Domicio Araujo Pereira Neto

INTELLIGENT PROGNOSIS OF THE HEALTH STATUS OF INDUSTRIAL SYSTEMS

1 2 9 0 UNIVERSIDADE D COIMBRA

UNIVERSIDADE D
COIMBRA

Domicio Araujo Pereira Neto

INTELLIGENT PROGNOSIS OF THE HEALTH STATUS OF INDUSTRIAL SYSTEMS

Dissertation in the context of the Master in Data Science and Engineering, advised by Prof. Alberto Cardoso and co-advised by Prof. Jorge Henriques and Prof. Paulo Gil, presented to the Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

Faculty of Sciences and Technology

Department of Informatics Engineering

Intelligent Prognosis of the Health Status of Industrial Systems

Domicio Araujo Pereira Neto

Dissertation in the context of the Master in Data Science and Engineering, advised by Prof. Alberto Cardoso and co-advised by Prof. Jorge Henriques and Prof. Paulo Gil, presented to the Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

UNIVERSIDADE D
COIMBRA

This page is intentionally left blank.

Abstract

The well-functioning and availability of machinery is crucial for the industry. Unexpected breakdowns and downtimes generate substantial financial losses to businesses, in addition to the waste of human and material resources. Considering the subject of Circular Manufacturing (CM), where a more sustainable, efficient and clean industry is promoted, Predictive Maintenance (PdM) emerges as a promising mean to uphold this evolution. Thanks to the Fourth Industrial Revolution (Industry 4.0), sensory data from industrial systems are increasingly available, and with Machine Learning (ML) it is possible to extract information from this data that can point to health problems in the machinery. In this work, a prognostics approach for predicting the health status of an emulated crane translation system through the diagnostics of a virtual rolling bearing is proposed. In the first semester, two preliminary approaches were developed: one Deep Learning (DL) based Remaining Useful Life (RUL) estimator for the PRONOSTIA bearing Dataset, and an unsupervised diagnostics approach for the crane translation system combining the Principal Component Analysis (PCA) with a clustering technique. Following what was learned at the literature's review, a Convolutional Neural Network (CNN) was tested for diagnostics with the Case Western Reverse University's (CWRU) rolling bearing vibration dataset and compared with the state-of-the-art for the same case. This network is then used to diagnose the health state of a rolling bearing within the emulated translation system. The synthetic bearing vibration data generated in the emulation is based on the PRONOSTIA dataset, from which a set of different degradation states was manually identified, having their Fourier coefficients extracted using the Discrete Fourier Transform (DFT). A signal generation module was also developed, making possible the creation of several bearing degradation scenarios. This module was integrated into the crane translation system emulator, where the generation of virtual bearing data is synchronized with real data from a crane. Finally, the previously mentioned CNN model was deployed to diagnose the virtual bearing's health status, which is then used for the translation system's health condition prognosis. The simulation results show that this PdM approach has the potential of preventing the unexpected breakdown of the system within the considered scenarios and might be further developed into a solution for real-world use cases, fulfilling one of the main objectives of the KYKLOS 4.0 project, in which this work is inserted.

Keywords

Circular Manufacturing, Industrial Systems, Predictive Maintenance, Machine Learning, Health Condition Prognosis

This page is intentionally left blank.

Resumo

O bom funcionamento e a disponibilidade das máquinas são cruciais para a indústria. Falhas e interrupções inesperadas geram perdas financeiras substanciais para as empresas, além do desperdício de recursos humanos e materiais. Considerando o tema da Manufatura Circular (MC), onde é promovido uma indústria mais sustentável, eficiente e limpa, a Manutenção Preditiva (MP) surge como um meio promissor para sustentar essa evolução. Graças à Quarta Revolução Industrial (Indústria 4.0), dados sensoriais de sistemas industriais estão cada vez mais disponíveis, e com *Machine Learning* (ML) é possível extrair informações desses dados que podem identificar problemas de saúde nas máquinas. Neste trabalho, é proposta uma abordagem prognóstica para prever o estado de saúde de um sistema emulado de translação de uma grua através do diagnóstico de um rolamento virtual. No primeiro semestre, duas abordagens preliminares foram desenvolvidas: um estimador de vida útil remanescente (RUL) baseado em *Deep Learning* para o dataset PRNOSTIA de rolamentos, e uma abordagem de diagnóstico não supervisionada para o sistema de translação de grua combinando a Análise de Componentes Principais (ACP) com uma técnica de *clustering*. Seguindo o que foi aprendido na revisão da literatura, foi testada uma Rede Neuronal Convolucional (RNC) para diagnóstico, com o conjunto de dados de vibração de rolamentos da Case Western Reverse University (CWRU), e comparada com o estado da arte para o mesmo caso. Essa rede foi, então, usada para diagnosticar o estado de integridade de um rolamento dentro do sistema de translação emulado. Os dados sintéticos de vibração dos rolamentos gerados na emulação são baseados no dataset PRONOSTIA, a partir do qual um conjunto de diferentes estados de degradação foram identificados manualmente, tendo os seus coeficientes de Fourier sido extraídos por meio da Transformada Discreta de Fourier (TDF). Foi desenvolvido, também, um módulo de geração de sinal, possibilitando a criação de diversos cenários de degradação de rolamentos. Este módulo foi integrado no emulador do sistema de translação, onde a geração de dados do rolamento virtual é sincronizada com dados reais de uma grua. Por fim, o modelo RNC mencionado anteriormente foi aplicado para diagnosticar o estado de saúde do rolamento virtual que é usado para o prognóstico da condição de saúde do sistema de translação. Os resultados da simulação mostram que esta abordagem de MP tem o potencial de prevenir a falha inesperada do sistema dentro dos cenários considerados e pode ser, adicionalmente, utilizada para desenvolver uma solução para casos de uso do mundo real, cumprindo um dos principais objetivos do projeto KYKLOS 4.0, em que este trabalho está inserido.

Palavras-Chave

Manufatura Circular, Sistemas Industriais, Manutenção Preditiva, Aprendizagem de Máquina, Prognóstico de Condição de Saúde

This page is intentionally left blank.

Acknowledgments

I am grateful to my advisor, Professor Alberto Cardoso, and co-advisors, Professor Jorge Henriques and Professor Paulo Gil, for all the patience and support during my thesis, and even before that, since the day I have joined their CISUC research group. I also would like to give thanks to my defense committee, who provided the feedback that was essential to steer this work in the right direction. Additionally, I would like to express my gratitude to the KYKLOS 4.0 project, that financed my studentship during these last two years.

Lastly, I would like to give special thanks and acknowledgement to my family, especially my parents, without whom I would not have gotten this far, to my wife, who supported me during all the difficult moments and never let me give up, my son, who gives me extra motivation to accomplish my job, and to all my family, friends, Church community, and above all, to God, who guided me thorough all these years. That my future as a professional might be worthy of the support that I have received from all of those around me.

This work has been carried out in the H2020 KYKLOS 4.0 project (Grant Agreement Number 872570), which is funded by the European Commission. This work was also partially financed by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the projects CISUC (UID/CEC/00326/2020) and CTS (UID/EEA/00066/2019).

This page is intentionally left blank.

Contents

This page is intentionally left blank.

Acronyms

ADCNN Adaptive Deep CNN. 18, 19, 51

AE Auto-encoder. 13, 14, 19, 20, 24

AI Artificial Intelligence. 1

ANN Artificial Neural Network. 9, 10, 14

CatGAN Categorical Generative Adversarial Network. 21, 22, 51

CE Circular Economy. 2

CM Circular Manufacturing. 1, 2, 60

CNN Convolutional Neural Network. 9, 10, 18, 19, 22–24, 43, 49–51, 56, 59, 60

CpS Cyber-Physical System. 1, 8

CV Cross-validation. 15–17, 19, 20, 27, 44, 49, 50, 56, 59

CWRU Case Western Reserve University. 17–19, 21, 22, 24, 44, 49–51, 56, 59, 60

DBN Deep Belief Network. 13, 14

DFT Discrete Fourier Transform. 39

DL Deep Learning. 2, 8, 9, 15, 19, 21, 23, 25, 26, 28, 42, 44, 47, 56, 57, 59, 60

DNN Deep Neural Network. 8

DT Digital Twin. 4, 22, 23, 38, 44

FFT Fast Fourier Transform. 20, 21, 51

GPR Gaussian Process Regressor. 27, 28

GRU Gated Recurrent Unit. 11, 12

IoT Internet of Things. 8

LSTM Long Short-Term Memory. 11, 12, 26, 28, 59

ML Machine Learning. 4, 8, 9, 15, 19, 21, 23, 26, 60

PCA Principal Component Analysis. 30–32

PdM Predictive Maintenance. 1–5, 7–10, 13, 14, 19, 21–23, 25, 35, 37, 42, 44, 46, 47, 49, 51, 53, 54, 56, 57, 59, 60

RBM Restricted Boltzmann Machine. 13, 14

RF Random Forest. 13, 23, 27, 28

RMSE Root Mean Square Error. 14, 27, 56

RNN Recurrent Neural Network. 10, 11, 26

RUL Remaining Useful Life. 8, 25, 35, 44–47, 51–54, 56, 57, 59

SAE Sparse Auto-encoder. 19, 20

SSAE Stacked Sparse Auto-encoder. 19–21

STFT Short-Time Fourier Transform. 21, 22, 24, 39–41, 43, 49–51, 56

SVM Support Vector Machine. 23, 27, 28

VAE Variational Auto-encoder. 20

This page is intentionally left blank.

List of Figures

This page is intentionally left blank.

List of Tables

This page is intentionally left blank.

Chapter 1

Introduction

The Fourth Industrial Revolution brought massive benefits to the industry thanks to the ever-growing technological advancements, increasing productivity and reducing costs. Industrial machines are at the center of this revolution, gradually transitioning from the old mechanical and analogical systems from the past to the new Cyber-physical Systems (CpS), which are composed of physical and software components that are highly intertwined, resulting in more efficient systems that reduce the need of human involvement in production. Keeping these machines in good functioning conditions is essential to any business. In this context, Predictive Maintenance (PdM), which consists of preventing failures by predicting when they will occur, is a subject of recurrent research that can provide the industry the means to avoid unexpected breakdowns and costs.

The present work was set to look into the problem of intelligent prognosis of the health status of industrial systems. As a result, an approach with the potential to be applied to a real-world use case is developed. This work is part of the KYKLOS 4.0[1] European project, that seeks to create a Circular Manufacturing (CM) ecosystem with novel technology, such as Artificial Intelligence (AI), to make intra-factory production more efficient, sustainable and clean.

1.1 Motivation

To give a solid notion about the relevance of PdM to the industry, a set of statistics from [Moyle, 2021] is presented below:

- The estimated annual cost of unplanned downtime for industrial manufactures is 50 billion dollars.
- Around 80% of maintenance time is spent on reacting to issues that arise instead of preventing them.
- As direct benefits of PdM, machine downtime can be reduced by 30%-50% and machine lifetime increased by 20%-40%.

From this information, it is possible to point that PdM is promising for modernizing industrial maintenance and saving costs. Nonetheless, PdM has also a major role in Circular

[1] `https://kyklos40project.eu/`

Manufacturing. CM is a set of Circular Economy (CE) strategies applied to manufacturing, which aims at achieving resources sustainability for industrial economy, promoting the economic growth without the negative impacts on the environment [Acerbi et al., 2021]. While many of the CM strategies are focused on the final product, there are several others centered on the manufacturing process, including PdM.

Although PdM is certainly important to the present and future industry, it has also a considerable amount of challenges that need to be overcome. The work of Wen et al. (2022) identifies some of the main challenges, as shown below:

1. Data insufficiency: PdM data-driven approaches rely on acquired data from industrial systems to be able to perform health condition diagnostics and prognostics, but data acquisition from real machines is complex, time-consuming and costly. Furthermore, the quality of the data is also important, it must be representative of the possible health conditions a machine can have, otherwise the solution will not be accurate enough.

2. Developed models' poor generalization capacity: Most developed models underperform when applied to other types of machines. This problem also concerns a single machine functioning under different operation regimes.

3. Late predictions: This is caused by a model's low predictive capacity. If a late prediction occurs, it might be impossible to avoid damage and losses.

4. Noise during real-time prognostics: Random disturbance from the environment may affect a system's components data acquisition process, requiring fast and efficient online prognostics algorithms.

5. Manual hyperparameter tuning and estimation: PdM algorithms, specially Deep Learning algorithms, require hyperparameters to be assigned and tuned, directly influencing the algorithm's performance. Most approaches rely on manual hyperparameter tuning.

6. Discrepancy of cross-domain prognosis: A common assumption during a PdM model's development is that the training and testing data are from the same distribution. Discrepancies exist in the real world, caused by noise from the environment and other factors, leading to deterioration in the prediction performance.

Most of the difficulties mentioned above were experienced during the main approach's development, especially the first one (data insufficiency), which led to the creation of a synthetic data generation module. Moreover, these challenges were taken in consideration during this work in order to maximize its quality and provide means to overcome them.

1.2 Problem Statement

The use case featuring in this work is from Spanish company ASTANDER, which performs ship repair and conversions, employing several cranes that move on rails (Figure 1.1).

The crane translation system is the object of the PdM research, it is composed of an electric motor that propels the crane on the rails back and forth. More than 200 variables were recorded from sensors installed on the crane. The recordings span 9 months, starting in October 2021 and finishing in June 2022. Although there was data available before

Figure 1.1: ASTANDER tower cranes.

October, it was deleted. Despite the fact that many sensor variables were available, there were only 8 variables related to the translation system, recorded with a sampling frequency of just $\frac{1}{60}Hz$ (one sample per minute), and no labelling indicating if the system was healthy or faulty. This situation made unfeasible the creation of a PdM algorithm using only the provided data. To mitigate this drawback, a virtual component of the translation system's electric motor was considered, namely, a rolling bearing.

Rolling bearings are mechanical components, present in practically every rotating equipment, from industrial machines to airplanes, making an essential part of a larger system. These components are also responsible for up to 44% of failures in some devices [Cerrada et al., 2018], and as such, it is a recurrent subject of PdM research. It is composed of four main parts (Figure 1.2): the outer race, the inner race, the ball, and the cage. All of this parts are subject to wear and faults, which are the consequence of various problems, such as operation conditions and manufacturing defects [SKF, 2017]. Therefore, the rolling bearing was chosen as the virtual component for the translation system, since the system certainly contains a number of these components.

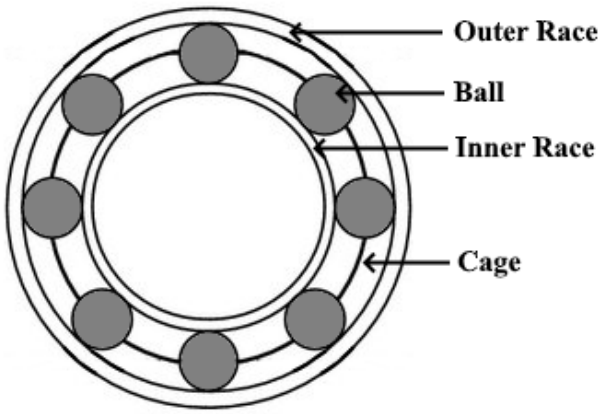

Figure 1.2: A representation of a rolling bearing and its parts.

It is worth mentioning that ASTANDER is one of the companies that are involved in the KYKLOS 4.0 project, in which this work is inserted.

1.3 Objectives

Following the problem statement, the main purpose of this work is to **contribute to the presented use case and to PdM research by developing a health condition prognostics approach that estimates the health state of a crane translation system**. Thus, the objectives are identified and described in the list below:

1. Perform a literature review on data-driven rolling bearing diagnostics approaches.
2. Develop a rolling bearing diagnostics model using Machine Learning and test it with a public benchmark.
3. Apply automatic architecture and hyperparameter optimization during the development of the aforementioned model.
4. Create a rolling bearing synthetic vibration data generator that can simulate various bearing degradation scenarios.
5. Create a crane translation system Digital Twin that is synchronized by real data from the crane and integrate it with the rolling bearing vibration data generator to create a PdM emulator.
6. Develop a prognostics approach for estimating the translation system's health condition based on the rolling bearing diagnostics and integrate it into the emulator.
7. Assess the diagnostics and prognostics performance using the PdM emulator.
8. Propose improvement strategies concerning the developed approach for the future.

1.4 Contributions

During the development of this work, various contributions were made. They are listed below:

1. Several modules were developed for the KYKLOS 4.0 PdM environment using python and MATLAB. These modules provide means to download, pre-process, and visualize data from the KYKLOS backend platform, in addition to train and test diagnostics and prognostics modules with benchmark datasets.
2. One published approach on rolling bearing prognostics, which was presented at the CONTROLO2022[2] conference (see Chapter 4).
3. Documentation of developed PdM modules and tools were provided for KYKLOS 4.0 reports.
4. The developed rolling bearing diagnostics model, with the distinctiveness of having architecture and hyperparameters that were optimized automatically.
5. The crane translation system Digital Twin and PdM emulator, implemented and presented in this work, that can be used to simulate multiple degradation and fault scenarios, in addition to having the potential to be improved and adapted to be used in the real world.

[2] `https://controlo2022.deec.fct.unl.pt/`

1.5 Structure

First, in chapter 2, the background concepts that feature in this work are introduced. Next, chapter 3 contemplates the state of the art based on the literature review. Then, in chapter 4, the initial work carried out during the first semester is presented, concerning one prognostics approach for the PRONOSTIA use case, and one experimental clustering approach for the ASTANTER use case. The description of the main proposed approach, the crane translation system PdM emulator, is explained in chapter 5. Furthermore, the results of the main approach are shown in chapter 6, where they are assessed and compared to the state of the art. Finally, the conclusion in chapter 7 shows the final reflections about what have been achieved in this work, the encountered difficulties, and suggestions for future work.

This page is intentionally left blank.

Chapter 2

Background Concepts

2.1 Predictive Maintenance

Before introducing the main aspects of PdM, it is necessary to take a brief look into the types of maintenance [Mobley, 2002]. A simple diagram is presented in Figure 2.1.

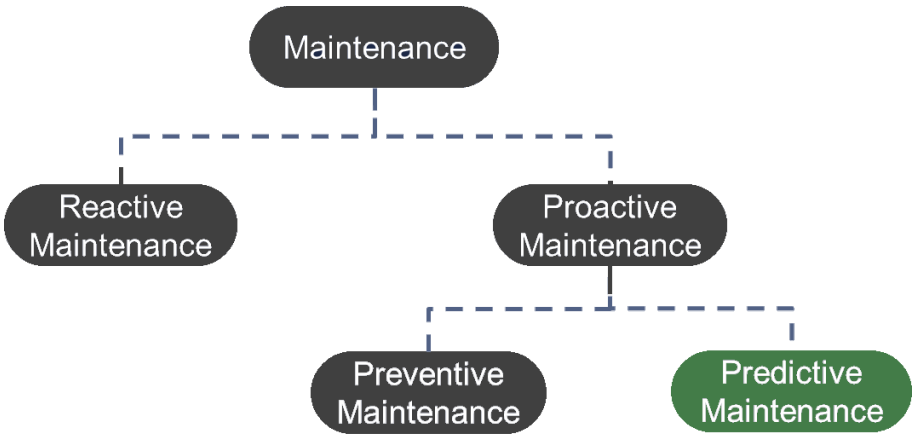

Figure 2.1: Principal maintenance types, highlighting PdM, that was used in this work.

Reactive Maintenance, also known as Run-to-failure, is the simplest type of maintenance. It means that an equipment only goes under maintenance if a breakdown occurs. The logic of run-to-failure can be summarized in one sentence: "Do not fix it if it's not broken". The attractiveness of this approach comes from the fact that no money is spent and no effort is made until a failure occurs. Although the method is straightforward and still frequently practiced, it is also the most expensive, demanding the existence of available spare parts and maintenance workers, and it also may cause longer disruptions.

The alternative to reactive maintenance is Proactive Maintenance, which consists of taking action before a failure occurs. The first type of proactive maintenance is Preventive Maintenance. In this line of action, a set of checks and minor fixes are carried out periodically. A common example comes from car regular servicing. Each car has to undergo servicing after a predefined number of kilometers to change oil, filters, and check for broken, malfunctioning, or worn parts that need replacement. The same principle is applied to industrial systems, where check-ups are performed with an interval that is estimated based on the average lifetime of the system's components. Even though preventive maintenance might offer cost savings between 12% to 18% when applied instead of reactive mainte-

nance [Moyle, 2021], it does not take into account the dynamic health condition evolution of industrial systems and components, caused by changing environments and operation regimes. This can lead to unnecessary replacements when the actual Remaining Useful Life (RUL) exceeds the predefined lifetime, or it may lead to the unexpected breakdown caused by the inaccuracy of the estimated lifetime or by the additional degradation that can naturally or abruptly occur.

