differ to that from the input data (number of channels) or past hidden states (which has the same quantity of feature maps as the ones produced by the convolution).

ConvRNN networks has been recently introduced and between the applications found for it are weather forecast and computer vision analysis. Currently in the literature, there are found 2 adaptations of RNN to ConvRNN, the ConvLSTM and the ConvGRU. Up to date, there has been no research involving this network architecture for PHM purposes, but it has promising capabilities for prognostics given temporal evolution of spatial data.

ConvLSTM

The Convolutional LSTM was proposed by Shi et. al [41] (2015) as a way to adequately sequences of weather maps for weather forecasting, without the need of a preprocessing to adapt it for a common recurrent network. As stated before, in this kind of architecture the dot product operations over the input and hidden states are replaced by convolutional operations. In equation 3.5 are shown the set of operations performed at each gate of the ConvLSTM cell considering this change.

$$\begin{aligned} f_t &= \sigma(U_f * h_(t-1) + W_f * x_t + b_f) \\ \mathrm{i}_t &= \sigma(U_\mathrm{i} * h_(t-1) + W_\mathrm{i} * x_t + b_\mathrm{i}) \\ o_t &= \sigma(U_o * h_(t-1) + W_o * x_t + b_o) \\ c_t &= f_t \odot c_{t-1} + \mathrm{i}_t \odot \tanh(U_c * h_{t-1} + W_c * x_t + b_c) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{3.5}$$

where * denotes the convolution operation, U and W are the convolution filter weight matrixes and b are the convolution biases. $\odot$ correspond to the Hadamard (pointwise) product.

3.3 Neural networks training process

As stated before, the main objective of an Artificial Neural Network is to approximate to an unknown function of high complexity, which is done by adjusting its parameters (interconnection weights and biases), that initially are randomly selected, to perform to its desired purpose. This parameter fitting is done by a training process, also referred as learning process.

For this process, the network is fed with (training) data ($\mathbf{x}$) that already has a known, correct, label ($\mathbf{y}$). The network's output is matched with the data's actual label, according to an equation that maps the error to a scalar value (E), named the *loss function*. The main objective of the learning process is then to minimise the loss function by adjusting the network's adaptive parameters (w).

$$w^* = \arg\min_w \mathrm{E}(w, x) \tag{3.6}$$

Loss function optimization is performed by updating w with respect to its gradient. The gradient is computed by the **backpropagation** algorithm, which calculates the derivatives of the error with respect to the parameters for every (network) layer using the chain rule of calculus.

The optimization procedure to find the error's global, or local, minima is iterative. Many optimization algorithms are based on the **Stochastic Gradient Descent** (SGD) method, which depends solely on a hyperparameter called the **learning rate** γ. With SGD, the gradient is computed using only a small number of, randomly selected, training samples and, according to the learning rate, it updates the network's parameters with it (see Equation 3.7). The loss function (error) gradient is computed with less, but random samples ($\nabla \mathrm{E}^*$) to decrease the number of computations required. Nevertheless, the random selection of samples, and an adequate learning rate (selected by the network developer), ensures the convergence to a local or global minima on the error.

$$w_{n+1} :\leftarrow w_n - \gamma \nabla \mathrm{E}^* \tag{3.7}$$

Regarding the loss function used, it depends on the type of problem. For regression problems, mean square error (MSE) and mean absolute error (MAE) are common examples. For classification problems, a cross entropy (negative log likelihood) loss function is commonly used.

3.3.1 Training validation

While training the network, it is necessary to evaluate the loss function after each epoch to see if it has already reached a minimum or not, and with that information decide when to end the training process. However, the model can very well overfit to the training dataset and be incapable to correctly interpret data outside of that domain.

To avoid overfitting, the training set is divided into 2 independent sets: an actual dataset for training and a validation set. This validation set, as well as the training set, is used to evaluate the model performance after each training epoch, but, because is not used for training, it can be used to monitor the generalization capability of the network. When the validation loss reaches a minima, it can be understand that the trained network has reached an optimal point in which its generalization capability are maximized. After this minimum point, the validation loss starts to gradually increase, indicating that the network is overfitting to the train set.

3.3.2 Regularization techniques

For a further prevention of overfitting, there are some other techniques that may help to *regularize* the network interpretation capabilities. A commonly used technique is **dropout** [32], which randomly disable neurons in the hidden layers while training, meaning their adaptive parameters (w) can't get updated, adding greater generalisation capabilities to these hidden layers.

Another form of regularization is **early stopping**, which takes into account the evolution of the validation loss during training to decide when to stop the process. In the implementation described in [8], early stopping considers a "patience period" in which if the validation loss doesn't improved, training is terminated (thus meaning that a minimum value has been reached).

More over, the general idea of early stopping can be implemented to the learning rate; by reducing by a factor of 2 to 10 following a *patience period* after a minimum value has been reached and not improved. With this, a further (minimal) reduction in loss may be possible, and, by implementing this, a smoother convergence to the minimal is ensured. This method has no particular name, so from now on it will be referenced as **learning rate reduction on plateau**.

3.4 System Health quantification

As presented in the literature review, there are 2 typical approaches for quantifying the health condition of a physical asset: by an Autoencoder network approach and by the Mahalanobis distance approach. Both are further explained below.

3.4.1 AutoEncoder approach

An AutoEncoder (AE) is a neural network that compress the input data into a lower dimensionality space, called the *latent space*, which then takes and tries to rebuild the same input data from it. These kind of models are based in 2 neural networks: an encoder and a decoder. The first one compress the input data into the latent space and the second one uses this latent space to rebuild the same input data to the decoder.

As these models reduce data into a lower dimensionality, AE are implemented when data compression is necessary. On the other hand, as the reconstructed input data will have

associated a reconstruction error, AE encoders can be used for anomaly detection considering as anomalies those cases where the reconstruction error surpasses a defined threshold.

Given that data reconstruction will always have a numerical error when compared with the original data, AE have been used for establishing a health index indicator that is directly related to the reconstruction error obtained with the network. In the context of PHM, health index based on AE have been developed and has obtained promising results [25], producing a health index in good correlation with a degradation process.

However, a problem that this implementation may encounter is when defining the base dataset for healthy operation in assets with a continuous and subtle degradation over time. In the work cited above, the definition of the healthy operation range was arbitrarily set by the researchers. In that case, if within that range selected as healthy data there were segments of operation under degradation, the AE could have learnt to correctly interpret a degradation process; therefore making the network useless for detection based on reconstruction error, as it will yield similar error values to those for healthy operation.

3.4.2 Mahalanobis Distance approach

Another method that has been adopted for assessing degradative process on physical assets is the Mahalanobis distance (MD) [16], which is a spatial metric for multivariate systems developed by the Indian statistician P.C. Mahalanobis [24]. Differently to other metrics, the Mahalanobis distance defines its measure considering the distribution of its composing variables, as well as the covariance between them. This has lead to its use as a way to quantify degradation (or health) with an index, by calculating the distance of data under actual operation, to operational data under a known healthy state. For example, Wang et. al used the Mahalanobis distance as a degradation indicator for rolling bearings for a degradation stage detection, with a later implementation of Kalman filter model for RUL estimation [38]. MD approaches have also been notoriously studied for the degradation assessment on electronic devices [19].

Formally, the Mahalanobis distance is a mathematical metric that defines the distance to the mean point of a distribution in an m-dimensional space. Considering a point $\vec{x}$ and a distribution with a vector of mean values ($\vec{\mu}$) and a variance-covariance matrix (S), the MD is calculated according to the operation described in equation 3.8. Is to be noted that as the system has m constituent variables, the MD needs to be scaled by m.

$$MD = \sqrt{\frac{1}{m}\left(\vec{x} - \vec{\mu}\right) S^{-1} \left(\vec{x} - \vec{\mu}\right)^T} \tag{3.8}$$

The mean value (μ) for each of the m variables, considering n independent observations, is calculated accordingly to equation 3.9. Samewise, the variance-covariance matrix (S) between each variable, is calculated through equation 3.10.

$$\mu_{\mathrm{i}} = \mathrm{E}[x_{\mathrm{i}}] = \frac{1}{n}\sum_{k=1}^{n} x_{k,\mathrm{i}} \tag{3.9}$$

$$S_{\mathrm{i},j} = \frac{1}{n-1} \sum_{k=1}^{n} \left(x_{k,\mathrm{i}} - \mathrm{E}[x_{\mathrm{i}}]\right) \left(x_{k,j} - \mathrm{E}[x_j]\right) \tag{3.10}$$

The mean vector μ and the variance-covariance S matrix are obtained from a reference distribution, called the *Mahalanobis space* (MS). The average MD for the points inside the MS will always be 1, and because of this, no assumptions on the distribution of the input variables is necessary beforehand. Moreover, due to this characteristic, categorical data can also be used as a composing variable of the MS.

To exemplify the MD metric definition, in figure 3.5 the Mahalanobis distance is compared to the euclidean distance (ED) by plotting the equal-distance curves from a point in the distribution shown. For the MD metric, distance distributes accordingly to the variance of each composing variable, while for the ED a perfect circle is obtained as no consideration is given to the distribution of the variables. In fact, the ED is a particular case of the MD, when all variables are completely independent from each other, as in this case the variance-covariance matrix would be equivalent to the Identity matrix I.

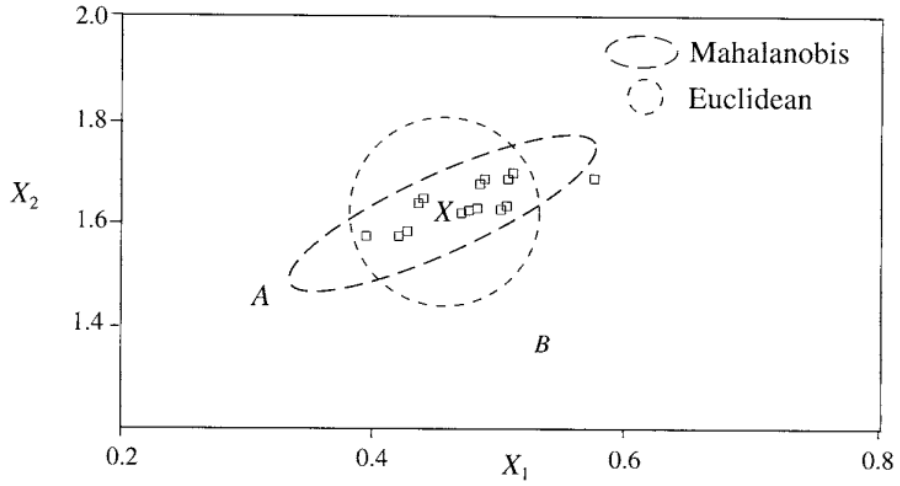


Figure 3.5: Comparison between a Mahalanobis and an Euclidean space

This metric has proved to be a useful technique for determining the similarity of a set of values from an unknown sample, to a set of values measured from known samples. Another useful characteristic that simplifies the implementation of this approach is that only 1 group of "normal" (or "healthy") observations is needed to obtain a correlation structure and to define the reference group. Selection of this healthy group is entirely at the discretion of the data-engineer, which nevertheless will have to be validated. To this, is essential to note that abnormal conditions should have higher MD values. In cases where "healthiness" cannot be precisely defined, multiple iterations can be required to obtain a valid MS: calculating all MDs and discarding observations with high MD, then recalculating until a suitable MS is found.

Chapter 4

Proposed Framework for degradation start detection and RUL prognosis

In the following chapter is presented the 2-stage Framework developed for RUL prognostics. As an introduction to the proposed approach, first is given the main motivation behind its development and the aim it has within the PHM field. Following, these 2 stages involved are thoroughly detailed and represented in a graphical implementation scheme, explaining the data flow and processing throughout the Framework. Finally, general remarks are given on the implementation of the Framework for this Thesis work and for real-world applications.

4.1 Main motivation and aim

In previous works aiming for RUL prognosis of physical assets is found a fundamental problem that has not received the appropriate attention it deserves. This problem corresponds to initiating with RUL predictions naively, whenever the asset is operating and without proper consideration if it is actually possible to obtain accurate predictions given the engine current health condition.

Specifically, when the asset is in a *healthy condition* (i.e. when no degradative process has started), operational sensor data do not show any useful trend that could correlate with an accurate RUL prediction. So, when RUL predictions are naively performed, undoubtedly there will be errors for the predictions that could turn into a total loss of trust in the model itself.

Therefore, to tackle this problem, is developed a 2-stage RUL prognostic Framework based on an initial degradation assessment stage, with a following stage for RUL estimation based on an Artificial Neural Networks that is used only when the asset is under an on-going degradation process. Each stage has a particular objective, corresponding to the following:

- **Degradation assessment stage**: quantifies the degradation of the asset. Establishes a degradation start criteria through a statistical threshold after which RUL can be predicted.

- **RUL prediction stage**: estimates RUL values based on sensor data only after the 1st stage has confirmed an on-going degradative process.

These stages, and their implementation, are further explained in the coming paragraphs. Block diagram are also schematized to better represent the Framework stages and the data operations involved.

4.2 Degradation assessment stage

The first stage of the Framework has the objective of assessing the health condition of the asset by calculating a degradation index based on the operational sensor data of the asset. This degradation index (or equivalently, health index) corresponds to the Mahalanobis distance (MD), which calculates the "divergence" of the current sensor data measurements to historical data known to be in a healthy condition. Given that he MD is set as degradation index, it shares one important property with it: **the average degradation index for healthy data has always a value of 1**.

As explained in the past Chapter, for calculating the MD it is required the Mahalanobis space (MS) of the composing variables, which contains the mean (μ) and the variance-covariance matrix (S) that are used as reference (eq. 3.8). In this application field, **the MS equals to the healthy dataset**, ergo, the healthy dataset is used as reference for the MD. However, as the MS is not known beforehand, due to the asset degrading slowly overtime, the MS must be found.

Therefore, to find the healthy portion of an asset's operational data and conform the MS, an unsupervised iterative algorithm is developed, proposed and explained in the section below.

4.2.1 Iterative algorithm for Mahalanobis Space definition

For defining a Mahalanobis space representative of the healthy operational data, a training set containing the complete life cycle of the asset in question is required beforehand. But before proposing the search algorithm, the following assumptions are made:

- Healthy operation is continuous until the MD values reach a certain threshold; after crossing it, a continuous degradation process follows until the asset reaches failure. The MS will be composed of all these data points before the threshold.
- MD values inside the MS distribute log-normally. Therefore, the logarithm of the mean MD will be always 0.
- A threshold is established to differentiate healthy measurements from abnormal ones. The threshold (MD_{th}) is set as an upper bound to the log-normal distribution; equivalent to σ standard deviations over the mean. The actual MD_{th} is obtained as follows:

$$\mathrm{MD}_{th} = \exp\left(\log(\mathrm{MD})_{\mu} + \sigma \cdot \log(\mathrm{MD})_{\sigma}\right) \tag{4.1}$$

- Random noise is expected within sensor data, therefore, the MD threshold must consider a sustained statistical tendency over-time to certainly confirm the occurrence of

a degradative process. Therefore, degradation starts after "k" out of "n" consecutive MDs are over the threshold, where $k \leq n$.

- All assets that have reached failure must have crossed the threshold at some point during their lifetime.

Based on the points above, the proposed algorithm for healthy MS definition follows an iterative procedure of calculating MD values for all instances and leaving out of the MS data coming after crossing the threshold (eq. 4.1), using the criteria "k" out "n" consecutive instances. This procedure is iteratively performed, starting from all instances being considered part of the MS, until the MS converges to an unchanging number of instances (i.e. cardinality, `Card`). This convergence is checked by the parameter epsilon (ε), which represents the change in the cardinality of the MS with respect to the cardinality of the iteration before, as depicted in equation 4.2. If no change occurs for 2 consecutive iterations the MS is considered to have converged.

$$\varepsilon = \frac{\texttt{Card}(\mathrm{MS_i}) - \texttt{Card}(\mathrm{MS_{i-1}})}{\texttt{Card}(\mathrm{MS_{i-1}})} = 0 \tag{4.2}$$

Considering a training set for a particular type of asset with multiple, independent runs to failure, the proposed training algorithm to obtain a healthy dataset from unlabelled data is represented as the following pseudo-code:

Algorithm 1: Iterative algorithm for Mahalanobis Space definition

```
Result: Mahalanobis space of healthy data
ε = 1;
Assume whole training set as MS;
while ε >tolerance do
    Calculate MD based on MS for each sample;
    Check degradation threshold on each sample;
    if threshold crossed in sample then
        Remove section after threshold from MS;
    else
        Maintain sample section in MS
    end
    Calculate
                                        ε
     ;
end
```

If for some sample in the training set the threshold is never reached, this may be due to a threshold set too high or a degradation start criteria too strong. In those cases, the sigma for the threshold or the start criteria should be reduced.

To use this "trained" Mahalanobis space on the calculation of MD values for new measurements, the variance-covariance matrix (S) and the mean vector (μ) of the space are used with equation 3.8. In figure 4.1 is graphically shown the whole training and implementation

scheme for the degradation assessment stage.

4.3 RUL prediction stage

After a degradation process start was detected in the first stage of the Framework, RUL estimation for the asset can actually be done. To perform this task, an Artificial Neural Network (ANN) model is used due to their ease of implementation and increased performance in comparison to other approaches..

The general structure and hyperparameters of the ANN are to be optimized for the specific case study. Nevertheless, as RUL is a scalar value in search for a scalar number, the ANN must have a multi-layer perceptron (MLP) ending in a single neuron as its output layer. The hyperparameters of the network are found through a bayesian optimization process (discussed in 4.5.3).

Once the optimal network structure is defined, the ANN is trained to the specific dataset in study, in order to obtain a working model useful for deployment. Moreover, as the different sensors may widely different order of magnitudes in their measurements, each sensor data is independently scaled to have the same scale for all sensor; thus, helping the notwork not to overfit or ignore any input parameter. This is further explained in section 4.5.

In regards to the input data, it will contain current measured data, but the structure for this data given to the ANN will depend on the chosen ANN architecture. To help the network provide an accurate prediction, a time-window with past consecutive sensor measurements is given. The extension of the time window will depend on the particular dataset studied, specifically, on the average lifetime of the assets and their sensor sampling rate.

4.4 Framework implementation scheme

In figure 4.2 is schematized the complete implementation of the proposed Framework for degradation assessment and RUL estimation. Data flow is indicated through arrows and trainable algorithms involved are coloured in green. Doted arrows between those algorithms in the training procedure and in the implementation procedure imply the reuse of the trained model. Yellow blocks show the 2 results that the Framework can output for a current measurement on an asset: showing that there is no degradation present on the equipment, or, the actual RUL estimation if a degradation process was already detected.

4.5 General implementation remarks

Several important remarks have to be made about the Framework development and implementation; starting from the required dataset characteristics, its preprocessing an labelling, to Framework coding, training and resources needed for it. It is also important to note that this Framework is independently applied to each particular case study, as sensor data and formats differ in each application.

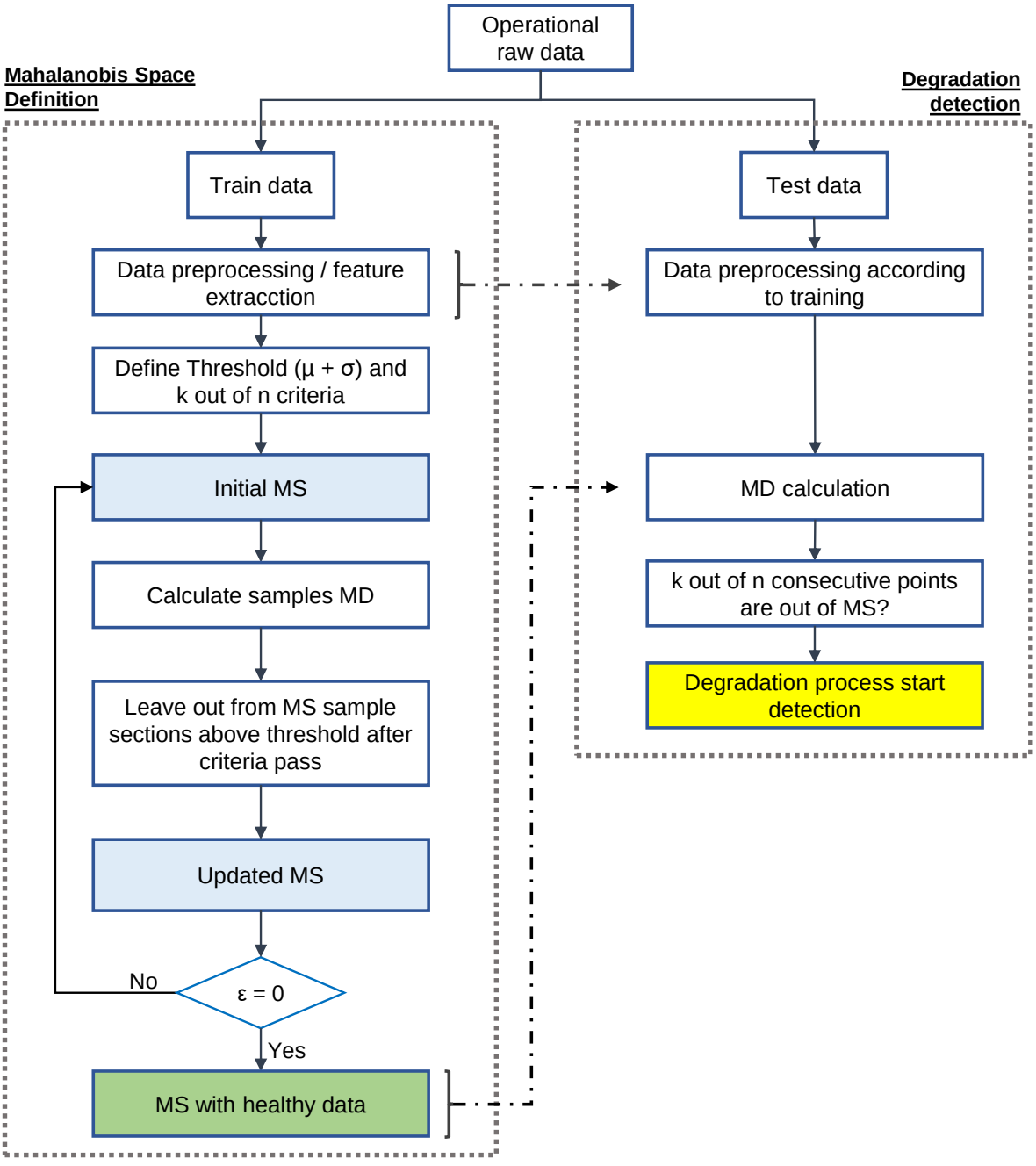


Figure 4.1: Degradation start detection algorithm and its implementation

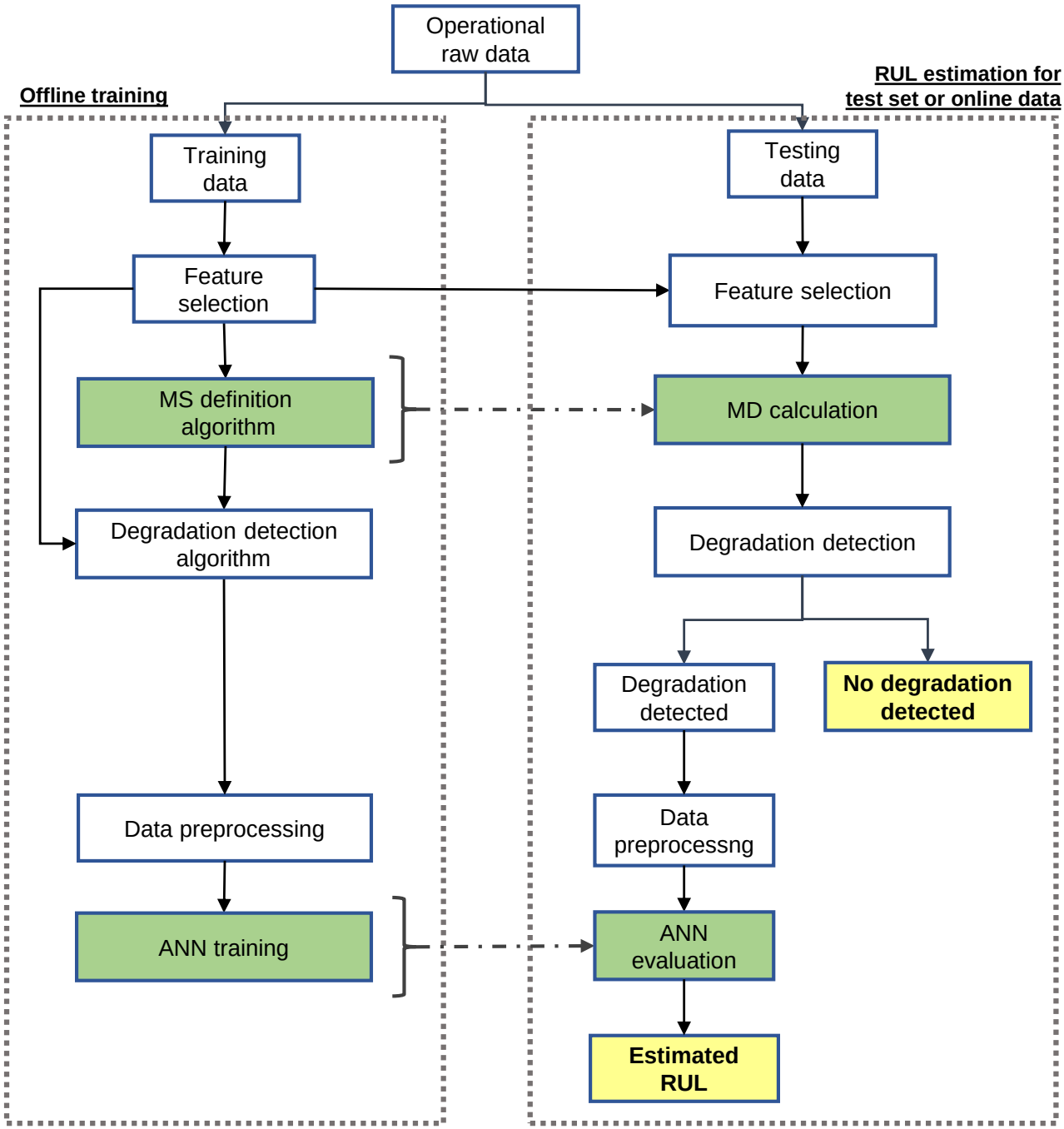


Figure 4.2: Complete Framework. Left side: training procedure. Right side: testing/implementation procedure. In green are marked the trainable algorithms. In yellow are the possible results that the Framework may give.

4.5.1 Dataset requirements

The most important requirement for a implementation of the proposed Framework is a useful dataset, as evidently would be impossible to apply it without one. Said useful dataset needs to posses several key aspects, which among them are the following:

- **Multiple independent samples**: from which the degradation process can be generalized when training the Framework.
- **Healthy data distinguishable from faulty data**: sensor data needs to show different operating conditions and correlate to the degradation process. Sensor data measurements that show constant readings, with no variation in both healthy and abnormal states, should be ignored.
- **Faults are not random**: failure in equipment studied should be driven by a degradation process and not by random chances or external causes.

4.5.2 Data preprocessing

In the first place, when training each stage of the Framework, data needs to be scaled to ease the training process, as parameters of the input data may differ in orders of magnitude between each other. Moreover, these differences may cause that the network overestimate the influence of some parameters over the others. The scaling therefore needs to balance each set of parameters (x) by setting them in the same range, and to achieve this, the Min-Max scaling approach is taken; which scales each set between 0 and 1 by considering the minimum ($x_{\min}$) and maximum ($x_{\max}$) values of the variable, as described in equation 4.3.

$$\overline{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{4.3}$$

Secondly, as this sets are scaled before training the stages, for the Framework implementation the corresponding sets in a new instance (like when testing) need to be scaled accordingly to the training set. To maintain the same scaling, the corresponding parameter x of the testing instance (χ) is scaled accordingly to the minimum and maximum value of the un-scaled training set, as shown in equation 4.4.

$$\overline{\chi} = \frac{\chi - x_{\min}}{x_{\max} - x_{\min}} \tag{4.4}$$

4.5.3 Hyperparameter optimization

Finding the optimal set of hyperparameters is crucial for producing an ANN with the best performance possible. To find this set usually a grid search is performed over a space of selected hyperparameters. This approach however rapidly increases in computational requirements as more hyperparameters are added to the network, so a sequential model-based optimization (SMBO) is followed.

SMBO, also known as bayesian hyperparameter optimization, are applied in cases where the cost function is expensive (time consuming) to evaluate [2]. This optimization builds a surrogate probabilistic function based on past evaluations of the model (ANN in this case), which is the one to actually optimize. specifically for this thesis work, the Tree-structured Parzen Estimator (TPE) method is used for hyperparameter.

4.5.4 Framework coding

The development, training and implementation of the proposed Framework is done computationally using `python` language. This is done individually for each different dataset studied.

A variety of `python` libraries are used for data manipulation and algorithm implementation, including: `numpy`, `pandas`, `scikit-learn`, `scipy`, among others. The proposed algorithm for the Mahalanobis space search is specially coded for this work, following the pseudo code shown in Algorithm 1.

All the scripts created are publicly available in the following git-hub repository: `https://github.com/Cris-Schaad/MD-degradation-detection-for-RUL-prognostics`.

4.5.5 Neural networks, training and optimization

The neural networks involved in the Framework are also coded in `python`, using google's `TensorFlow` and `keras` libraries, which are specially designed for Deep Learning applications. As the ANN has to predict a single scalar value, it is trained with the root mean square error (RMSE) (eq. 4.5) between the predicted value and the true value (label) as loss function. During the network training process, this cost function has to be minimised, and this is done through the backpropagation method using the Adam optimization algorithm [18].

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{\text{i}=1}^{N} \text{d}_\text{i}^2} \tag{4.5}$$

where $\text{d}_\text{i} = y_\text{i}^* - y_\text{i}$ is the difference between the predicted value and the true value to predict. N is the number of samples in evaluation. The TPE hyperparameter optimization is done using the `hyperopt` library for python.

4.5.6 Computational resources

As neural networks require significant computational loads, a capable computer is needed for the implementation of the proposed Framework. For this Thesis work, a computer with the following specifications is employed:

- Operating system: Ubuntu 16.04 LTS
- GPU: Nvidia Titan X pascal (12 GB)
- CPU: Intel Core i7-6700K CPU @ 4.00GHz (quad-core)
- RAM: 32 GB
- Internal storage: 2 TB

Chapter 5

Case Study Validation

The proposed Framework is first studied with the C-MAPSS dataset, a turbofan engine degradation dataset created in a simulation environment. C-MAPSS has become one of the most popular benchmarking dataset for RUL prognostics in the PHM field.

Each stage of the implemented Framework is thoroughly detailed and discussed. The final RUL predictions results are analysed and compared to a framework following the same data processing scheme, but without degradation assessment beforehand.

5.1 C-MAPSS dataset

C-MAPSS is a simulated dataset produced for remaining useful life prognostics which has become a popular benchmarking problem as it was introduced for PHM's first international conference data challenge in 2008 [29]. C-MAPSS dataset consist of multi-sensor data of turbofan engines during normal operation (flights), obtained through the simulation software "Commercial Modular Aero-Propulsion System Simulations" (C-MAPSS) under support of NASA's Integrated Vehicle Health Management (IVHM) program.

C-MAPSS dataset consist of 4 sub-datasets: FD001, FD002, FD003 and FD004, each one with an independent training and testing set. These 4 datasets come with the recorded readings of 21 independent sensors that are distributed throughout the engine, measuring different relevant parameters, which are shown in table 5.1. A diagram of the simulated engine is also shown in figure 5.1, identifying its main components: the inlet fan, the low pressure compressor (LPC), the high pressure compressor (HPC), the high pressure turbine (HPT), the low pressure turbine (LPT) and the outlet nozzle.

Sensors readings given correspond to a cruise snapshot of every flight cycle after reaching a steady state operation. Alongside the sensory data, is also given a snapshot of the operational conditions under which the engines were run. 3 operational conditions were given: altitude (between the range of 0 and 42.000 ft), mach number (between 0 and 0.84) and throttle resolver angle (between 20 and 100). The 4 sub-datasets in C-MAPSS differ on the number of operating conditions (altitude and temperature) and types of failure modes to which they subjected the engines (see table 5.2)

Table 5.1: C-MAPSS sensors

Sensor	Description	Unit
#1	Total temperature at fan inlet	°R
#2	Total temperature at LPC outlet	°R
#3	Total temperature at HPC outlet	°R
#4	Total temperature at LPT outlet	°R
#5	Pressure at fan inlet	psia
#6	Total pressure in bypass-duct	psia
#7	Total pressure at HPC outlet	psia
#8	Physical fan speed	rpm
#9	Physical core speed	rpm
#10	Engine pressure ratio (P50/P2)	–
#11	Static pressure at HPC outlet	psia
#12	Ratio of fuel flow to Ps30	pps/psi
#13	Corrected fan speed	rpm
#14	Corrected core speed	rpm
#15	Bypass Ratio	–
#16	Burner fuel-air ratio	–
#17	Bleed Enthalpy	–
#18	Demanded fan speed	rpm
#19	Demanded corrected fan speed	rpm
#20	HPT coolant bleed	lbm/s
#21	LPT coolant bleed	lbm/s

Another important point in C-MAPSS datasets it that the engines in the training sets are all driven until failure, but the engines in the testing sets are driven until a certain point before failure. For these engines, the actual remaining useful life is provided for the last operational measurement present in the data. The purpose of this problem is then to estimate the remaining useful life of the test engines, which for the C-MAPSS case correspond to the remaining flight cycles until engine failure.

5.2 General remarks and preprocessing

First of all, the implementation of the proposed Framework is done independently on each of the 4 sub-datasets, as they have different operating conditions and number of fault modes (which can be actually seen in the differences of the obtained clusters shown in the coming paragraphs).

In regards to the engines total lifetime values, they are measured in complete flight cycles; the same manner in which the sensor data was recorded. The distribution for the engine total lifetime on both the training and testing sets for each of the 4 datasets are plotted on figure 5.2, where it is evidenced a similar distribution between training test set. This is a positive characteristic as it implies that the complete dataset is within the same domain. Between

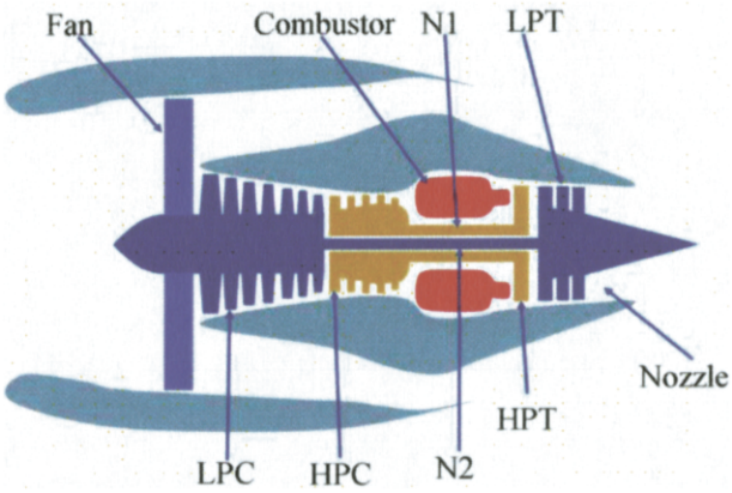


Figure 5.1: C-MAPSS engine scheme

Table 5.2: C-MAPSS datasets features

Dataset	**Op. conditions**	**Fault Modes**	**Train Samples**	**Test Samples**
FD001	1	1	100	100
FD002	6	1	258	259
FD003	1	2	100	100
FD004	6	2	248	248

each dataset however, there are clear differences with the distributions; where datasets FD003 and FD004 are reaching to longer lifetimes than FD001 and FD002.

Previous studies on C-MAPSS dataset have usually limited the maximum RUL values that could be predicted to a value of 125 flight cycles. That is, if at an instance the RUL of an engine was more than 125 flight cycle, its label was set to 125. This limitation for predicting a maximum value of 125 was proposed by Heimes [11], where it showed with neural network that predictions above that value were significantly inaccurate due to operational data in the early (healthy) stages did not show any clear degradation pattern. This limitation suggest the implementation of an initial degradation pattern detection, as the proposed Framework does.

Finally, the operational data for each sensor in each of the training set is inspected. It is detected that in FD001 and FD003 there are several sensors with constant values during the whole engine lifetime and constant across all the samples in the sub-dataset, so these are removed from the analysis as no useful information is provided. Sensors with high correlation between them are also removed as it is redundant information for the MS. From the 21 sensors, 7 are left out of further analysis on all 4 subdatasets. According to table 5.1, the 7 ignored sensors are: #1, #5, #6, #10, #16, #18, #19.

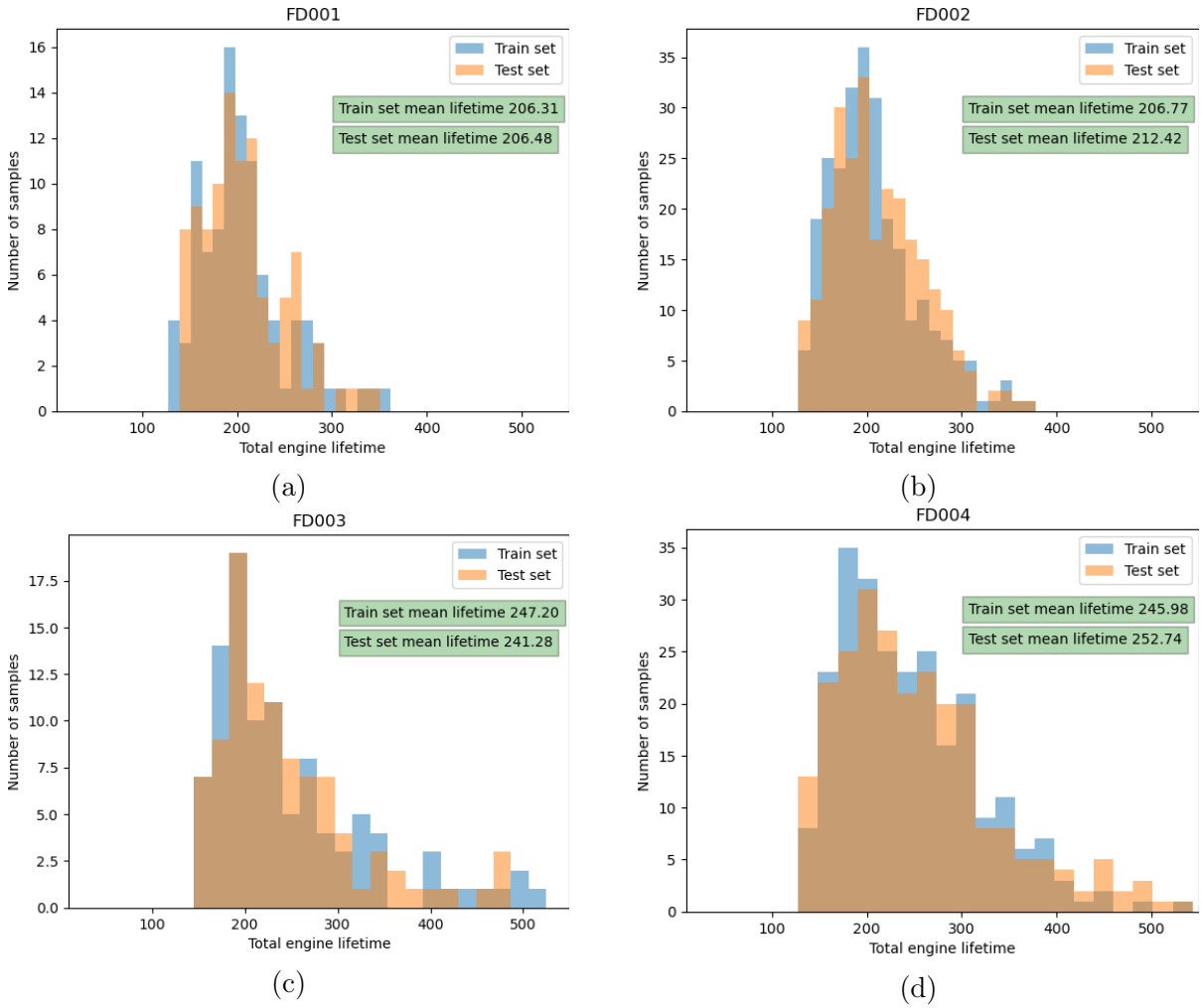


Figure 5.2: Distribution of engines total lifetime in all 4 C-MAPSS datasets

5.3 Degradation assessment stage

For the implementation of the degradation assessment stage, first are needed appropriate hyperparameters to give a corresponding MD measure. The selected hyperparameters are then review at first with the training set before it is applied on the testing set.