Finally, Predictive Maintenance comes as a solution to the drawbacks of the first two approaches. PdM constitutes the set of strategies that aim to prevent failures by predicting them and scheduling maintenance based on this information. As already mentioned, PdM benefits the industry by minimizing failures, maximizing availability, reducing costs on spare parts and labor, increasing machine lifetime, among others. Figure 2.2 present the main types of PdM approaches [Zonta et al., 2020].

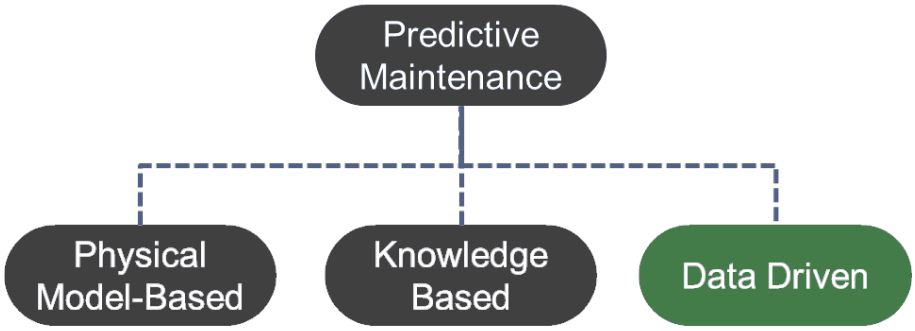

Figure 2.2: Principal PdM approach types, highlighting Data Driven, that was used in this work.

Physical Model-Based PdM relies on the modelling of components' behavior and failures through mathematical and statistical methods. It requires profound understanding of the system and its possible faults, usually through specifications given by the manufacturer or by experts knowledge. For example, the author of Tinga (2013) proposed an approach for modelling failures of military vehicles by analyzing the correlation of the operation regimes to the observed failures, defining parameters for the operation regime factors that were found to be linked to the faults. The obstacle to this kind of approach is how difficult the modelling can be if there is no previous detailed information about the physical system and associated faults, moreover, acquiring the necessary data can be time-consuming and costly.

Knowledge Based techniques are composed of rule-based systems, like knowledge graphs and fuzzy systems, built from knowledge bases, which store data acquired from the domain in form of statements that are compared with new observations. As an example, the work of Cao et al. (2022) presents a system that can be used for creating knowledge-based models for industrial systems by automatically generating rules using a degradation model extracted from the provided data.

Data Driven PdM methods are mainly based on Machine Learning (ML) and Deep Learning (DL), which extract information from data acquired from machines, e.g., collected from systems through builtin or retrofitted sensors, creating models that can diagnose or predict a machine's health condition, working like a black box function. This type of PdM approach is steadily growing more popular thanks to the increasing availability of data, driven by advancements in Internet of Things (IoT) and CpSs, which provides an integrated connectivity in manufacturing plants. Moreover, powerful ML and DL models, such as Deep Neural Networks (DNNs), are able to learn hidden information that bolster

diagnostics and prognostics capabilities.

The success of data driven approaches depends on the quantity and quality of the acquired data, making the data acquisition process the most challenging. The data must represent as many operational and health conditions as possible. When acquiring real data is unfeasible or insufficient, generating data from simulations becomes an alternative, although applying these approaches in the real world may result in poor performance [Wen et al., 2022]. Furthermore, unsupervised learning is a viable solution when no information (labelling) about the system's health condition is available [Amruthnath and Gupta, 2018]. For example, clustering techniques can identify different health condition regimes without the need of labels, however, additional information about these regimes is necessary to determine what they represent, e.g., healthy or faulty states.

2.2 Deep Learning

In ML, the techniques which allow machines to receive raw data and automatically uncover the representations that are necessary for classification or detection constitute Representation Learning. DL methods are part of Representation Learning, having multi-level representation that results from composing non-linear modules. At each level, starting with the raw input at the first level, the representation is transformed into a more abstract level. The composition of an adequate number of transformations allow very complex functions to be learned [Lecun et al., 2015].

The earliest perspective on DL were on biologically-inspired systems, namely, the brain [Goodfellow et al., 2016]. Nevertheless, DL now exceeds this perspective, being more generally associated with the composition of the previously-mentioned multi-level learning, no longer being necessarily inspired by neuroscience.

In this section, the principles of the most common DL methods applied to PdM are presented, specially those applied to rolling bearing diagnostics and prognostics, as they will appear in Chapter 3.

2.2.1 Convolutional Neural Networks

The Convolutional Neural Network (CNN) is a class of Artificial Neural Network (ANN). The inspiration for CNNs comes from the visual cortex of animals. Originally applied to object recognition, CNNs are now used in other domains, like object tracking, text processing, action recognition, among others [Aloysius and Geetha, 2018]. A typical CNN architecture is shown in Figure 2.3.

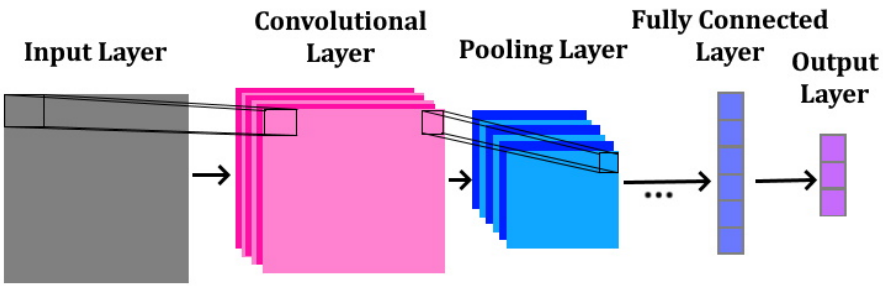

Figure 2.3: Representation of a CNN architecture.

Bellow, a brief explanation of each layer is presented [O'Shea and Nash, 2015]:

- The **convolutional layer** is the basic unit of the CNN. The output of this layer is the result of the scalar product between the filters' weights learned during training and a region of the input. The filters learn to identify specific features in the input. Each filter is convolved across the entire input volume, resulting in a two-dimensional activation map. The combination of all activation maps is called the feature map.
- The **pooling layer** performs downsampling of the given feature map. The most common pooling operations are the max pooling and average pooling, where filters of usually 2 x 2 are applied through the spatial dimensions of the input, outputting the maximum or average value of the given input region. The pooling layer reduces the amount of data to be processed, the number of parameters, and overfitting.
- The **fully connected layer** is analogous to the layers of ANNs, were a layer has neurons connected to every neuron in the adjacent layers. It maps the extracted features of the previous layers to the target output.

CNNs architectures for PdM are usually used for one or two-dimensional data [Lee et al., 2016]. For example, a 2D CNN can be used for processing time-frequency features resulting from a spectrogram, or vertically-stacked one-dimensional signals, and a 1D CNN might be used for processing time series, with only one dimension, which is time.

2.2.2 Recurrent Neural Networks

The Recurrent Neural Network (RNN) is a type of ANN that can learn sequential information by feeding the network layers with their past outputs [Medsker and Jain, 2001]. As data-driven PdM data is mainly composed of time series, RNNs and their variations are widely used for learning degradation patterns and making health condition prognosis. Figure 2.4 shows an unfolded RNN, providing an insight about its functioning, while Figure 2.5 presents a RNN unit.

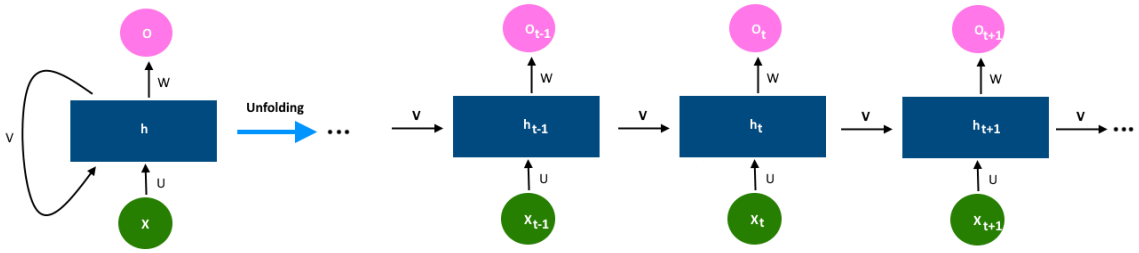

Figure 2.4: Representation of a RNN unfolded.

The following equations demonstrate how the RNN unit works.

$$h_t = \sigma_h(U_h x_t + V_h h_t - 1 + b_h) \tag{2.1}$$
$$o_t = \sigma_c(W_c h_t + b_c) \tag{2.2}$$

Where U, V and W are weight matrices, and b are bias vectors.

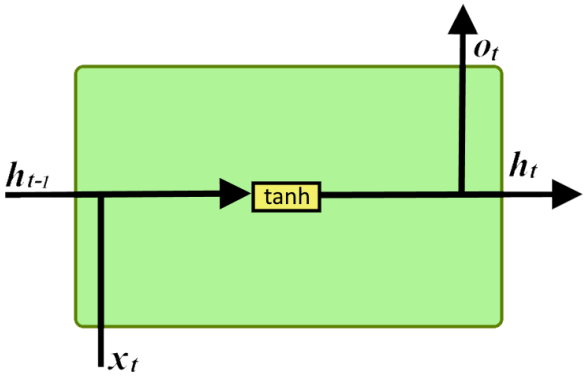

Figure 2.5: Representation of a RNN unit.

Although RNNs are capable of learning sequential information, they suffer from vanishing and exploding gradients, as a result of the back-propagation through time algorithm [Serradilla et al., 2020]. This problem causes the RNN to forget long-term dependencies. Nonetheless, RNN variations were created to solve this issue, namely, the Long Short-Term Memory (LSTM) and the Gated Recurrent Unit (GRU).

The LSTM was specifically designed to overcome the exploding and vanishing gradient problem of the RNN [Van Houdt et al., 2020]. The LSTM unit (Figure 2.6) comprises a cell and three gates: input, output, and forget. The forget gate allows the LSTM to reset its state. The cell is then able to remember values over various time intervals, while the other gates control the cell's information flow.

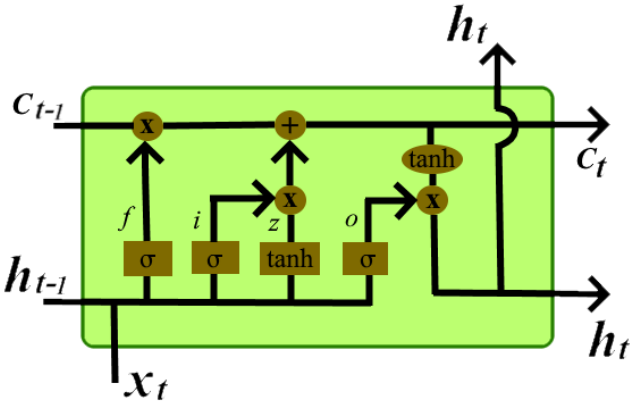

Figure 2.6: Representation of a LSTM unit.

The following equations demonstrate how the LSTM unit works.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t]) + b_f \tag{2.3}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t]) + b_i \tag{2.4}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t]) + b_o \tag{2.5}$$
$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.6}$$
$$C_t = f_T \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.7}$$
$$h_t = o_t \odot tanh(C_t) \tag{2.8}$$

Following now a brief summary of the LSTM unit. First, the input gate i combines the current step input x_t, the output h_{t-1} of the LSTM unit at the previous time step, and the cell value c_{t-1}, also from the last time step. Next, the forget gate controls which information will be removed from the previous cell states c_{t-1}, therefore, the value of f is calculated using input x_t, previous output h_{t-1}, the cell state c_{t-1} and the bias associated with the forget gate. Then, the cell value c_t is calculated by combining the block input z, input gate i and forget gate f with the previous cell value c_{t-1}. Finally, the output h_t is calculated, combining current cell value c_t with the output gate o.

The presented LSTM unit is considered to be the standard (or vanilla) variant. Many other variants have been developed since its creation. The work of Greff et al. (2017) presents an extensive experimentation of LSTM variations, mainly by combining or removing gates, and through hyperparameter optimization. This study concluded that the forget gate and the activation function of the output gate are the most important components of the LSTM, while some modifications to the other components, for example, by combination or removal, do not bring substantial improvements, but also no decrease in performance, possibly explaining why the GRU performs well without a memory cell.

The GRU, first proposed by Cho et al. (2014), was designed to make each unit capable of learning dependencies of different time scales adaptively. The GRU, like the LSTM unit, has gates that control the information flow, but without the memory cells (Figure 2.7). One of the main differences between the LSTM and the GRU is that the LSTM unit regulates, through the output gate, the memory content that is seen or used by other units. The GRU does not control the degree of exposure to the previous states. Nevertheless, it is not possible to point what type of unit is the best one in general, making experimentation for each use case necessary [Chung et al., 2014].

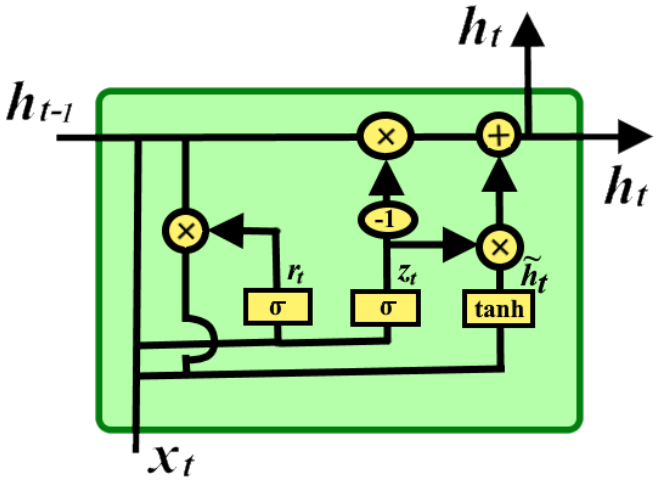

Figure 2.7: Representation of a GRU unit.

The GRU unit equations are presented below.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) + b_z \tag{2.9}$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) + b_r \tag{2.10}$$
$$\tilde{h}_t = (W_h) \cdot [r_t \odot ht - 1, x_t] + b_h \tag{2.11}$$
$$h_t = (1 - z_t) \odot ht - 1 + z_t \odot \tilde{h}_t \tag{2.12}$$
$$\tag{2.13}$$

2.2.3 Auto-encoders

An Auto-encoder (AE) is a multi-layered neural network that is trained for dimensionality reduction and feature extraction, achieving state-of-the-art performance in many areas [Wang et al., 2016]. Figure 2.8 shows the common structure of an auto-encoder.

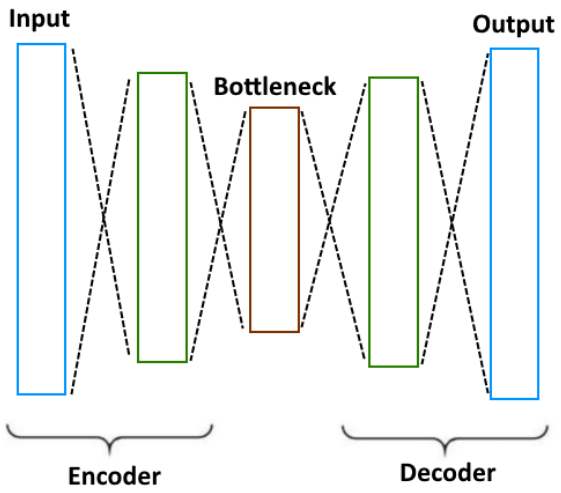

Figure 2.8: Representation of an auto-encoder.

The auto-encoder architecture is composed of an encoder followed by a decoder, nevertheless, the AE is trained as a whole, where the encoder reduces the dimensionality of the data through its layers, until it reaches the bottleneck, from where the latent representation (or vector) of the original data can be extracted. The decoder reconstructs the original data from the latent vector and the error between the original data and the reconstructed data is used to train the AE.

AEs might be used for PdM in various ways. For example, in the work of Sakurada and Yairi (2014) the authors propose an anomaly detection approach using an AE, which is trained using only data from healthy systems. The anomaly detection is based on the AE's reconstruction error, which naturally will be higher for faulty data, as it has not been seen during training. Another example comes from the work of Yu et al. (2021), where an AE is used for extracting features from the NASA's turbofan dataset, which are then used for training a Random Forest (RF) model for prognostics.

2.2.4 Deep Belief Networks

Before explaining what a Deep Belief Network (DBN) is, it is necessary to introduce the Restricted Boltzmann Machine (RBM). A RBM is a neural network that has only two layers: the visible layer and the hidden layer (Figure 2.9).

RBM training is unsupervised, which is composed of two steps. First, the forward pass, where the input data in the visible layers is passed to the hidden layer, where the latent vector is calculated by:

$$h = \sigma(v \times W^T + a) \tag{2.14}$$

where W and a are the weight matrix and bias vector, respectively, and σ is the activation

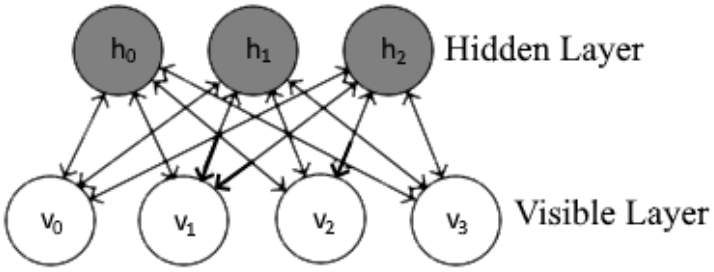

Figure 2.9: Representation of an RBM.

function. This calculation generates a latent representation of the visible layer's values, which are then used as input in the backward pass as follow:

$$v = \sigma(h \times W^T + b) \tag{2.15}$$

where b is the bias vector associated to the visible layer. This backward pass attempts to reconstruct the original values of the visible layer from the latent vector. As such, RBMs are also used for dimensionality reduction, similarly to the AEs.

DBNs are made by stacking RBMs [Hinton, 2009], where the hidden layer of one RBM is the visible layer of the next. Figure 2.10 shows a representation of a DBN architecture.

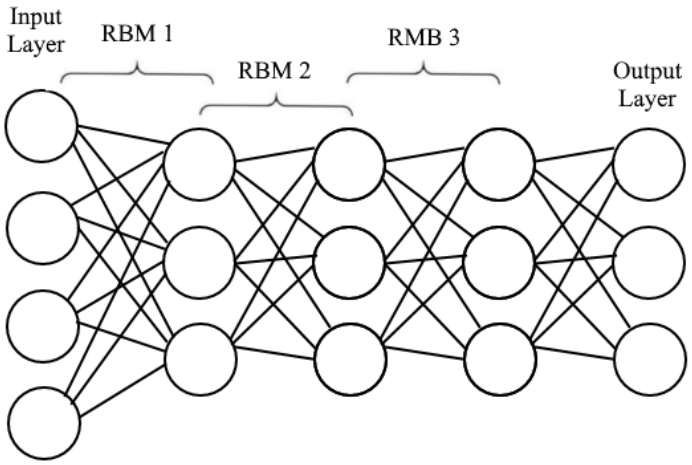

Figure 2.10: Representation of a DBN.

The DBN's structure is very similar to a multi-layer ANN, but as RBM's training is unsupervised, it is necessary to combine the DBN with a network that uses supervised learning, for example, by adding a softmax layer [Hua et al., 2015]. This way, the stacked RBMs extract features that are mapped by the softmax layer to the target output.

One example of DBN use in PdM comes from the work of Chen et al. (2018), where a DBN is used for health condition prognostics of a cutting tool. The approach is compared with an ANN, outperforming it in terms of Root Mean Square Error (RMSE), which is the most common metric for performance assessment in prognostics.

2.3 Hyperparameter Optimization

One of the main challenges for the implementation of DL models is finding good hyperparameter values. Some examples of hyperparameters in DL that require tuning include: learning rate (α), momentum (β), number of hidden layers, number of neurons in each hidden layer, mini-batch size, among others. This task is usually performed manually, but this can lead to far-from-optimum performance, as DL models are highly affected by the choice of hyperparameters.

The alternative to manual assignment is automatic search. A summary of the most used methods are presented below [Bergstra et al., 2011].

- **Grid search** is an exhaustive method. First, a set of values for each hyperparameter is defined manually. The algorithm will train the model for every combination between the previously defined sets. This method suffers from dimensionality problems, as larger parameter sets increase the number of total combinations to be tested. Another challenge is the manual definition of the parameter sets, requiring some knowledge about the parameters so reasonable values can be chosen.
- **Random search**, as opposed to grid search, is not exhaustive. The algorithm tests random combinations of hyperparameters, both discrete and continuous. Random search is more effective when just a small number of parameters are found to be relevant to the model's performance.
- **Bayesian optimization** works by transforming the search of hyperparameters in a black-box function optimization problem. The algorithm constructs a probabilistic model from the hyperparameters values and updates it by testing a new combination that will bring the most information about the model. Bayesian optimization has been shown to achieve better results than grid search and random search for several ML methods, including neural networks, and with less samples [Wu et al., 2019].

2.4 Validation Techniques

Several techniques for assessing the performance of ML models exist. The most common methods are listed below.

- Resubstitution: is the most simple method, it uses all dataset for training and for testing. It is extremely prone to overfitting and the results are biased, since supposedly good performance metrics may actually conceal the fact the model just memorized the output for each input.
- Train-test split: the dataset is split in two subsets, one for training, and the other for testing. It results in less overfitting and bias than the previous method, but it is also flawed, as the split is chosen randomly, and for a single train-test split the results can be biased.
- K-fold Cross-validation (CV): it is based on several train-test splits, where the dataset is divided in K folds, and for K iterations the model is trained with $K-1$ folds and tested with the remaining one. At the end, the performance is assessed as the average between all K iterations. The K-fold CV is considered more reliable than the single train-test split and the resubstitution method. Figure 2.11 bellow shows a graphical representation of a 5-fold CV split.

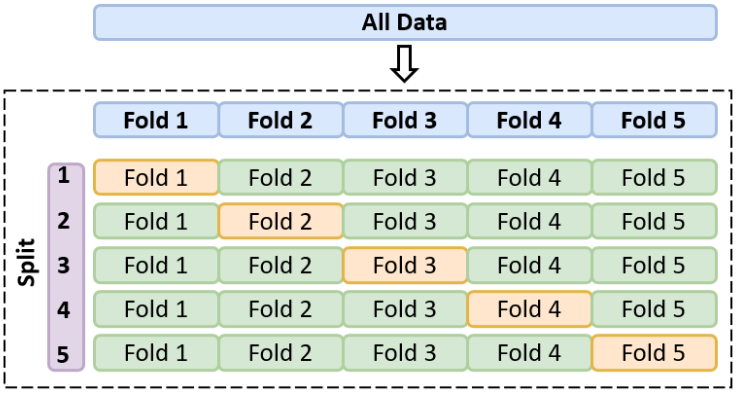

Figure 2.11: 5-fold Cross-validation graphical representation.

Chapter 3

State of the Art

Rolling bearing health condition monitoring is in the center of this work, as it is the chosen component for virtualization in the crane translation system emulator. The objective is to diagnose the bearing, which enables the estimation of the overall system health condition. Thus, a literature review was performed, focusing on rolling bearing diagnostics.

The works cited in this chapter were developed for the Case Western Reserve University (CWRU) dataset [CWRU, n.d.], as it is the most popular benchmark for rolling bearing diagnostics, allowing a better comparison and performance assessment of this work's approach. Nonetheless, the techniques and principles featured in this chapter are frequently used with other rolling bearing datasets, both public and private [Cerrada et al., 2018]. Another selection criteria is the validation method. Approaches validated using Cross-validation were given preference over other works that used only single train-test split or resubstitution, which can lead to biased results [Rauber et al., 2021].

3.1 Case Western Reserve University

In this section, a brief description of the CWRU dataset is given. The test bed, seen at Figure 3.1, is a 2 hp reliance electric motor. Acceleration data was acquired by sensors installed at three different distances from the bearings. Faults were seeded with electro-discharges at the inner and outer raceways and at the ball, with magnitudes ranging from 0.007 to 0.040 inches in diameter.

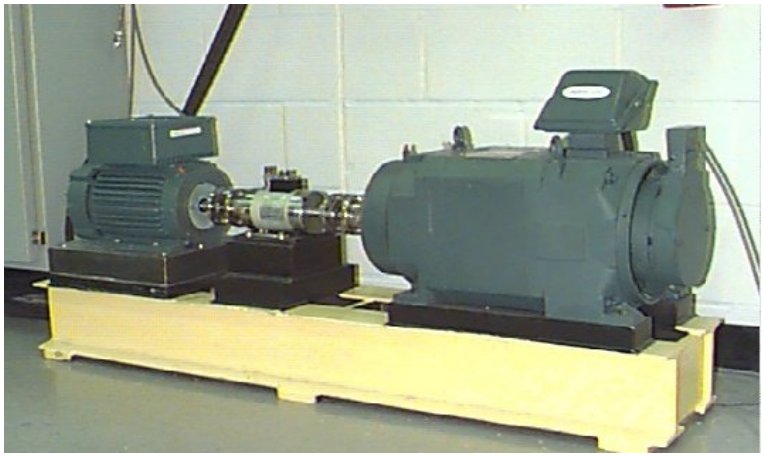

Figure 3.1: CWRU test bed [CWRU, n.d.].

The data was sampled at 12 kHz and 48 kHz, but only the dataset sampled at 12 kHz was used, as it is more common to find approaches that use it. In terms of operation regimes, the motor load varies between 0 and 3 hp, and the RPM between 1730 and 1797. Table 3.1 presents more information about the dataset. The nomenclature for each set is defined by the first letters of the location of the fault in the bearing, the fault diameter, and in the case of the outer race faults, the position of the fault in the outer race.

Table 3.1: CWRU dataset description.

Fault Diameter	Load (HP)	Speed (RPM)	Inner Race	Ball	Outer Race and Position Relative at Load Zone		
					Centered	Orthogonal	Opposite
0.007"	0	1797	IR007_0	B007_0	OR007@6_0	OR007@3_0	OR007@12_0
	1	1772	IR007_1	B007_1	OR007@6_1	OR007@3_1	OR007@12_1
	2	1750	IR007_2	B007_2	OR007@6_2	OR007@3_2	OR007@12_2
	3	1730	IR007_3	B007_3	OR007@6_3	OR007@3_3	OR007@12_3
0.014"	0	1797	IR014_0	B014_0	OR014@6_0	*	*
	1	1772	IR014_1	B014_1	OR014@6_1	*	*
	2	1750	IR014_2	B014_2	OR014@6_2	*	*
	3	1730	IR014_3	B014_3	OR014@6_3	*	*
0.021"	0	1797	IR021_0	B021_0	OR021@6_0	OR021@3_0	OR021@12_0
	1	1772	IR021_1	B021_1	OR021@6_1	OR021@3_1	OR021@12_1
	2	1750	IR021_2	B021_2	OR021@6_2	OR021@3_2	OR021@12_2
	3	1730	IR021_3	B021_3	OR021@6_3	OR021@3_3	OR021@12_3
0.028"	0	1797	IR028_0	B028_0	*	*	*
	1	1772	IR028_1	B028_1	*	*	*
	2	1750	IR028_2	B028_2	*	*	*
	3	1730	IR028_3	B028_3	*	*	*

For training and testing the diagnostics models, usually 4 classes are considered, the first representing normal condition, and the rest according to the bearing faulty component: inner race, ball, and outer race.

3.2 Convolutional Neural Networks

In this section, CNN based approaches are presented. In [Guo et al., 2016], the author proposes a hierarchical model named Adaptive Deep CNN (ADCNN). The hierarchical aspect comes from the combination of two sets of 2D CNNs, one for diagnosing the fault type and the other for evaluating the magnitude of the fault. Training is performed using adaptive learning rate and momentum to improve the learning process. Figure 3.2 presents the architecture of the ADCNN.

Figure 3.2: ADCNN's architecture.

The architecture is simple and consists of three convolutional-max-pooling layers, followed by two fully connected layers. As the CNN is two-dimensional, the raw input data is rearranged to 32x32 matrices. The ADCNN achieved, for diagnostics, a mean accuracy of 97.9%, using 10-fold CV.

Another CNN based model with interesting features is presented in the work of Eren et al. (2019). A 1D CNN is proposed, also using only raw data as input. Although the validation accuracy reached only 93.2%, it showed that the 1D CNN can also be effectively applied to rolling bearing diagnostics, having the advantages of being less complex than other DL approaches, with few hyperparameters, and it is also faster, making it reliable for real-time PdM. The architecture is simpler than the one presented before, as it can be seen in Figure 3.3.

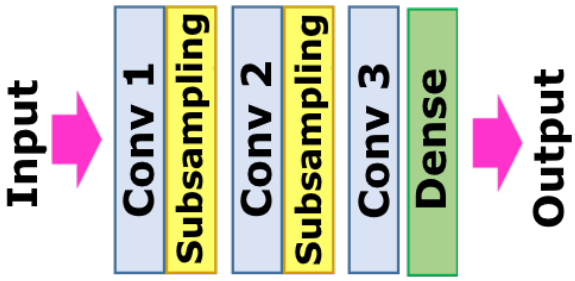

Figure 3.3: 1D CNN architecture.

No hyperparameter tuning was performed in that work. Applying hyperparameter optimization, including the number of layers, might substantially improve the model's performance.

Although most of the approaches for the CWRU dataset do not use CV, some works are worth mentioning, specially the following, which achieved 100% accuracy.

A 1D CNN is proposed by Yuan et al. (2018), having only 2 convolutional-max-pooling layers followed by a single fully-connected layer. The author of Hoang and Kang (2019) presented a 2D CNN that processes images generated by stacking the raw signal. The architecture contains 2 convolutional-subsampling-layers and one fully-connected layer. Moreover, in [Zhang et al., 2020], a slightly modified 2D CNN was proposed to process raw data transformed into images, where two fully-connected layers, each one combined with dropout layers, follow 3 convolutional-max-pooling layers. The dropout layers mitigate overfitting by eliminating a percentage of features randomly. None of these approaches applied automatic hyperparameter optimization.

3.3 Auto-encoders

As it was discussed in chapter 2, AEs are able to extract deep abstract features from raw data that bring valuable information for training ML models. Recently in the work of Wang et al. (2022), a Stacked Sparse Auto-encoder (SSAE) combined with softmax classification is proposed. A Sparse Auto-encoder (SAE) is a variant of the classic AE that has sparse penalty in training. This penalty deactivates a number of neurons from the hidden layers to improve the learning process, similar to what happens when a dropout layer is used in other neural networks.

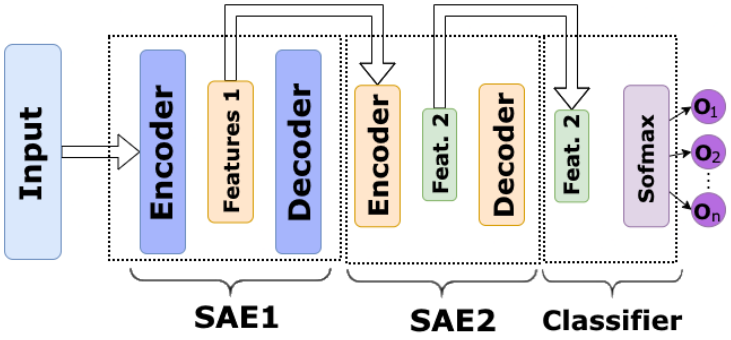

Figure 3.4: SSAE architecture.

In terms of architecture (Figure 3.4), two SAEs are stacked, where the features extracted from the first SAE are used to train the second one, and the features from the latter are used to train the classifier. Before training, the raw data was segmented and the Fast Fourier Transform (FFT) of each segment was calculated and used as input. Cross-validation results were very satisfactory, with an average accuracy of 99.15%.

3.4 Hybrid Approaches

Combining different types of models are often a way to take the most of their capabilities. From the hybrids approaches, properly validated with CV, the work of Dewangan and Maurya (2022) stands out. The model is based on the combination of a Variational Autoencoder (VAE) with convolutional layers. VAE works considerably different from classic AEs, being part of the family of variational Bayesian methods, the VAE learns to model the probability distribution of the data, including the variance parameter, which can lead to better performance in reconstruction in comparison to the deterministic AE [An and Cho, 2015].

What the author proposes is a modified VAE, where both the encoder and decoder have convolutional layers. The architecture can be seen in Figure 3.5.

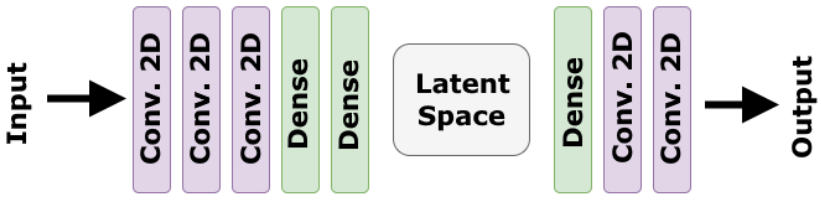

Figure 3.5: Deep Convolutional VAE's architecture.

The classifier trained with the extracted features have 4 hidden layers and one softmax layer. With 5-fold CV the model achieved a mean accuracy of 99.93%, being the best observed approach so far (not considering those validated with single train-test split and resubstitution).

3.5 Frequency and Time-frequency Analysis

One of the attractive capabilities of DL model is that no previous feature extraction is needed for training, as models may achieve high accuracy with raw data. Even so, PdM researchers have experimented a variety of techniques to extract features that can improve the models performance.

The example of the SSAE cited in section 3.3 used the FFT, which provides a frequency analysis of a finite signal. An example of a single-sided amplitude spectrum calculated using FFT and one segment of the CWRU dataset is presented in Figure 3.6. The Y-axis is the amplitude and the X-axis is the frequency range from 0 to half the sampling frequency.

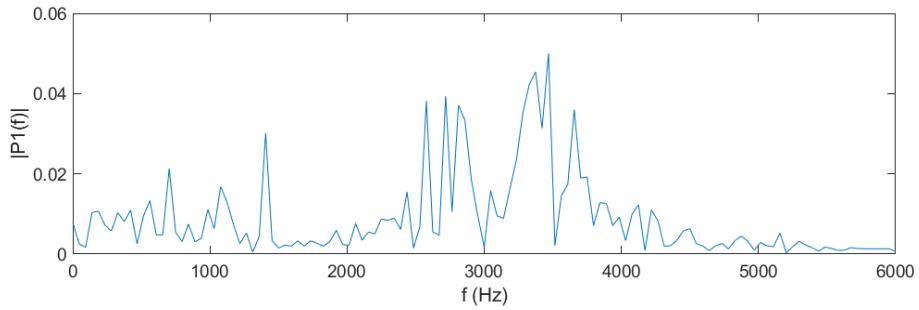

Figure 3.6: Frequency analysis of a CWRU dataset segment using FFT.

Rolling bearing faults have discriminative frequency signatures that can be exploited by PdM ML models [Li et al., 2019]. Another popular technique is the Short-Time Fourier Transform (STFT), a time-frequency analysis method. The result of the STFT is three dimensional and can be represented by an image, where it is possible to see the evolution of frequency components' magnitude (dB) through time. A good example is seen in Figure 3.7, where a STFT is performed with the three different fault types of the CWRU dataset. By analyzing the results, the differences become visible. The warmer the color, higher is the magnitude, and analogously, the cooler the color, the lower is the magnitude.

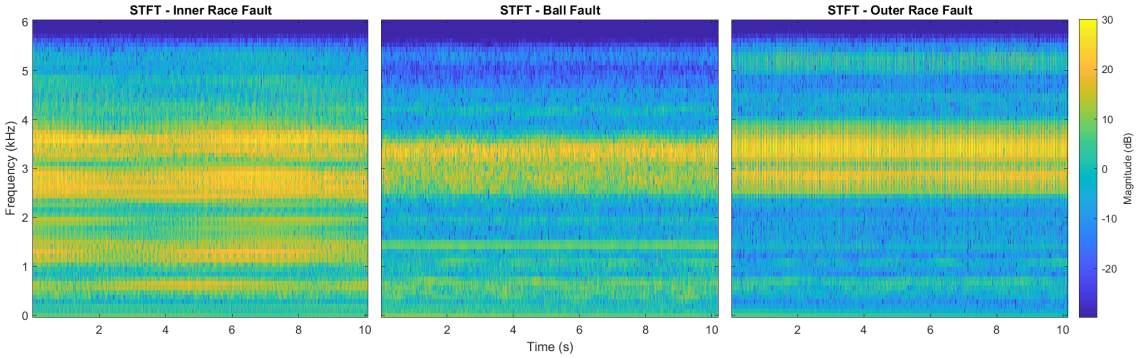

Figure 3.7: STFT of the three fault types of CWRU bearing dataset.

In the work of Tao et al. (2020), the authors combine STFT with a Categorical Generative Adversarial Network (CatGAN), naming it the ST-CatGAN. The GAN is a combination of two neural networks, one is the generator and the other is the discriminator. The generator creates synthetic samples from noise and the discriminator tries to determine if a sample is real, i.e., from the original dataset, or fake (generated). A well trained generator can be used to expand datasets with synthetic data, which can be a valuable resource when current data is not enough and acquiring more data is difficult or unfeasible. The CatGAN

is a variant that uses a classifier instead of the discriminator, that is trained with both real and generated data, improving its performance. Moreover, the authors pre-process the CWRU dataset with the STFT before training the model, which means the generator creates synthetic STFT samples.

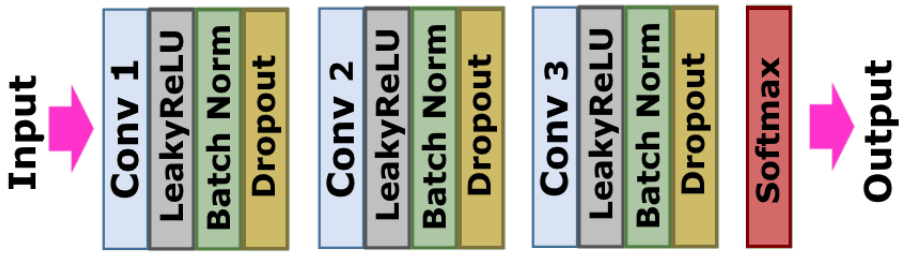

Figure 3.8: ST-CatGAN classifier architecture.

STFT data is processed by the 2D CNN classifier (Figure 3.8). One of the distinctive features of this architecture is the use of batch normalization layers, that scale the output of the previous layer to have mean 0 and variance 1, enabling faster training. This approach achieved an accuracy of 99.34% for diagnostics with no motor load, but failed achieved similar results for varying loads (1 hp to 3 hp), reaching only 91.89%. Nevertheless, this work shows the potential of STFT to extract valuable features from vibration data.

3.6 System Emulation with Synthetic Data Generation

A common obstacle to develop PdM solutions is the difficulty of acquiring data from physical systems. As mentioned before, the quality of a dataset depends on various factors, including the representativeness of different health conditions. The occurrence of faults and failures can be rare, making the acquisition of a diverse dataset unfeasible. Another solution is to induce a fault artificially, but it may also be impractical for various reasons, as lack or expensiveness of spare parts, or if the machine's unavailability causes financial losses to the business.

System emulation and synthetic data generation is a popular alternative when the given data is insufficient. Digital Twins (DT) are one of the main techniques of system emulation [Falekas and Karlis, 2021]. A DT is a virtual representation of a physical system with the objective of mirroring the physical counterpart and reproducing its processes. There are three levels of integration between the physical object and the DT, as described by Kritzinger et al. (2018):

- Digital Model: does not have automatic data exchange between the physical and digital objects. Changes in the state of the physical object do not lead to changes on the digital object and vice versa. The level of complexity and detail of the digital model may vary.
- Digital Shadow: has one-way automatic data flow between the physical and the digital objects. The state of the digital object is affected by changes in the physical object's state, not vice versa.
- Digital Twin: has two-way data flow between the objects and a change in one of the objects' state leads to a change in the other.

In the work of Selçuk et al. (2021), a DT of a motor and gearbox system was used to generate synthetic vibration data. The individual components' behavior was modelled, followed by the overall system behavior. The vibration signal is modelled by a subsystem that converts the motor and gearbox rotational displacement at the output to translational motion through masses and spring, where the vibration is measured from the spring-damper chain by virtual sensors. A vibration dataset was generated, simulating gearbox tooth fault and vibration sensor drift error. Several time, frequency, and time-frequency domain features were extracted for training ML models for diagnostics. The best result was achieved by a RF-based approach, with 99% accuracy. This is a example of a digital model, as there is no data flow between the virtual model and the physical system.