5.3.1 Hyperparameter selection

The degradation assessment stage requires 2 hyperparameters: the sigma-threshold and the degradation start criteria, which are the required inputs for the iterative MS defining algorithm. Their election is mostly arbitrary, depending on the dataset, so to ease the selection process the degradation start criteria is first imposed and then, based on this, the sigma-threshold is selected.

C-MAPSS engines mean lifetime ranges between 200 and 250 flight cycles, it is estimated that a 5-consecutive cycles for the degradation start criteria is a window small enough in comparison to the total lifetime, but long enough to contain valuable temporal information about the general trend of the MD value. However, as the actual MD values show a signifi-

cant noisy progress, it is finally imposed a criteria of **5-out-of-5 consecutive MD above threshold** for defining both the MS and the degradation process start.

The selection of the sigma-threshold must comply with 2 key aspects. On the first hand, the threshold should correctly discern between healthy operation and the following degradative operation, which has an increasing trend over time. On the other hand, all samples (in the training set) need to cross at some point the imposed threshold during their lifetime, as all these engines operate until reaching failure. As the 4 datasets have different operating and failing conditions, a particular sigma-threshold was selected.

These selected hyperparameters guarantee that all samples in the training set reach a degradation status, given that all of these end up failing. In Table 5.3 are presented the selected hyperparameters for each subdataset and also shown the corresponding MD threshold obtained from equation 4.1 (after running the MS search algorithm).

Table 5.3: Hyperparameters for the MS search algorithm

Dataset	**σ Threshold**	**k-out-of-n criteria**	**MD Threshold**
FD001	0.35	5-out-of-5	1.03
FD002	0.45		1.05
FD003	0.65		1.09
FD004	0.55		1.07

5.3.2 MS distribution

With the selected degradation start criteria, the MS is obtained after running the MS search algorithm. From this, it is obtained that the MS size in comparison to the complete training sets is between 55% and 60%. In figure 5.3 is plotted the MS proportion evolution during the MS defining algorithm for each of the subdatasets, where it can be seen how it starts from the complete training set as MS and converges to the range mentioned before.

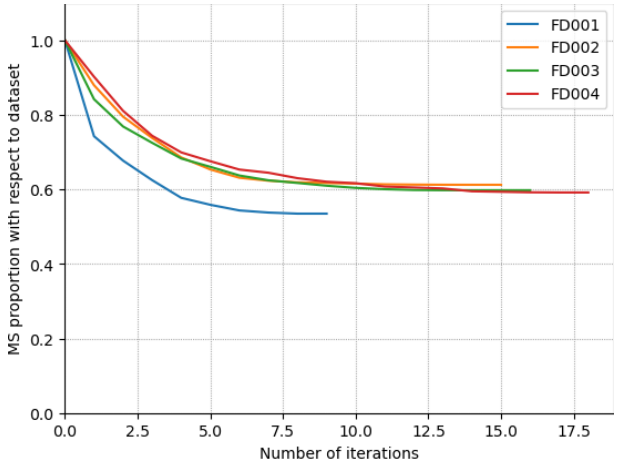
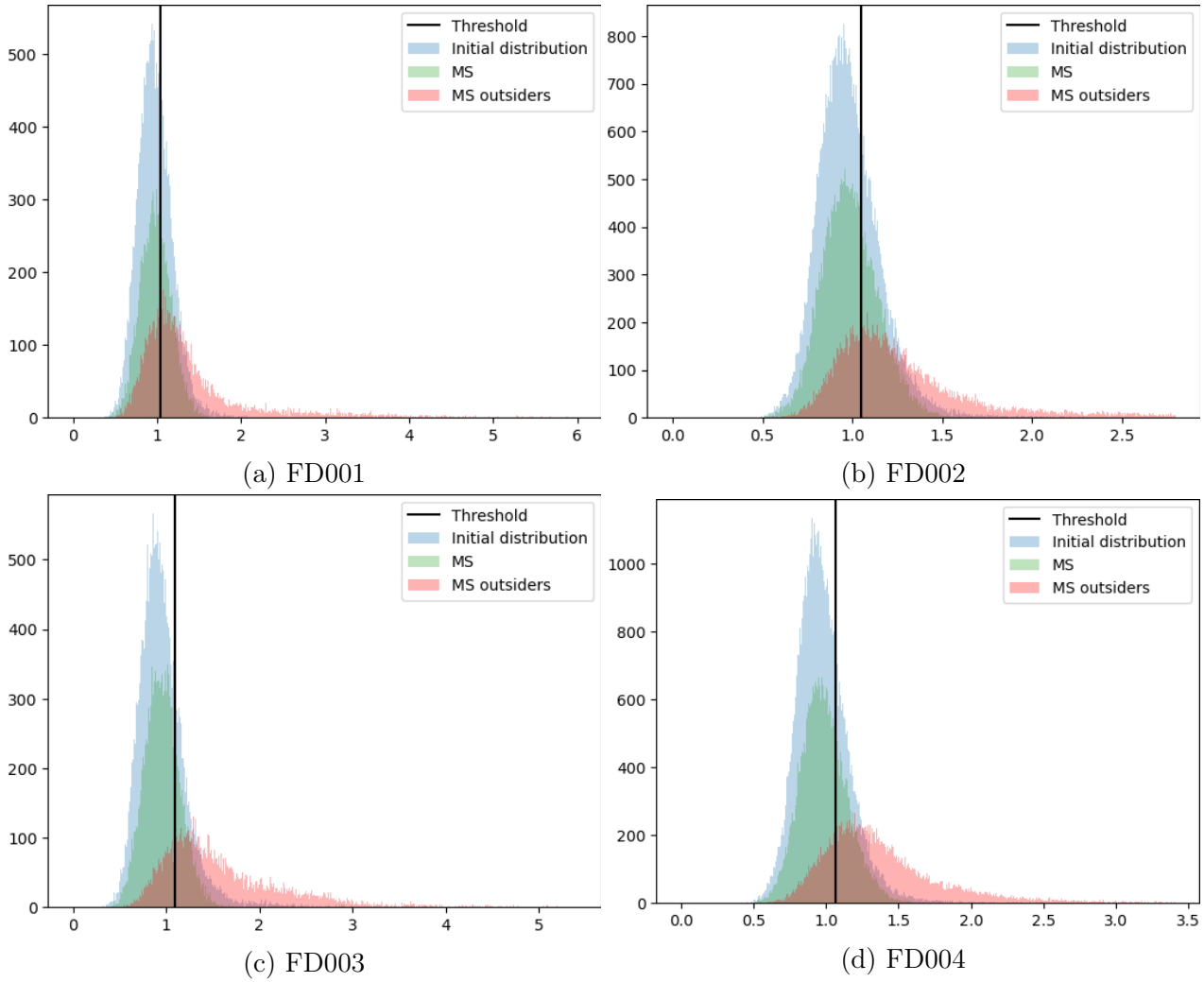


Figure 5.3: Mahalanobis Space (MS) size evolution for each iteration on the MS defining algorithm

Moreover, by the application of this algorithm, the distribution of MDs inside the MS changes. In the plots of figure 5.4 are shown the starting distribution (with all training samples in the MS), and the final distribution obtained for the MS. Also, it is shown the distribution of MD values after the degradation start threshold has been crossed. This distribution is evidently skewed to higher MD values, however, still overlaps with the MS. This is mainly due to the strict 5-out-of-5 MS leaving criteria and to random noise present in both groups.



(a) FD001

(b) FD002

(c) FD003

(d) FD004

Figure 5.4: MD distribution: initially and after the MS search algorithm

5.3.3 MD curves and degradation detection

Before continuingwith the next stage, it is important to review the obtained MD values for all of the samples in the training sets, so in figure 5.5 are plotted the MD values, at each flight cycle, for all samples in the training sets. In these plots are identifying in green colour the portion under healthy operation and in red the portion after degradation detection, while in black is the threshold. The MD progression shown throughout the engine operation is the expected curve for a degrading asset: having initially a flat curve around a constant value and then changing to an exponential trend nearing the failure point. It is also observed that

the imposed threshold and degradation start criteria is able to correctly distinguish between these 2 operative conditions in the majority of cases. Nevertheless, for dataset FD003, there are samples that do not show the expected MD curve; instead having significantly high noise during start of operation and then changing to the actual trend. The cause for this was not further inspected, due to only happening to a few samples in only that dataset.

To further analyze the MD obtained, in figure 5.6 are plotted the resulting MD curves for the longest and the shortest running samples in each of the training sets. These sample are selected because they are the extreme cases, and therefore can show the capabilities of the MD approach under these extremes. Overall, from these plots, it is evidenced that the MD threshold can correctly differentiate between normal and under degradation operation in the majority of cases. For the longest running samples of FD002, however, the degradation process is detected well before reaching the actual increasing trend, due to higher MD values during start of operation. This implies that there may be future tweaking needed for this approach. This also appreciated for the longest running sample on FD003, where initially there are noisy values at the start that provoke an early degradation start detection, when the actual degradative process can be actually identified later in operation.

5.4 RUL prognostic stage

As in the previous stage, the RUL prognostic ANN also need a training step to adequate to the each of the 4 datasets in study. This optimal network structure is searched for with Bayesian optimization, and the obtained model will be used for the 4 datasets, but independently trained in each case.

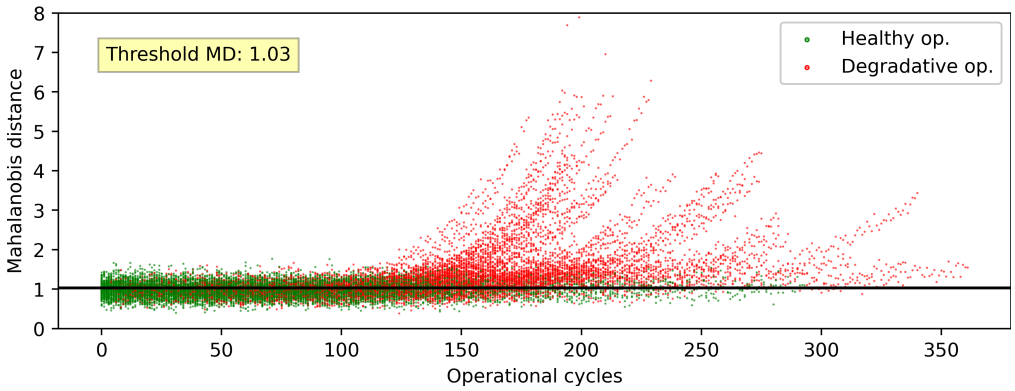
5.4.1 Network and hyperparameter selection

With Bayesian optimization, an extensive hyperparameter search is done. For the C-MAPSS case, it also involves searching for the adequate type of neural network to use: a CNN or an RNN. For the search process, a range of values to select is given for each hyperparameter, which are presented in table 5.4 for the CNN model and in table 5.5 for the RNN (LSTM) model.

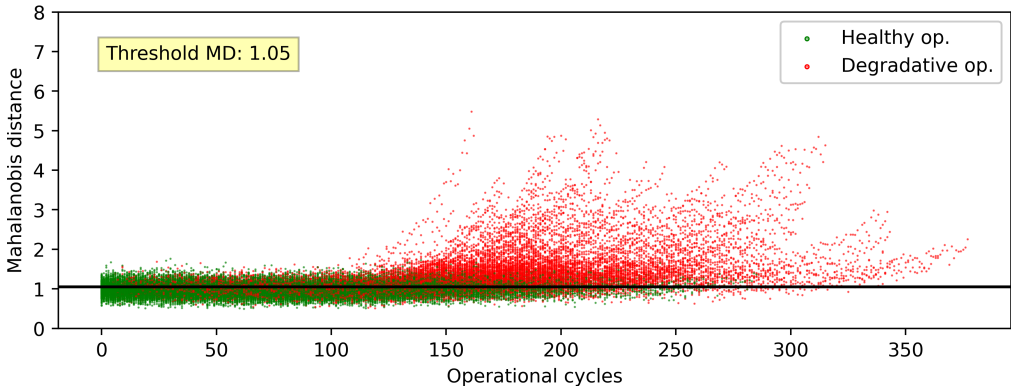
Table 5.4: CNN model hyperparameter search space for C-MAPSS

Hyperparameter	Search range
# CNN layers	[1 - 5]
# CNN filters	[8 - 256]
CNN filter height	[1 - 6]
CNN filter width	[1 - 6]
CNN activation function	[`tanh`, `relu`]

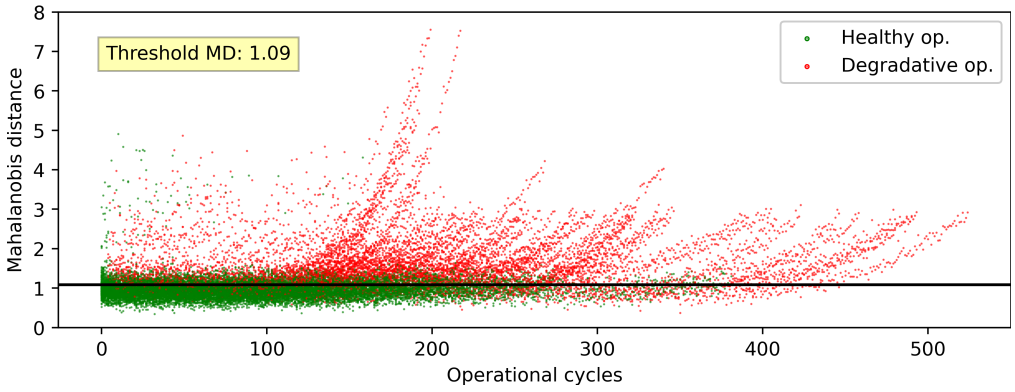
For the Bayesian optimization of the primary networks, a simple MLP structure of 2 layers with 256 neurons each, and a ReLU as activation functions was used. However, when already decided for a primary architecture, another Bayesian optimization is done to search



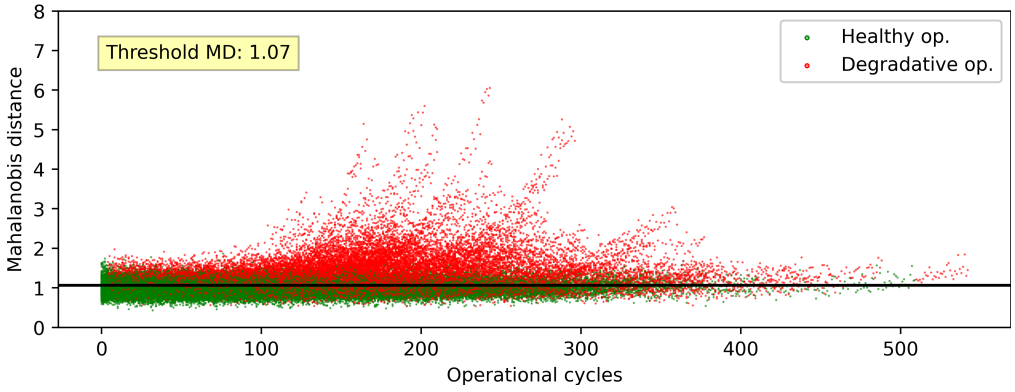
(a) FD001



(b) FD002

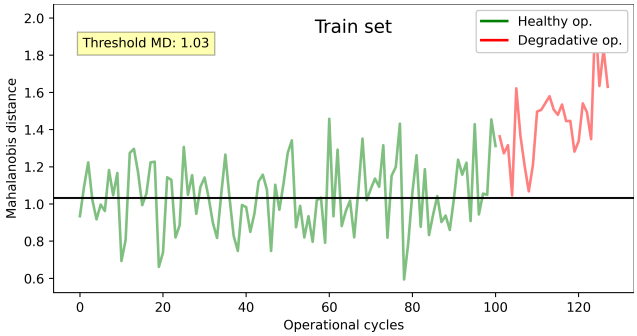


(c) FD003

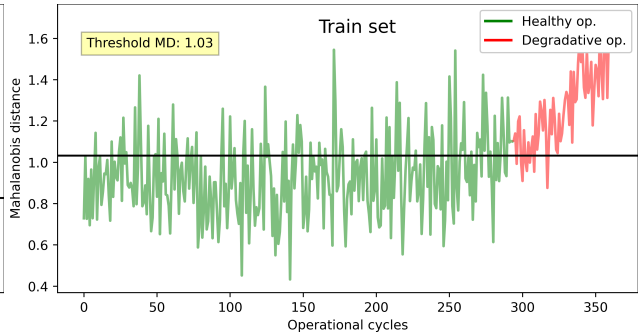


(d) FD004

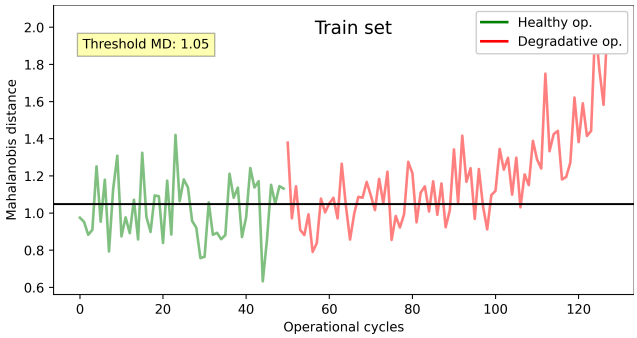
Figure 5.5: MD values for all samples in training sets. In green is identified the healthy operation portion, in red the degradative portion and in black the threshold.



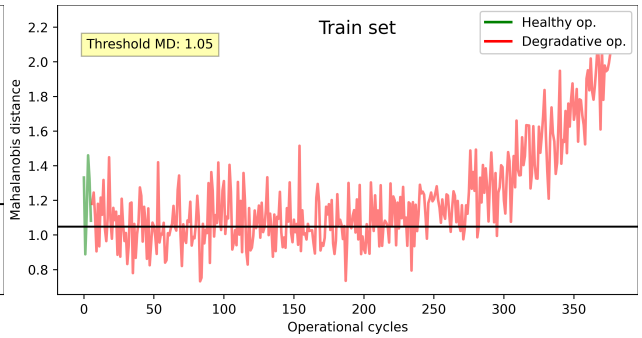
(a) Shortest sample train set FD001



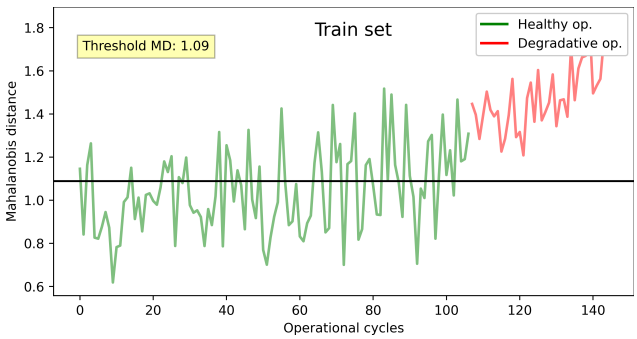
(b) Longest sample train set FD001



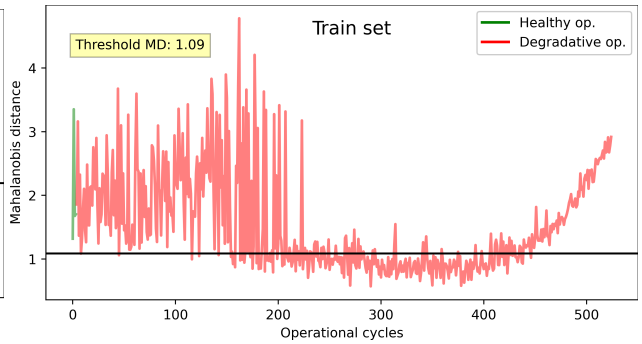
(c) Shortest sample train set FD002



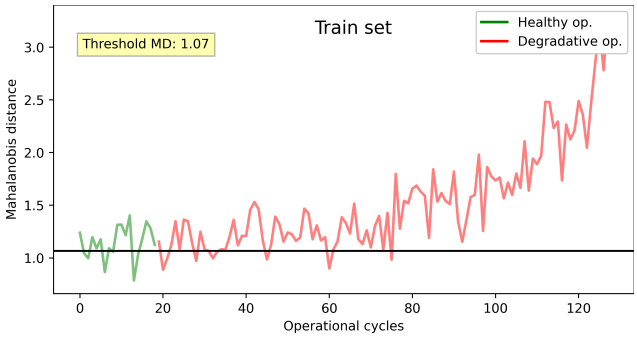
(d) Longest sample train set FD002



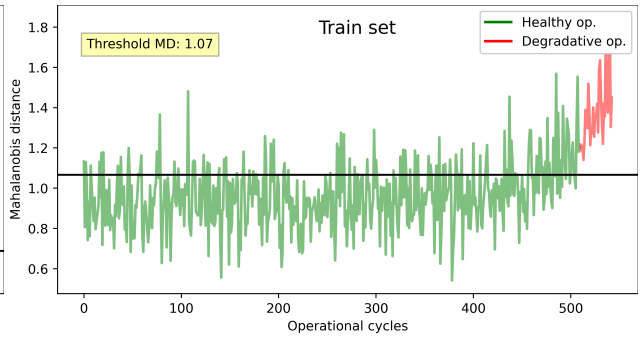
(e) Shortest sample train set FD003



(f) Longest sample train set FD003



(g) Shortest sample train set FD004



(h) Longest sample train set FD004

Figure 5.6: MD curves for shortest and longest sample in each subdataset. In green is identified the healthy operation portion, in red the degradative portion and in black the threshold.

Table 5.5: RNN model hyperparameter search space for C-MAPSS

Hyperparameter	Search range
# LSTM layers	[1 - 5]
# LSTM hidden units	[8 - 256]
LSTM activation function	[`tanh`, `relu`]

for the best hyperparameters for the MLP, as well as the dropout rate used before its input layer, added for network regularization. In table 5.6 are shown the ranges for each of these parameters in search.

Table 5.6: MLP hyperparameter search space for C-MAPSS

Hyperparameter	Search range
Dropout rate	[0 - 0.5]
# MLP layers	[1 - 5]
# MLP hidden units	[8 - 256]
MLP activation function	[`tanh`, `relu`]

For the bayesian hyperparameter optimization processes described, 100 iterative calculations are given for each of the 2 models to review, as well for the MLP optimization. For every iteration, the ANN is trained until it converges to a minimum loss value. Because of the heavy resources needed for this procedure, it is only assessed with dataset FD001, and more over, 20% of the training set is used as validation set for this step. The network architecture found to have the best performance is used for the remaining datasets. [1]

It is to be noted, that only the data after degradation detection is used to train the networks during the Bayesian optimization; data which is preprocessed by scaling each sensor in the range between 0 and 1. A time window of 19 flight cycles is imposed as input data, as it contains sufficient length to get the general trend of the data and get with it an accurate prediction. For the CNN model, the data will form an input image of shape 19×14 pixels, while for the RNN model the input data will have a sequential format of 19 vectors of dimension 14.

5.4.2 Selected model

After the first 2 searches, the CNN architecture is selected over the RNN as it performs as good as it but it takes less time to train. The optimal CNN found has 4 convolutional layers, producing 168 feature maps at each layer with ReLU as activation function. The filter used to form these is of shape 1×5 pixels, and zero padding is used to maintain the dimensionality of the input image. The feature maps from the last layer are then flattened, and connected to the following MLP.

[1]The found network will be independently implemented for each sub dataset.

For the MLP, the optimal configuration consists of 2 layers with 200 neurons each and ReLU as activation function. In table 5.7 is presented the complete list of hyperparameters to use. Also, in figure 5.7 is displayed graphically the data flow throughout the model, showing the data shape format at each layer.

Table 5.7: CNN model hyperparameter search space for C-MAPSS

Hyperparameter	Value
# CNN layers	4
# CNN filters	168
CNN filter shape	1×5
CNN activation function	`relu`
Dropout	25 %
# MLP layers	2
MLP hidden units	200
MLP activation function	`relu`

5.5 Results and discussion

With both stages of the Framework trained, it is implemented to the testing set to validate its performance. The results obtained are presented for each stage individually.

5.5.1 Degradation assessment stage

First of all, in figure 5.8 are plotted the resulting MD curves for all of the samples in the 4 testing sets. Unlike for the MD curves in the training set (figure 5.5), it can be clearly seen that not all the testing engines reach that increasing trend which correlates with the degradative operation just before failure. This notable difference is expected as no engine in these test sets reach failure. Even more, as these samples end at a random cycle before failure, there are samples that do not reach the degradation start threshold. In table 5.8, is presented the number of samples that reach and do not reach the criteria. As is meant with this framework, the undegradaded samples are not analyzed by the following RUL prediction stage.

Table 5.8: C-MAPSS testing set samples with and without degradation detected

Dataset	**Number of testing samples**	**Number of samples with detected degradation**	**Number of samples with no detected degradation**
FD001	100	52	48
FD002	259	130	129
FD003	100	55	45
FD004	258	127	131

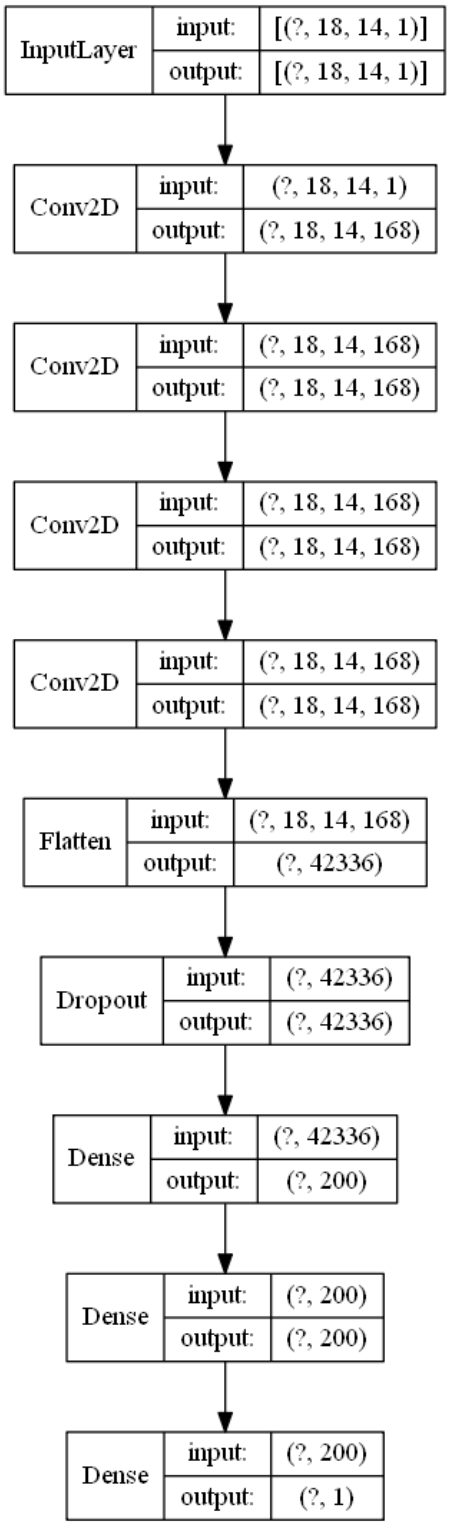
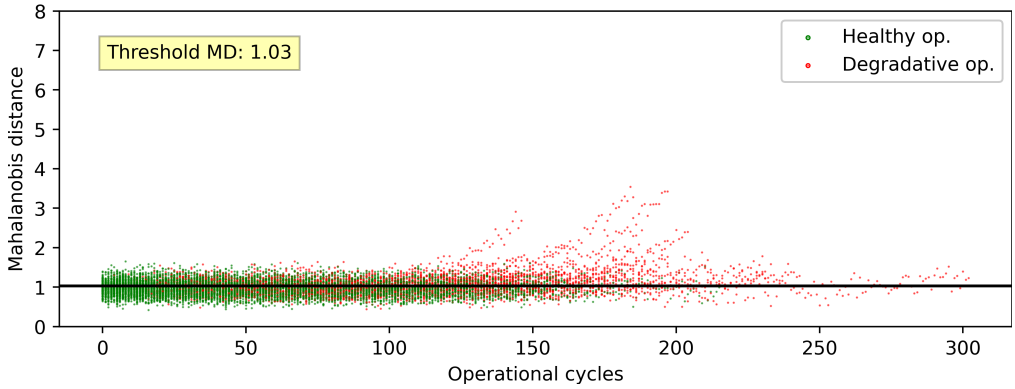
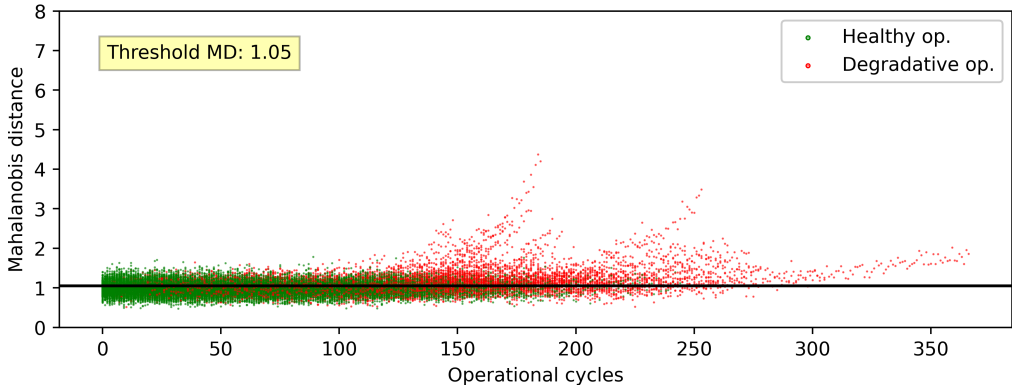


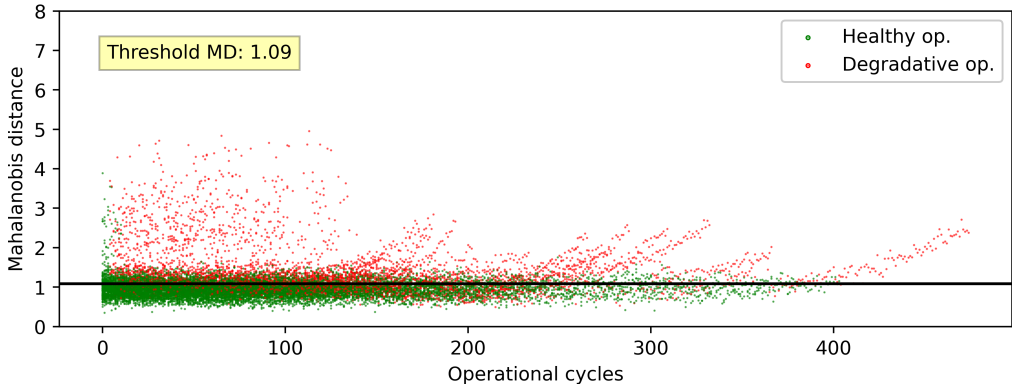
Figure 5.7: Complete CNN architecture for the C-MAPSS RUL ANN. Data shape for the input and output at each layer is displayed



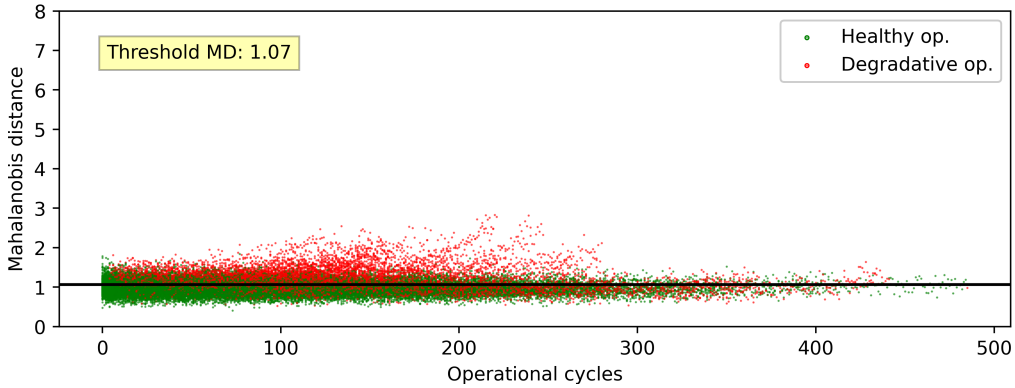
(a) FD001



(b) FD002



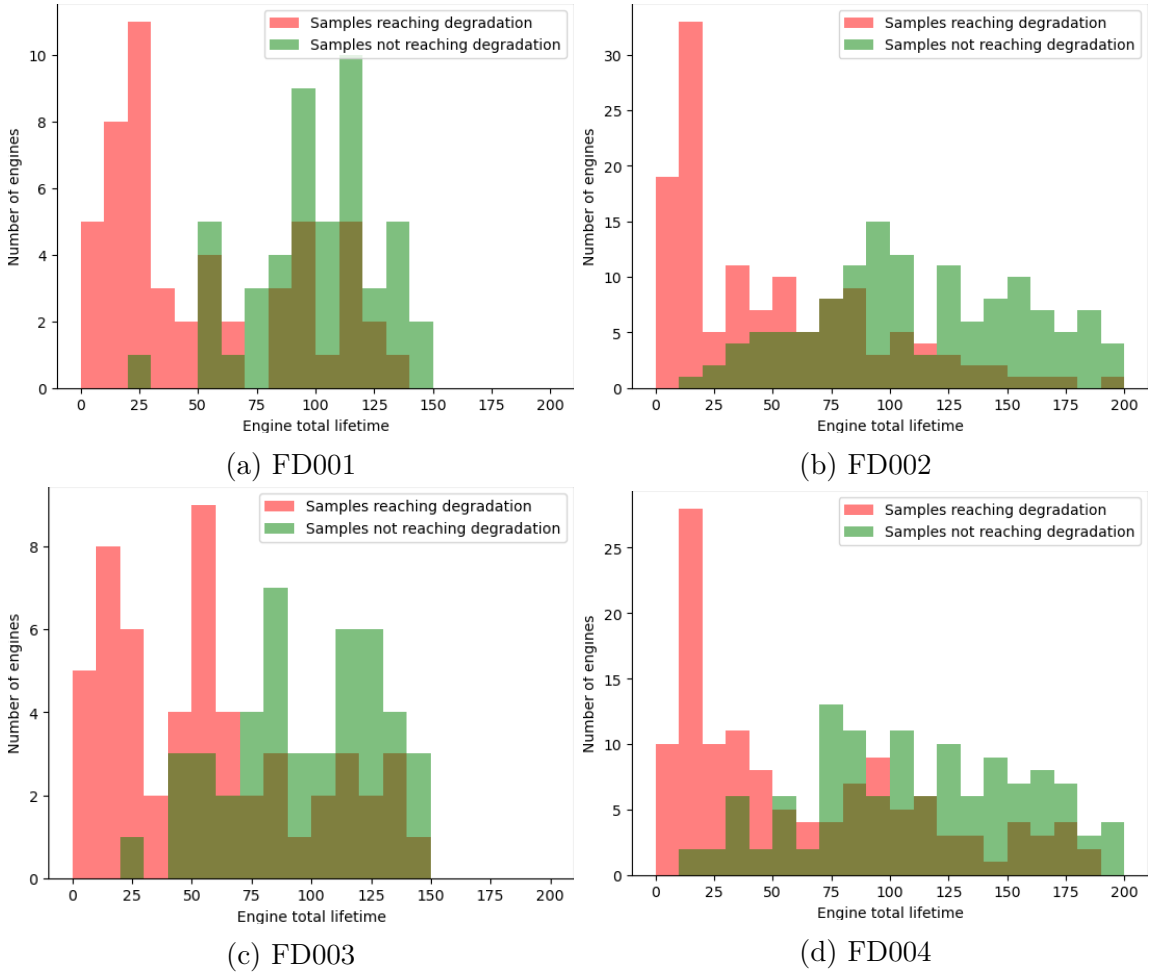
(c) FD003



(d) FD004

Figure 5.8: MD values for all samples in training sets. In green is identified the healthy operation portion, in red the degradative portion and in black the threshold.

Moreover, as shown by the RUL distribution of all the test set samples (figure 5.9), degradation start threshold tends to filter out (in green) the samples that have higher RUL, while sample with lower RUL cross the degradation threshold. Once again, it can be confirmed that the proposed method is capable of correctly discerning between samples with a healthy condition to those operating closer to failure.