In terms of rolling bearing simulation, Farhat et al. (2021) propose a rolling bearing DT for fault severity classification. The DT models a bearing fault detection test bench and is used to generate synthetic vibration data, using the same operation conditions of the real bench experiments, to train a ML model to diagnose the physical bearing. With a Support Vector Machine (SVM), the DT-based approach achieved a 84% accuracy, and for comparison the conventional approach, which is trained with real data, achieved 90% accuracy.

3.7 Synthesis of the State of the Art

After reviewing the literature, it is possible to trace a plan for the experiments that will lead to the final approach. As stated in Chapter 1, the objective of this work is to create a model capable of estimating the health condition of a crane translation system. Although there are multiple sensor signals acquired from the translation system, there are no health condition labels available, thus limiting the solution to unsupervised ML algorithms, but, as the sampling frequency is too low ($\frac{1}{60}Hz$), those algorithms are also impracticable, since there are not enough information with such frequency.

Considering the circumstances mentioned above and what was presented in this chapter, the preferred course of action was to emulate the translation system, in other words, to create a Digital Twin. Although the real data available is not sufficient for PdM algorithms, they are worth for determining the crane operation regime. By the types of DTs presented in section 3.6, it would be possible to develop a digital shadow, where an automatic data flow from the physical system would change the state of the digital object.

The rolling bearing is the component chosen for virtualization in the DT, for its presence in piratically every rotating machinery, including the crane translation system. Modelling the electrical motor and its faults was also considered [Foito et al., 2015], but it is a complex task that could not be featured in this work because of time constraints. Nonetheless, several abstractions had to be made to simulate and generate the synthetic vibration data, as it can be seen in Chapter 5.

In terms of DL model, CNNs is highly popular and accurate, yet choosing the CNN as model implies that the architecture must be designed. Different architectures were presented in this chapter, giving practical knowledge of the most used types of layers. Max-pooling and subsampling layers are the ones used to perform subsampling, although some approaches do not use them. Batch normalization and dropout layers are also used between the main convolutional layers, but also not unanimously. What can be pointed as common is the number of layers (considering each group started by a convolutional layer as a single layer): they are commonly 2 or 3. As there is no predetermined solution that can achieve good results, and to make a contribution in terms of automatic hyperparameter

and architecture optimization, Bayesian Optimization is going to be used to test different layer and parameter combinations, instead of relying on manual tuning.

The use of AEs for diagnostics, by itself or combined with a CNN, could also be a promising solution. Nonetheless, training AEs is highly time consuming. Moreover, CNN models were shown to achieve accuracy above 90% for rolling bearing diagnostics. Thus the implementation of AEs is only worthwhile if the raw data or extracted features were insufficient.

For feature extraction, the usage of the STFT for time-frequency analysis can be beneficial and improve the solution's performance, being also viable in terms of computational time. In the experimentation phase, the STFT is used and the results are compared to the model with raw data.

All model tests are carried out with the CWRU dataset to assess the diagnostics performance. The best model is used to diagnose the synthetic vibration data generated by the translation system emulator.

Chapter 4

Initial Work

This chapter presents what was implemented during the initial phase of this work. It comprised two approaches, one for the PRONOSTIA use case [Nectoux et al., 2012], and the other for the ASTANDER use case. The former was developed as an experiment to create a DL model capable of mapping input data directly to RUL values, while the latter was developed right after the data was made accessible, though no concrete information was given about the meaning of each sensor variable, making it a purely experimental approach. Both approaches did not make it into the main approach, i.e., the crane translation system PdM emulator. Nevertheless, these approaches still have value in terms of PdM research, and can be used for future work, if the necessary requirements, that were not fulfilled during this work, are met.

4.1 PRONOSTIA Use Case

Before it was known that the crane data sampling frequency could not be increased and labelling would not be available, the development of a supervised DL model for prognostics was prioritized. The objective was to have a model that could map raw data directly to target RUL values (labels). Thus, the PRONOSTIA dataset was chosen as public prognostics benchmark. This approach was ultimately presented at the CONTROLO2022 conference and published [Neto et al., 2022].

4.1.1 Dataset Description

This dataset was published as part of the IEEE PHM 2012 Prognostic Challenge. It contains 17 rolling bearing run-to-failure experiments. The test bed used for the experiments can be seen in Figure 4.1.

The distinctive aspect of this dataset is that experiments feature natural bearing degradation, i.e., there are no artificially induced faults, the bearings start healthy at each experiment and degrade over time until a failure occurs. Moreover, there is three experiment operation regimes:

- Op. regime 1: 1800 rpm and 4000 N.
- Op. regime 2: 1650 rpm and 4200 N.
- Op. regime 3: 1500 rpm and 5000 N.

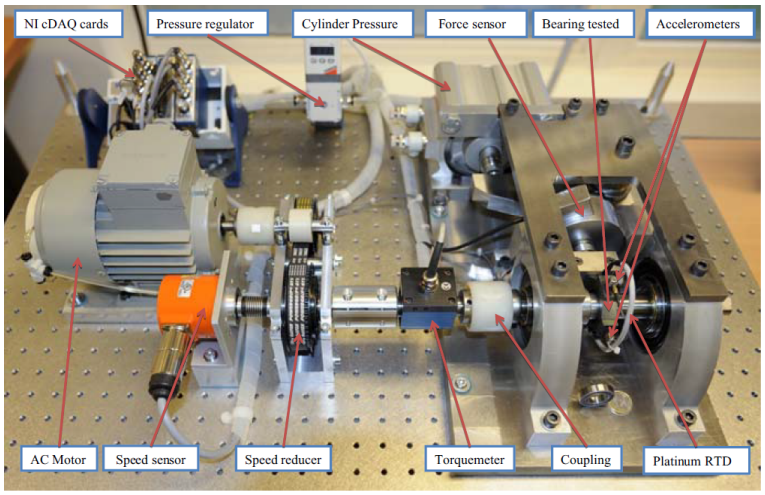

Figure 4.1: PRONOSTIA-FEMTO test bed [Nectoux et al., 2012].

The full description of how the datasets are grouped between operation regimes and training and test set are shown in table 4.1.

Table 4.1: PRONOSTIA-FEMTO datasets description.

	Operation Regime		
	Op. Regime 1	Op. Regime 2	Op. Regime 3
Training set	Bearing1_1	Bearing2_1	Bearing3_1
	Bearing1_2	Bearing2_2	Bearing3_2
Test set	Bearing1_3	Bearing2_3	Bearing3_3
	Bearing1_4	Bearing2_4	
	Bearing1_5	Bearing2_5	
	Bearing1_6	Bearing2_6	
	Bearing1_7	Bearing2_7	

The vibration data is measured by accelerometers at a sampling frequency of 25.6 kHz. A visual representation of one of the datasets (Bearing1_1) can be seen at Figure 4.2.

No information about what are the types of failures is available, making the PRONOSTIA-FEMTO dataset a benchmark mainly for prognostics, although some unsupervised diagnostics approaches exist.

4.1.2 Prognostics Model

The LSTM was chosen as the DL model to be trained and compared with other non-deep ML models. As stated in Chapter 2, the LSTM is an improved variant of the RNN, capable of learning sequential information, which is adequate for this dataset, as it represents bearing degradation through time. Figure 4.3 shows the architecture obtained from the optimization experiments, using Bayesian Optimization.

In terms of hyperparameters obtained from the optimization experiments, giving a total of 50 iterations, the following results were obtained, as listed below.

- Number of LSTM hidden units: 64.
- Number of neurons in the fully connected layer: 56.

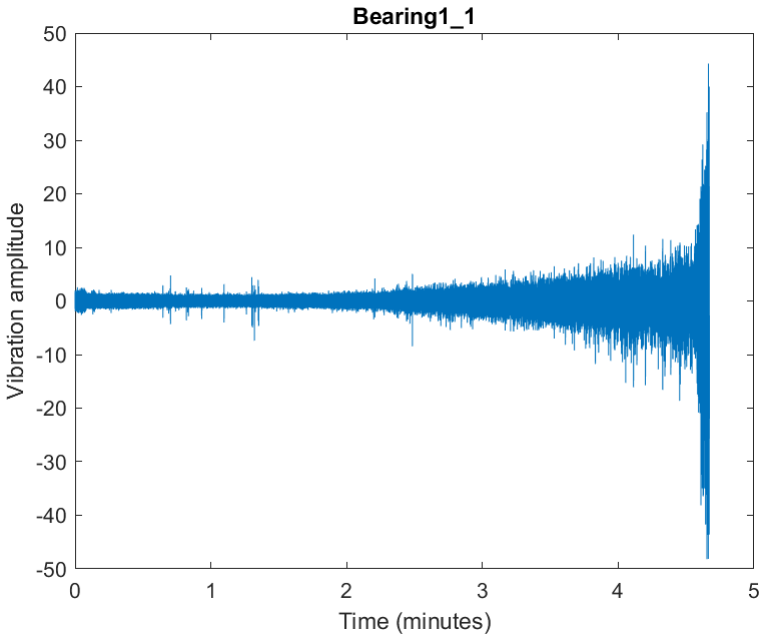

Figure 4.2: Bearing1_1 vibration data, from the PRONOSTIA dataset.

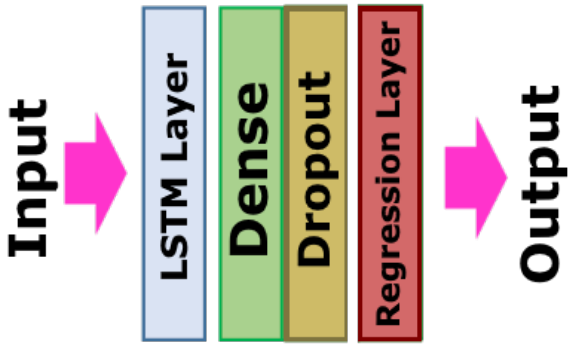

Figure 4.3: LSTM architecture used for RUL prognostics.

The input data was separated into 256-sample segments. There is no need for Cross-validation in this use case, as the training and test sets are already defined for benchmark comparison. The three best models among all the trainable models provided by the MATLAB's Classification Learning app were chosen. These models were also optimized with Bayesian Optimization. They are the following: Random Forest, Gaussian Process Regressor, and Support Vector Machine.

The usual metric for assessing prognostics approach is the Root Mean Square Error (RMSE), as it is given below.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.1}$$

$$RMSE = \sqrt{MSE} \tag{4.2}$$

Figure 4.4 shows the results of RUL prediction using the trained LSTM model for two

subsets.

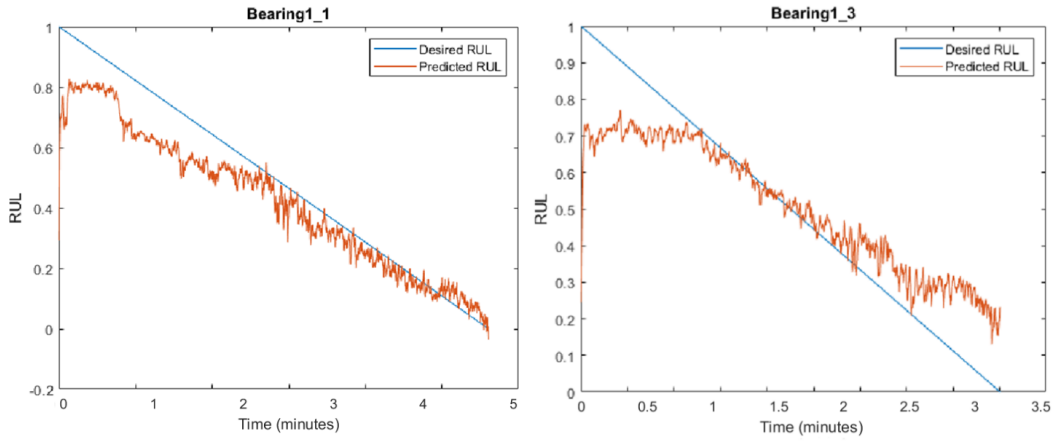

Figure 4.4: RUL predictions for the Bearing1_1 and Bearing1_3 subsets.

Finally, table 4.2 shows the results for all test subsets in terms of RMSE and comparison with the three best non-deep models.

Table 4.2: Comparison between the developed model (LSTM) and other non-deep approaches.

Model	RMSE
LSTM	**0.254**
SVM	0.288
GPR	0.334
RF	0.353

The results were considered satisfying, as the LSTM model outperformed the non-deep counterparts and was able to predict the RUL fairly accurately. However, as it is known, real data from the crane neither had sufficient sampling frequency, nor labelling, which made the adaptation and deployment of the developed model to be unfeasible, specially for the former condition, as even if there were no labelling, there are unsupervised DL approaches for the PRONOSTIA dataset, including LSTM models [Qin et al., 2022]. The next section presents an experimental unsupervised approach developed using an earlier version of the crane data.

4.2 ASTANDER Clustering

For the specific ASTANDER use case, the dataset is assumed to be composed of a number of physical attributes/variables (e.g., temperature, vibration), that can be used to characterize the health status of a given equipment or part, as well as its degradation over time. It should be mentioned that the health status of a given component at a given time, to be diagnosed and predicted, needs at an initial phase further information, such as a time stamp and the health status class of the component.

The dataset is currently composed of 8 variables collected using a sampling time of one minute: Altivar fault code (AFC), Drive state (DS), Drive thermal state (DTS), Motor current (MC), Motor thermal state (MTS), Motor torque (MT), Output velocity (OV), Resistor thermal state (RTS). A set of $N = 13123$ instances have been used in the present

analysis, corresponding approximately to 9 days of operation. The first acquisition was performed on 2021-09-16, 09:03:00. Table 4.3 summarizes the characteristics of each raw variable (discrete or continuous) and the respective range of possible values.

Table 4.3: ASTANDER dataset characterization.

Description	Discrete	Continuous	Units
1| Altivar fault code	{0, 22, 23, 39}		*
2| Drive state	{0, 1, 2, 4, 5, 6, 11}		*
3| Drive thermal state	{}		*
4| Motor current		{0} ∪ [878, 1450]	*
5| Motor thermal state	{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11, 12, 13}		*
6| Motor torque		{0} ∪ [9, 409] ∪ [65424, 65535]	*
7| Output velocity		{0} ∪ [45, 1005] ∪ [64533, 65535]	*
8| Resistor thermal state	{0, 1, 23}		*

From the analysis of the ASTANDER data, and in accordance with the ASTANDER designation "code & state", it can be observed:

- 5 variables assume discrete values (code/state): 1|AFC, 2|DS, 3|DTS, 5|MTS, 8|RTS.
- 3 variables assume continuous values: 4|MC, 6|MT, 7|OV.

Figure 4.5 presents the values taken by these variables over time. As can be observed, the selected variables significantly differ in range, thus recommending a normalisation during the pre-processing stage.

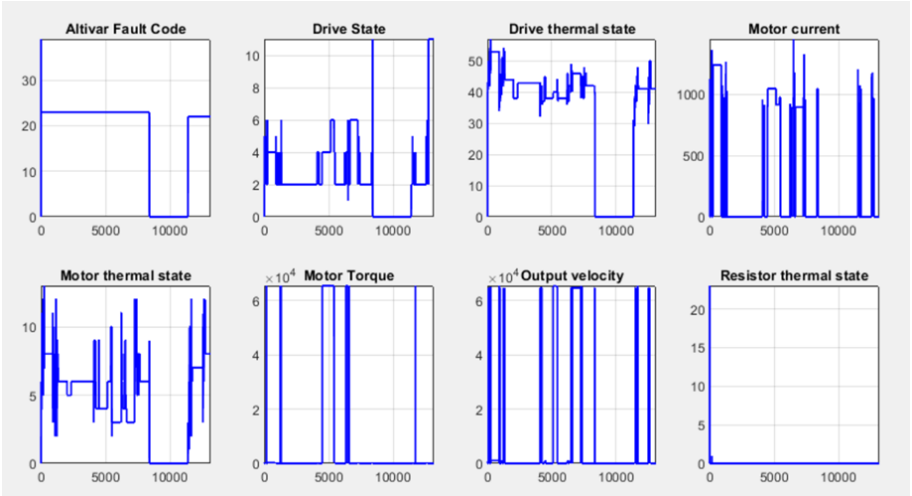

Figure 4.5: ASTANDER: raw data variables.

4.2.1 Pre-processing

Data pre-processing is an essential step to improve raw data quality for subsequent data analysis. These operations typically include noise reduction, as well as the handling of outliers and missing values, normalization and dimensionality reduction.

Outliers are measurements that significantly deviate from the expected normal pattern of sampled data. Detection and accommodation of this kind of artefacts are crucial when collected data from sensors are subsequently used to assess running conditions, such as a system part health condition or for data-based decision-making, as in the case of timely maintenance.

As can be observed from Table 4.3, the variable 6| Motor Torque presents two main distinct ranges of operation, in particular {9 .. 409} and {65424 .. 65535}. The first range comprises 91.76% of the data, while the second one 8.24%. Therefore, the occurrences corresponding to the second range (low percentage of abnormal values or stemming from some sort of fault, which needs to be elucidated) were considered at this point as outliers, being excluded from the analysis. As a result, the operation range for the variable 6| Motor Torque results in $0 \cup [9, 409]$.

Similarly, variable 7| Output Velocity presents two main ranges: 42, 1005 and 65533, 65535. The first range comprises 97.68% of the data, and the second one just 2.32%. Likewise, variable 6| Motor Torque, the occurrences corresponding to the second range were considered at this point as outliers, being excluded from the analysis. As such, the operation range for the variable 7|Output Velocity consists of $0 \cup [45, 1005]$.

Considering the elimination of the data corresponding to these abnormal ranges, a dataset composed of N=10907 was obtained. (approximately 84% of the original dataset). Moreover, after excluding the samples corresponding to the outliers, the variable 8| Resistor thermal state only assumes values equal to zero. Therefore, this variable was excluded from the current analysis. Figure 4.6 shows the remaining 7 variables, normalized in the range [0, 1].

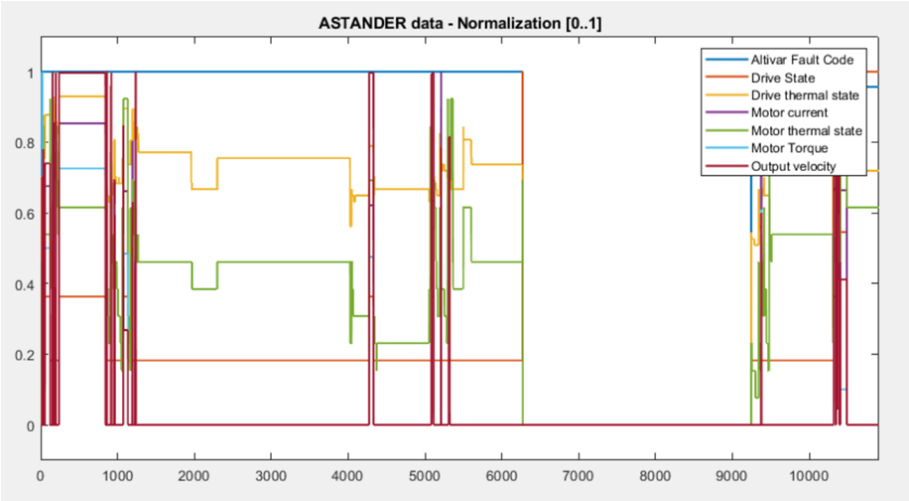

Figure 4.6: ASTANDER: normalized data.

4.2.2 Dimensionality Reduction

Dimensionality reduction facilitates the interpretation of the data, and enables to capture/extract relevant features to be used in the characterization of distinct operation modes. In the present analysis, the standard Principal Component Analysis (PCA) method was applied to perform data reduction. Three components were used {PC1, PC2, PC3} mainly for visualization purposes (3D space) and because they roughly explain the majority of the underlying information. Figure 4.7 presents the data reduction (scores) obtained

for the ASTANDER data.