(a) FD001

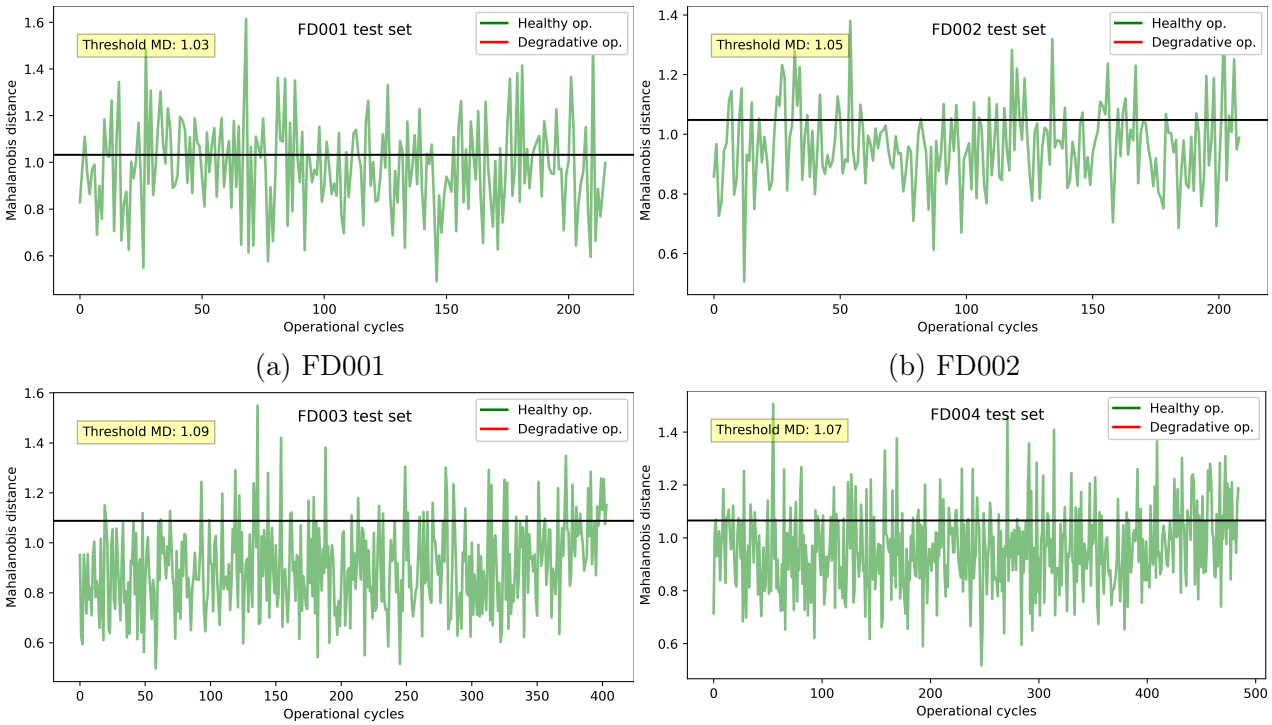
(b) FD002

(c) FD003

(d) FD004

Figure 5.9: Distribution of test set samples by actual RUL with and without degradation detected

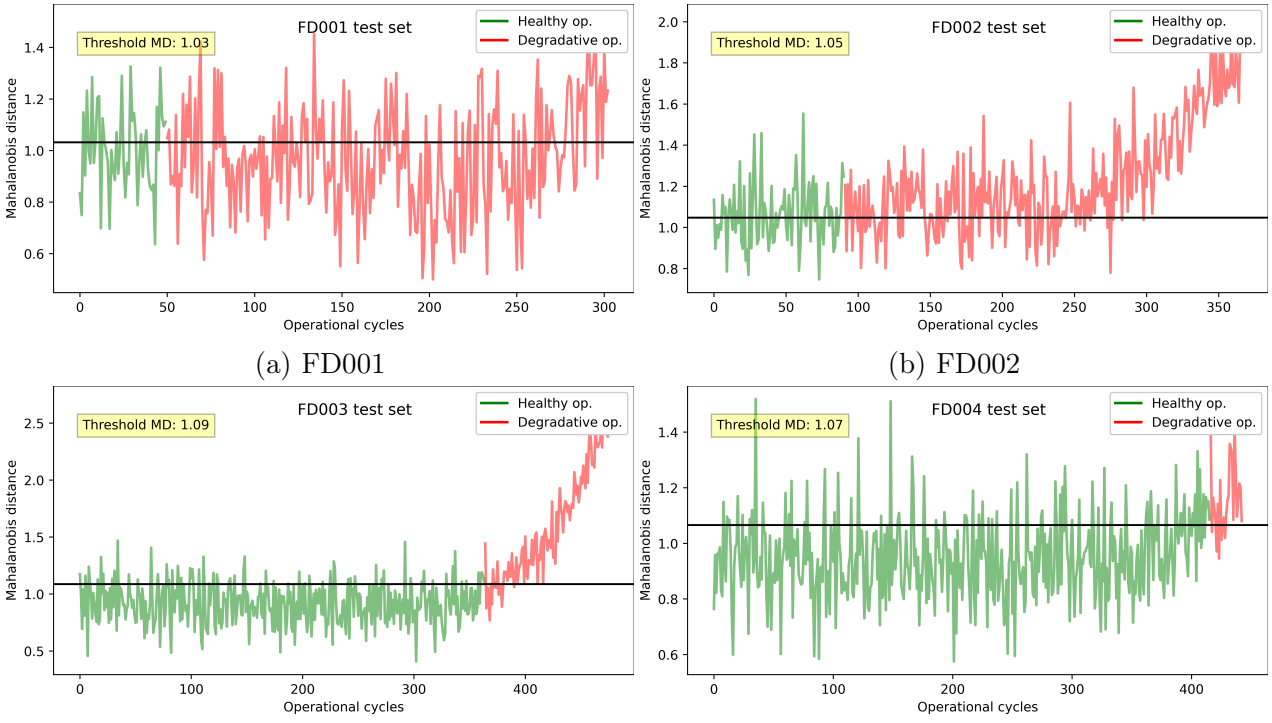
Lastly, to further assess the degradation detection criteria, in figures 5.10 and 5.11 are plotted MD curves for individual samples, specifically, for the longest running engines without degradation detected and for the longest running engines with degradation detected. For the first undegraded samples, it can be confirmed that the MD curve is flat with no clearly tendency yet, despite being a highly noisy. For the second case with longest samples with degradation detected, for the examples of datasets FD003 and FD004 the degradation start criteria correctly identifies the degradation starting point. In the FD002 example the degradation start is detected well before the actual change in trend occurs and in the FD001 case, degradation detection when there is no clear trend in the complete signal. This 2 last cases imply that the detection algorithm sometimes fails and it may need future improvements.



(a) FD001 (b) FD002

(c) FD003 (d) FD004

Figure 5.10: MD curves for the longest running engines that did not reach degradation.



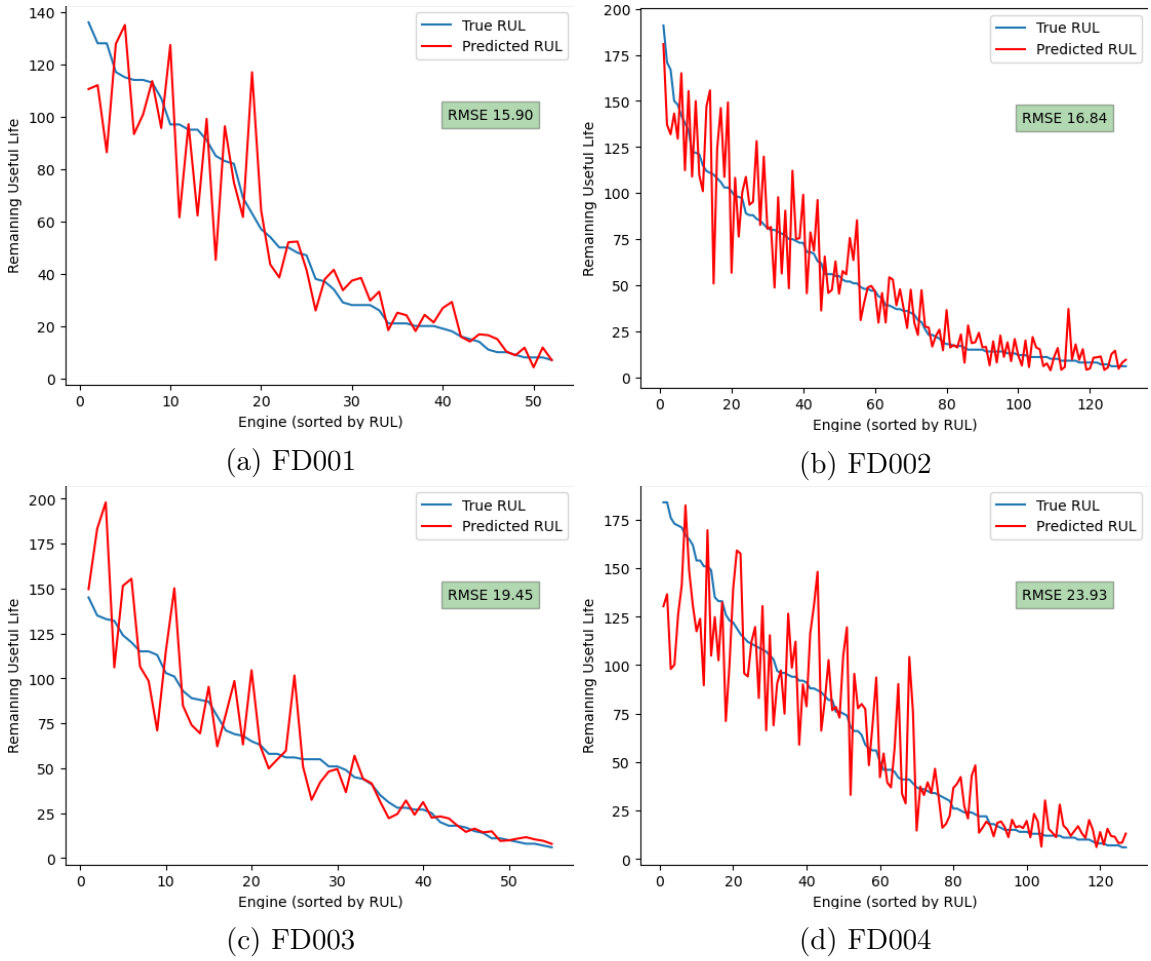
(a) FD001 (b) FD002

(c) FD003 (d) FD004

Figure 5.11: MD curves for the longest running engines that did reach degradation.

5.5.2 RUL prognostics stage

The test samples that crossed the degradation threshold in the stage before are then processed by the ANN to perform RUL estimation, as they should have noticeable trends to make an accurate prediction. The resulting predictions for the last temporal instance given for the engine are plotted in figure 5.12, ordering the samples from higher to lower RULs on the x axis.



(a) FD001

(b) FD002

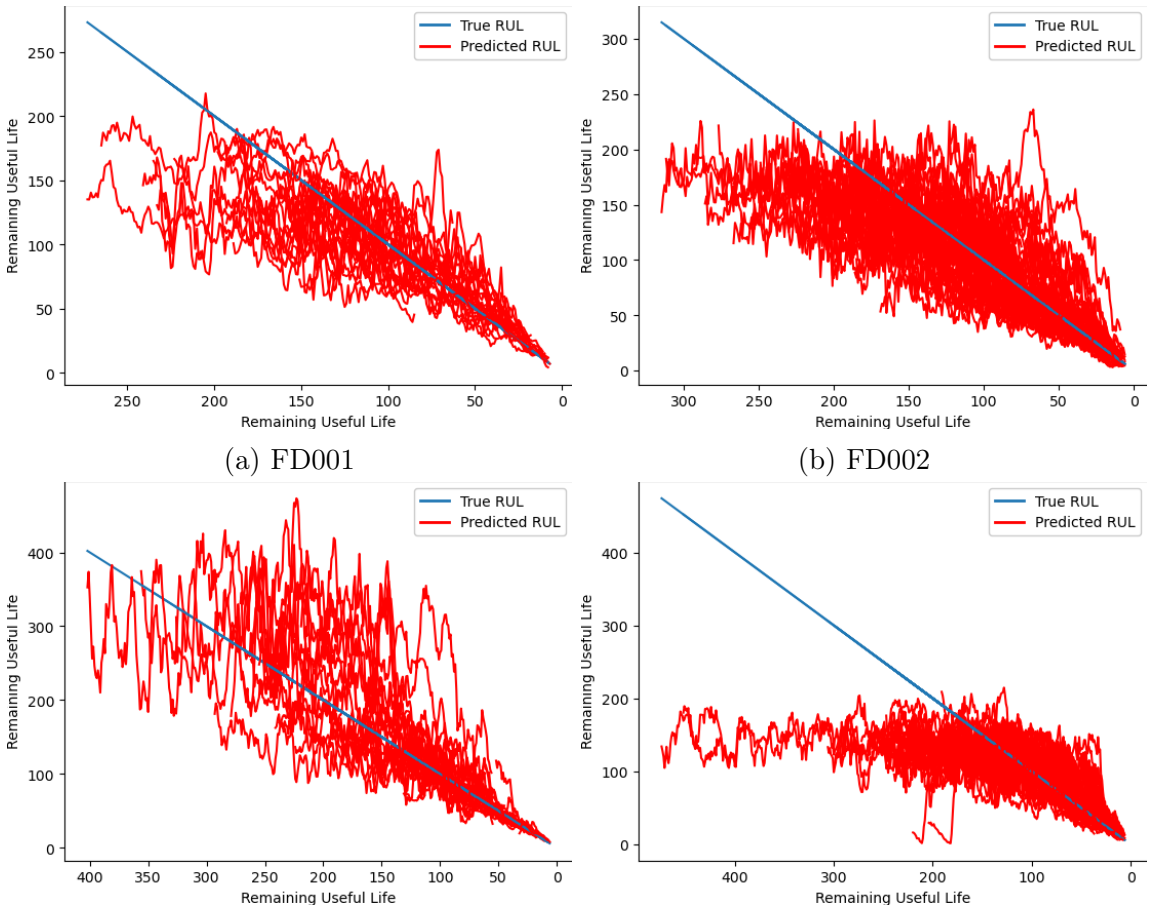
(c) FD003

(d) FD004

Figure 5.12: RUL prediction on test set samples under degradation

These plots show that there is a correspondence between the predicted RUL and the true RUL of the engines. Despite these curves having a characteristic noisy behaviour, the accuracy of the predictions during the end stages of operation of the engine tends to increase, giving results that are very close to its real value. This is en extremely important property for the model as it is at these stages were the most accurate predictions are needed, specially for an aircraft's turbofan engine.

The tendency of increasing accuracy towards the engine's end stages of life can also be seen as a general behaviour when plotting the predictions for all instances of the testing samples under degradation, as shown in figure 5.13. Moreover, it can be seen how the network has "trouble" predicting values above 200 RUL, which is specially noticeable for the FD004 case, as a few early degradation detection occurred which resulted in a mostly flat RUL curve.



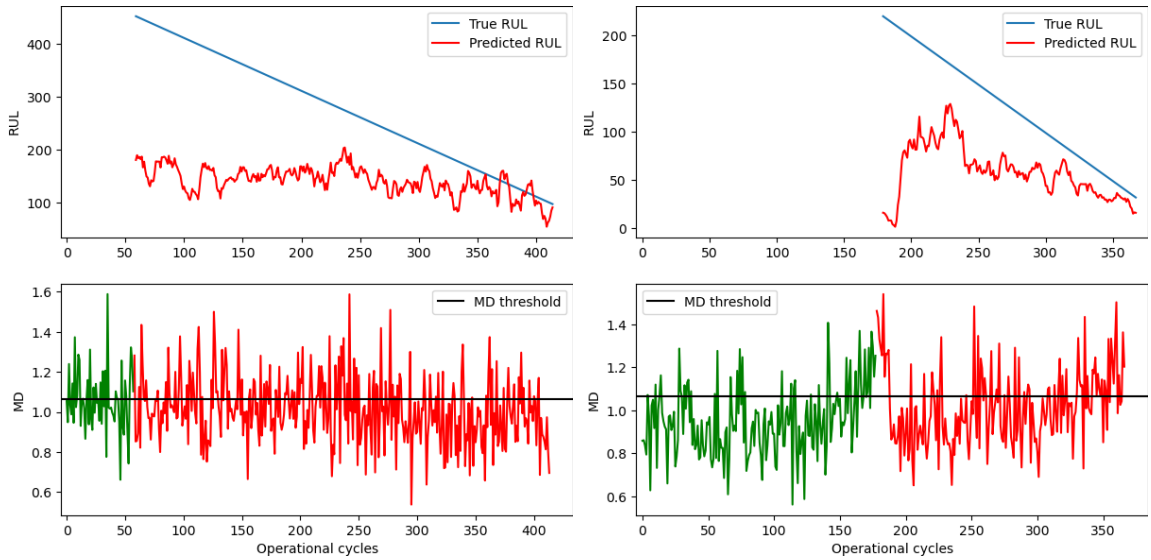
(a) FD001 (b) FD002

(c) FD003 (d) FD004

Figure 5.13: RUL prediction on test set samples under degradation for all time instances

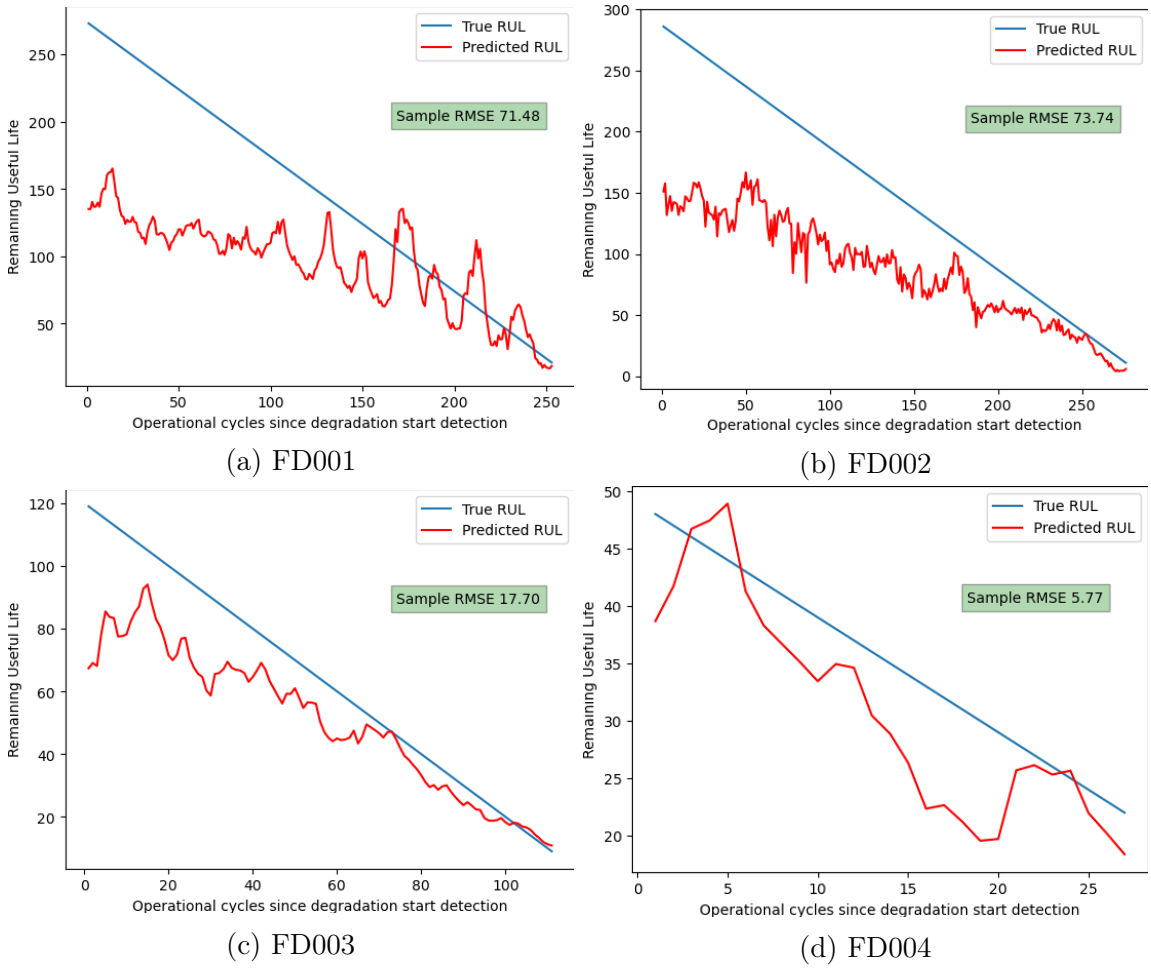
From the plots above, there are 2 types of "anomalous" predictions within the FD004 test set that deserve to be analyzed in more detail: samples with relatively high real RUL but low predicted value, and 2 samples that show a sudden and significant increase in the predicted RUL value. Samples 41 and 180 of the testing set represent this cases respectively, and in figure 5.14 are plotted their individual RUL predicted in parallel to the their MD curve. For the first case, it can be easily checked that its detected degradation starting point is a "false positive", as their is no actual increasing trend in the MD curve when it satisfied the criteria. For the second case, the sudden jump in RUL correlates well with a sudden drop in the MD. This may be due to a random health regeneration, a characteristic randomly added to the samples to simulate corrective maintenance over the engines. This could also explain some of the noisy components generally present on the predictions.

Continuing with an analysis for individual samples, in figure 5.15 are plotted the real and predicted RUL curves for the longest running engines that actually crossed the degradation threshold during their lifetime; same samples for which their MD curve was plotted in figure 5.11. As with the results presented before, these samples display the same property of increasing accuracy towards their end of life. However, the RUL predictions for the samples of FD001 and FD002 evidently underestimate the real curve, which directly correlates to the early degradation start detection that happened for these engines.



(a) Sample 41 (b) Sample 180

Figure 5.14: Samples of interest in FD004 test set



(a) FD001 (b) FD002

(c) FD003 (d) FD004

Figure 5.15: RUL prediction for longest samples (total op. cycles) in test set

5.5.3 Comparison with other approaches

The main objective of the C-MAPSS data challenge was to predict RUL for all the engines in the test sets, quantifying the accuracy of the predictions by the RMSE between the true and predicted RUL at the final temporal instance given. However, RUL predictions with this Framework can only be made when the asset is detected to be operating under a degradative condition. Because of this, results from the Framework are not completely comparable to other proposed approaches that predicted for all samples.

Due to this last issue, the proposed Framework effectiveness is compared with the same CNN architecture developed but without degradation assessment and healthy data filtering out. Therefore, with this baseline CNN the complete training set is used and all the testing samples are analyzed. Its results are compared with the Framework in two ways:

- Comparing the predictions for the samples that reach the degradation start criteria with the Framework
- Comparing the predictions for the samples that did not reach degradation start criteria with the Framework.

In the following table, the RMSE mean and standard deviation values after 10 training runs are giving for both the Framework and the baseline CNN, for each of the 4 dataset, as a measure of the accuracy of the predictions.

Table 5.9: RUL RMSE comparison between studied approaches.

Dataset	**Proposed Framework (unhealthy samples)**		**Baseline CNN (unhealthy samples)**		**Baseline CNN (healthy samples)**	
	Mean	std.	Mean	std.	Mean	std.
FD001	17.2	0.6	17.7	1.3	27.9	1.6
FD002	18.4	1.4	17.5	0.6	40.9	0.6
FD003	25.1	4.1	25.7	3.5	43.2	4.8
FD004	26.2	1.1	26.7	0.8	35.5	1.3

In the first place, when comparing the predictions for the unhealthy samples, it can be easily identified that there is no meaningful difference between the approaches. From this, it is implied that there is no increase in the accuracy of the predictions by filtering out healthy operation data. This similarity between the approaches imply that healthy data does not posses meaningful information to increase the accuracy of prediction on the remaining useful lifetime of the asset, which supports one of the main motivations for the degradation assessment approach.

Secondly, the RUL predictions with the Baseline CNN for the healthy samples (ignored by the Framework) show that there is a significant increase in the estimation error. This again supports the main hypothesis for the Framework that healthy data does not provide meaningful information for accurate RUL predictions, and therefore should be ignored.

Finally, focusing on the dataset itself, there are significant differences on the prediction's accuracy between each of the subdatasets. This can be primarily explained by the number of failure modes present on the engines, which for datasets FD001 and FD002 was only 1 meanwhile for FD003 and FD004 there were 2. The added failure mode increased RUL prediction error by 50% approximately. On the other hand, the number of operating conditions (1 for FD001 and FD003 and 2 for FD002 and FD004) does not significantly affect the model accuracy.

5.6 Concluding remarks

The results obtained for the proposed Framework leads to the conclusion that the Mahalanobis Distance can be effectively used for quantifying degradation on a physical asset, without the need to individually analyze each sensor reading. Even more, it defines a space for healthy data without the need to manually establish ranges for each sensor. Therefore, as it can classify between healthy and unhealthy condition, this approach can serve as a condition assessment technique.

Considering degradation start criteria, it is an effective implementation that serves for 2 main points: it eliminates healthy data that does not provide meaningful information for future predictions, and it establishes a threshold after which RUL predictions can actually be made. However, the criteria for MD degradation start detection was not always precise, as it in some cases it gave a "false positive" by indicating degradation start when the overall trend was still flat or started from a higher value. Therefore, another technique for degradation start detection may be an interesting subject for further research.

In regards to the C-MAPSS dataset, for some samples in dataset FD003, the MD metric gave values with extremely high noise at operation start. The reason for this was unknown and only occurring in a few cases. A study on why this occurred and how to avoid it in the future may also result be useful, Moreover, for this dataset, as it produced bu simulations, it was given random noise and random performance increases to emulate maintenance actions on the engine. This increases were also reflected in the MD curves, so MD metric could be used to assess maintenance effectiveness on physical assets as well.

On a final note, even though CNN are specialised for image classification and recognition, this architecture gave better performance than the LSTM commonly used for time series analysis. The difference between these models was not significant, but as CNN offers better computing performance, it was selected over the other. Given this, it is proven that the CNN architecture offers great capabilities for regression problems involving temporal data.

Chapter 6

FEMTO Case Study

FEMTO is another common benchmarking dataset for RUL estimation on physical assets, which in this case are rolling bearings. As C-MAPSS, FEMTO was presented as a data challenge for the 2012 IEEE PHM international conference. In this chapter is described the implementation and the results obtained with the proposed Framework for degradation assessment and RUL estimation.

6.1 FEMTO dataset

The FEMTO dataset was produced experimentally on an accelerated ageing platform called PRONOSTIA, that enables accelerated degradation of bearings under constant and/or variable operating conditions, developed by the FEMTO-ST Institute (Besançon - France). This platform is dedicated to test and validate fault detection, diagnostic and prognostic approaches on bearings, providing real experimental data that characterises their degradation through their whole operational life (until failure). Through the PRONOSTIA platform it was possible to conduct bearing's degradation in only a few hours of operation under load, without the need to manually generate defects on the bearings; therefore obtaining normally occurring defects due to the degradation on the balls, rings and cage. As shown on figure 6.1, the experimental platform is composed of three main parts: a rotating part, a degradation generation part (with a radial force applied on the tested bearing) and a measurement part.

The dataset consist of vibrational and thermal data of 17 rolling bearings run until failure, under the following 3 operation conditions:

- First operating conditions: 1800 rpm and 4000 N
- Second operating conditions: 1650 rpm and 4200 N
- Third operating conditions: 1500 rpm and 5000 N

From the total 17 bearings, 7 are under condition 1, 7 under condition 2 and 3 under condition 3, each named accordingly (from "Bearing1_1" to "Bearing 3_3"). As stated before, their vibrational and thermal data was recorded. However, thermal data was not registered for all of the examples so it is excluded from the analysis.

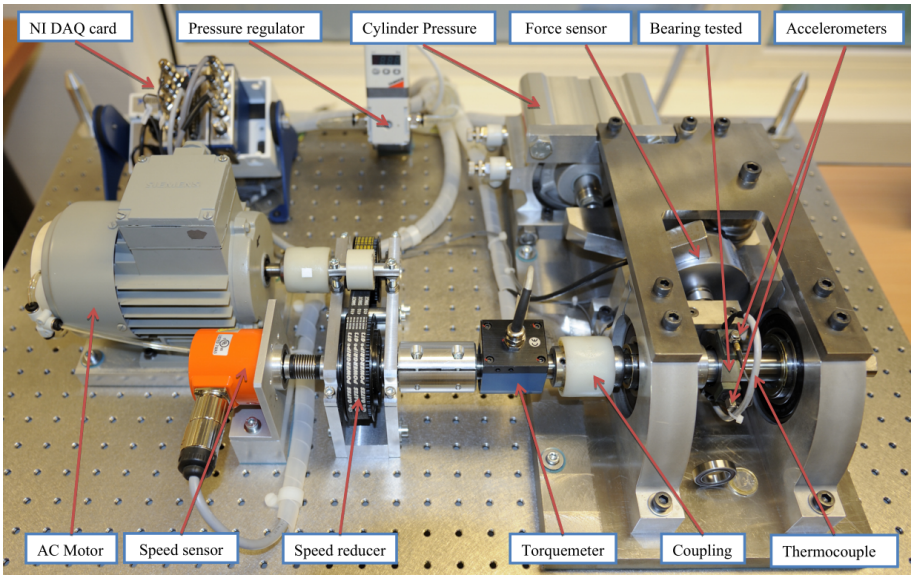


Figure 6.1: PRONOSTIA experimental platform

The vibrational data given with the dataset correspond to the acceleration perceived on both horizontal and vertical axis of the bearing, being the horizontal axis the one in which the load was applied. Records were taken every 10 seconds, for lapse of 0.1 seconds with a sampling rate of 25600 hertz. Therefore, every 10 seconds, 5120 (2560 x 2) data points were obtained. In figure 6.2 is plotted as an example the raw vibration on the horizontal and vertical axis for bearings 1_1, 2_1 and 3_1. The increased vibrations over time is a characteristic present in the other samples as well.

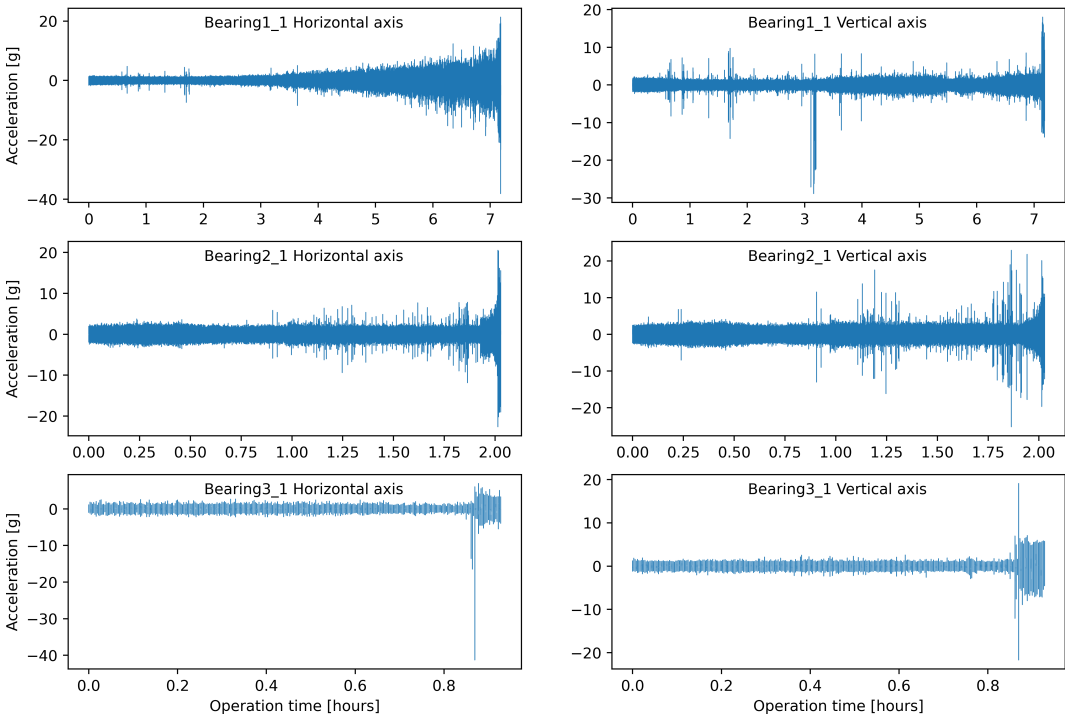


Figure 6.2: Bearings 1_1, 2_1 and 3_1 raw signals

However, there are 3 bearing samples in the FEMTO dataset that do not share this increasing vibrational characteristic, samples 1_2, 2_3 and 2_5 shown in figure 6.3). The first sample shows a highly erratic vibrations, while the last 2 show the opposite pattern of the typical samples: high vibrations at first and an brief ending (implying sudden failure). Due to these, these samples are left out of further analysis.

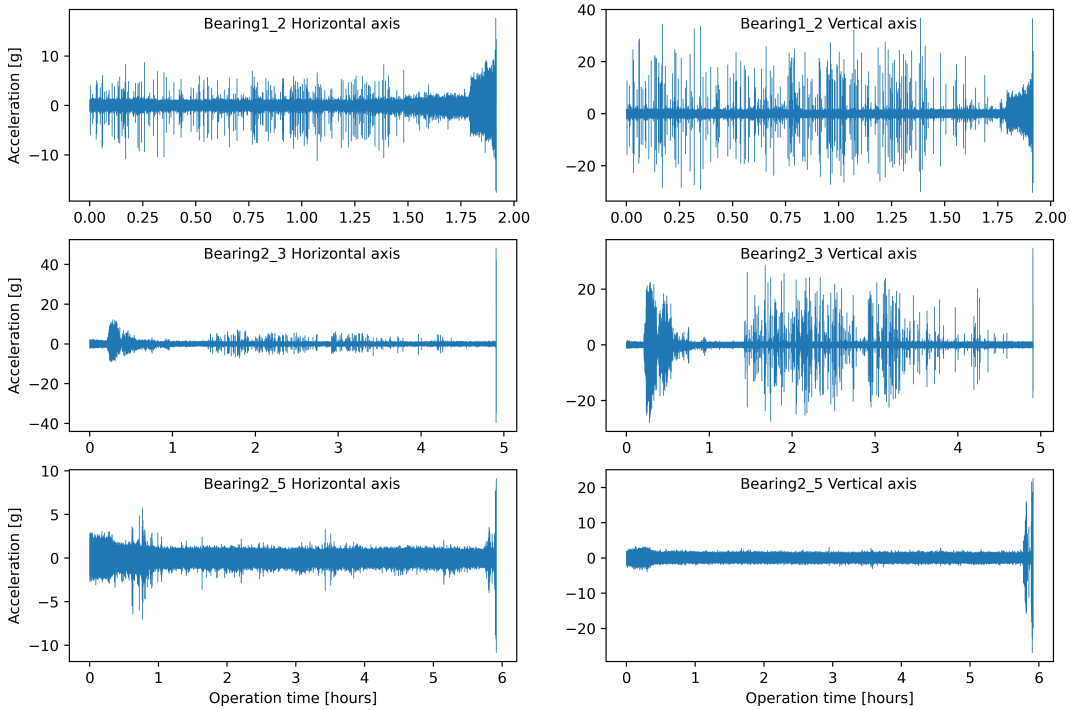


Figure 6.3: Bearings 1_2, 2_3 and 2_5 raw signals

6.2 General remarks and data preprocessing

The total lifetime distribution of the tested bearings range from 1 hour to over 7 hours, as shown in figure 6.4 [1]. From this it can be inferred that no singular distribution can correlate to it. Therefore, this could be interpreted as 2 different distributions: an uniform distribution for the lower end (ranging between 1 and 3 hours); and a second one that resembles a beta distribution (ranging from 4 to over 7 hours). This may imply that 2 different failure modes can occur on the bearings.

Previous works on this dataset have shown that vibrations on the horizontal axis provides a better representation of the degradation process than the vertical acceleration [22, 42], so only this measurement will be considered for this work as well. These studies have used both raw vibrational data [42] and their time-frequency domain representations [22] for RUL prognostics. Nevertheless, the second approach opens the possibility for image processing algorithms such as CNN, which have previously shown good performances for RUL prognostics tasks. Therefore, for this case study, spectrograms of vibrational signal are used to perform RUL estimation.

[1] This include the 3 ignored samples

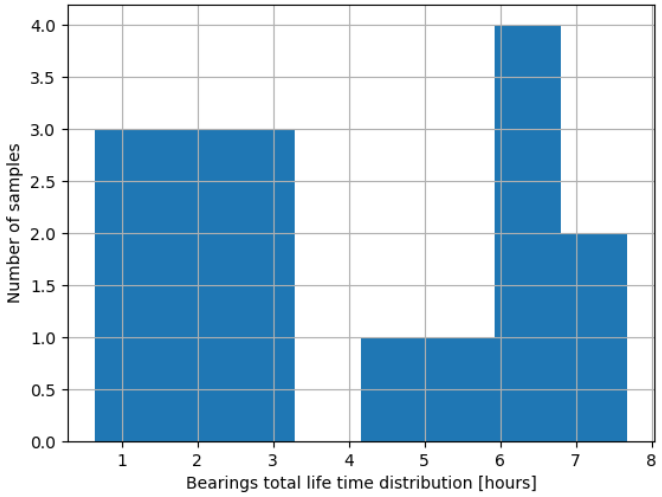


Figure 6.4: Bearings total lifetime histogram

These spectrograms are obtained through short-time fourier transform (STFT), a common technique used for audio signal possessing. The STFT transform the data from the time domain into a 2D representation of the intensity of the composing frequencies; with one axis giving the range of constituent frequencies and on the other its evolution through time. This technique divides the raw signal into segments (*windows*) of length L, and over each segment computes a discrete Fourier transformation of the raw segment multiplied by a window function h that is symmetrical and non zero-valued inside of said segment, as described in equation 6.1. The obtained spectrogram (X) represents a range frequencies (ω) at time (t) for a segment of the raw signal ($x[n]$). Those inspected frequencies ω are up to half the original sampling rare of the signal. The window function employed for the STFT is the Hann function, described in equation 6.2.

$$X[t, \omega] = \sum_{n=-\infty}^{\infty} x[n]h[n-t] \exp^{-j\omega n} \tag{6.1}$$

$$h(n) = 0.5 - 0.5 \cdot \cos\left(\frac{2\pi n}{L-1}\right) \tag{6.2}$$

The length of the time window L has significant influence over the quality of the spectogram; a window too large has a better resolution of the constituent frequencies but too little on the time scale, a window too small provides an spectogram with better resolution on the time scale but too little on the frequencies. For this dataset, a time window of 2.5 [ms] (64 data points) with no overlapping between each segment is selected, as it provides a good trade-off between the time and frequencies resolutions; giving temporal of information for every 2.5 [ms] for resolutions up to 12,800 [hz] in bands of 400 [hz].

For the actual implementation of the proposed Framework, a single spectrogram is formed from a single sensor measurement, that is, an spectogram for every 10 seconds. Given the

selected window of 2.5 [ms] for the STFT, each spectrogram has a dimension of 40×32 pixels, as presented in figure 6.5.