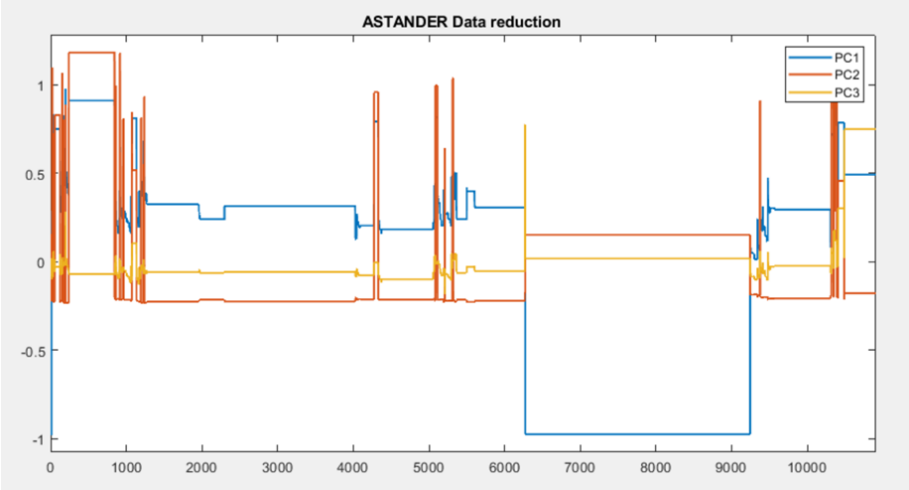

Figure 4.7: ASTANDER: data reduction {PC1,PC2,PC3}.

4.2.3 Diagnostics

To perform correct diagnostics, namely fault detection and isolation, an annotated dataset must be available. Therefore, together with the collected variables, a label associated to each sample time needs to be known, enabling to characterize the operation mode or the health status of the equipment. Based on the condition status, a supervised model can be trained with the available data and used, afterwards, to infer the health condition status and location of faults.

However, in the current ASTANDER use case, the dataset is not provided with labels. As such, it is not possible to characterize the available data, in particular to assess the health status or to distinguish the several operating modes or discriminate possible faults in the equipment.

This limitation of lacking labelled data led to using an unsupervised approach, i.e., only based on the input variables. For this purpose, clustering techniques are employed here, allowing to discriminate operating regimes, which can be normal or indefinite.

4.2.4 Clustering

In the present analysis, the standard K-means method was applied to perform clustering based on PCA scores. In order to select the number of clusters (NK), a critical parameter of the method, the Dunn's index was computed considering the number of cluster NK in the range $\{2, 3, \ldots, 10\}$. The obtained values are shown in Figure 4.8.

As can be observed from Figure 4.8, the value for $NK = 4$ seems to be a suitable compromise between the accuracy of the clustering process (assessed by Dunn's index) and complexity, determined by the number of clusters.

Figure 4.8: Clustering process – Dunn's method indexes.

4.2.5 Clusters Analysis

Figure 4.9 presents the selected normalised 7 variables of the ASTANDER use case (left Top), as well as the corresponding 3 PCA variables (left Bottom), for 3D visualization. In the right, it is shown the ASTANDER data and the respective four clusters ($NK = 4$). It should be noted that the different colours identify the distinct clusters in all figures: C1=GREEN, C2=BLACK, C3=BLUE, C4=RED .

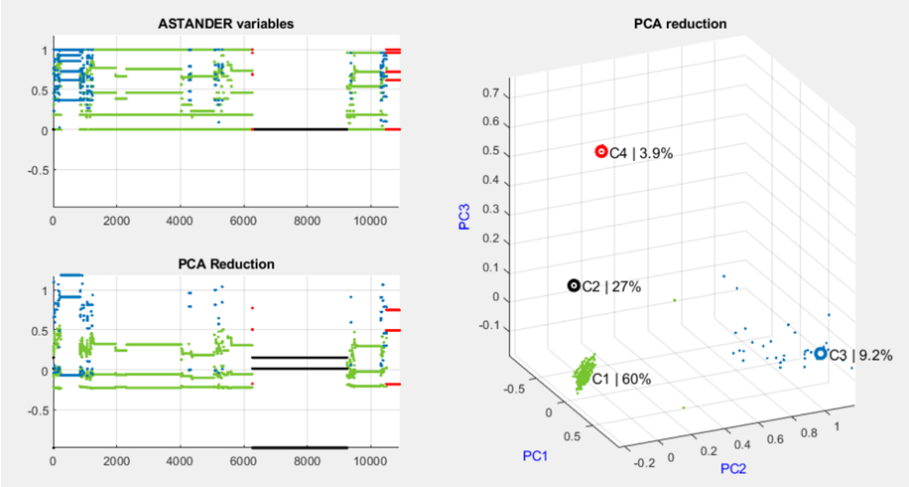

Figure 4.9: ASTANDER data analysis resulting from clustering process, $NK = 4$.

From Figure 4.9, it can be observed that the chosen 4 clusters can effectively group the data in 4 distinct operating modes (3D figure, right). Moreover, from the figures on the left (time domain), it can be observed that the operating modes C1 and C4 can be identified as "normal" operating modes. On the other hand, the operating mode C3 occurs mainly in the beginning of the acquisition process and in some few small functioning time intervals, suggesting that it is associated with a "transition" regime or possible to a fault. Finally, the operating mode C2 is clearly related to "stop" operation mode, since all variables are equal to zero.

For comparison purposes the same analysis was carried out considering $NK = 6$, as it is the first value greater than 4 which has a lower Dunn's index value, answering to the question

if a higher number of cluster should be used. As can be observed from Figure 4.10, the results are somewhat similar. In fact, the addition of two new clusters has enabled to split clusters C1 and C3 into subclasses. Basically, both "stable" and "transition" classes have been divided in two subclasses.

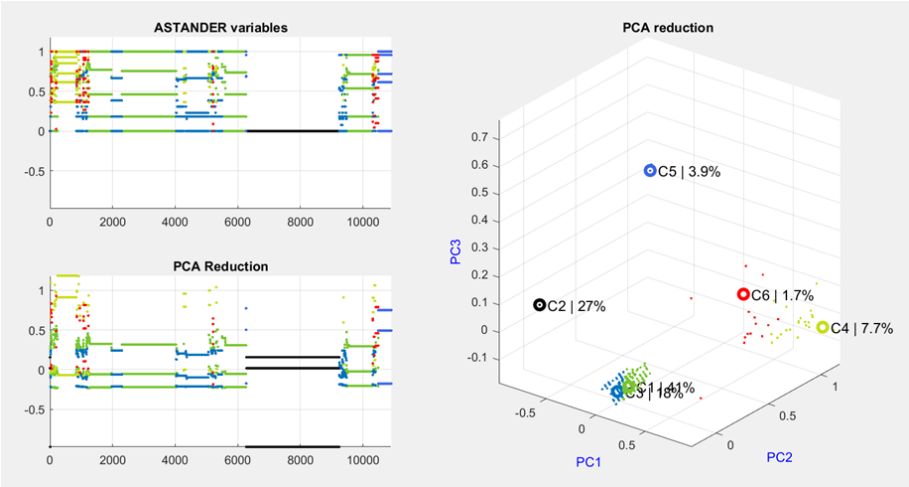

Figure 4.10: ASTANDER data analysis resulting from clustering process, $NK = 6$.

As a result, taking into account Dunn's index (Figure 4.8) and the previous comparison analysis (Figure 4.9 and Figure 4.10), a number of $NK = 4$ is assumed. The number of occurrences for each cluster is detailed in Table 4.4.

Table 4.4: Cluster distribution $NK = 4$.

	C1 OP1	C2 STOP	C3 TRANSITION	C4 OP2
Occurrences	6503	2978	999	427
Percentage	60.0%	27.0%	9.1%	3.9%

Taking into account the presented percentages in Table 4.4, it can be concluded that operating regime C1 is much more frequent (60%) than the operating regime C4 (3.9%). This may suggest the existence of distinct classes, in particular, related to normal and faulty conditions. Only a deeper understanding of this use case can clarify this hypothesis.

4.2.6 Global Analysis of the Variables

Based on the individual analysis (see Table 4.5 and Figure 4.11), it can be concluded that the specific values of the input variables have potential to distinguish between cluster C_i with $i = 1, 2, 3, 4$.

The distinction for each cluster can be described as following:

- For C2 all variables are zero (corresponding to a stop period).
- For C1 and C4 the variables 1, 2, 3 and 5 can be used to distinguish the two operating regimes.
  - Mode values for C1= $\{AFC = 23; DS = 2; DTS = 43; MTS = 6\}$.
  - Mode values for C4= $\{AFC = 22; DS = 11; DTS = 41; MTS = 8\}$.

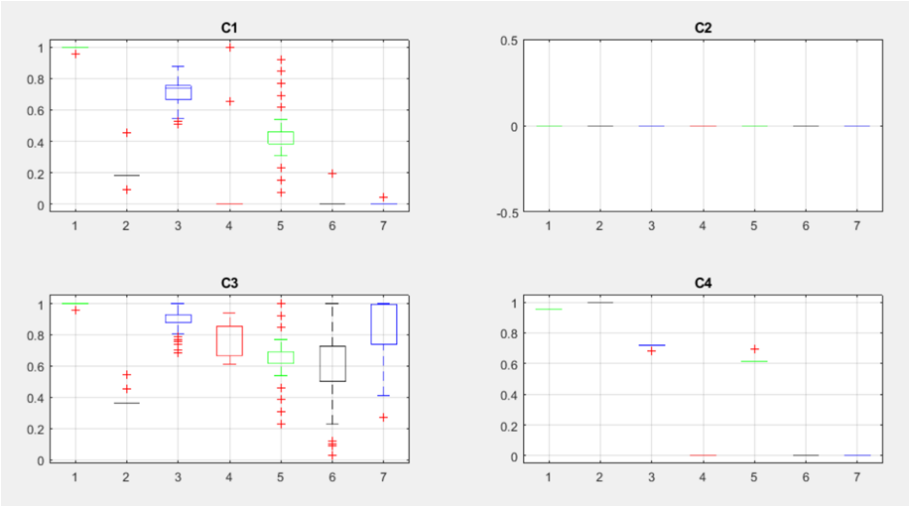

Figure 4.11: Correlation between clusters and variables.

Table 4.5: Cluster characterization.

	Cluster 1 OP1	Cluster 2 Stop	Cluster 3 Transition	Cluster 4 OP2
1 Altivar fault code	23	0	23	22
2 Drive state	2	0	4	11
3 Drive thermal state	43	0	51	41
4 Motor current	0	0	1238	0
5 Motor thermal state	6	0	8	8
6 Motor torque	0	0	219	0
7 Output velocity	0	0	1001	0
8 Resistor thermal state	*	*	*	*

- For C3 the variables 4, 6 and 7 can be used to distinguish this operating regime from the others.
  - Mode values for C3= $\{MC = 1238; MT = 219; OV = 1001\}$.
  - Modes values common to C1, C2 and C3 are $\{MC = 0; MT = 0; OV = 0\}$.

In parallel to this particular analysis, a general similarity measure (e.g., Euclidean distance) can be automatically used to compute the distance of the current triplet (PC1, PC2, PC3) scores to each one the clusters. Given these three distances, the triplet will belong to the cluster of the closest centre (the most similar cluster).

4.2.7 Real Time Diagnostics

The previous analysis suggest that the aforementioned initial hypothesis can be validated, i.e., particular values of the variables can be used to distinguish the different operating regimes. In fact, assuming that the distinct operating regimes have been computed a priori, reflected in the characterization of the several groups C1, C2, C3 and C4, it is straightforward to implement the present strategy for real time identification of the underlying operating regime.

The diagnostic process, i.e., operating regime identification, is composed of the following three steps:

1. The raw data is collected and pre-processed (outliers detection and accommodation and normalization).
2. A dimensionality reduction is performed using the pre-processed variables.
3. The reduce data is compared with the several clusters centres and the group where the reduced data belongs to is evaluated.

Figure 4.12 presents the implementation of such tool for real time identification of operating regimes. Moreover, assuming that the several operating regimes can be classified as "normal" or "faulty" regimes, the application of this strategy to detect fault regimes is straightforward.

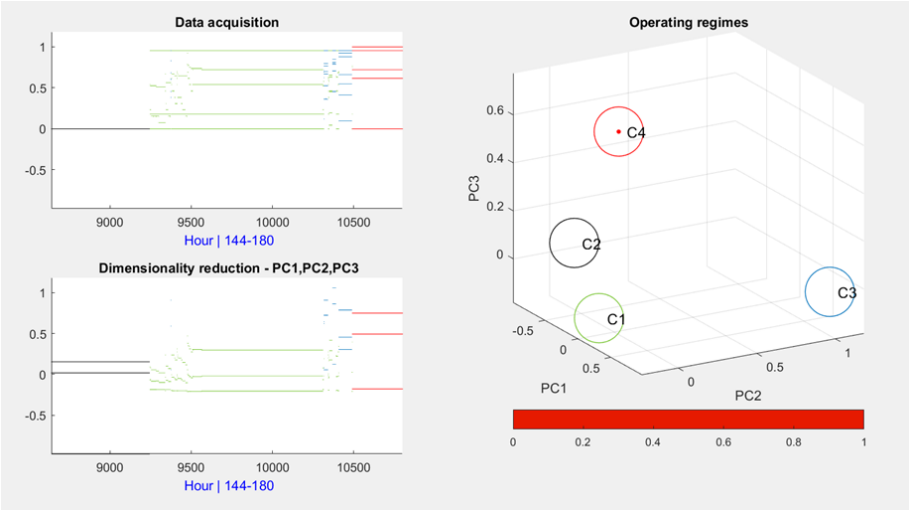

Figure 4.12: ASTANDER diagnostics tool.

4.2.8 Synthesis

Results presented in this section have been obtained assuming an unsupervised approach. In fact, the real data provided by the ASTANDER partner presents no labels, thus not allowing the validation of supervised strategies, as well as, the development of PdM strategies and RUL estimation. Therefore, additional information describing cranes is required (e.g., details of the equipment, the sensor localization, and respective units), together with the availability of labelled data, in order to come up with a reliable framework aiming to the identification of faulty regimes and health status prognostics.

This page is intentionally left blank.

Chapter 5

Main Approach Description

In this chapter, all the steps taken for the creation of the PdM approach for the crane translation system and their respective methods are presented. Figure 5.1 features the workflow followed during this development process.

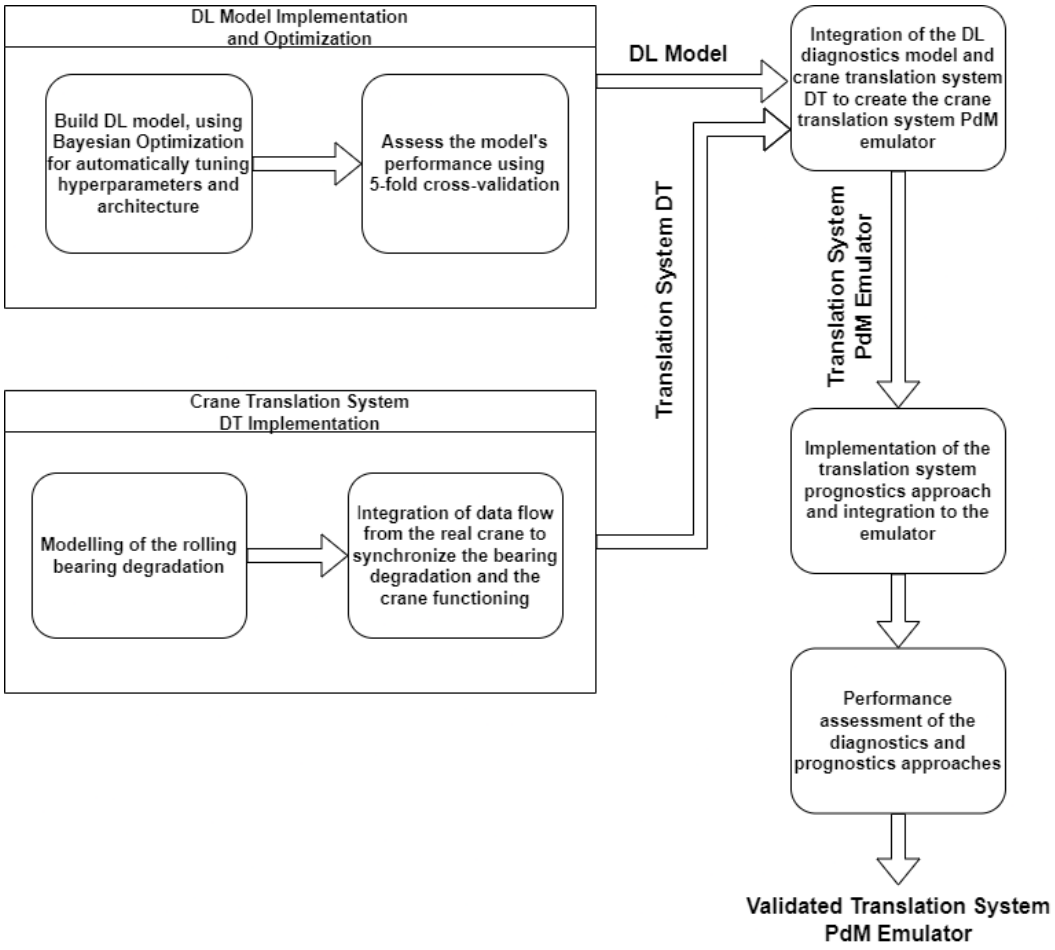

Figure 5.1: Workflow for developing the crane translation system PdM approach.

5.1 Crane Translation System Digital Twin Implementation

The objective of the crane translation system DT is to model the functioning and degradation of the system components, in this case, the rolling bearings. The idea is to create a digital shadow, where the functioning state of the digital object is directly connected to the state of the physical crane. More than 200 sensor variables were provided by ASTANDER through a database that was updated every minute. Although, just a small fraction of this variables are related to the translation system and actually provide information that can be used to determine the system operation state. The most relevant variables are presented in Table 5.1.

Table 5.1: Relevant sensor variables for detecting the crane translation system's operation state.

Variable	Unit	Range
Motor current	A	[0 197]
Motor torque	N·m	[0 66]
Output velocity		[0 65535]
Joystick translation		[-5099 5056]

There are still a few other variables related to the translation system, namely: driver state, driver thermal state, motor thermal state, and resistor thermal state. These variables do not have a unit, they represent numerical states, and the meaning of those states are ultimately unknown to this work. Thus, only the variables in Table 5.1 were considered.

With the available sampling frequency, the operation state can be distinguished between idle and moving. The rolling bearing degradation is synchronized with this information, thus no degradation occurs when the system is idle.

5.1.1 Modelling of the Rolling Bearing Degradation

Modelling of the rolling bearing degradation was performed by resorting to the PRONOSTIA dataset. The natural degradation of the rolling bearings in this dataset is the main reason why it was chosen in this work for creating the synthetic vibration data generator, as it enables the simulation of various degradation scenarios, both in terms of time and degradation magnitude. The other reason is the high sampling frequency, that allows precise time and frequency analysis, revealing more information about the underlying health conditions. Figure 5.2 shows the steps taken to perform the degradation modelling.

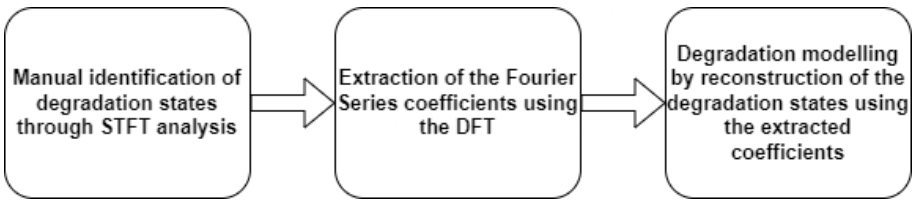

Figure 5.2: Diagram of the rolling bearing degradation modelling steps.

The first step is to manually identify the segments of the degradation states that are

going to be modelled. From the PRONOSTIA dataset, the subset named Bearing1_1 was chosen for its progressive degradation and lack of abrupt faults. Nonetheless, Bearing1_4 and Bearing2_3 subsets were also selected, this time for their sudden faults. Therefore, combinations of progressive degradation and abrupt faults can be made. Figure 5.3 presents the mentioned PRONOSTIA subsets.