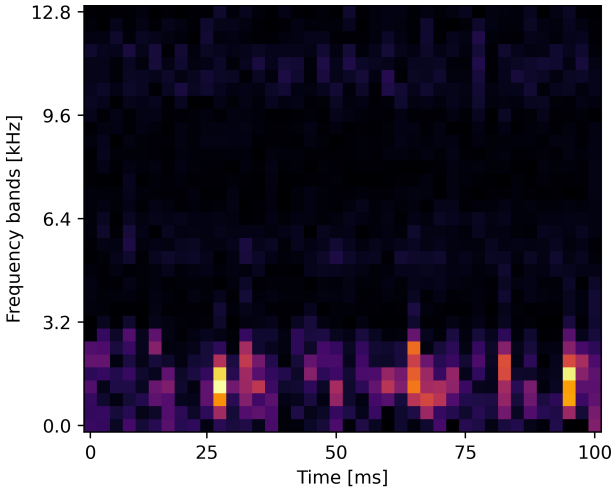


Figure 6.5: Sequence of spectograms for the last minute of operation of bearing 1_1

These spectrograms are to be used for RUL estimation, however, for the degradation assessment stage they could not be used due to significant amounts of raw data, which would require important computational time to determinate the correlation matrix necessary for the Mahalanobis space (in the each frequency band was considered as composing variable for example). Therefore, for the first degradation detection stage, only statistically relevant features from the raw signal are used, specifically the following 8: mean, standard deviation, minimum, maximum, root mean square, peak-to-peak, crest factor, skewness and kurtosis. These features are taken from each measurement of 0.1 seconds, and their equations are summarised in table 6.1.

Table 6.1: Selected features and their equations

Feature	**Equation**
Mean ($\bar{x}$)	$\frac{1}{N}\sum_{i=1}^{N} x_i$
Standard Deviation (σ)	$\sqrt{\frac{\sum_{i=1}^{N}(\bar{x}-x_i)^2}{N}}$
Root Mean Square (RMS)	$\sqrt{\frac{\sum_{i=1}^{N}{x_i}^2}{N}}$
Peak-to-peak (PTP)	$\max(x_i) - \min(x_i)$
Crest factor (cf)	$\frac{\lvert\text{PTP}\rvert}{RMS^2}$
Skewness	$\frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i-\bar{x}}{\sigma}\right)^3$
Kurtosis	$\frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i-\bar{x}}{\sigma}\right)^4$

6.3 Degradation assessment stage

In this section is described the implementation of the degradation assessment stage using the raw data features discussed above to do so. First are selected the appropriate hyperparameters for degradation detection and then the obtained results are analyzed. As only 14 samples are left out in the dataset, the same hyperparameters are applied to all the samples.

6.3.1 Hyperparameter selection

There are 2 hyperparameters to be found at this stage: the sigma-threshold and the degradation start criteria. The same methodology followed for C-MAPSS is followed here: first a degradation start criteria is imposed and then the sigma-threshold is selected.

As the shortest bearing lifetime left in the dataset consists of 230 (bearing 2_7) measurements while the longest consists of 2767 (bearing 1_1), and given that some samples have sudden vibrational increases leading up to failure, it is imposed that a degradation start criteria of 4-out-of-4 consecutive MD values above threshold is suitable enough to detect in time both slow-paced and fast-paced degradation processes.

The selection of the sigma-threshold is arbitrarily as well, but with 2 restrictions to consider: first, all samples must cross the degradation failure at some point because all samples in the dataset reach failure and second, the detected degradation start must coincide with change of trend of the MD curve (from flat to exponential / linear increase). Given these restriction, it is found that a sigma-threshold of $\sigma = 2$ satisfy them for all the bearing samples. In table 6.2 are summarised the selected parameters.

Table 6.2: Mahalanobis Space algorithm Hyperparameters for degradation detection on FEMTO

Hyperparameter	**Value**
Degradation start criteria	4-out-of-4
σ Threshold	2

6.3.2 MS and degradation start detection

By using the using the selected hyperparameters with the Mahalanobis space search algorithm, the healthy portion of the dataset is obtained. After 9 iterations, the MS converges to about the 84% of the complete dataset, as seen in the plot of figure 6.6. With this, it is obtained an **MD threshold of 1.71**.

In table 6.3 are presented the operational time for degradation start detection and the corresponding RUL after that point, for all the 14 samples. The resulting degradation starting points differ widely, producing samples with predictable RUL ranging from 40 seconds upwards to 8,730 seconds; 3 orders of magnitude in difference. In the third column, is presented the proportion between the total time under degradation and the total operative lifetime, which also shows high variance. Some samples live shortly under this process, and other live up to 54% of their total lifetime under this condition.

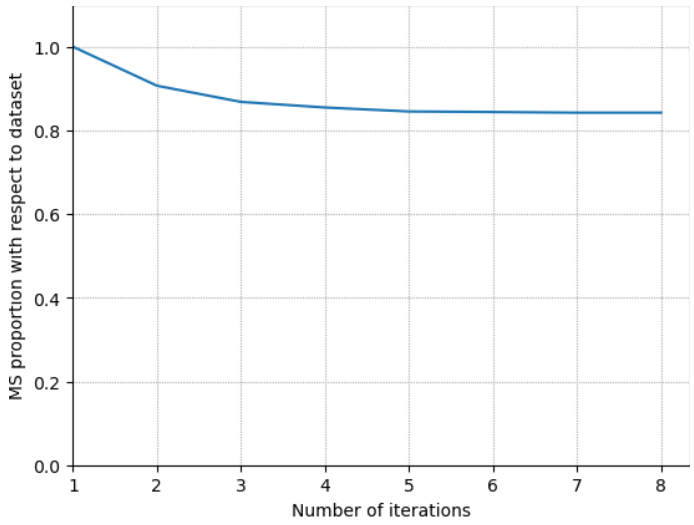


Figure 6.6: Single spectrogram used

These difference on the bearing samples can be further analyzed with figure 6.7, where the obtained MD curves are plotted; identifying in green the instances in the healthy state and in red the instances after degradation start detection. Nevertheless, the detection algorithm can detect the change in the curves trend with good accuracy in the majority of cases, with exception of bearing 2_2 where early detection seems to have occurred. Also, from these plots is evidenced that in many cases the MD drastically increases before failure, reaching values well above he MD threshold. This implies that the degradation process quickly ends in failure.

Table 6.3: FEMTO bearings MD degradation detection

Sample	MD deg. start detection (s)	RUL after deg. detection (s)	% of lifetime after deg. start
Bearing 1_1	19.810	7.860	28,4
Bearing 1_3	17.560	5.410	23,6
Bearing 1_4	10.850	550	4,8
Bearing 1_5	24.140	490	2,0
Bearing 1_6	16.330	8.150	33,3
Bearing 1_7	22.120	470	2,1
Bearing 2_1	8.470	640	7,0
Bearing 2_2	7.540	430	5,4
Bearing 2_4	3.450	4.060	54,1
Bearing 2_6	6.930	80	1,1
Bearing 2_7	2.270	30	1,3
Bearing 3_1	4.930	220	4,3
Bearing 3_2	14.350	2.020	12,3
Bearing 3_3	4.260	80	1,8

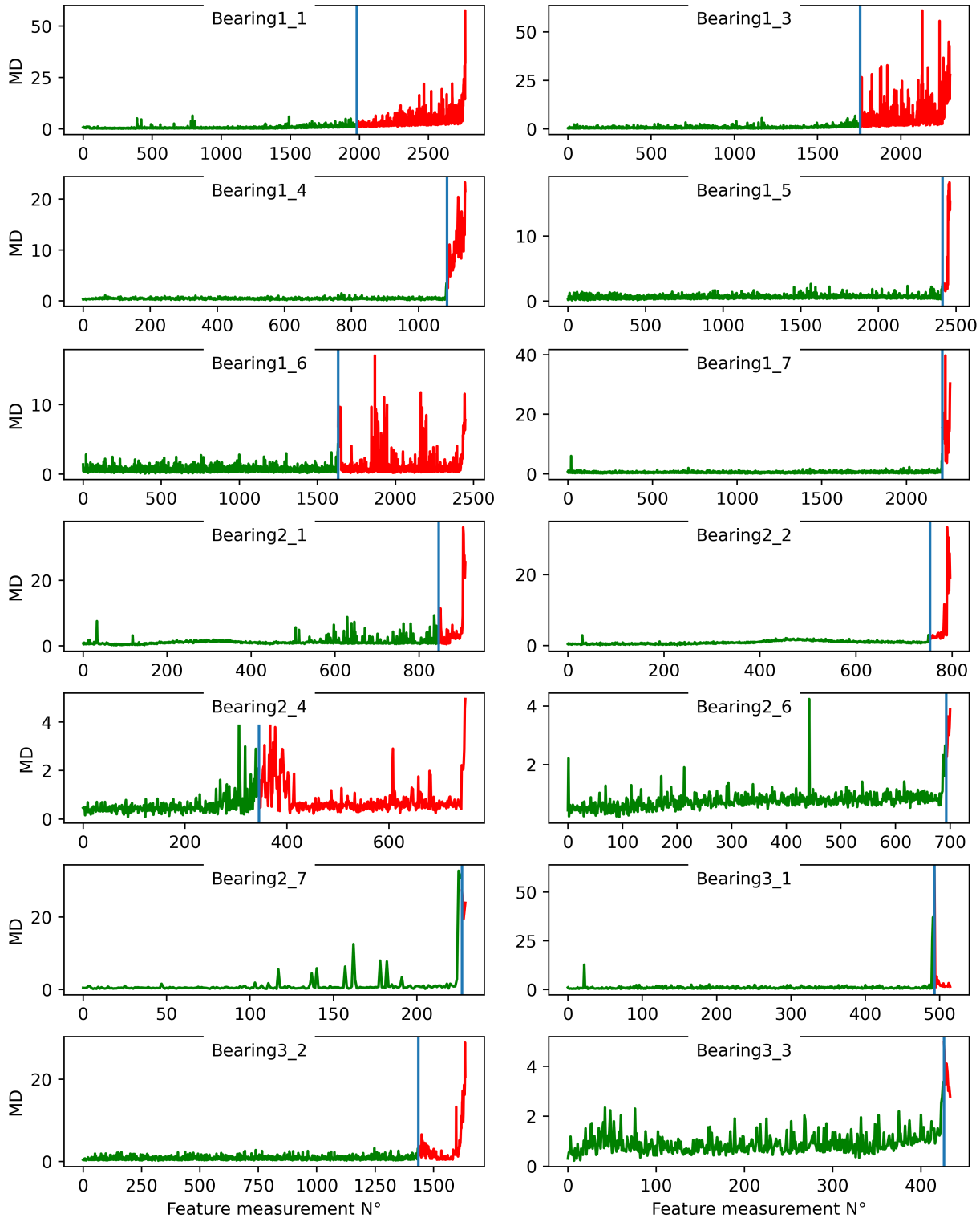


Figure 6.7: MD curves for each FEMTO sample. In green are coloured healthy instance and in red are under degradation, while the blue line identifies the detected degradation starting point.

6.4 RUL prognostic stage

In this section is described the implementation of the RUL prognostic stage. First is detailed the input data format to be used, and then the hyperparameters are optimized through bayesian search. Then the selected ANN is tested on the 14 samples of the datasets. For this case, the network is tested through the leave-one-out method, that is, one sample is left as testing set and trained with the remaining 13. This is done 14 times for each sample.

6.4.1 Input data

For RUL prognosis has been decided to use spectrograms of the raw vibrational data as input data to the model. Moreover, to include more temporal information about the ongoing trend, sequences of spectrograms are to be used. Specifically, a sequences containing the last minute of operation is to be used, that is, a sequence of 6 consecutive spectrograms, as shown on figure 6.8.

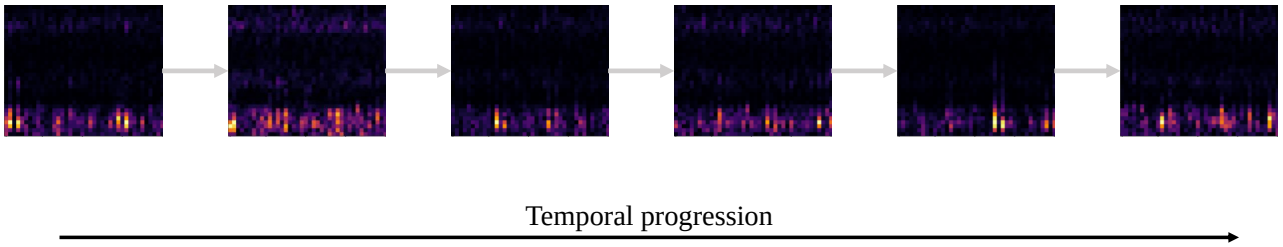


Figure 6.8: Spectrogram sequence to be used as input data

Moreover, to help with the training process, the whole training set spectrograms are scaled at each frequency range between 0 and 1 using min-max scaling (eq. 4.3). The test set spectrograms are scaled accordingly.

6.4.2 Network and hyperparameter selection

Give the input data format of sequences of images, the convolutional LSTM (ConvLSTM) ANN is selected for the RUL prognostics stage. As a general implementation pattern, only the last temporal output of the ConvLSTM is later processed in an MLP for RUL estimation, as this output should have already encoded information of the past timesteps.

The selection of the optimal hyperparameters for this network is done through bayesian optimization, using the RMSE between the prediction and real RUL as cost function to minimize. For the ConvLSTM netowrk, the hyperparameters to optimize are the convolutional filter shape, the number of fiters, the activation function and the total number of ConvLSTM layers. For the MLP, the hyperparameters are the hidden unites per later, the dropout rate at first layer, the activation function and the total numbers of layers. In table 6.4 are summarized the search range imposed of each of the hyperparameters to optimize,

A maximum of 100 epochs is imposed for the search of the optimal configuration. In each epoch, the selected ConvLSTM network is trained until it converges to a minimum loss value,

Table 6.4: ConvLSTM model hyperparameter search space for FEMTO

Hyperparameter	Search range
ConvLSTM	
# ConvLSTM layers	[1 - 5]
# ConvLSTM filters	[8 - 256]
ConvLSTM filter height	[1 - 12]
ConvLSTM filter width	[1 - 12]
ConvLSTM activation function	[`tanh`, `relu`]
MLP	
Dropout rate	[0 - 0.5]
# MLP layers	[1 - 5]
# MLP hidden units	[8 - 1024]
MLP activation function	[`tanh`, `relu`]

as is typically done. In regards to the training data, it is assessed with the training set when the bearing 1_1 is left out as testing sample (this sample is never used nor evaluated during this optimization). From this training set, the 30% of it is left out as validation set.

6.4.3 Selected Model

The optimal configuration found for this ANN is composed by 1 ConvLSTM layer producing 32 feature maps from a filter of shape 10×11, followed by a 5 layer MLP with 656 neurons per layer that takes the last temporal output of the ConvLSTM. The full detail of the obtained hyperparameters, and the graphic data flow throughout the model with the specific data format at each layer, is shown in the diagram of figure 6.9.

Table 6.5: ConvLSTM selected hyperparameters for FEMTO

Hyperparameter	Value
ConvLSTM	
# ConvLSTM layers	1
# ConvLSTM filters	32
ConvLSTM filter height	10
ConvLSTM filter width	11
ConvLSTM activation function	`relu`
MLP	
Dropout rate	0.32
# MLP layers	5
# MLP hidden units	656
MLP activation function	`relu`

For the training process of the network, a patience of 3 epochs is given for learning rate reduction (by 50%, starting from 0.001), an a patience of 5 epochs is given for early stopping. 30% of the training data is left out as validation, in each of the 14 cases to study.

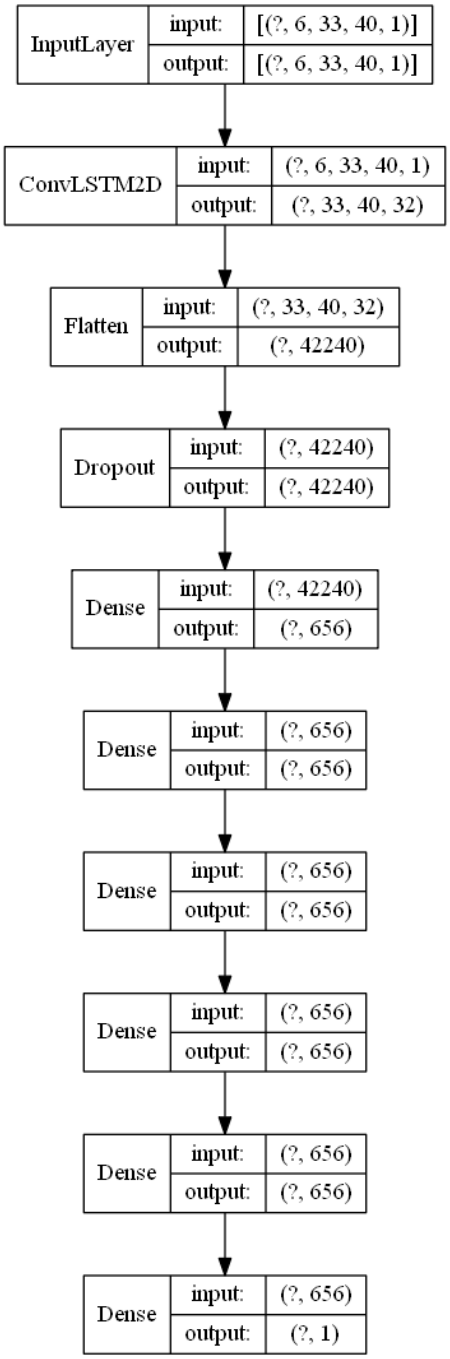


Figure 6.9: ConvLSTM architecture for the FEMTO RUL ANN. Data shape for the input and output at each layer is displayed

6.4.4 RUL estimation results

After the degradation start detection done by the first stage, RUL predictions are made with the selected ConvLSTM configuration. This dataset is evaluated with the leave-one-out method, performing 10 training iterations for each of the 14 cases. The corresponding results are presented in table 6.6.

Table 6.6: FEMTO bearings estimated RUL with ConvLSTM

Sample	**RMSE (s)**			Max RUL (s)
	Best	Average	std.	
Bearing 1_1	2,868	3.243	301	7,860
Bearing 1_3	970	2,369	722	5,410
Bearing 1_4	212	446	528	550
Bearing 1_5	253	687	627	490
Bearing 1_6	3,196	3,682	304	8,150
Bearing 1_7	150	183	34	470
Bearing 2_1	108	365	186	640
Bearing 2_2	84	110	25	430
Bearing 2_4	841	1,770	848	4,060
Bearing 2_6	11	136	109	80
Bearing 2_7	370	2,731	1,733	30
Bearing 3_1	1,174	1,906	477	220
Bearing 3_2	521	1,060	560	2,020
Bearing 3_3	89	853	1,156	80

The accuracy of these predictions evidently differs between each sample, and, even more, differ significantly between each iteration for the same sample. Due to this high variance, in the table it is also shown a column with the best prediction obtained, that is, with the lowest RMSE. The value of the RMSE also depends on the sample size, so in the 4th column is added the maximum RUL value to predict for the sample. Considering the best results obtained however, the error is still significant in comparison to the true RUL value. This can be further observed in the plots for each sample shown in figure 6.10.

For the samples with the highest errors, bearings 1_1 and 1_6, the predictions fail to adjust to the real value at the start of the predictions, when a degradation start was detected by the first stage. Given the general trend at first, early detection is implied. For bearing 2_7 and 3_3, the error is also important considering their true RUL, but in these cases sudden failure may be the cause, given their late degradation detection and low true RUL value.

The rest of the samples have a better adjustment to the real RUL curve, following the decreasing trend throughout time until the bearing fails. Nevertheless, noisy spikes occurs in the majority of cases, which would difficult a real time assessment in an industrial deployment. Smoothing techniques may be required.