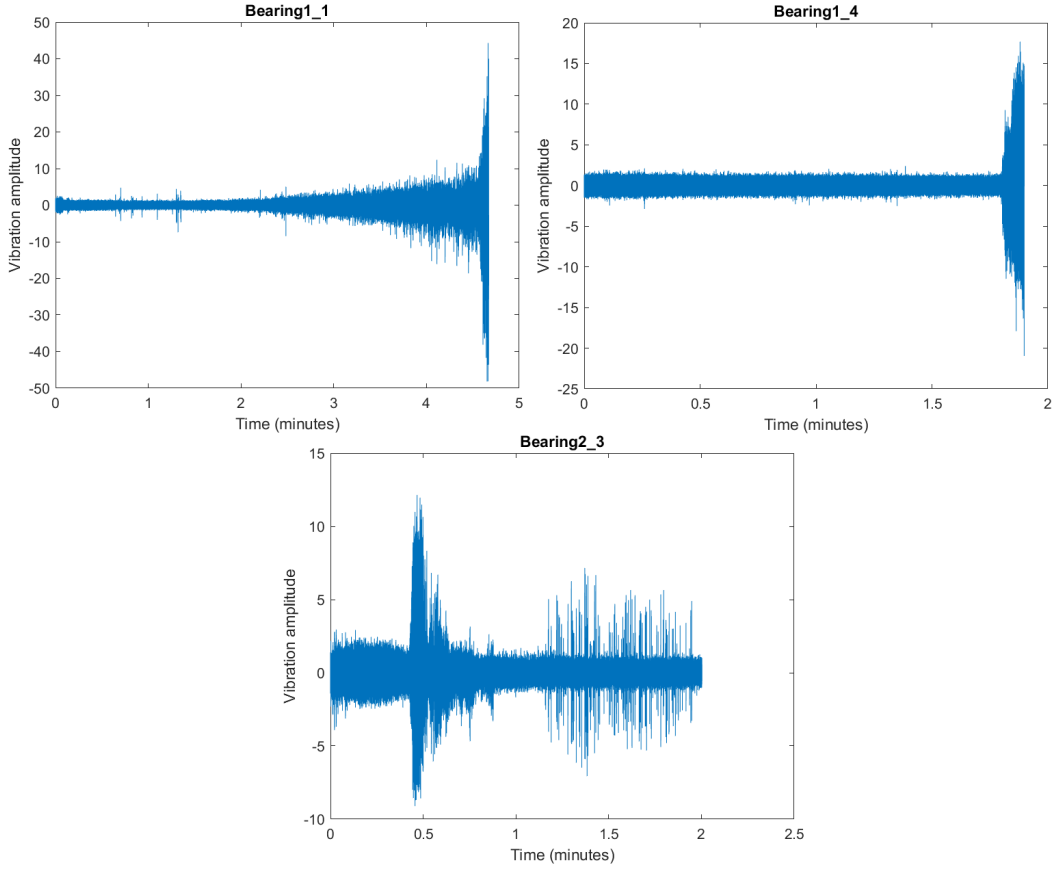

Figure 5.3: Vibration data from subsets Bearing1_1, Bearing1_4, and Bearing 2_3.

To identify the different degradation state segments, the STFT was performed for the entirety of each subset. Figures 5.4 to 5.6 show the identified states, outlined and numbered in red.

Differences between the identified states are visible both in the raw data and in the time-frequency analysis, in terms of vibration amplitude and most relevant frequencies, respectively. Now, in order to generate synthetic vibration signals with the same characteristics of those states, it is necessary to extract the Fourier series coefficients, which are later used for reconstructing the signals through the sum of trigonometric Fourier series:

$$x[n] = \sum_{m=0}^{M} C_m cos(m\omega_0 n + \theta_m) \tag{5.1}$$

where C_m and θ_m are the amplitude and phase coefficients, respectively. The previous coefficients are calculated from the Discrete Fourier Transform (DFT) coefficients, as follow:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-jk\Omega_0 n} \tag{5.2}$$

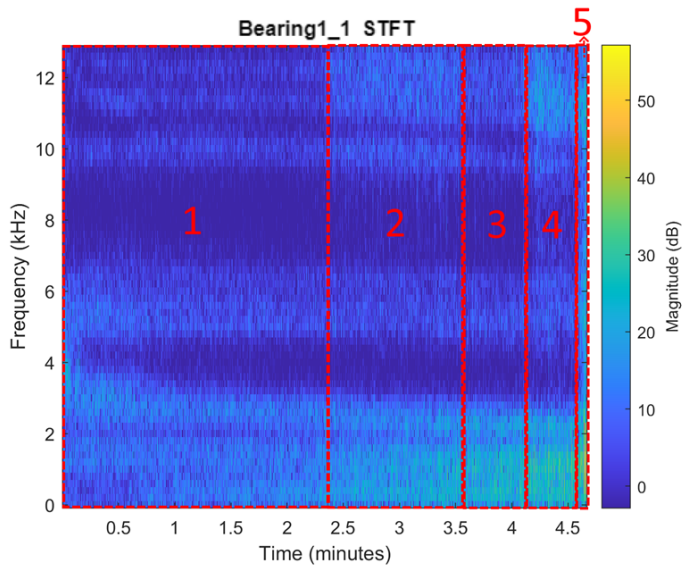

Figure 5.4: Bearing1_1 STFT analysis and degradation states identification.

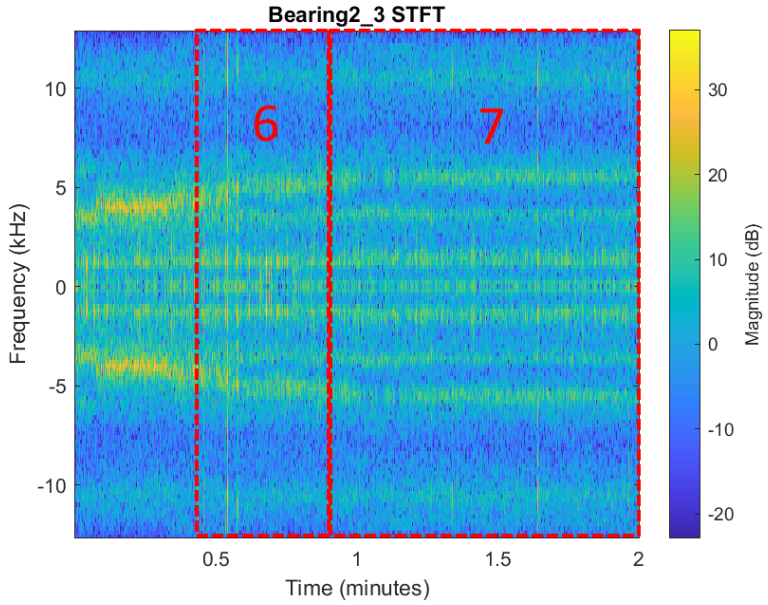

Figure 5.5: Bearing2_3 STFT analysis and degradation states identification.

$$c_m = \frac{1}{N} X[m], \quad m = 0, 1, 2, ..., N-1 \tag{5.3}$$

$$C_m = \begin{cases} |c_m|, & m = 0 \\ 2|c_m|, & m > 0 \end{cases} \tag{5.4}$$

$$\theta_m = \angle c_m \tag{5.5}$$

where in equation 5.4, $m = 0, 1, 2, ..., \frac{N-1}{2}$ when N is odd, and $m = 0, 1, 2, ..., \frac{N-2}{2}$ when N is even.

To emulate the environment noise, each value of the reconstruction is multiplied by a random value that follows a uniform distribution between 0.9 and 1.1, which means that the

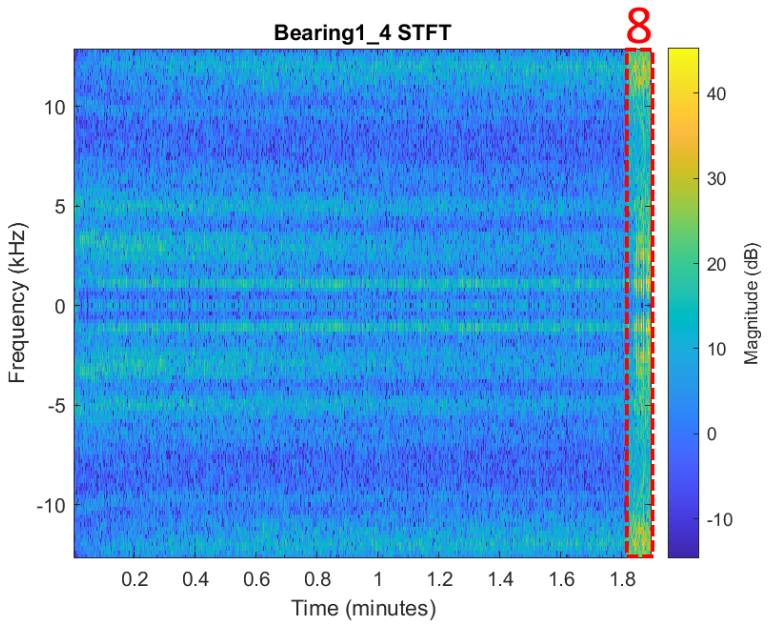

Figure 5.6: Bearing1_4 STFT analysis and degradation states identification.

signal varies by up to 10% at each point. Finally, the generation of the synthetic vibration signal is performed according to the defined degradation scenarios. Four scenarios were designed in this work:

1. Only state 1 included. This simulates an ideal scenario where no degradation to the rolling bearing occurs.

2. Progressive degradation, starting at state 1 and ending at state 5. Does not comprise sudden faults (states 6 to 8). This simulates the natural bearing degradation, without additional anomalies.

3. Similar to the previous scenario, but includes additional states 6 and 7. The occurrence of these states simulates abrupt faults that cause additional damage, but no breakdown.

4. Similar to the previous scenario, but includes state 8. This state simulates a major failure that causes imminent breakdown.

These scenarios are controlled through a set of parameters: total duration of the vibration signals (from healthy to faulty), start instant of each degradation state, duration of each transition between states (one value per transition), anomaly occurrence probability for each time step, probability distribution between anomalies (states 6 to 8).

Transitions between states are performed linearly and the superposition of states are calculated by a weighted sum, where the sum of all weights is equal to 1. When an anomaly is set to occur at the current simulation time step, all other weights are set to 0 and the anomalous state's weight is set to 1. Figure 5.7 illustrates the transitions between states, representing only 4 states.

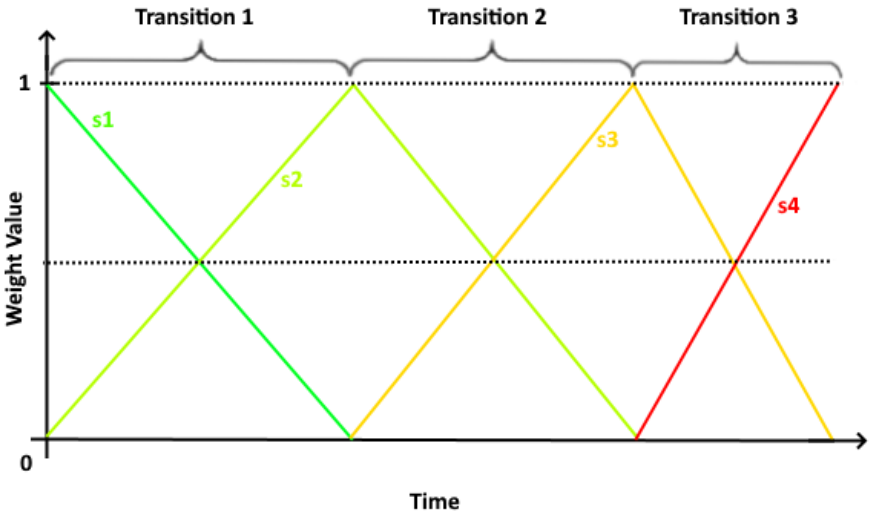

Figure 5.7: Representation of degradation states transitions.

5.1.2 Integration of Data Flow from the Real Crane

Data from the real crane is available from an online database updated every minute. Although the sampling frequency is not high enough to be directly used for diagnostics and prognostics, it can be used to define the operation regime of the crane, but in a minimalist way, by distinguishing between movement and idleness. Thus, the vibration data generation is synchronized with the crane's movements. When the crane is idle, the vibration is set to 0, and when it is moving, the vibration generation is resumed.

The definition of crane translation movement through the provided real data is straightforward in this work. First, for each sensor variable (see Table 5.1), the value ranges associated with idleness and movement are identified. Then, a simple rule is applied to classify the operation condition: if two consecutive samples of a sensor variable, or combination of variables, are within the value ranges defined for movement, then the crane is said to be moving during one minute, else, it is idle. Whether the movement rule is applied to a single variable or to multiple variables depends on a parameter that can be defined at any moment.

It is worth to be mentioned that the generated data has the same sampling frequency of the original dataset, which is 25.6 kHz, and generating this amount of data in real time is impractical in terms of computational requirements. For this reason, only a second of vibration data is generated at every minute.

At this point, the crane translation system digital shadow is complete. It is able to generate vibration data according to the given degradation scenarios. Several aspects of the generation process are parameterized, like total duration, number of degradation states, transition between states, and anomaly occurrence probability. This parameterization allows the generation process to be adapted to the desired simulation goals. This model and the DL model developed for fault diagnostics are then integrated to create the crane translation system PdM emulator.

5.2 Deep Learning Model Implementation and Optimization

Following the conclusions of Chapter 3, it was decided that a CNN model would be implemented to perform fault diagnostics from rolling bearing vibration data. Experiments were developed to find the best architecture, hyperparameters, and feature extraction technique. Table 5.2 shows the experiments that were carried out according to the CNN type and feature extraction technique.

Table 5.2: Experiment scenarios used to find the best CNN architecture.

Experiment Number	Model	Feature extraction
1	1D CNN	None
2	2D CNN	
3	1D CNN	STFT
4	2D CNN	

For each experiment, a set of parameters were optimized by the Bayesian Optimization algorithm. These parameters defined the model's architecture and training, as seen in the Table 5.3 below.

Table 5.3: Parameters optimized using the Bayesian Optimization algorithm.

N°	Parameter	Values
1	Number of convolutional layers	{1, 2, 3, 4}
2	Number of filters of the first convolutional layer	{16, 32, 64}
3	Pooling layer	{0, 1}
4	Batch normalization layer	{0, 1}
5	Dropout layer	{0, 1}
6	Number of fully connected layers	{1, 2}
7	Number of neurons of the first fully connected layer	{32, 64, 128, 256}
8	Learning rate	$\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$

It is important to note how each parameter works in the optimization algorithm:

1. Defines the number of convolutional layers in the network, from 1 to 4. Max-pooling, batch normalization, and dropout layers may be concatenated after each convolutional layer, in this very order.

2. Defines the number of filters of the first convolutional layer, from 16 to 64, but only powers of 2. For the nth layer, the number of filters is equals to 2^{f_0+n-1}, where f_0 is the number of filters defined by the parameter 2.

3-5. Boolean variables that determines if a convolutional layer is followed by a max-pooling, batch normalization, and/or dropout layer.

6. Similar to parameter 1, defines the number of fully connected layers at the end of the network.

7. Defines the number of neurons of the first fully connected layer as a power of 2 between 32 and 256. The number of neurons of the nth fully connected layer is

defined analogously to what is done with parameter 3, but is given by 2^{f_0-n+1}, halving the number of neurons in each consecutive layer.

8. Defines the initial learning rate as a power of 10 between 10^{-5} and 10^{-1}.

The value ranges of the parameters (excluding parameters 3 to 5) were selected based on literature, computational requirements, and computational power available for the experiments. Also, the number of models trained by the optimization algorithm (iterations) was set to 50 for each experiment, where 8 were trained simultaneously at each given time.

5.2.1 Model Performance Assessment with 5-fold Cross-validation

After the experiments were carried out, a 5-fold CV was used to obtain performance metrics from the best model of each experiment. The metrics are accuracy and Macro-F1 score, as given below:

$$accuracy = \frac{correct \;\; classifications}{all \;\; classifications} \tag{5.6}$$

$$F1 = 2\frac{precision * recall}{precision + recall} \tag{5.7}$$

$$MacroF1 = \frac{1}{N}\sum_{i=1}^{N} F1_i \tag{5.8}$$

Accuracy is the main metric for comparison in diagnostics approaches, although it may be biased if the dataset is not balanced, as in the case of the CWRU dataset. The Macro F1-score gives the same importance to all classes, thus if the performance in a rare class is poor, the F1-score will reflect it.

After validating the performance of the best model of each experiment, the best model was selected to be integrated into the translation system PdM emulator. Experiment results are shown in the next chapter.

5.3 Crane Translation System PdM Emulator Implementation

With the completion of the previous modules, the emulator can be implemented. The objective of the emulator is to apply the developed DL model to perform diagnostics within the degradation scenarios defined for the digital shadow. Moreover, the diagnostics results are used by the health condition prognostics module, which is explained in the next section. In total, the emulator can be divided into 4 subcomponents, as seen in Figure 5.8 below.

The first two components belong to the translation system DT, while the third is the DL model used for diagnostics. The external input is real data from the crane, obtained through the database, whilst the external output is the estimated RUL value. There is no major difference on how the emulator ultimately works compared to the DT model and DL model combined. Once data is received from the online database, the movement

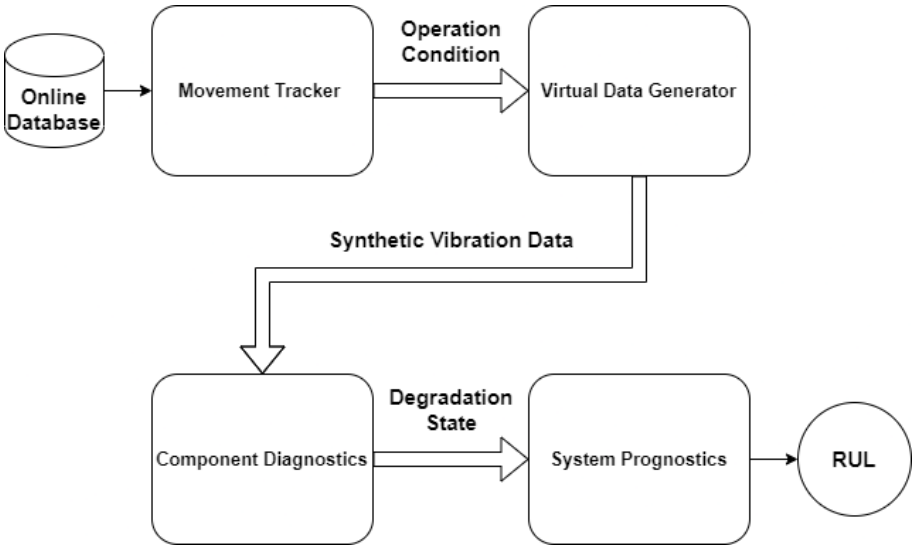

Figure 5.8: Emulator subcomponents and respective inputs and outputs.

tracker determines if the crane is idle or moving at the given moment. Next, the virtual data generator is activated when the crane is moving, generating one second of vibration data, following the defined degradation scenario. Then, the component diagnostics module classifies the vibration data according to the identified degradation state. Finally, the system prognostics module estimates the impact of the current degradation on the RUL. This last module is discussed next.

5.3.1 System Prognostics Module

In this module, the objective is to output the RUL of the translation system. For this, an approach was developed for calculating the RUL at each time step, taking into account the current degradation state of the system components, in this case, the rolling bearing. This module has a high level of abstraction, using a simple logic to calculate the RUL. First, each component degradation state is associated with a damage coefficient. At any given time step, the RUL is calculted as follow:

$$RUL_t = RUL_{t-1} - (1 + dc_i) \tag{5.9}$$

Where dc_i is the damage coefficient of the degradation state i. This equation works for discrete time, where t is time given in minutes. In simple terms, for each minute of functioning time, the system ages one minute, plus the value of the damage coefficient. If the dc_i is 1, the system will age 2 minutes for every minute of functioning. If the degradation state is misclassified by the diagnostics module, the estimated RUL will drift away from the true RUL. To evaluate the performance of the prognostics module, the RMSE is calculated between the estimated RUL and the true RUL.

Although the diagnostics module is used to identify discrete degradation states, the true RUL is calculated taking into account the transition between states. Thus, instead of using the dc_i, the coefficient is a combination given by the equation bellow.

$$dc_t = d_a \times a_t + d_b \times b_t \tag{5.10}$$

Where d_a and d_b are the damage coefficients of transitioning states a and b, respectively, and a_t and b_t are the transitioning weights of states a and b at the current time step t, respectively, and the sum of both weights is equals to 1. The idea behind the presented strategy is to mimic the behavior of component degradation in real life, even though it is through a high level of abstraction.