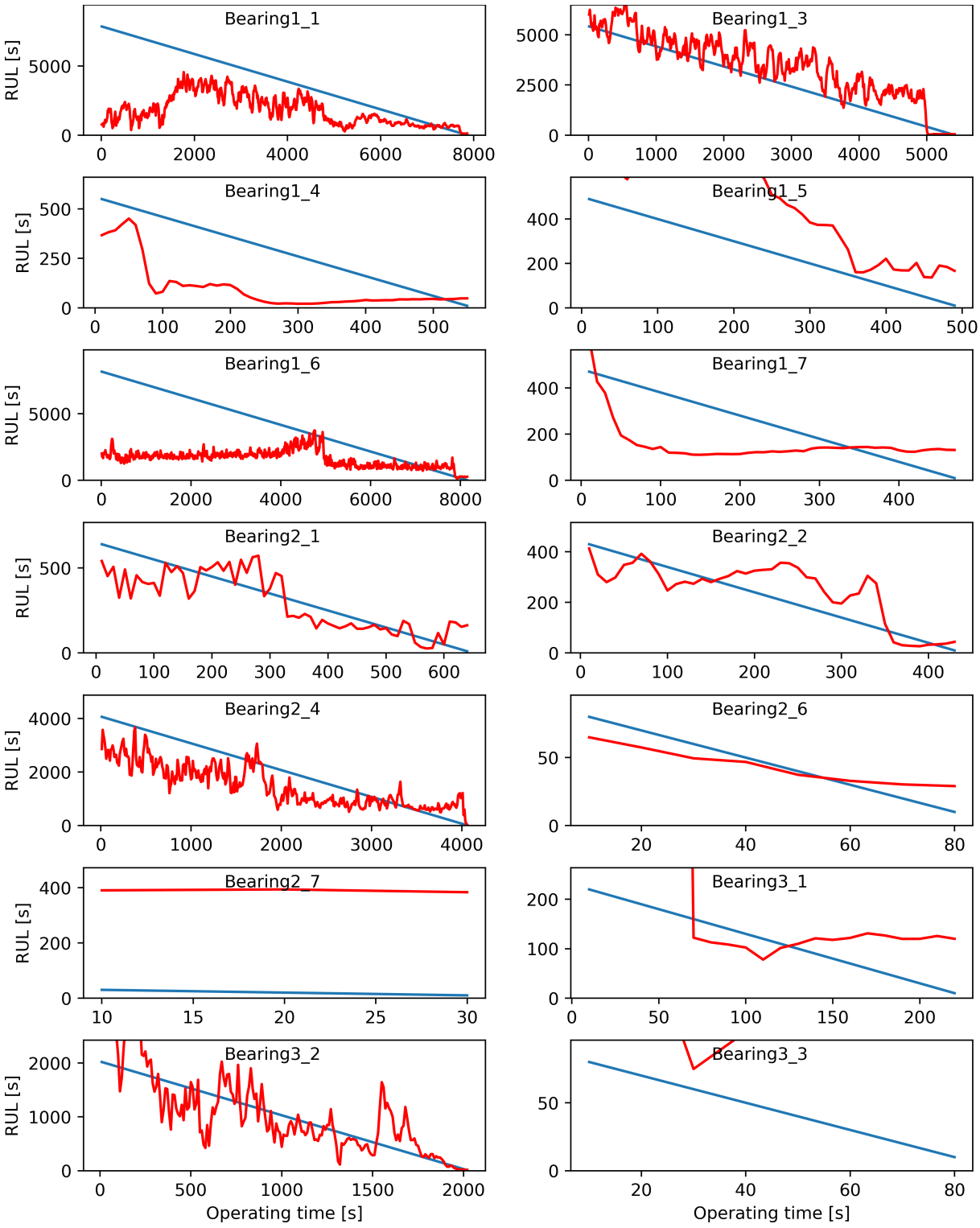


Figure 6.10: RUL predictions from degradation process start for each sample

6.5 Discussion and comparison

The proposed Framework is intended to perform RUL prediction only when a degradative process on the asset is detected, avoiding the high inaccuracies commonly occurring at the first stages when the asset was in a healthy state. However, for sample 1_1, this problem still occurred even though the MD curve implied an on going degradation process. Therefore, it is necessary to discuss if the obtained MD values may have been the problem or if it was due to the ANN used.

To analyze this, the same ConvLSTM used before is implemented for direct RUL predictions without the first degradation assessment stage. The network configuration is the same, as well as the training procedure followed. The best prediction for each sample are shown in the plots of figure 6.11.

It can be seen that in many of these cases the predictions during the early stages correlate well with the actual value, following the same linear trend overtime. This is contrary to the flat trend that was expected, implying that the spectrograms contained useful information to make these predictions, even though the MD measure produced a constant value. This could again imply that spectral data does have information that the selected vibrational features for the MD calculation does not.

Nevertheless, these RUL prediction without the initial degradation assessment have significantly worse accuracy during the end stage, ending with a drastic drop to 0 at the end stage. Therefore, the proposed Framework would still offer a better solution for RUL prognostics if accuracy in the end stage is taken into account.

6.6 Concluding Remarks

From the implementation of the proposed Framework on the FEMTO data challenge, the following remarks are made:

- MD curves based on vibration features can represent the health state of the bearing, but spectral features should also be considered as they proved to contain useful information by producing accurate RUL predictions at early stages of operation.
- Spectrograms, and sequences of spectrograms, are useful datatype for RUL predictions.

In regards to the FEMTO dataset, considering the small amount of examples and the unique pattern between each of them made of this dataset a difficult case study. Generalization of the degradation start criteria was difficult, and 3 samples were to be excluded for having a raw signal significantly different to the rest.

Even more, there are some contradictions between the dataset specifications and the data, for example, the experiment ending criteria. It was mentioned that these were ended when an acceleration of 20 [g] was reached, but in many cases this was surpassed. Moreover, some running's showed signs of spontaneous failure, something that would be impossible to accurately predict due to its random nature.

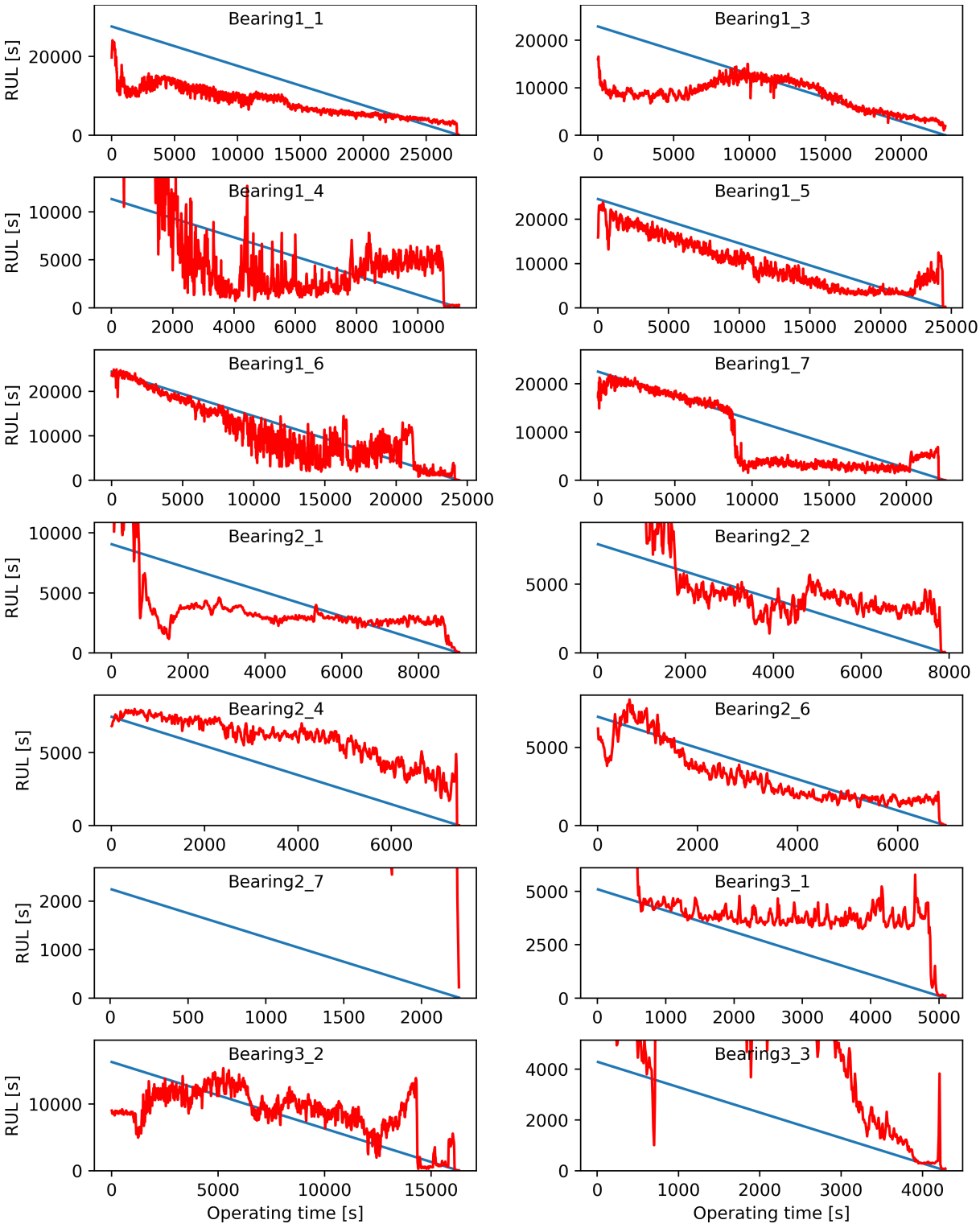


Figure 6.11: Direct RUL predictions from operation start for each sample

Chapter 7

Conclusion

In this Thesis was successfully developed a data-driven Framework for remaining useful life (RUL) prognostics of physical assets. This Framework was divided into 2 consecutive stages serving an specific purpose. The first one, to assess the health condition of the asset and detect a possible degradative process acting on, using the Mahalanobis distance (MD) as a metric for this. While the second stage produced RUL estimation through an Artificial Neural Network, only if the asset was detected to be in a degradative state by the first stage. By this way, this approach tackles a common problem found with RUL prognosis; the significant inaccuracies during the early and healthy stages of operation of an asset.

This Framework was tested on 2 different datasets: C-MAPSS and FEMTO, a turbofan and bearing degradation dataset respectively. From these implementations, the following points can be concluded:

- The Mahalanobis distance can effectively quantify a health index for an asset that is in correspondence with the operational evolution of an asset. This index has a nearly flat trend during the healthy stages, when no noticeable degradation has acted on the asset. After a degradation process takes relevance, the MD measure changes into an exponential trend, that increases until the asset reaches a failing point.
- The operational data used to conform the Mahalanobis space (MS) requires an initial exploration step to select the group of most representative features while maintain a degree of independency between each of them. In problems with vibrational data, it was found that spectral data may also be a useful feature to include.
- The developed algorithm for unsupervised MS search can define a healthy data space while only requiring normal and failure data, without any label known apriori. Using this MS as a baseline, a degradation assessment can be made over an operating asset by calculating the MD at each sensor measurement. Moreover, this MS serves to establish when the asset has progressed into a degradative state when the sensor measurements are outside the healthy space consistently overtime.
- Performing RUL estimations when the asset has reached a degradative state shows improved improves accuracy. Thus, the proposed Framework accomplishes the main objective set for its development. Moreover, it was found that filtering out healthy

operational data has no negative impact on the predictions accuracy, proving that healthy data does not give any more useful information for RUL estimation than data under degradation.

On the other hand, there are some problems and limitations found with the Framework that need to be discussed. This are the following:

- Early detection of the degradation starting point was an occurrence found in both C-MAPSS and FEMTO implementations. Samples with initially high MD values were detected to be under degradation even though there was no increasing trend found. The RUL predictions when this happens is highly inaccurate, which is contrary to the main objective proposed.
- The selection of the σ confidence interval and the degradation start criteria is mostly arbitrary, done by trial and error. This resulted in a time consuming task.
- The second stage for RUL estimation requires large amounts of data for training the network, and. moreover, data that is representative of all the possible operational conditions for the asset. This restricts the applicability of the Framework to cases where it is available data for a large amount of assets.

Considering the above points, it is concluded that the Framework may be a useful tool to implement in cases where large quantity of samples, but nevertheless, further improvements may be necessary.

7.1 Future work

From this Thesis work several ideas can be extracted that would be of interest for further research and development. Among these, are the following:

- Redefine the degradation start criteria through a probabilistic/bayesian approach, trying with this to avoid early detection due to high starting MD values and also detect increasing trends before crossing the degradation threshold.
- Study the relevance of spectral features as a component for the Mahalanobis metric in problems with vibrational data, and its capability to represent a degradation index.
- Explore the possibility of using the Mahalanobis distance for maintenance operation assessments on physical assets.

Bibliography

[1] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International conference on database systems for advanced applications*, pages 214–228. Springer, 2016.

[2] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*, pages 2546–2554, 2011.

[3] E Peter Carden and Paul Fanning. Vibration based condition monitoring: a review. *Structural health monitoring*, 3(4):355–377, 2004.

[4] Zhen Chen, Tangbin Xia, and Ershun Pan. Remaining useful life estimation based on a segmental hidden markov model with continuous observations. In *ASME 2017 12th International Manufacturing Science and Engineering Conference collocated with the JSME/ASME 2017 6th International Conference on Materials and Processing*, pages V003T04A060–V003T04A060. American Society of Mechanical Engineers, 2017.

[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[7] João Paulo Pordeus Gomes, Roberto Kawakami Harrop Galvão, Takashi Yoneyama, and Bruno P Leão. A new degradation indicator based on a statistical anomaly approach. *IEEE Transactions on Reliability*, 65(1):326–335, 2015.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[9] Rafael Gouriveau, Kamal Medjaher, and Noureddine Zerhouni. *From Prognostics and Health Systems Management to Predictive Maintenance 1*. Wiley, 1st edition, 2016.

[10] Liang Guo, Naipeng Li, Feng Jia, Yaguo Lei, and Jing Lin. A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing*,

240:98–109, 2017.

[11] Felix O Heimes. Recurrent neural networks for remaining useful life estimation. In *2008 international conference on prognostics and health management*, pages 1–6. IEEE, 2008.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] ISO 13381-1: Condition monitoring and diagnostics of machines – prognostics –. Standard, International Organization for Standardization, 2015.

[14] Olivier Janssens, Viktor Slavkovikj, Bram Vervisch, Kurt Stockman, Mia Loccufier, Steven Verstockt, Rik Van de Walle, and Sofie Van Hoecke. Convolutional neural network based fault detection for rotating machinery. *Journal of Sound and Vibration*, 377:331–345, 2016.

[15] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical systems and signal processing*, 20(7):1483–1510, 2006.

[16] Xiaohang Jin, Yu Wang, Tommy WS Chow, and Yi Sun. Md-based approaches for system health monitoring: A review. *IET Science, Measurement & Technology*, 11(4):371–379, 2017.

[17] Samir Khan and Takehisa Yairi. A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing*, 107:241–265, 2018.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Sachin Kumar, Nikhil M Vichare, Eli Dolev, and Michael Pecht. A health indicator method for degradation detection of electronic products. *Microelectronics Reliability*, 52(2):439–445, 2012.

[20] Jay Lee, Fangji Wu, Wenyu Zhao, Masoud Ghaffari, Linxia Liao, and David Siegel. Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. *Mechanical systems and signal processing*, 42(1-2):314–334, 2014.

[21] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018.

[22] Xiang Li, Wei Zhang, and Qian Ding. Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction. *Reliability Engineering & System Safety*, 182:208–218, 2019.

[23] Jianhui Luo, Madhavi Namburu, Krishna Pattipati, Liu Qiao, Masayuki Kawamoto, and SACS Chigusa. Model-based prognostic techniques [maintenance applications]. In *Proceedings AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference.*,

pages 330–340. IEEE, 2003.

[24] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[25] 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. *arXiv preprint arXiv:1608.06154*, 2016.

[26] Richard J McDuff, Patrick K Simpson, and David Gunning. An investigation of neural networks for f-16 fault diagnosis. i. system description. In *IEEE Automatic Testing Conference. The Systems Readiness Technology Conference. Automatic Testing in the Next Decade and the 21st Century. Conference Record.*, pages 351–357. IEEE, 1989.

[27] Ceena Modarres, Nicolas Astorga, Enrique Lopez Droguett, and Viviana Meruane. Convolutional neural networks for automated damage recognition and damage type identification. *Structural Control and Health Monitoring*, 25(10):e2230, 2018.

[28] Emmanuel Ramasso. Investigating computational geometry for failure prognostics. 2014.

[29] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008.

[30] 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 journal of operational research*, 213(1):1–14, 2011.

[31] Ahmet Soylemezoglu, Sarangapani Jagannathan, and Can Saygin. Mahalanobis-taguchi system as a multi-sensor based decision making prognostics tool for centrifugal pump failures. *IEEE Transactions on Reliability*, 60(4):864–878, 2011.

[32] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[33] Jing Tian, Carlos Morillo, Michael H Azarian, and Michael Pecht. Motor bearing fault detection using spectral kurtosis-based feature extraction coupled with k-nearest neighbor distance analysis. *IEEE Transactions on Industrial Electronics*, 63(3):1793–1803, 2015.

[34] Kwok L Tsui, Nan Chen, Qiang Zhou, Yizhen Hai, and Wenbin Wang. Prognostics and health management: A review on data driven approaches. *Mathematical Problems in Engineering*, 2015, 2015.

[35] Irem Tumer and Anupa Bajwa. A survey of aircraft engine health monitoring systems. In *35th joint propulsion conference and exhibit*, page 2528, 1999.

[36] David Verstraete, Andrés Ferrada, Enrique López Droguett, Viviana Meruane, and Mo-

hammad Modarres. Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings. *Shock and Vibration*, 2017, 2017.

[37] Tianyi Wang, Jianbo Yu, David Siegel, and Jay Lee. A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In *2008 international conference on prognostics and health management*, pages 1–6. IEEE, 2008.

[38] Yu Wang, Yizhen Peng, Yanyang Zi, Xiaohang Jin, and Kwok-Leung Tsui. A two-stage data-driven-based prognostic approach for bearing degradation problem. *IEEE Transactions on Industrial Informatics*, 12(3):924–932, 2016.

[39] Qidong Wei and Dan Xu. Remaining useful life estimation based on gamma process considered with measurement error. In *2014 10th International Conference on Reliability, Maintainability and Safety (ICRMS)*, pages 645–649. IEEE, 2014.

[40] Qiong Wu and Changsheng Zhang. Cascade fusion convolutional long-short time memory network for remaining useful life prediction of rolling bearing. *IEEE Access*, 8:32957–32965, 2020.

[41] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[42] Boyuan Yang, Ruonan Liu, and Enrico Zio. Remaining useful life prediction based on a double-convolutional neural network architecture. *IEEE Transactions on Industrial Electronics*, 66(12):9521–9530, 2019.

[43] Mei Yuan, Yuting Wu, and Li Lin. Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network. In *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, pages 135–140. IEEE, 2016.

[44] Zhengxin Zhang, Xiaosheng Si, Changhua Hu, and Yaguo Lei. Degradation data analysis and remaining useful life estimation: A review on wiener-process-based methods. *European Journal of Operational Research*, 271(3):775–796, 2018.

[45] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X Gao. Deep learning and its applications to machine health monitoring: A survey. *arXiv preprint arXiv:1612.07640*, 2016.

[46] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. In *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 88–95. IEEE, 2017.