5.4 Emulator Validation Experiments

In order to validate and assess the performance of the diagnostics and prognostics model, a set of experiments were defined, one for each scenario (see section 5.1.1): (i) no degradation, (ii) progressive degradation, (iii) progressive degradation with sporadic non-breaking anomalies, (iv) progressive degradation including all types of sporadic anomalies.

In terms of real crane data for synchronizing the vibration data generation, a week of samples was chosen and replicated 52 times to form a full year of recording. The chosen week, dating from April 3rd to April 9th of 2022, encompasses 7 full days of work, from Sunday to Saturday (Figure 5.9). The purpose of replicating this week to a full year is to simulate the crane in non-stop service, which would be the situation of maximum degradation of any component in the crane, highlighting how the PdM approach would work.

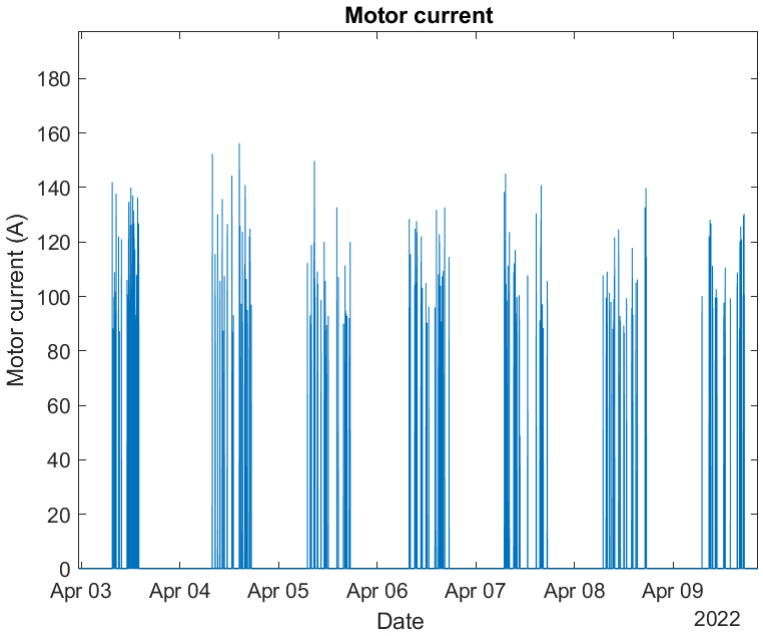

Figure 5.9: Motor current samples from the selected week of recordings.

Even though it is possible to combine different variables from the translation system to determine if the crane is moving or idle, only the joystick translation signal was chosen as input. This decision is based on the low sampling frequency, which would make a slight temporal offset between multiple signals lead to misclassifications of the operation regime. Thus relying on the commands from the crane operator is a better option.

For all experiments, the system initial RUL was set to 120 hours, which is the estimation of the translation system's total working time for one year under the work schedule created from the previously mentioned 7-day work week. Once the real or estimated RUL reach 0, the simulation stops. A few more simulation parameters are shown bellow at Table 5.4.

Table 5.4: Simulation parameters for validating the emulator.

Parameter	Value	Notes
Degradation state 1 start	0% of the initial RUL	For scenarios 2 to 4, scenario 1 has only the first state
Degradation state 2 start	48% of the initial RUL	
Degradation state 3 start	79% of the initial RUL	
Degradation state 4 start	93% of the initial RUL	
Degradation state 5 start	99% of the initial RUL	
Anomaly occurrence probability	2% every minute	0% for scenarios 1 and 2
Prob. distribution between anomalous degradation states	20%, 80%, 0%	Scenario 3 only
	17.5%, 77.5%, 5%	Scenario 4 only
Degradation coefficients	0, 0.5, 1, 2, 4, 20, 8, 500	For degradation states 1 to 8

The degradation state start parameters were based on how the natural degradation of a rolling bearing occurs. The initial and healthier states last longer than the ones close to breakdown. The probability of anomaly occurrence was also set to a low value to imitate the rarity of anomalies, even so, the value could not be much lower than 2%, or else, the anomalies would not have an impact considering the simulations time span. In terms of the probability distribution of the anomalies, the values were set to be inversely proportional to the damage caused by the respective anomaly. Finally, the degradation coefficients associate a magnitude to the degradation caused by a given state, which were assigned manually according to the perceived impact of such states.

It is important to note that, in order to simulate a breaking failure, once state 8 happens, the simulation changes the parameters. The anomaly occurrence probability is set to 100% and the anomaly probability distribution is also changed, setting state 8 to 100%. This mechanism mimics a serious and irreversible failure, if the system continues to operate, it breaks down quickly.

What is expected from the PdM approach is that the RUL estimation is sufficiently close to the true RUL, so that, for example, if an alarm system is deployed when a certain RUL value is reached, a system breakdown is prevented. If the estimation deviates far away from the true value, the system can break before the alarm is activated, or else, an false alarm can occur. The next chapter presents the results from the DL model optimization and the PdM emulator experiments results.

This page is intentionally left blank.

Chapter 6

Main Approach Results

In this chapter, the results of the CNN architecture and hyperparameter optimization with the Bayesian Optimization are presented, pointing out the best architecture found for each of the defined experiments: (i) 1D CNN with raw data, (ii) 2D CNN with raw data, (iii) 1D CNN with STFT data, (iv) 2D CNN with STFT data. The CWRU dataset was used to train and test the models, which includes all the 12 kHz subsets, previously presented at Table 3.1, plus 4 normal condition subsets. The class distribution of the used dataset is seen in Figure 6.1 below.

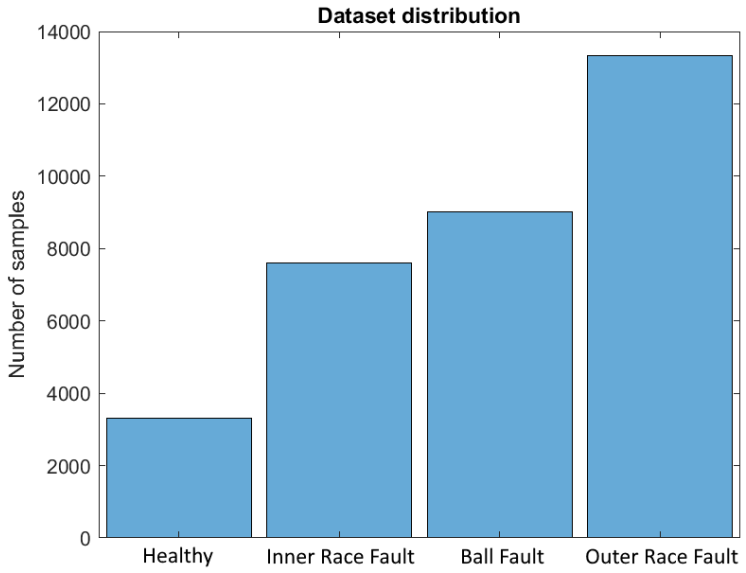

Figure 6.1: Class distribution of the used CWRU dataset.

The dataset is unbalanced, hence the importance of using the macro-F1 score, in addition to the accuracy metric. The 5-fold Cross-validation technique is stratified, i.e., each subset maintains the same class distribution ratio than the original dataset.

Next, the results of the crane translation system PdM emulator experiments are presented. The experiments are defined by the degradation scenario: (i) no degradation, (ii) progressive degradation with no random anomalies, (iii) progressive degradation with random anomalies and no breaking failures, (iv) progressive degradation with random anomalies and breaking failures.

6.1 CNN Architecture and Hyperparameter Optimization Results

Tables 6.1 and 6.2 show the results from the optimization experiments using the Bayesian Optimization algorithm. The first table presents the performance metrics of the best obtained models for each experiment, and the second table shows the architecture and hyperparameters chosen by the algorithm for these models.

Table 6.1: Cross-validation results of the best models of each architecture optimization experiment.

Experiment	Model	Data	5-fold CV Accuracy	5-fold CV Macro-F1 Score
1	1D CNN	Raw	92.93	93.29
2	2D CNN		71.14	66.29
3	1D CNN	STFT	97.85	97.98
4	2D CNN		**98.64**	**98.69**

Table 6.2: Architecture and hyperparameter optimization results.

Experiment	1	2	3	4
Data	Raw		STFT	
Model	1D CNN	2D CNN	1D CNN	2D CNN
N° of conv. layers	4	3	3	2
N° of filters (1st conv. layer)	16	32	64	32
N° of fully connected layers	3	1	2	2
N° of neurons (1s f.c. layer)	32	32	64	32
Has pooling layer	Yes	No	Yes	No
Has batch normalization layer	Yes	Yes	Yes	No
Has dropout layer	No	No	No	No
Initial learning rate	0.001	0.001	0.001	0.001
Training time	02:03:25	00:12:22	00:04:02	00:47:22

Now, Table 6.3 compares the results obtained with experiment (2D CNN model with STFT data) to the state of the art for the CWRU dataset.

Starting with the results from Table 6.1, the 2D CNN model from optimization experiment 4 achieved the best accuracy and Macro-F1 score, with 98.64% and 98.69%, respectively. The second best model is the 1D CNN from experiment 3, followed by the models from experiment 1 and 2, in this order. Both models trained with data processed with the STFT had accuracy close to the best state of the art approaches, and were considerably superior to the models trained with raw data, pointing to the already known advantages of the STFT to vibration data analysis. Nevertheless, the 1D CNN achieved good results, specially in experiment 3, which was less than 1% apart from the superior 2D CNN from experiment 4, but with a training time more than 10 times shorter.

In terms of architecture and hyperparameters defined by the optimization algorithm, the best 2D CNN from experiment 4 has 4 layers: 2 convolutional layers, which are not fol-

Table 6.3: Comparison between experiment 4 2D CNN model with the state of the art, using the CWRU dataset.

Approach	Data	Cross-validated	Accuracy	Reference
Experiment 4 2D CNN model	STFT	Yes	98.64	
ADCNN	Raw	Yes	97.90	[Guo et al., 2016]
1D CNN	Raw	Yes	93.20	[Eren et al., 2019]
Deep convolutional variable-beta VAE	Raw	Yes	99.93	[Dewangan and Maurya, 2022]
SSAE	FFT	Yes	99.15	[Wang et al., 2022]
ST-CatGAN	STFT	No	91.89	[Tao et al., 2020]

lowed by any max-pooling, batch normalization, or dropout layer, and 2 fully connected layers. The complete lack of post-convolutional layers was unexpected, since these layers are frequently used for improving accuracy, reducing overfitting and training time. A few additional ad-hoc tests were made, modifying the 2D CNN by adding all post-convolutional layers to the 2 convolutinal layers, but these modifications failed to achieve better accuracy and macro-F1 score results, yet the training time was reduced to 10 minutes, as expected.

6.2 Diagnostics Model Selection for Emulator Integration

Considering that both 1D and 2D CNNs that were trained with the STFT-processed CWRU data had similar performance compared to the state of the art, both were validated to diagnose the synthetic vibration data generated by the PdM emulator. The training and test sets were extracted from all 8 degradation states subsections identified in the PRONOSTIA dataset and processed with the STFT. The 2D CNN achieved 94.08% accuracy, while the 1D CNN achieved 97.92%, similarly to what was obtained with the CWRU dataset. Moreover, the 2D CNN took 10 minutes to be trained, while the 1D CNN took only two minutes. Thus, the 1D CNN was chosen to be integrated into the translation system PdM emulator, being a more lightweight and effective solution when compared to its 2D counterpart. The simulation results from the emulator are shown in the next section.

6.3 Translation System Emulator Simulation Results

6.3.1 Scenario 1

This first scenario is a baseline, no rolling bearing degradation is present. The system ages the same amount of time it has been functioning, until it reaches the limit stipulated by the initial RUL, equals to 120 hours (7200 minutes). Figure 6.2 shows the final simulation result.

In the first subplot, a blue line represents the true bearing degradation state, which is continuous, and the red dots represent the diagnostics classification, which is discrete, as the diagnostics is carried out for every one minute of functioning. Moments of idleness are not represented. The normal degradation in the second subplot represents the system's aging without additional degradation, the real degradation represents the true RUL evolution,

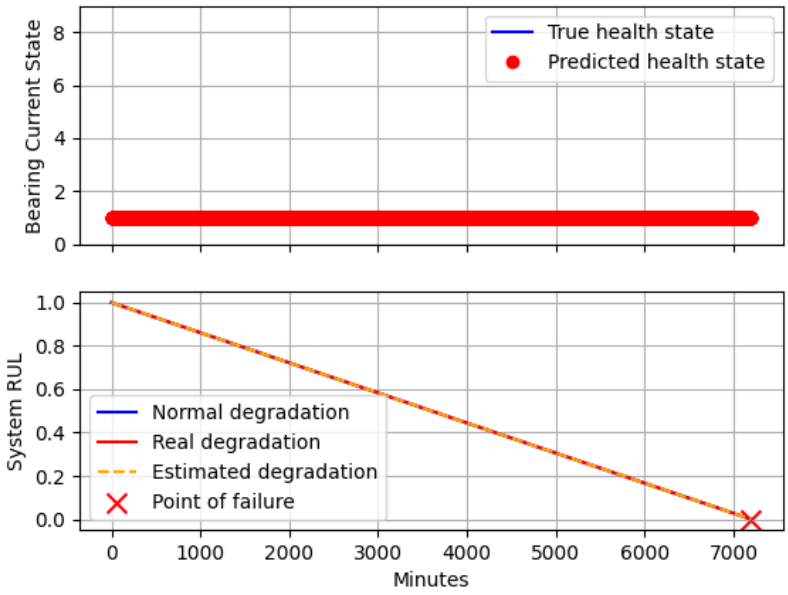

Figure 6.2: Scenario 1 final simulation results.

and the estimated degradation is the RUL estimated by the prognostics approach. In this case, all values overlap each other, as expected, given that the diagnostics and prognostics approaches succeeded at identifying the bearing degradation state and estimating an accurate RUL throughout the simulation. Diagnostics accuracy and macro-f1 score reached the best possible values of 100% and 1, respectively. RMSE was also the best attainable, equals to 0.

6.3.2 Scenario 2

For the second scenario, progressive bearing degradation was enabled, but without random anomalies. With bearing degradation, the system is expected to reach a critical point before the 120 hour mark. Figure 6.3 bellow presents the final results for scenario 2.

The first subplot shows how the bearing degradation continuously progresses to a higher level. The diagnostics model attempts to associate the input vibration data to the most likely discrete degradation state. From the diagnostics results, the prognostics is performed, calculating the impact of the current estimated degradation state to the RUL.

By analysing the scenario 2 point failure at Figure 6.4, it is possible to conclude that any alarm based on the estimated RUL and set approximately before the 7% estimated RUL mark can prevent the system's breakdown. With no alarm system or other types of contingency measures, the breakdown would match the 30% normal degradation RUL mark (blue line), in other words, if 120 hours were set for performing the translation system preventive maintenance and the rolling bearing suffered this type of degradation, the breakdown would occur 36 hours before the scheduled maintenance.

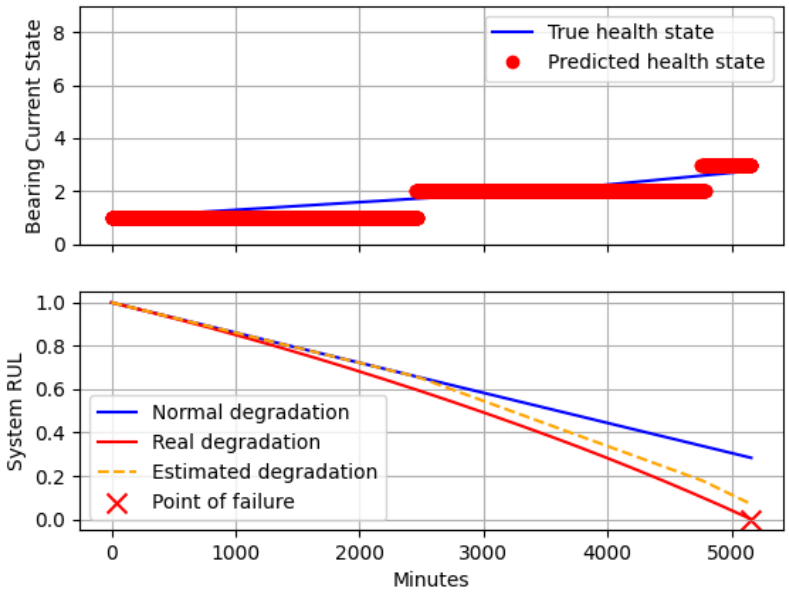

Figure 6.3: Scenario 2 final simulation results.

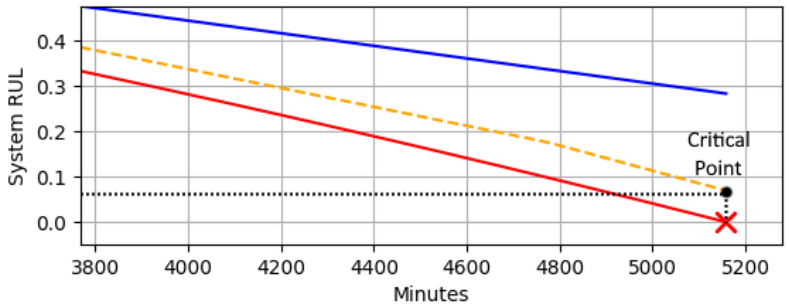

Figure 6.4: Scenario 2 point of failure analysis.

6.3.3 Scenario 3

This scenario is similar to the previous one, but it features random anomalies (degradation states 6 and 7). Those anomalies cause additional damage to the system, making the true RUL to decrease faster. It can be seen as an emulation of situations like translation system overload or overheating, which are know to increase bearing degradation. Figure 6.5 below presents the obtained results.

With the additional damage caused by the anomalies, the system reaches the point of failure 8 hours (480 minutes) earlier, compared to the previous scenario. Nevertheless, the RUL estimation was once again close to the true values, and the difference between the estimation at the point of failure was practically the same as in scenario 1 (Figure 6.6).

The losses without PdM in this scenario would be even greater than the last one, as in terms of preventive periodical maintenance, the system achieved the point of failure with approximately 35% (2500 minutes) of the initial RUL.

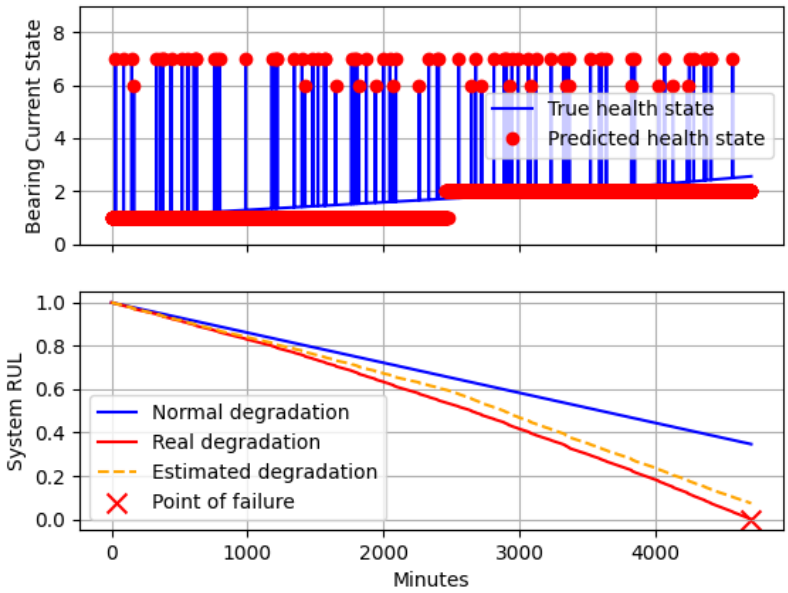

Figure 6.5: Scenario 3 final simulation results.

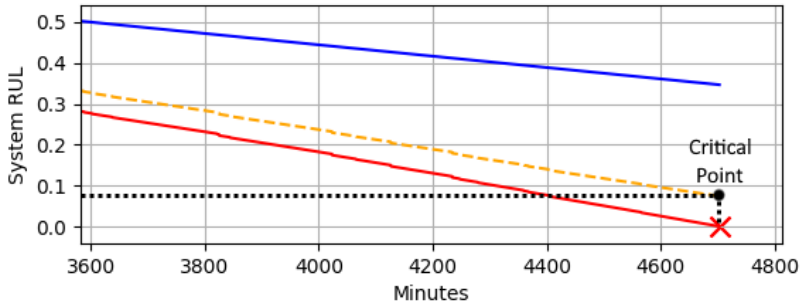

Figure 6.6: Scenario 3 point of failure analysis.

6.3.4 Scenario 4

For the final scenario, the degradation state 8, which consists of a major failure, may occur alongside the other two anomalous states. Although the occurrence of such failure in real life would be rare, the probability in this scenario was set to 5% for each time step, for demonstration purposes. Once this fault happens, the emulator is locked to the same degradation state, generating the same type of vibration data continuously. The results are presented in Figure 6.7, as follow.

The system came to a breakdown at around 80% of the initial RUL. Although it seems that the point of failure was reached instantaneously, it took around 7 minutes, counting from the degradation state 8 onset. This progression can be seen in Figure 6.8 below.

This situation, where a radical degradation state occurs, with the system continuing to run, can be seen as either lack of knowledge from the operators or just something that could happen unnoticed until the system fails, which is plausible when considering car failures caused by worn out components, or a massive and complex system as a tower crane. Nonetheless, once again an alarm system integrated into the PdM model could

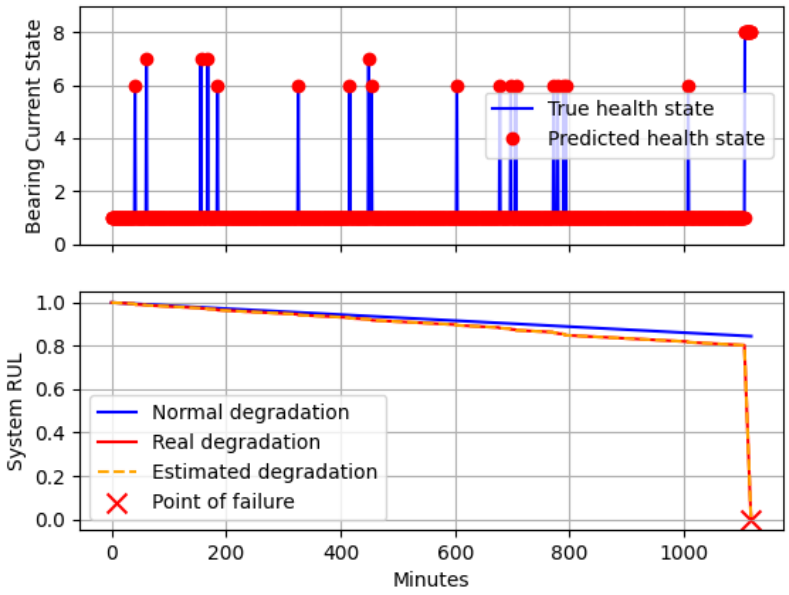

Figure 6.7: Scenario 4 final simulation results.

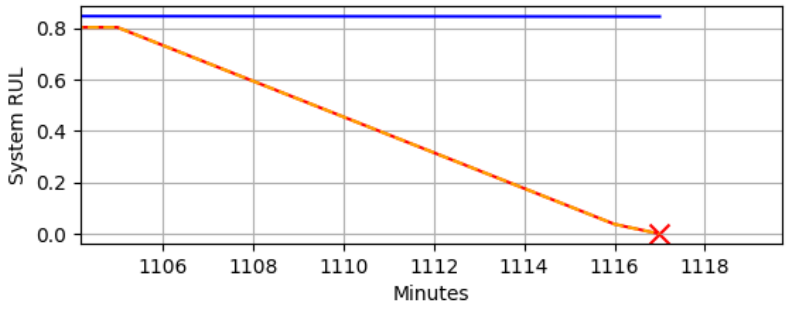

Figure 6.8: Scenario 4 point of failure analysis.

prevent the system's collapse.

6.3.5 Diagnostics and Prognostics Performance

In this section, the metrics of the diagnostics and prognostics approaches used during the simulations are presented. It is important to note that in the degradation state classification performance assessment, the true value of the degradation state, which is continuous, is rounded and compared to the discrete classifications made by the diagnostics model. Table 6.4 below presents the results obtained in the four simulation scenarios.

The results above are satisfactory. Although the accuracy is lower for scenarios 2 and 3, it can be easily explained by the fact that the diagnostics model sporadically classified the current transitioning degradation state as the previous or the next discrete state, but when the current state was rounded, it did not match the obtained classification. However, by visually inspecting the simulation results, it is possible to conclude that the classifications were accurate, specially for the anomalous states 6, 7, and 8, where the model achieved perfect accuracy.

Table 6.4: Performance metrics for the diagnostics and prognostics approaches in the simulations.

Scenario	Accuracy	Macro-F1 Score	RMSE
1	100	100	0
2	81.89	81.25	0.0475
3	81.89	73.09	0.0433
4	100	100	0

In terms of prognostics results, the perfect results of scenarios 1 and 4 can be explained by the two reasons. First, in scenario 1, there is no degradation and the diagnostics model classifies all the degradation states correctly. Thus, there is no RUL estimation error. Then, in scenario 4, the simulation ends before any progressive degradation starts, and once more, the diagnostics model has a 100% for discrete states, resulting in a perfect RMSE of 0. Even so, the prognostics results from scenarios 2 and 3 are also fairly good, which can be confirmed by visual inspection.

6.4 Main Findings and Suggestions for Future Work

6.4.1 CNN Model Tested with CWRU Dataset

The most solid contributions in this work come from the DL model developed for bearing diagnostics and tested with the CWRU dataset. The 1D and 2D CNN models with STFT-processed data achieved nearly state-of-the-art performance, affirming the applicability of the CNN and the STFT for bearing diagnostics made from vibration data. Another contribution that can be pointed out is the use of Bayesian Optimization for finding good architectures and hyperparameters, considering that most approaches rely on manual tuning and exhaustive experimentation. Yet, the optimization process was constrained by the available computational power, limiting the number of optimized parameters and the value ranges to be tested.

This work also gave emphasis to the validation process, namely in the form of the Cross-validation technique, since the majority of published approaches use methods that are prone to be biased, such as the single train-test split or resubstitution methods, which make the obtained results to be questionable. Moreover, works based on the CWRU dataset tend to rely only on the accuracy metric for performance assessment, although this can also be a biased measure, as the dataset is unbalanced. Thus, in this work, the macro-F1 score was also used as a classification metric, taking into account the dataset unbalance and the performance achieved for each fault class.

Results obtained by the best model were considered to be enough for it to be applied to the diagnostics of synthetic vibration data in the translation system PdM emulator. Even so, combining the CNN model with other models, such as the auto-encoders, to create a hybrid approach can be beneficial to improve the diagnostics capabilities even further, and to optimize such complex solutions, the Bayesian Optimization technique used in this work can also be advantageous. Finally, other time, frequency, and time-frequency analysis techniques may be tested to enhance the feature extraction process.

6.4.2 Translation System PdM Emulator

The outcomes of the simulations performed with the emulator were also satisfactory, both in terms of the diagnostics and prognostics. However, interpreting the true meaning of the obtained results and metrics from a real-world perspective is difficult, as this approach is a conceptual design with a high level of abstraction, with no equivalent in the literature that could give a baseline for comparison. Nonetheless, the proposed emulation methodology can be improved in several ways.

First, if no technical restrictions are present, upgrading the data acquisition mechanisms to increase the sampling frequency would allow the data from the crane to be directly used for PdM, for example, using the already present motor current signal to diagnose faults in the electric motor, although if no fault labelling is provided the solutions are limited to unsupervised approaches. Also, if vibration sensors were to be installed at the crane's translation system to acquire rolling bearing vibration data, it would be possible to develop DL approaches for fault detection when enough data was collected. If the data acquisition mechanism currently installed at the crane could not be upgraded, providing technical specifications about the translation system components could allow the development of physical-based digital twin models to emulate the system more accurately.

If the methodology used for creating the emulator were to be expanded into a more realistic approach, improvements should be made into the bearing generation process and prognostics technique. With available bearing specifications, it is possible to create digital twin models from laboratory-built test rigs [Peng et al., 2022], allowing detailed experimentation on bearing faults. Regarding the prognostics approach, it can be improved by developing a method for estimating the real association between the bearing degradation state and the translation system health condition and RUL. This can be done using the aforementioned test rig for data acquisition, which could be extended to other system components, and DL for mapping the input data to the observed health condition, providing a more reliable and complete PdM solution.

This page is intentionally left blank.

Chapter 7

Conclusion

The initial objective of this work was to implement a PdM approach that could be directly deployed, in real time, to perform the diagnostics and prognostics of the health condition of a tower crane translation system. As such, two initial approaches were developed. The first, implemented as a generic approach that could later be adapted to the real-world use case, was developed using the PRONOSTIA rolling bearing dataset, featuring a LSTM model that directly mapped the input vibration data to the target RUL values. This approach came to be published and presented in a conference, but it could not be used for the crane use case, as the data sampling frequency and lack of labelling made its adaptation and deployment unfeasible. Next, a unsupervised clustering technique was applied to the available crane data, as attempt to create what could become a anomaly detection approach. In fact, a number of operation states were identified, but with the absence of additional information about the sensor variables and the actual operation states, the model remained as purely experimental.

As an alternative solution to the works that were limited by the previously mentioned problems, an approach for rolling bearing diagnostics and crane translation system health condition prognostics was proposed, resorting to synthetic data generation. The bearing diagnostics model was first developed and tested with the CWRU dataset. Multiple experiments were designed to find a combination of CNN architecture and hyperparameters that could match the state-of-the-art performance for the same dataset. This process was done using Bayesian Optimization, which provided an automatic and efficient solution to the problem of manually tuning DL models. Another effort that can be valued in this work, in terms of comparison between models tested with the CWRU dataset, is the emphasis on better validation procedures, such as Cross-validation, considering that most approaches do not give attention to this step, possibly compromising the validity of the obtained results.

After achieving nearly state-of-the-art accuracy with the optimized CNN model, the model was integrated with a crane translation system digital model to diagnose synthetic rolling bearing vibration data. The translation system digital model uses real data from a crane to synchronize the bearing vibration data generation, which is based on the PRONOSTIA bearing dataset, from where frequency components of distinct degradation states were extracted and used for generating the synthetic signal.

By integrating the designed CNN model and the vibration data generation module, the crane translation system PDM emulator was created. The CNN model classifies the vibration signal into predefined degradation states, followed by a prognostics model which calculates the impact of the degradation on the system RUL. Four simulation scenarios

were set to test the emulator, from which satisfactory results were obtained, both in terms of bearing fault diagnostics and system health condition prognostics.

This work has contributed to the KYKLOS 4.0 project, as well as to Circular Manufacturing, by providing multiple PdM approaches, resources, and tools, such as the diagnostics CNN model, which achieved competitive results when compared to the state of the art. All of the presented methodologies can be reused in future works, in the context of the KYKLOS 4.0 project and PdM in general.

Several suggestions were made for the future. Concerning the bearing diagnostics model tested with the CWRU benchmark dataset, combining the CNN with other DL models, such as the auto-encoder, to create hybrid models, can further improve the feature extraction and overall diagnostics capabilities. Designing more extensive architecture and hyperparameter optimization experiments may also be advantageous, as the experiments in this work were constrained by the available computational resources.

Regarding the proposed crane translation system PdM approach, the high abstraction level on which the model was based limits the direct application of such approach to the real-world use case. Possible solutions for improving the model's realism and precision include upgrading the data acquisition mechanisms to increase the sampling frequency, so that the sensor data could be directly used for estimating the system's health condition, and not only to define if the system is moving or idle. With additional real sensors, for collecting bearing vibration data, a more accurate fault diagnostics could be performed. As an alternative to changes in the data acquisition, a laboratory-built test rig with similar specifications to the real system was pointed. This last suggestion could provide a way to better emulate the faults in the system components, not only in the rolling bearing, and supplying data that could be used to train ML and DL models with direct applicability to the real system.

This page is intentionally left blank.

References

Acerbi, F., Sassanelli, C., Terzi, S., and Taisch, M. (2021). A systematic literature review on data and information required for circular manufacturing strategies adoption. *Sustainability (Switzerland)*, 13(4):1–27.

Aloysius, N. and Geetha, M. (2018). A review on deep convolutional neural networks. *Proceedings of the 2017 IEEE International Conference on Communication and Signal Processing, ICCSP 2017*, 2018-January:588–592.

Amruthnath, N. and Gupta, T. (2018). A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance. *2018 5th International Conference on Industrial Engineering and Applications, ICIEA 2018*, (August 1993):355–361.

An, J. and Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011*, pages 1–9.

Cao, Q., Zanni-Merk, C., Samet, A., Reich, C., de Beuvron, F. d. B., Beckmann, A., and Giannetti, C. (2022). KSPMI: A Knowledge-based System for Predictive Maintenance in Industry 4.0. *Robotics and Computer-Integrated Manufacturing*, 74(November 2021):102281.

Case Western Reserve University (n.d.). Bearing Data Center, `https://engineering.case.edu/bearingdatacenter`, visited in August 2022.

Cerrada, M., Sánchez, R. V., Li, C., Pacheco, F., Cabrera, D., Valente de Oliveira, J., and Vásquez, R. E. (2018). A review on data-driven fault severity assessment in rolling bearings. *Mechanical Systems and Signal Processing*, 99:169–196.

Chen, Y., Jin, Y., and Jiri, G. (2018). Predicting tool wear with multi-sensor data using deep belief networks. *International Journal of Advanced Manufacturing Technology*, 99(5-8):1917–1926.

Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. *Proceedings of SSST 2014 - 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. pages 1–9.

Dewangan, G. and Maurya, S. (2022). Fault Diagnosis of Machines Using Deep Convolutional Beta-Variational Autoencoder. *IEEE Transactions on Artificial Intelligence*, 3(2):287–296.

Eren, L., Ince, T., and Kiranyaz, S. (2019). A Generic Intelligent Bearing Fault Diagnosis System Using Compact Adaptive 1D CNN Classifier. *Journal of Signal Processing Systems*, 91(2):179–189.

Falekas, G. and Karlis, A. (2021). Digital twin in electrical machine control and predictive maintenance: state-of-the-art and future prospects. *Energies*, 14(18).

Farhat, M. H., Chiementin, X., Chaari, F., Bolaers, F., and Haddar, M. (2021). Digital twin-driven machine learning: Ball bearings fault severity classification. *Measurement Science and Technology*, 32(4).

Foito, D., Maia, J., Fernaõ Pires, V., and Martins, J. F. (2015). Double Three-phase Induction Machine Modeling for Internal Faults Simulation. *Electric Power Components and Systems*, 43(14):1610–1620.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Greff, K., Srivastava, R. K., Koutnik, J., Steunebrink, B. R., and Schmidhuber, J. (2017). LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232.

Guo, X., Chen, L., and Shen, C. (2016). Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement: Journal of the International Measurement Confederation*, 93:490–502.

Hinton, G. E. (2009). Deep belief networks. *Scholarpedia*, 4(5):5947.

Hoang, D. T. and Kang, H. J. (2019). Rolling element bearing fault diagnosis using convolutional neural network and vibration image. *Cognitive Systems Research*, 53:42–50.

Hua, Y., Guo, J., and Zhao, H. (2015). Deep Belief Networks and deep learning. *Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things, ICIT 2015*, pages 1–4.

Kritzinger, W., Karner, M., Traar, G., Henjes, J., and Sihn, W. (2018). Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022.

Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Lee, D., Siu, V., Cruz, R., and Yetman, C. (2016). Convolutional Neural Net and Bearing Fault Analysis. *Int'l Conf. Data Mining*, pages 194–200.

Li, X., Zhang, W., and Ding, Q. (2019). Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction. *Reliability Engineering and System Safety*, 182(November 2018):208–218.

Medsker, L. R. and Jain, L. C. (2001). Recurrent Neural Networks: Design and Applications. *CRC Press*, pages 11–13.

Mobley, R. K. (2002). 1 - Impact of Maintenance. In Mobley, R. K., editor, *An Introduction to Predictive Maintenance (Second Edition)*, Plant Engineering, pages 1–22. Butterworth-Heinemann, Burlington, second edition edition.

Moyle, C. (2021). 12 Manufacturing Maintenance Statistics to Consider When Planning for 2022. https://parsable.com/blog/quality/12-manufacturing-maintenance-statistics-to-consider-when-planning-for-2020/, visited in August 2022.

Nectoux, P., Gouriveau, R., Medjaher, K., Ramasso, E., Morello, B., Zerhouni, N., and Varnier, C. (2012). PRONOSTIA : An Experimental Platform for Bearings Accelerated Degradation Tests. (June).

Neto, D., Henriques, J., Gil, P., Teixeira, C., and Cardoso, A. (2022). A deep learning approach for data-driven predictive maintenance of rolling bearings. In Brito Palma, L., Neves-Silva, R., and Gomes, L., editors, *CONTROLO 2022*, pages 587–598, Cham. Springer International Publishing.

O'Shea, K. and Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv e-prints*, pages 1–11.

Peng, F., Zheng, L., Peng, Y., Fang, C., and Meng, X. (2022). Digital Twin for rolling bearings : A review of current simulation and PHM techniques. *Measurement*, 201(July):111728.

Qin, Y., Zhou, J., and Chen, D. (2022). Unsupervised Health Indicator Construction by a Novel Degradation-Trend-Constrained Variational Autoencoder and Its Applications. *IEEE/ASME Transactions on Mechatronics*, 27(3):1447–1456.

Rauber, T. W., da Silva Loca, A. L., Boldt, F. d. A., Rodrigues, A. L., and Varejão, F. M. (2021). An experimental methodology to evaluate machine learning methods for fault diagnosis based on vibration signals. *Expert Systems with Applications*, 167(April 2020).

Sakurada, M. and Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. *ACM International Conference Proceeding Series*, 02-December-2014:4–11.

Selçuk, Y., Ünal, P., Albayrak, , and Jomâa, M. (2021). A workflow for synthetic data generation and predictive maintenance for vibration data. *Information*, 12(10).

Serradilla, O., Zugasti, E., and Zurutuza, U. (2020). Deep learning models for predictive maintenance: a survey, comparison, challenges and prospect.

SKF (2017). Bearing damage and failure analysis, https://www.skf.com/binaries/pub12/Images/0901d1968064c148-Bearing-failures---14219_2-EN_tcm_12-297619.pdf, visited in August 2022.

Tao, H., Wang, P., Chen, Y., Stojanovic, V., and Yang, H. (2020). An unsupervised fault diagnosis method for rolling bearing using STFT and generative neural networks. *Journal of the Franklin Institute*, 357(11):7286–7307.

Tinga, T. (2013). Predictive maintenance of military systems based on physical failure models. *Chemical Engineering Transactions*, 33(November):295–300.

Van Houdt, G., Mosquera, C., and Nápoles, G. (2020). A review on the long short-term memory model. *Artificial Intelligence Review*, 53(8):5929–5955.

Wang, J., Zhan, C., Yu, D., Zhao, Q., and Xie, Z. (2022). Rolling bearing fault diagnosis method based on SSAE and softmax classifier with improved K-fold cross-validation. *Measurement Science and Technology*.

Wang, Y., Yao, H., and Zhao, S. (2016). Auto-encoder based dimensionality reduction. *Neurocomputing*, 184:232–242.

Wen, Y., Fashiar Rahman, M., Xu, H., and Tseng, T. L. B. (2022). Recent advances and trends of predictive maintenance from data-driven machine prognostics perspective. *Measurement: Journal of the International Measurement Confederation*, 187(September 2021):110276.

Wu, J., Chen, X. Y., Zhang, H., Xiong, L. D., Lei, H., and Deng, S. H. (2019). Hyper-parameter optimization for machine learning models based on Bayesian optimization. *Journal of Electronic Science and Technology*, 17(1):26–40.

Yu, H., Chen, C., Lu, N., and Wang, C. (2021). Deep auto-encoder and deep forest-assisted failure prognosis for dynamic predictive maintenance scheduling. *Sensors*, 21(24):1–14.

Yuan, Y., Ma, G., Cheng, C., Zhou, B., Zhao, H., Zhang, H.-T., and Ding, H. (2018). Artificial Intelligent Diagnosis and Monitoring in Manufacturing. pages 1–18.

Zhang, J., Sun, Y., Guo, L., Gao, H., Hong, X., and Song, H. (2020). A new bearing fault diagnosis method based on modified convolutional neural networks. *Chinese Journal of Aeronautics*, 33(2):439–447.

Zonta, T., da Costa, C. A., da Rosa Righi, R., de Lima, M. J., da Trindade, E. S., and Li, G. P. (2020). Predictive maintenance in the Industry 4.0: A systematic literature review. *Computers and Industrial Engineering*, 150(April 2019).